Insu-YOLO: An Insulator Defect Detection Algorithm Based on Multiscale Feature Fusion

Yifu Chen, Hongye Liu, Jiahao Chen, Jianhong Hu and Enhui Zheng *

School of Mechanical and Electrical Engineering, China Jiliang University, Hangzhou 310018, China; p22010854011@cjlu.edu.cn (Y.C.); liuhongye1998@163.com (H.L.); p21010854010@cjlu.edu.cn (J.C.); zjuhjh@126.com (J.H.)
* Correspondence: ehzheng@cjlu.edu.cn

Abstract: To keep the balance of precision and speed of unmanned aerial vehicles (UAVs) in detecting insulator defects during power inspection, an improved insulator defect identification algorithm, Insu-YOLO, which is based on the latest YOLOv8 network, is proposed in this paper. Firstly, to lower the computational complexity of the network, the GSConv module is introduced in the backbone and neck network. In the neck network, a lightweight content-aware reassembly of features (CARAFE) structure is adopted to better utilize the feature information for upsampling, which enhances the feature fusion capability of Insu-YOLO. Additionally, Insu-YOLO enhances the fusion between shallow and deep feature maps by adding an extra object detection layer, thereby increasing the accuracy for detecting small targets. The experimental results indicate that the mean average precision of Insu-YOLO reaches 95.9%, which is 3.95% higher than the YOLOv8n baseline model, with a memory usage of 9.2 MB. Moreover, the detection speed of Insu-YOLO is 87 frames/s which achieves the purpose of real-time identification of insulator defects.

Keywords: YOLOv8 network; insulator defect detection; power inspection; image processing

1. Introduction

Insulators, which serve as insulating controls on power towers, play an important role in supporting and separating electrical conductivity during power transmission. However, with the problem of long time exposure of transmission lines to the changing outdoor environment, various unknown natural factors may lead to insulator problems, such as flashover damage and self-explosion. Therefore, insulator defect detection has become one of the significant steps in ensuring the safe operation of transmission lines. The traditional manual power inspection methods require a lot of labor and time, resulting in low inspection efficiency and also a certain risk for the inspectors. In recent years, the popularity of drone inspection technology in the power industry has greatly alleviated the burden on inspection personnel and gradually replaced manual inspection. By obtaining a large number of power inspection images or videos using high-resolution cameras mounted on drones, insulator defects can be detected, thus improving the efficiency and quality of electric power inspection.

The issue of insulator defect identification in aerial images captured by drones is worth discussing. Traditional machine learning methods commonly adopt Hough transformation, canny edge extraction, ant-colony clustering, and other algorithms. However, these traditional methods are generally suitable for cases where insulator defects present distinctive features and simple backgrounds for inspection images. In fact, the aerial images are often affected by various types of noise and their background is complex. Furthermore, inconsistent insulator defect characteristics make it difficult for traditional algorithms to identify insulator defects. Nowadays, deep-learning-based object detection methods have become the current research focus in insulator defect detection, as they have addressed...
the problems of the poor robustness of feature extraction and low real-time detection to some extent.

Deep-learning-based methods for insulator defect detection can be broadly categorized into one-stage and two-stage object detection algorithms. Typically, one-stage methods are efficient detection frameworks that offer higher frames per second (FPS), while two-stage methods tend to have higher computational complexity but provide higher detection accuracy. Recently, the emergence of transformer-based detection methods [1–4] has provided an end-to-end solution for traditional detection methods. It is undeniable that transformer-based detection methods exhibit better performance in terms of detection accuracy, but the high computational complexity and memory requirements of transformer-based methods make it challenging to deploy them effectively on edge computing devices such as Jetson Nano.

For the specific task of insulator defect detection addressed in this paper, where the platform is an edge computing device mounted on a UAV, the inference speed of the detection algorithm is of the utmost importance. In this scenario, a desirable requirement is typically an FPS greater than 40. In previous research, in order to obtain higher detection accuracy, more attention has been paid to the optimization of the two-stage detection methods. Lu et al. [5] adopted GIoU as the bounding box loss function in Faster RCNN to overcome sensitivity to multiscale insulator detection. Moreover, Soft-NMS is used during the post-processing stage to avoid detection failures posed by insulator overlapping. Zhao et al. [6] improved the feature pyramid network and applied it to Faster RCNN, resulting in better detection of insulator defects such as breakage or detachment. Zhou et al. [7] enhanced the Mask RCNN model by incorporating an attention mechanism into the backbone network, allowing the model to focus more on small objects for improved localization. Additionally, a rotation mechanism was introduced in the loss function to accurately determine the location of defects by considering various rotation angles. Although these methods achieved remarkable performance in terms of detection accuracy, their inference speed remains a significant challenge.

To address the aforementioned issue, one-stage algorithms have been introduced. Yang et al. [8] modified the feature pyramid network of YOLOv3 from unidirectional feature fusion to bidirectional fusion with the aim of improving the detection accuracy for small targets. Hao et al. [9] introduced the cross-stage partial and residual split attention networks in the backbone network of YOLOv4, which demonstrated enhanced feature extraction capability. They also employed a bidirectional feature pyramid network with a simple attention mechanism to improve the accuracy of insulator defect detection in aerial images with complex background interference. However, although they improved the one-stage detection algorithm to address the detection of small objects and challenging cases, it led to significant increases in algorithm memory consumption and computational complexity. Xu et al. [10] replaced the backbone network of YOLOv4 with the lighter Mobilenet-V1 architecture. They introduced the spatial and channel squeeze and channel excitation (scSE) attention mechanism module to enhance the feature extraction capability of the model. Additionally, they incorporated depth-wise separable convolution to reduce the overall number of network parameters. Guo et al. [11] proposed MSFT-YOLO based on YOLOv5 for detecting small target defects on the surface of steel. They enhanced the model by introducing a transformer-based TRANS module in both the backbone and the neck network, which effectively fused global information from the feature maps. Additionally, they incorporated a weighted bidirectional feature pyramid network to enable information fusion at different scales. To improve the accuracy and speed of detecting insulator defects during UAV power inspections, Han et al. [12] enhanced the backbone network of YOLOv4 by designing the D-CSPDarknet53 network, which reduced both the model’s parameters and computational complexity. Additionally, they incorporated SA-Net (shuffle attention neural networks) into the feature fusion network to enhance the model’s attention to target features. Furthermore, they introduced multi-head outputs to improve the detection accuracy of small target insulator defects. Although the improved model
effectively enhances detection accuracy and speed, there is still room for optimization in terms of the algorithm’s memory consumption and computational complexity.

In aerial inspection images, insulators usually occupy a large area, while defects only account for a small portion, as shown in Figure 1. To address the aforementioned issues, this paper proposes an improved network, Insu-YOLO, based on YOLOv8n. The contribution of the proposed method can be summarized as follows.

1. The GSConv modules are adopted in the backbone and neck networks of the Insu-YOLO model, which can reduce the number of parameters and complexity of our model.
2. The original upsampling modules in Insu-YOLO are replaced with a content-aware reassembly of features (CARAFE) structure, which ensures that the model retains the ability to extract information from small targets without losing corresponding detailed features due to interpolation-based upsampling. Additionally, the previous SPPF module is replaced with the SimCSPSPPF structure to further enhance the representational power of the model.
3. To enhance the detection capability of the model for challenging cases such as small targets and larger variance of aspect ratios, an additional object detection layer is added in Insu-YOLO, which can fuse shallow feature maps with deeper ones to optimize the detection performance of insulator defects.

Figure 1. Insulator disc defect in insulator string.

2. Related Work
2.1. YOLOv8 Basic Model

YOLOv8 is the latest object detection framework proposed by the Ultralytics company, in early 2023. Compared with the previous versions of YOLO models, YOLOv8 has achieved state-of-the-art results in basic tasks such as object detection, semantic segmentation, and image classification. The structure of YOLOv8 is shown in Figure 2.

In the backbone network, YOLOv8 still adopts the CSPNet structure [13]. The input image is scaled to a size of $640 \times 640 \times 3$. After the preprocessing step, the image is passed through the backbone network, which generates three multiscale feature maps of different dimensions: $20 \times 20 \times 256$, $40 \times 40 \times 128$, and $80 \times 80 \times 64$. Referring to the idea of ELAN in YOLOv7 [14], an effective C2f structure is introduced. As shown in Figure 3, after inputting the feature maps into the C2f module, they are passed through a Conv module composed of Conv2d, batch normalization (BN), and the SiLU activation function. Subsequently, they are split in the channel dimension, using the split module, into two parts, each with half the number of channels as the output channels. One part goes through n bottleneck modules, where the value of n depends on the scale of YOLOv8. Finally, the
resulting feature maps are concatenated and processed through another Conv module to obtain the final output.

The neck network of YOLOv8 still uses the Path Aggregation Network and Feature Pyramid Network (PAN-FPN) structure, where the C2f module is also adopted to enhance the model’s ability to fuse the global information of feature maps.

In the detecting head structure, YOLOv8 utilizes the currently predominant decoupled head to accelerate the network convergence speed while improving the detection accuracy. Furthermore, YOLOv8 replaces the traditional anchor-based method with an anchor-free approach, which allows the model to predict the location and size of objects directly without using pre-designed anchors.

In this work, YOLOv8 is adopted to serve as the insulator defect detection model and improvements are made on this basis to balance detection speed and accuracy.

Figure 2. The structure of basic YOLOv8.

Figure 3. The structure of the C2f module. T/F in the bottleneck module indicates whether the shortcut is used or not. T denotes true and F denotes false.
2.2. Characteristics of GSConv

When it comes to drone-based power line inspections, detection speed and accuracy are equally important. On the one hand, large-scale models such as ResNet [15], vision transformer [16], etc., can achieve high detection accuracy, but their detection time is too long to satisfy the timeliness. On the other hand, lightweight networks such as Xception [17], MobileNets [18–20], and ShuffleNets [21,22] greatly improve the detection speed by depth-wise separable convolution, but their lower detection accuracy renders them unsuitable for power inspection tasks. Considering the above problems, Li et al. [23] proposed the GSConv structure, as shown in Figure 4. In this structure, the input feature map is passed through a module consisting of a 2D convolutional layer, batch normalization (BN), and the activation function SiLU, resulting in a feature map with half the number of channels as the final output channel. After passing through the DWConv module, the two resulting outputs are concatenated along the channel dimension and then subjected to a shuffle operation to obtain the final output. Experimental results indicate that the adoption of the GSConv modules can reduce the model complexity while maintaining accuracy and enhancing the detection performance of insulator defects.

Figure 4. The structure of the GSConv module. The Conv2d_BN_SiLU module includes a 2D convolutional layer, a batch normalization (BN) layer, and an activation function SiLU. The DWConv module refers to the depth-wise separable convolution.

2.3. Characteristics of CARAFE

In the original YOLOv8, the feature pyramid structure utilizes the nearest-neighbor interpolation upsampling method. However, this method only determines the upsampling kernel based on the spatial location of pixels, which makes it difficult to exploit the feature information. To address this problem, Wang et al. [24] proposed the content-aware reassembly of features (CARAFE) structure. This structure has a larger receptive field, allowing for better aggregation of contextual information, while also utilizing the feature information and performing upsampling based on the input feature map. In addition, the CARAFE structure is more lightweight and only introduces a small amount of computation and parameters, making it easy to integrate into various model structures. Specifically, the input feature map is used to predict the upsampling kernels, which are independent for each location. Then, the feature reassembly is performed based on the predicted upsampling kernels. In this work, the nearest-neighbor interpolation upsampling structure in the YOLOv8 neck network is replaced with the CARAFE module, which can improve the feature fusion for insulator defects without adding extra parameters or computation.

2.4. Characteristics of Small Object Detection Layer

In the UAV inspection pictures, not only large-scale insulator strings but also small-scale insulator defects are included. For YOLOv8 with a large downsampling ratio, it is difficult for the model to learn the feature information of tiny objects in the deeper feature maps. In addition, the insulators are arranged densely in the power transmission line and may occlude one another. Furthermore, the complex background of the aerial images can introduce noise, which further increases the difficulty of detecting small objects. Insu-YOLO incorporates an additional small object detection layer, enabling feature fusion of shallow feature maps with deeper ones. Despite the increase in computational cost and
reduction in detection speed caused by adding a small object detection layer, there is a significant enhancement effect for small target detection.

3. Improved YOLOv8 Model Network Structure

In order to optimize the performance of detecting insulator defects in drone inspection images, the Insu-YOLO model is proposed, whose structure is shown in Figure 5. Insu-YOLO introduces the GsConv modules in the backbone and neck networks to reduce the number of parameters of the model and optimize its ability to extract insulator defect features, thereby improving the detection accuracy. In addition, Insu-YOLO adopts a lightweight content-aware reassembly of features (CARAFE) structure in the neck network, which enables the model to better utilize the feature information during upsampling and enhance the feature fusion capability of insulators. To enhance the accuracy of the model in detecting tiny defects, Insu-YOLO adds an additional small object detection layer to concatenate shallow and deeper feature maps before detection. Inspired by YOLOv6 v3.0 [25], this paper replaces the SPPF module with the SimCSPSPPF block, which brings a certain degree of performance improvement to our model.

Figure 5. The structure of Insu-YOLO.
The bounding box loss function used in the original YOLOv8 is CIoU, which does not take the direction between the ground-truth and predicted bounding boxes into account, resulting in a slower convergence rate during model training. Therefore, Insu-YOLO introduced the SIoU [26] as the loss function. It considers the vector angle between the ground-truth and predicted boxes, and redefines the penalty metric. SIoU consists of the following four parts.

1. **Angle cost.** The angle in this cost function is the angle between the line connecting the center points of the ground-truth and the predicted boxes. The formula is given as follows.

   \[
   \Lambda = 1 - 2 \cdot \sin^2\left(\arcsin\left(\frac{c_h}{\sigma}\right) - \frac{\pi}{4}\right)
   \]

   \[
   \frac{c_h}{\sigma} = \sin \alpha
   \]

   \[
   \sigma = \sqrt{(b^\text{gt}_cx - b_cx)^2 + (b^\text{gt}_cy - b_cy)^2}
   \]

   \[
   c_h = \max(b^\text{gt}_cy, b_cy) - \min(b^\text{gt}_cy, b_cy)
   \]

   where \(\sin \alpha\) is the ratio of the opposite side to the right-angled side in the right triangle, \(\sigma\) is the distance between the center point of the ground-truth and predicted boxes, \(c_h\) is the height difference between the center point of the ground-truth and predicted boxes, \(b^\text{gt}_cx\) and \(b^\text{gt}_cy\) are the center coordinates of the ground-truth box, and \(b_cx\) and \(b_cy\) are the center coordinates of the predicted box.

2. **Distance cost.** The distance cost is related to the minimum bounding rectangle of the ground-truth and predicted boxes. The formula of it is as follows.

   \[
   \Delta = \sum_{t=x,y} (1 - e^{-\gamma \rho_t}) = 2 - e^{-\gamma \rho_x} - e^{-\gamma \rho_y}
   \]

   \[
   \rho_x = \left(\frac{b^\text{gt}_cx - b_cx}{c_w}\right)^2, \rho_y = \left(\frac{b^\text{gt}_cy - b_cy}{c_w}\right)^2, \gamma = 2 - \Lambda
   \]

   where \(c_w\) and \(c_h\) are the width and height of the minimum bounding rectangle.

3. **Shape cost.** The definition of the shape cost is illustrated in the following formulas.

   \[
   \Omega = \sum_{t=x,y} (1 - e^{-w_t})^\theta = (1 - e^{-w_x})^\theta + (1 - e^{-w_y})^\theta
   \]

   \[
   w_x = \frac{|w - w^\text{gt}|}{\max(w, w^\text{gt})}, w_y = \frac{|h - h^\text{gt}|}{\max(h, h^\text{gt})}
   \]

   where \(w, h, w^\text{gt},\) and \(h^\text{gt}\) are the width and height of the ground-truth and predicted boxes, respectively. The value of \(\theta\) controls the degree of attention to the shape cost.

4. **IoU cost.** IoU refers to the intersection rate between the ground-truth and predicted bounding boxes, which is defined as the ratio of the intersection to the union of the two boxes, and is calculated using the following formula.

   \[
   \text{IoU} = \frac{|A \cap B|}{|A \cup B|}
   \]

   where \(A\) is the predicted box and \(B\) denotes the ground-truth box.

Combining the above four cost functions, the final SIoU loss function is calculated by the following equation.

\[
L_{\text{SIoU}} = 1 - \text{IoU} + \frac{\Delta + \Omega}{2}
\]
4. Experiments

This section, which presents the experimental results of the proposed model and other baseline models, is divided into four sections. Section 4.1 describes the datasets used in our experiment. Section 4.2 introduces the experimental environment and the experimental hyperparameters. Section 4.3 illustrates the evaluation metrics used for the experimental results. In Section 4.4, the ablation experiments are carried out, and also the experimental results of different models on the datasets are compared and discussed.

4.1. Dataset Preparation

In this article, experiments are conducted on two datasets, “CPLID” and “IDID”, to facilitate performance comparison of the proposed model with other baseline models. Several sample images of the two datasets are shown in Figure 6.

Figure 6. Sample images of the CPLID and IDID datasets. Images of insulator defects are contained in both datasets. There are some insulator images with complex backgrounds and small area of defects in CPLID dataset, as shown in figure (a,b). There are some insulator images with obvious defects in IDID dataset, as shown in figure (c,d).

(1) CPLID. The dataset “Chinese Power Line Insulator Dataset” (CPLID) [27] was collected by the State Grid Corporation of China. It contains 848 aerial images of composite insulators. In order to optimize the generalization ability of the Insu-YOLO in detecting insulators of different materials, this paper also introduces the intelligent defect detection dataset of power inspection provided by the eighth “TipDM Cup” data mining challenge in 2020, which contains 40 aerial images of glass insulators. The final dataset consists of 284 defective insulator images and 604 normal insulator images. Due to the insufficient number of defective insulator images, corresponding data augmentation operations are performed in this paper, including brightness enhancement, color deepening, contrast enhancement, etc. Finally, 2876 insulator aerial images are obtained, which contain the labels insulator defect and insulator.

(2) IDID. The public dataset “Insulator Defect Image Dataset” (IDID) [28] contains 1600 high-resolution insulator images. To make the model learn more insulator features, data augmentation operations such as brightness enhancement, color deepening, and contrast enhancement are conducted to expand the dataset, resulting in 2800 insulator images.
Before the training starts, 10% of the images from each of the above two datasets are taken as the test set, and then the remaining images are separated into the training and validation sets with the ratio of 9:1. Then, the Labeme tool is used to annotate the training and validation sets to obtain label files conforming to the PASCAL VOC dataset format. Finally, these files are converted into labels that are suitable for the YOLO format, i.e., category, normalized values of the horizontal and vertical coordinates of the center point, and the normalized values of the width and height of the bounding box. The specific division of the two datasets is shown in Table 1.

Table 1. Division of the “CPLID” and “IDID” datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total Number</th>
<th>Training Set</th>
<th>Validation Set</th>
<th>Testing Set</th>
<th>Image Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPLID</td>
<td>2876</td>
<td>2329</td>
<td>259</td>
<td>288</td>
<td>1152 × 864</td>
</tr>
<tr>
<td>IDID</td>
<td>2800</td>
<td>2268</td>
<td>252</td>
<td>280</td>
<td>4928 × 3264</td>
</tr>
</tbody>
</table>

4.2. Experimental Environment and Hyperparameters Settings

The experiments were performed on the Windows 10 operating system. An NVIDIA GeForce RTX 3090 of GPU and a 12th Gen Intel(R) Core I9-12900K of CPU were used for training and testing. The versions of CUDA and cuDNN were 11.3 and 8.2.1, respectively. The models were built based on the Pytorch 1.11.0 framework with the Python 3.8 programming language.

In our experiments, stochastic gradient descent was used as the optimizer. Its initial learning rate was set to be 0.01, and the learning rate was updated by cosine annealing schedule. The momentum and weight decay values of the optimizer were set to 0.937 and 0.0005, respectively. Each model was trained for 200 epochs, and a batch of 16 images was taken as input to the network during each round. Regarding the parameter settings for data augmentation, the fractions for augmenting the hue, saturation, and value of the input image were 0.015, 0.7, and 0.4, respectively. During preprocessing, there was a 50% chance that the image would be horizontally flipped. Additionally, the mosaic data augmentation technique was employed to enhance the model’s generalization capability and accuracy. More detailed settings of the hyperparameters are shown in Table 2. The improved Insu-YOLO model experiments were performed on an NVIDIA GeForce RTX 3090. The experimental results indicate that Insu-YOLO achieves an inference time of 87 frames/s on the CPLID dataset and 43 frames/s on the IDID dataset.

Table 2. Detailed hyperparameters used in the experiment.

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Size</td>
<td>640 × 640</td>
</tr>
<tr>
<td>Training Epochs</td>
<td>200</td>
</tr>
<tr>
<td>Batch Size</td>
<td>16</td>
</tr>
<tr>
<td>Optimizer</td>
<td>SGD</td>
</tr>
<tr>
<td>Initial Learning Rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Momentum</td>
<td>0.937</td>
</tr>
<tr>
<td>Weight Decay</td>
<td>5 × 10⁻⁴</td>
</tr>
<tr>
<td>Fraction of Hue Augmentation</td>
<td>0.015</td>
</tr>
<tr>
<td>Fraction of Saturation Augmentation</td>
<td>0.7</td>
</tr>
<tr>
<td>Fraction of Value Augmentation</td>
<td>0.4</td>
</tr>
<tr>
<td>Probability of Image Flip Up–Down</td>
<td>0</td>
</tr>
<tr>
<td>Probability of Image Flip Left–Right</td>
<td>0.5</td>
</tr>
<tr>
<td>Probability of Image Mosaic</td>
<td>1.0</td>
</tr>
<tr>
<td>Probability of Image Mixup</td>
<td>0</td>
</tr>
</tbody>
</table>

4.3. Evaluation Metrics

In order to evaluate the performance of the model, precision (P), recall (R), average precision (AP), mean average precision (mAP), and F1 score were used as evaluation
metrics. Furthermore, FPS, memory usage, and GFLOPs were also taken into account. The formulas are as follows.

\[
P = \frac{TP}{TP + FP} \tag{11}
\]

\[
R = \frac{TP}{TP + FN} \tag{12}
\]

\[
AP = \int_0^1 p(r)dr \tag{13}
\]

\[
mAP = \frac{1}{n} \sum_{i=1}^{n} AP_i \tag{14}
\]

\[
F1 - Score = \frac{2 \times P \times R}{P + R} \tag{15}
\]

In the above-mentioned formulas, TP (True Positive) represents the number of true positive samples that are classified as positive correctly, FP (False Positive) represents the quantity of false positive samples that are classified as positive incorrectly, and FN (False Negative) is the number of false negative samples that are classified as negative incorrectly. Precision refers to the ratio of examples that are correctly classified as positive out of all samples that are predicted as positive. Recall refers to the ratio of positive samples that are correctly classified as positive out of all actual positive samples. Average precision (AP) reflects the capability of the model to recognize a specific class by calculating the area under the precision–recall curve, while mean average precision refers to the average of APs for all categories. The F1 score represents the weighted harmonic mean score of precision and recall, and a higher F1 score indicates a more effective experimental method.

4.4. Experimental Results

(1) Compared with other models on CPLID dataset. To perform further validation on the detection performance of Insu-YOLO, it is compared with other models on the CPLID dataset, and the experimental results are shown in Table 3.

Table 3. Comparison of detection performance of various models on the CPLID and IDID datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>AP (Def.)</th>
<th>AP (Ins.)</th>
<th>CPLID mAP</th>
<th>FPS</th>
<th>Memory (MB)</th>
<th>P</th>
<th>R</th>
<th>mAP</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT-DETR [1]</td>
<td>90.4</td>
<td>98.8</td>
<td>94.6</td>
<td>32</td>
<td>66.1</td>
<td>91.0</td>
<td>87.9</td>
<td>91.0</td>
<td>0.894</td>
</tr>
<tr>
<td>YOLOv3t</td>
<td>88.2</td>
<td>95.4</td>
<td>91.8</td>
<td>85</td>
<td>9.2</td>
<td>81.3</td>
<td>78.2</td>
<td>85.3</td>
<td>0.797</td>
</tr>
<tr>
<td>YOLOv5n</td>
<td>87.5</td>
<td>99.5</td>
<td>93.5</td>
<td>103</td>
<td>5.2</td>
<td>92.6</td>
<td>94.3</td>
<td>97.4</td>
<td>0.931</td>
</tr>
<tr>
<td>YOLOv6n [25]</td>
<td>88.5</td>
<td>97.7</td>
<td>93.1</td>
<td>94</td>
<td>8.6</td>
<td>83.2</td>
<td>82.4</td>
<td>88.4</td>
<td>0.828</td>
</tr>
<tr>
<td>YOLOv7t [14]</td>
<td>88.1</td>
<td>98.5</td>
<td>93.3</td>
<td>65</td>
<td>12.3</td>
<td>83.8</td>
<td>89.7</td>
<td>84.3</td>
<td>0.866</td>
</tr>
<tr>
<td>YOLOv8n</td>
<td>88.7</td>
<td>99.5</td>
<td>94.1</td>
<td>118</td>
<td>6.2</td>
<td>93.8</td>
<td>93.9</td>
<td>97.5</td>
<td>0.939</td>
</tr>
<tr>
<td>BF-YOLO [8]</td>
<td>89.0</td>
<td>94.0</td>
<td>91.5</td>
<td>11</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ID-YOLO [9]</td>
<td>92.1</td>
<td>99.1</td>
<td>95.6</td>
<td>63</td>
<td>227</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>YOLOv7 [29]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Insu-YOLO</td>
<td>92.2</td>
<td>99.5</td>
<td>95.9</td>
<td>87</td>
<td>9.2</td>
<td>97.6</td>
<td>96.7</td>
<td>99.1</td>
<td>0.971</td>
</tr>
</tbody>
</table>

Def. denotes insulator defects. Ins. denotes insulator string. The letter “t” in YOLOv3t denotes “tiny”, while the letter “n” in YOLOv5n represents “nano”, and so on. F1 denotes F1-Score.

In terms of mean average precision (mAP) for object detection, Insu-YOLO shows a certain degree of improvement compared to the other models. Compared to the transformer-based RT-DETR model, Insu-YOLO achieves a 1.05% increase in mAP. When compared to YOLOv3t and YOLOv5n, Insu-YOLO exhibits improvements of 4.14% and 3.35%, respectively. In comparison to YOLOv6n and YOLOv7t, Insu-YOLO demonstrates enhancements of 2.69% and 2.47%, respectively. Moreover, when compared to the baseline model YOLOv8n, Insu-YOLO achieves a 1.70% improvement. These results indicate that the enhanced Insu-YOLO model achieves higher precision in detecting insulators and their defects.
Detection speed is one of the important indicators for evaluating model performance. Due to the high memory consumption and computational complexity of RT-DETR, its detection speed is limited and only reaches 32 frames/s. In contrast, Insu-YOLO achieves a detection speed of 87 frames/s, which is an improvement of 2.35% and 33.85% compared to YOLOv3t and YOLOv7t, respectively. However, it shows a decrease of 15.53% and 7.45% compared to YOLOv5n and YOLOv6n. It is worth mentioning that the baseline model YOLOv8n has the fastest detection speed among all the models, reaching 118 frames/s, which is an improvement of 26.27% over Insu-YOLO. Nevertheless, Insu-YOLO still meets the requirements for real-time detection.

When it comes to memory usage, the transformer-based RT-DETR model has the largest value of all models, reaching 66.1 MB, which may pose challenges for deployment on mobile devices for testing purposes. On the other hand, one-stage YOLO models exhibit relatively smaller memory usage, with YOLOv5n occupying only 5.2 MB, making it the smallest among the models. The Insu-YOLO model has a model size of 9.2 MB, which represents a 48.4% increase compared to the baseline model YOLOv8n. Nevertheless, Insu-YOLO still meets the requirements of a lightweight model and remains suitable for deployment and experimentation on mobile devices.

Compared with the state-of-the-art (SOTA) models, Insu-YOLO achieves 95.9% on mAP, which is 4.81% and 0.31% better than BF-YOLO and ID-YOLO. In terms of the average precision of detecting defects, Insu-YOLO is able to achieve 92.2%, which is lower than BF-YOLO and ID-YOLO by 5.74% and 6.96%, respectively. However, as for insulator detection, compared to BF-YOLO and ID-YOLO, the average precision of Insu-YOLO can reach 99.5%, which is an increase of 16.78% and 8.03%, respectively. In addition, in terms of detection speed, Insu-YOLO is 7.9 times and 1.4 times faster than BF-YOLO and ID-YOLO. Furthermore, when it comes to memory usage, Insu-YOLO only occupies 9.2 MB, which is much smaller than ID-YOLO.

(2) Experiments on the IDID dataset. In order to further validate the generalization ability of our proposed model, this section conducts experiments on the IDID dataset. The experimental results are presented in Table 3.

In terms of the mean average precision (mAP) of the models, Insu-YOLO achieves 99.1%, which is a 1.64% improvement over the baseline model YOLOv8n. It shows increases of 16.18% and 1.75% compared to YOLOv3t and YOLOv5n, respectively, and improvements of 12.10% and 17.56% compared to YOLOv6n and YOLOv7t. In comparison to the transformer-based model RT-DETR, Insu-YOLO exhibits an 8.90% improvement in mAP. These results indicate that the improved Insu-YOLO model also maintains excellent generalization and detection accuracy on the IDID dataset.

Regarding the F1 score, Insu-YOLO exhibits significant improvements compared to other models. In comparison to RT-DETR, Insu-YOLO shows an 8.61% increase in F1 score. It demonstrates improvements of 21.83% and 4.30% compared to YOLOv3t and YOLOv5n, respectively. Similarly, it achieves enhancements of 17.27% and 12.12% when compared to YOLOv6n and YOLOv7t. Compared to the baseline model YOLOv8n, Insu-YOLO achieves a 3.41% increase in the F1 score. The experimental results demonstrate that the improved Insu-YOLO model exhibits better quality and performance.

When evaluated based on detection speed, Insu-YOLO achieves 43 frames/s, which is a 15.69% decrease compared to the baseline model YOLOv8n. Compared to RT-DETR, Insu-YOLO demonstrates a detection speed that is approximately 2.53 times higher. In comparison to YOLOv6n and YOLOv7t, Insu-YOLO shows improvements of 10.26% and 4.88% in detection speed, respectively. However, compared to YOLOv3t and YOLOv5n, Insu-YOLO experiences a decrease in detection speed of 34.85% and 15.69%, respectively. Nevertheless, Insu-YOLO is still capable of achieving real-time detection.

Compared with the SOTA model, the mAP of Insu-YOLO achieves 99.1%, which is 5.64% higher than the improved YOLOv7 [29]. As for F1 score, Insu-YOLO can reach 0.971, improving by 3.30% comparing to the improved YOLOv7 model, which demonstrates that Insu-YOLO also has robustness and generalization on this dataset. However, when
it comes to inference speed, the improved YOLOv7 model reaches 95 frames/s, which is approximately 2.2 times faster than Insu-YOLO. Nonetheless, Insu-YOLO still satisfies the requirements for timely detection.

(3) Comparison of Detection Performance on the COCO Dataset. In this paper, various models were tested on the COCO dataset, as shown in Table 4. Since the proposed Insu-YOLO model is improved based on YOLOv8n, under a uniform configuration, our method achieves a balance between detection speed and accuracy compared to previous SOTA models. Although YOLOv7n and Insu-YOLO are similar in terms of the number of parameters and computational complexity, our method exhibits slight performance improvement.

Table 4. Comparison of detection performance on the COCO dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>#Param.</th>
<th>FLOPS</th>
<th>FPS</th>
<th>AP_{test} (%)</th>
<th>AP_{val} (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YOLOXs</td>
<td>9.0 M</td>
<td>26.8 G</td>
<td>102</td>
<td>40.5</td>
<td>40.5</td>
</tr>
<tr>
<td>PPyOLOEs</td>
<td>7.9 M</td>
<td>17.4 G</td>
<td>208</td>
<td>43.1</td>
<td>42.7</td>
</tr>
<tr>
<td>YOLOv5n</td>
<td>1.9 M</td>
<td>4.5 G</td>
<td>159</td>
<td>-</td>
<td>28.0</td>
</tr>
<tr>
<td>YOLOv5s</td>
<td>7.2 M</td>
<td>16.5 G</td>
<td>156</td>
<td>-</td>
<td>37.4</td>
</tr>
<tr>
<td>YOLOv7n</td>
<td>6.2 M</td>
<td>13.8 G</td>
<td>286</td>
<td>38.7</td>
<td>38.7</td>
</tr>
<tr>
<td>YOLOv8n</td>
<td>3.2 M</td>
<td>8.7 G</td>
<td>-</td>
<td>37.3</td>
<td>-</td>
</tr>
<tr>
<td>YOLOv8s</td>
<td>11.2 M</td>
<td>28.6 G</td>
<td>-</td>
<td>-</td>
<td>44.9</td>
</tr>
<tr>
<td>Ours</td>
<td>6.4 M</td>
<td>13.8 G</td>
<td>174</td>
<td>39.5</td>
<td>39.1</td>
</tr>
</tbody>
</table>

(4) Ablation experiments. This section examines the effects of the GSConv, CARAFE, and SimCSPSPF modules and the addition of a small object detection layer on the model performance, and conducts corresponding experiments on the “CPLID” dataset. The corresponding experimental results are illustrated in Table 5. G-YOLO is the original model with the GSConv module. C-YOLO represents the model with only the CARAFE module. GC-YOLO denotes the model with the GSConv and CARAFE modules, and GCS-YOLO means the model with the GSConv, CARAFE, and SimCSPSPF modules.

Table 5. Comparison of detection performance of models under different improvements.

<table>
<thead>
<tr>
<th>GS</th>
<th>CARAFE</th>
<th>Sim</th>
<th>Det. Layer</th>
<th>F1</th>
<th>mAP</th>
<th>FPS</th>
<th>Memory (MB)</th>
<th>GFLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>88.7</td>
<td>99.3</td>
<td>0.955</td>
<td>94.0</td>
<td>118</td>
</tr>
<tr>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>89.3</td>
<td>99.5</td>
<td>0.961</td>
<td>94.4</td>
<td>130</td>
</tr>
<tr>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>89.0</td>
<td>99.5</td>
<td>0.859</td>
<td>94.2</td>
<td>135</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>89.9</td>
<td>99.5</td>
<td>0.962</td>
<td>94.7</td>
<td>96</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>90.2</td>
<td>99.5</td>
<td>0.964</td>
<td>94.9</td>
<td>93</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>92.2</td>
<td>99.5</td>
<td>0.969</td>
<td>95.9</td>
<td>87</td>
</tr>
</tbody>
</table>

GS denotes the GSConv block. Sim denotes the SimCSPSPF block. Det. Layer denotes the additional detection layer. Def. denotes insulator defects. Ins. denotes insulator strings.

As can be learned from Table 5, the mAP values of G-YOLO, C-YOLO, GC-YOLO, GCS-YOLO, and Insu-YOLO are higher than that of the baseline model YOLOv8n, reaching 94.4%, 94.2%, 94.7%, 94.9%, and 95.9%, respectively. These improved models have increased the average precision for defect detection by 0.68%, 0.34%, 1.35%, 1.69%, and 3.95%, respectively. As for detecting insulators, the average precision of these models is maintained at around 99.5%. For G-YOLO, the adoption of the GSConv block improves feature extraction capability to a certain extent. Compared with an ordinary convolutional module, the GSConv module is able to enhance model accuracy while reducing model complexity, where memory usage and computation are reduced to 5.7 MB and 7.6 GFLOPs, respectively. The experimental results show that G-YOLO achieves a 0.63% improvement in F1 score, a 10.2% improvement in detection speed, an 8.1% reduction in memory usage, and a 6.2% decrease in the number of floating-point operations (GFLOPs). C-YOLO replaces the original upsampling in the neck network with a CARAFE structure, which has a larger receptive field and can better utilize the semantic information in the feature map,
thus performing upsampling more efficiently. Although there is only a relatively weak improvement in the mean average precision of the model, its lightweight nature further enhances the detection speed, by 14.4\% compared to the original YOLOv8n. Combining the advantages of the GSConv and CARAFE modules, GC-YOLO shows a significant improvement in the average precision of defect detection compared to G-YOLO and C-YOLO. However, due to the increasing number of network layers in GC-YOLO, its inference time has greatly increased, resulting in an 18.6\% decrease in its detection speed compared to the YOLOv8n baseline model. GCS-YOLO uses the SimCSPSPPF module instead of the SPPF module. As for the results, the introduction of the SimCSPSPPF module in GCS-YOLO increases its memory usage and floating-point operations (GFLOPs) by 48.4\% and 14.8\% compared with YOLOv8n, respectively, while reducing the detection speed by 21.2\%. Nonetheless, GCS-YOLO still shows some improvement in detection performance, with a 1.69\% increase in average precision for defect detection and a 0.94\% increase in F1 score. Finally, Insu-YOLO adds an additional object detection layer based on the foundation of GCS-YOLO. The results indicate that fusing features from the first three C2f modules in the backbone network significantly improves the detection performance for tiny targets. Compared to the original YOLOv8n model, there is a 3.95\% increase in average precision for defect detection, and a 1.47\% improvement in F1 score. However, this improvement also comes with certain costs, as the introduction of an additional object detection layer leads to a deeper network and increased detection time. In particular, Insu-YOLO has a 70.4\% increase in floating-point operations (GFLOPs) and a 26.3\% decrease in detection speed. Despite this, Insu-YOLO is able to reach 87 frames/s in detection speed, which still satisfies the needs for timely detection.

The loss curves of bounding box regression during training for the different models can be seen in Figure 7. Models with different improvements show faster loss reduction in the early stage of training than the YOLOv8n baseline model. At the end of training, the loss value of Insu-YOLO is lower than that of other models, which indicates that the introduction of the GSConv, CARAFE, and SimCSPSPPF modules can optimize the feature extraction capability. In addition, the additional object detection layer can effectively enhance the ability to identify tiny defects, thereby improving detection performance.

![Figure 7. Loss curves of bounding box regression during training for different models.](image)

In order to compare the attention regions of various models in detecting targets, this paper utilizes the gradient-weighted class activation mapping (Grad-CAM) technique [30] to generate heatmaps based on the feature maps of the eighth layer of the network, which corresponds to the fourth C2f module of the backbone network. The comparison of the feature heatmaps generated by the different models on the test images is shown in Figure 8. As shown in the pictures, in the second and third rows, the region of interest of the
YOLOv8n and GCS-YOLO models is only focused on a small part when detecting the front and back insulators, respectively. The attention regions of G-YOLO and GC-YOLO are relatively scattered, with some regions focusing on power transmission towers and other background information. The attention regions of the C-YOLO and our proposed Insu-YOLO models can relatively completely cover the entire insulator target. Compared to C-YOLO, the attention regions of Insu-YOLO are more complete, resulting in a higher detection accuracy. As seen from the images in the last row, when detecting defects, the attention regions of the YOLOv8n, G-YOLO, and GCS-YOLO models are relatively scattered, including some irrelevant information. Although the area of interest of the C-YOLO and GC-YOLO models can cover the defect locations relatively well, there is a portion of attention regions of the C-YOLO model that contain the back insulator, while the attention regions of the G-YOLO model include the entire string of the insulator in front. In contrast, Insu-YOLO can focus more intensively on the defect parts, resulting in higher recognition accuracy.

Figure 8. Comparison of the feature heatmaps of various models at the eighth layer of the network. The first row (a–g) shows the original images. The second row (h–m) shows the heatmaps when detecting the front insulator. The third row (n–s) shows the heatmaps when detecting the back insulator. The last row (t–y) shows the heatmaps when detecting insulator defects. From the performance of the heatmaps, Insu-YOLO, compared to the previous models, allows the receptive fields of the model to be further expanded and has more aggregated features due to the introduction of GSConv and the upsampling module Content-Aware ReAssembly of FEatures (CARAFE), as shown in figure (m,s). For small-target defect detection, Insu-YOLO allows the fine-grained features of small-target defect detection to be enhanced by using a multi-scale approach. It is obvious to figure out from the comparison graphs in the fourth row that the attention region in figure (y) becomes more concentrated.

According to the results above, Insu-YOLO is able to focus more accurately and completely on the positions of insulators and defects than other models, thereby facilitating better feature extraction. Compared with the YOLOv8n baseline model, Insu-YOLO can achieve an F1 score of 0.969, an mAP value of 95.9%, and an inspection speed of 87 frames/s, demonstrating its excellent performance in insulator defect detection tasks.

5. Conclusions

To further improve the detection accuracy of insulator defects in power tower inspection images, this paper proposes an Insu-YOLO model for tiny-defect detection based
on the latest YOLOv8n model. By introducing the GSConv module into the backbone and neck networks, the model is able to improve the detection accuracy while reducing model complexity. Moreover, by replacing the original upsampling in the neck network with a more lightweight content-aware reassembly of features (CARAFE) block, the model is able to fully utilize the contextual information in feature maps and perform content-based upsampling, thereby enhancing the ability to fuse features for insulator defects. The original SPPF module is also replaced with the SimCSPSPPF module to optimize the representational capability. Finally, to improve the detection performance on small targets, an additional object detection layer is added to further enhance the fusion of shallow and deep feature maps. The experimental results validate the outstanding performance of Insu-YOLO. Specifically, Insu-YOLO reaches 95.9% on mAP and 0.969 on F1 score using only 9.2 MB of memory. Additionally, it achieves a detection speed of 87 frames/s, which satisfies the real-time detection requirements for insulator defects. In future research, it is necessary to further enhance the accuracy in detecting tiny defects while optimizing the detection speed, which is of great significance and application prospects for unmanned aerial vehicle power inspection tasks.

Author Contributions: Conceptualization, Y.C. and H.L.; methodology, H.L.; software, Y.C.; validation, Y.C., H.L. and J.C.; formal analysis, Y.C.; investigation, Y.C.; resources, H.L.; data curation, J.C.; writing—original draft preparation, Y.C.; writing—review and editing, Y.C. and H.L.; visualization, J.C.; supervision, J.H. and E.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no funding.

Data Availability Statement: Dataset link: https://github.com/InsulatorData/InsulatorDataSet, accessed on 19 November 2018.

Conflicts of Interest: The authors declare no conflict of interest.

References


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.