Journal of Real-Time Image Processing (2023) 20:104
https://doi.org/10.1007/s11554-023-01360-1

RESEARCH

Yolo-inspection: defect detection method for power transmission lines based on enhanced YOLOv5s

Lihui Lu[1,2] · Zhencong Chen[1] · Rifan Wang[1] · Li Liu[1] · Haoqing Chi[1]

Received: 19 May 2023 / Accepted: 21 August 2023 / Published online: 2 September 2023

Abstract
Accurate identification of defective components in transmission lines and timely feedback to inspectors for timely maintenance can ensure the stable operation of the power system. A defect detection system based on "edge-cloud-end" collaboration is introduced to solve the problems of high bandwidth consumption and response delay in the cloud server-based approach. The system transfers the operation of image detection to the edge device, which reduces the data transmission and improves the response speed of the system. To balance the detection speed and accuracy of the algorithm, the YOLO-inspection algorithm applied on edge devices is proposed. The algorithm uses GhostNetV2 to reconstruct the C3 module in the YOLOv5 model, which reduces the computational complexity and captures the correlation between distant pixels so that it is more targeted to the critical region of the defective target. Meanwhile, based on the feature fusion network, a dynamic adaptive weight assignment module and cross-scale connectivity are designed to effectively reduce information loss and help the network learn fine-grained features. The improved algorithm is deployed on the NVIDIA Jetson Xavier NX platform, and the model is optimally accelerated using TensorRT. Experimental results show that the method proposed in this paper can accurately identify defective samples, and the YOLO-inspection algorithm has superior generalization ability under the harsh conditions of low light and snowfall weather conditions. On the edge computing platform, the mean average precision (mAP) can reach 94.3%, and the inference speed can reach 63 frames per second (FPS). It can be proved that the method has good detection performance.

Keywords Transmission lines · Defect detection · Lightweight · UAV

1 Introduction

Drones have become one of the primary methods used by power companies to detect defects in power transmission lines and ensure uninterrupted electricity delivery. The defect detection algorithm's performance is the most critical factor in the entire inspection task. It directly determines the efficiency of the inspection. The mode of cloud detection is particularly prominent in UAV inspection tasks because of its powerful computing power. This mode has higher detection accuracy, but the transmission of a large amount of data will bring some delay and increase the cloud server's storage burden. With the rapid development of edge devices, the cloud computing mode is being moved to the edge gradually, avoiding the delay caused by data uploading to enable faster responses. However, this approach also has shortcomings. For example, the high-precision model detection speed cannot meet the requirements of real-time performance, and the accuracy of the model with fast detection speed is not ideal.

✉ Lihui Lu
lulihui@qfnu.edu.cn

Zhencong Chen
czc_qfnu@163.com

Rifan Wang
wwrf741852@163.com

Li Liu
ll13963608592@163.com

Haoqing Chi
chq970903@163.com

[1] School of Engineering, Qufu Normal University, Rizhao 276826, China

[2] Rizhao Huilian Zhongchuang Institute of Intelligent Technology, Rizhao 276826, China

Thus, researching a object detection method that balances detection speed and accuracy is of significant importance.

Currently, there are two main types of object detection algorithms. The first type is a one-stage object detection model, mainly including SSD [1] and YOLO [2–9]. The other type is a two-stage object detection model, mainly including Mask RCNN [10], Cascade RCNN [11], and FasterRCNN [12]. Among them, the two-stage network model has higher detection accuracy, but the model is too large to meet the real-time detection requirements and cannot be deployed on edge devices with limited computing power. Zhang et al. [13] applied the SPTL-Net shared convolutional neural network to improve Faster RCNN and achieve accurate detection of foreign objects on transmission lines. Li et al. [14] improved the faster RCNN detection network using the region of interest (ROI) mining, K-means clustering algorithm, and defocus function to achieve precise detection of bird nests on transmission lines. Although these studies achieve high detection accuracy, their inference speed is not ideal. The single-stage object detection model can maintain good detection accuracy while ensuring a high inference speed. Bao et al. [15] constructed the BC-YOLO model, which is based on YOLOv5 and introduces the CA attention mechanism and BiFPN feature fusion network to improve the model's detection ability for the damper. Liu et al. [16] used MnasNet to improve the feature extraction network in SSD and used two multi-scale feature fusion methods to fuse multiple feature maps to detect insulators and spacers on transmission lines. These methods have achieved high detection accuracy and are suitable for defect detection on transmission lines in cloud servers. However, the parameters of the network model designed by them are too large, leaving space for improvement. Therefore, some studies have begun to use lightweight models to improve the network structure to obtain faster inference speed. Huang et al. [17] introduced a lightweight network in YOLOv5s by adding a small-scale detection layer and designing a receptive field module to detect insulator defects, replacing the SPP module. Zhang et al. [18] used the GhostConv module to replace traditional convolution and then used high-resolution features for network prediction. Finally, they used knowledge distillation to ensure the accuracy of the detection model. Shen et al. [19] replaced the feature extraction network of the SSD network with the MobileNetv1 model and introduced the FPN model to detect foreign objects in transmission lines. These studies have made significant progress in detecting defects in transmission lines using UAVs. Still, there are some drawbacks. Lightweight networks sacrifice the accuracy of the model for speedup. In complex detection environments, the models' accuracy may not meet the requirements for practical applications. In addition, the computational power of edge devices is underutilized, limiting the speed of model inference.

Based on the above research, balancing detection accuracy and detection speed is comprehensively taken into consideration. The main work and contributions are as follows:

1. A collaborative "edge-cloud-end" analysis system for identifying transmission line defects is proposed. The edge device only transmits image frames containing defect information, which solves the problem of high delay when transmitting a large number of images to the cloud server and improves the efficiency of the whole inspection system.
2. YOLO-inspection is proposed for UAVs transmission line inspection. The algorithm improves the feature fuse network structure and assigns certain weight coefficients to feature maps of different scales to effectively aggregate features. The GhostNetV2 [20] model is used to reconstruct the network, reducing the model volume while maintaining detection accuracy. TensorRT is used to optimize and accelerate the model, fully utilizing the computing power of edge devices to improve inference speed.
3. A DTLFD (Diversified transmission line fault datasets) dataset of common defects in five types of transmission lines was produced, and 4300 image data were manually labeled. By simulating the low light image and the image during snowfall weather, it can be found that the generalization performance of YOLO-inspection algorithm is better.

2 Related work

The defect detection method of UAVs inspection transmission line for edge computing has achieved initial success. Most studies are based on single-stage object detection algorithms for innovation and application. Through research, we have found that the detection accuracy and inference speed of the algorithm model can be balanced in some simple ways to achieve an ideal effect. Since the release of YOLO, it has occupied a hot topic and became a very popular object detection algorithm.

YOLOv5 was initially released with four different structures: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The main difference is the width and depth of the model. In 2021, Ultralytics released version 6.0 of YOLOv5, proposing a new network YOLOv5n that detects faster. From the perspective of balancing detection accuracy and reasoning speed, YOLOv5s is selected as the initial network model.

The YOLOv5 network model structure is mainly composed of three parts, including the backbone, neck, and head. The network model of YOLOv5 is shown in Fig. 1. The feature extraction network is mainly composed of the C3 module, CBS (Conv Batch Normalization Sigmoid Linear

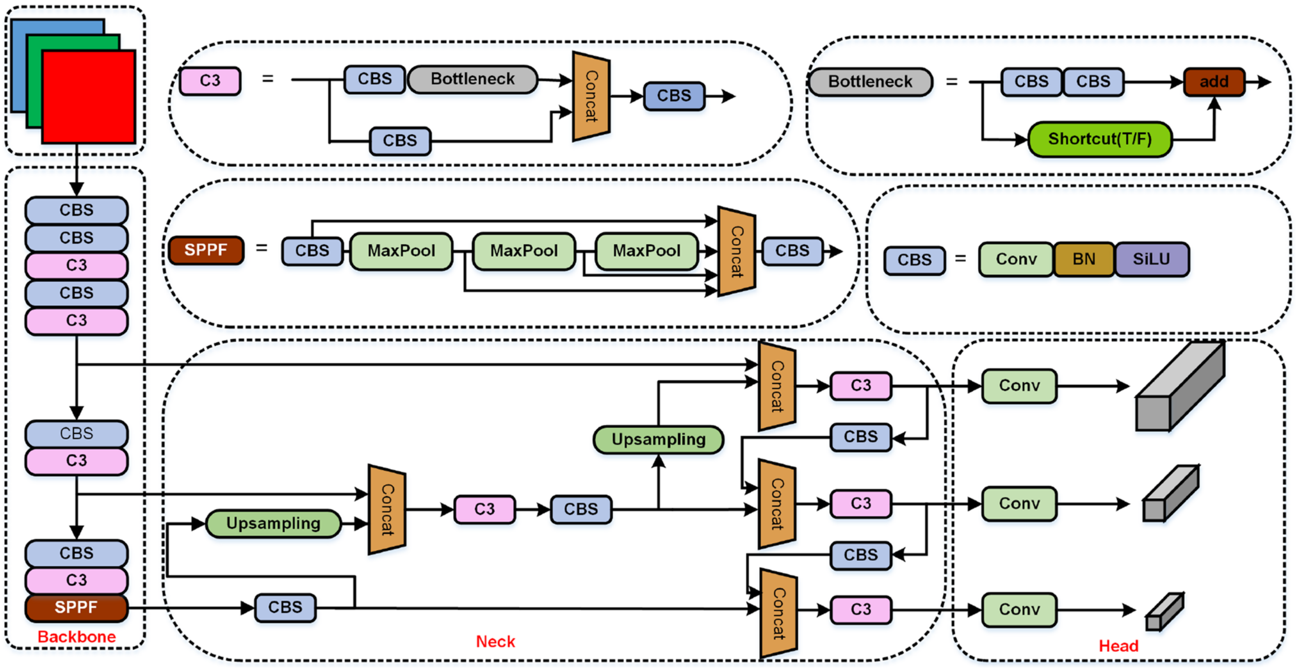

Fig. 1 The schematic illustration for YOLOv5s

Unit) module, and SPPF (Spatial Pyramid Pooling Feature) module. The C3 module in YOLOv5 adopts the idea of cross-stage partial connection and residual structure, aiming at improving the feature expression ability of the model and reducing the number of model parameters. Cross-stage partial connection divides the input feature map into two parts. One part goes directly through the convolution operation, while the other part is first connected through a cross-stage partial connection layer and then added to the feature map after the convolution operation. This approach enhances the diversity of feature maps and improves their expression ability. The CBS module is composed of a convolutional layer, a batch normalization (BN) layer, and a SiLU activation function. SPPF is a technique used to extract features from input images at multiple scales. The SPPF module is used to replace the previous SPP module. These pooling layers extract features from input images of different scales so that the network can capture small and large objects in the image. The feature fusion network aims to combine shallow graphic features with deep semantic features to obtain more comprehensive features. The shallow features are obtained from the feature extraction network by the feature fusion network of YOLOv5, and then connected with deep semantic features. To facilitate the fusion of the feature maps from the feature extraction network, the feature fusion network interpolates the feature map by upsampling, increasing the scale of the feature map. The feature maps are downsampled by the feature fusion network, first to obtain feature maps of different scales, and second to achieve better fusion of shallow graphic features and deep semantic features, rather than simply concatenating them. Through feature fusion operations, targets of different sizes and proportions can be better captured, thereby improving detection accuracy. The detection head of YOLOv5 includes three sub-networks, each of which is used to detect targets of different sizes. These sub-networks output the confidence of the prediction box, as well as the location of the prediction box.

3 Edge intelligence analysis system

One of the main challenges in investigating data processing optimization schemes in the field of object detection of transmission line defects with cloud servers at the core is how to effectively deal with the transmission latency problem triggered by large amounts of image data. In this context, the rapid processing of data is critical to achieving real-time and effective detection. However, in traditional methods, huge image data are usually transferred to a cloud server for processing, which requires both time for uploading the data and involves time for data processing in the cloud, which may lead to severe delays, weakening detection efficiency and responsiveness. To cope with this problem, this system adopts an edge-cloud-end collaborative processing strategy, which takes the edge device as the key link and transfers the image analysis and processing operations to the edge device. The edge-cloud-end co-processing system solves the latency problem through the following three aspects:

(1) Proximity computing: Edge computing deploys computing resources where the data are generated, thus avoiding the network latency incurred by the data during transmission. This allows data to be processed in real time on devices close by, thus reducing latency.
(2) Reduced network traffic: Edge computing allows data to be pre-processed, filtered, and analyzed locally, with only the necessary results transmitted back to the cloud. This reduces the amount of data that needs to be transmitted, which reduces network traffic and further reduces transmission latency.
(3) Immediate response: Edge devices have the ability to process data in real time, so responses can be generated and acted upon more quickly. This is especially beneficial for drone inspection missions that require a quick response.

Overall, the architecture of the edge-cloud-end collaborative system is shown in Fig. 2, while the processing flow is detailed in Fig. 3. The UAV inspection system consists of four key components: the UAV, the gimbal/camera, the edge device and the cloud server. Starting from the capture of transmission line video data by the UAV camera, defects in the inspection images are identified and localized in real time using the YOLO-inspection model embedded in the NVIDIA Jetson Xavier NX. Whenever an image frame containing defective information is captured, the system immediately writes the current detected frame into the database together with the defect type, following the principle of transmitting only image frames containing defective information, thus significantly reducing the amount of data transmitted.

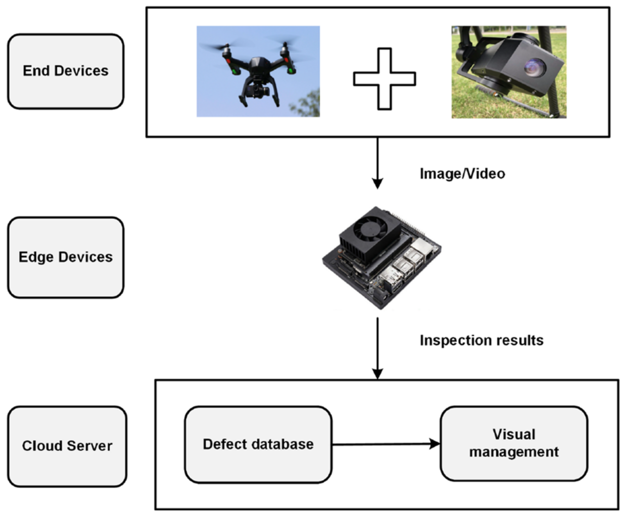

Fig. 2 Design diagram of “edge-cloud-end” collaborative UAV inspection system

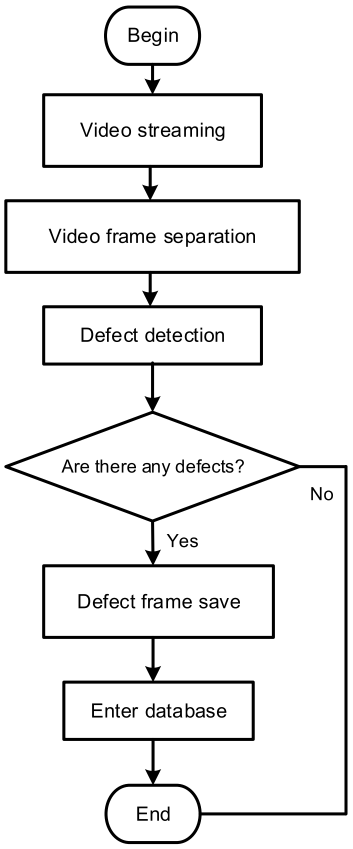

Fig. 3 Workflow diagram of edge-cloud-end collaborative system

On the cloud server, in addition to being able to view the current inspection status and detection results in real time, the powerful analytical capabilities of cloud computing can also be used to analyze a large amount of data in depth. This end-to-end process gives full play to the real-time nature of edge computing and the in-depth analysis advantages of cloud computing, thus achieving efficient and accurate transmission line defect detection and management.

4 Construction of YOLO-inspection lightweight network model

During the inspection of transmission lines by UAVs, there are usually a number of conditions that can affect the model's ability to detect defective targets, including vegetation interference, weather, structural complexity of power lines, and lighting variations. Therefore, it is necessary to design a defect detector with high detection performance. While maintaining high accuracy, this detector must also meet the requirements of real-time detection.

Based on the structure of the YOLOv5 series model, this paper optimizes the module and feature fusion network in the YOLOv5 model to solve the problems faced by UAV inspection. The YOLO-inspection model is proposed and its structure is shown in Fig. 4. On the whole, the YOLO-inspection model reconstructs the C3 module and completes the lightweight processing of the network model. Compared with the original C3 module, it can improve the inference speed of the model on the basis of maintaining accuracy. From the perspective of a feature fusion network, on the basis of increasing cross-scale connections, a dynamic adaptive feature weighting module is designed, which can fuse features from different levels and perform weighting processing.

4.1 GhostNetV2-C3

Most of the existing deep learning network models are overparameterized, which leads to the slow inference speed of the model. By visualizing the feature map of YOLOv5s, similar feature maps can be found between different layers in the same stage, as shown in Fig. 5. These similar feature maps can be generated by cheap operations, which can reduce the computational complexity of the network model. This study reconstructed the C3 module of the YOLOv5 model based on the GhostNetV2 model and created the GhostNetV2-C3 module.

The GhostNetv2 network is a network structure based on the Ghost module and the DFC attention mechanism. The

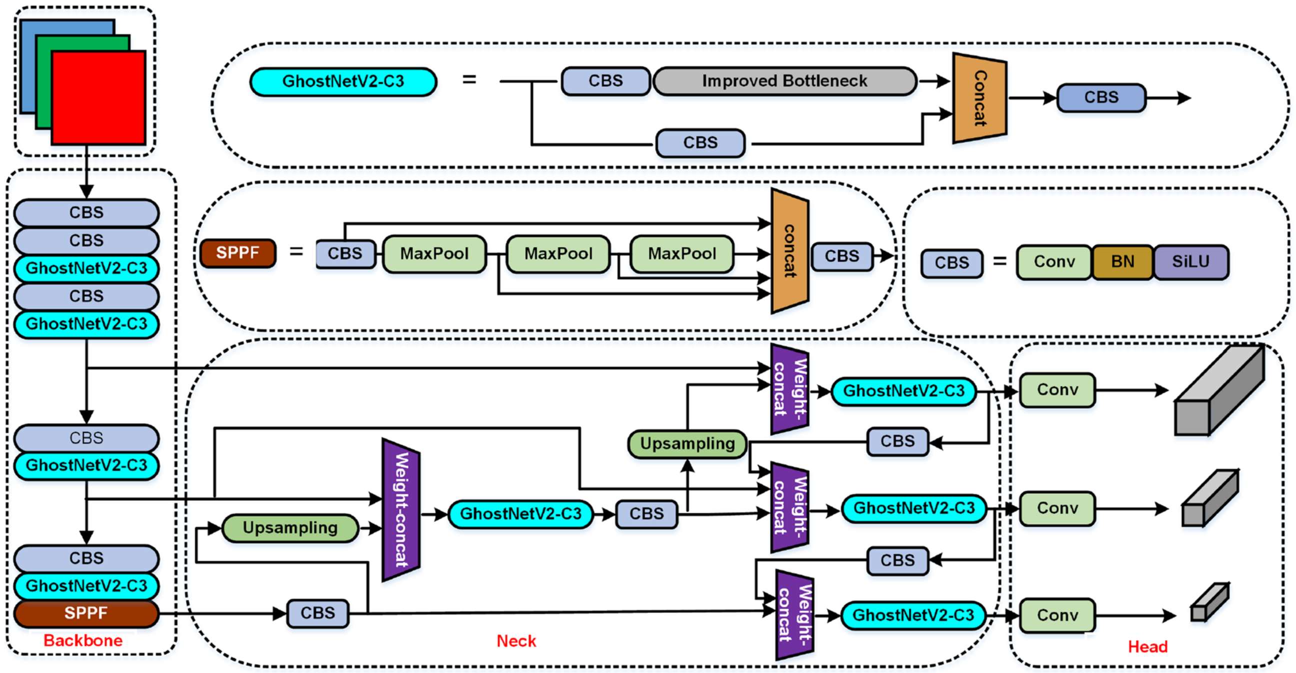

Fig. 4 YOLO-inspection model structure

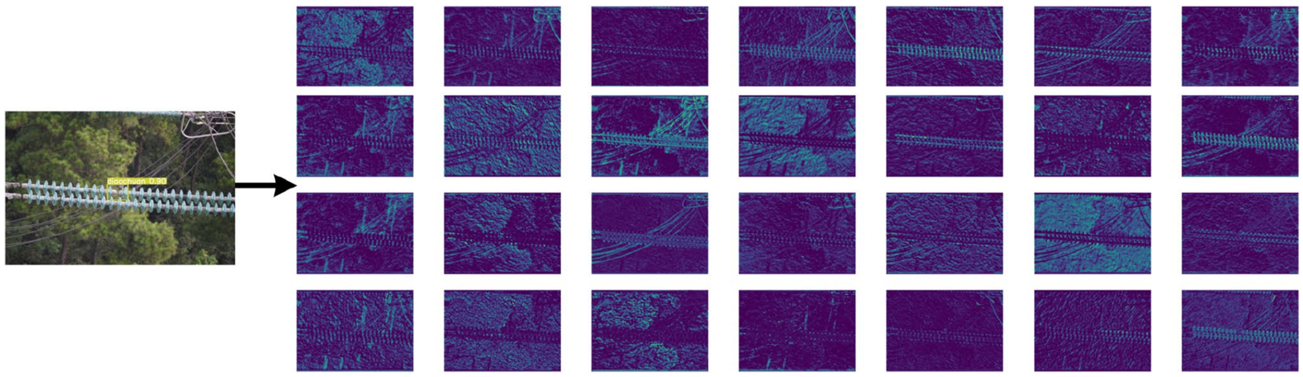

Fig. 5 Visualization of feature maps of YOLOv5s after passing through the first C3 module

Ghost module first generates the basic feature map through 1 × 1 convolution, then obtains the redundant feature map through linear operation, and finally concatenates the two feature maps together. This method of using linear operations to generate redundant feature maps can significantly reduce the number of model parameters. But it also weakens the model's feature representation ability, which may hinder the improvement of model performance.

Due to the weak ability of the model to capture spatial information, the dynamic feature correction (DFC) attention mechanism is used to capture the remote spatial information of the model. Figure 6a is the structure diagram of the DFC attention mechanism. The mechanism decomposes the fully connected layer into horizontal and vertical fully connected layers to aggregate pixels in the two-dimensional feature map. Under the attention mechanism of DFC, the Ghost module can aggregate local information and establish remote dependencies. Since direct parallelism between the Ghost module and the DFC attention mechanism reduces the efficiency of the Ghost module, downsampling the input feature maps in the DFC module in the horizontal and vertical directions can reduce the size of the features and allow them to operate on smaller features. The bottleneck structure of GhostNetV2 is composed of Ghost module and DFC module, as shown in Fig. 6b, c, where DWConv refers to channel-by-channel convolution. When the step size is 1, the Ghost model and the DFC maintain branch parallel input to further transmit the features to the next Ghost module for output. The first Ghost and DFC parallel module mainly captures the remote dependence of pixels in different spatial locations and increases the number of input feature map channels. The second Ghost module is used to reduce the number of channels in the input feature map and match the number of channels in the original input network. The difference between step size 1 and step size 2 is the use of DWConv, which is mainly used to compress the width and height of the two Ghost modules before and after compression, and change the shape of the input feature layer. Fig. 6d shows the final structure of the GhostNetV2-C3 module, where the improved Bottleneck structure consists of Fig. 6b or c.

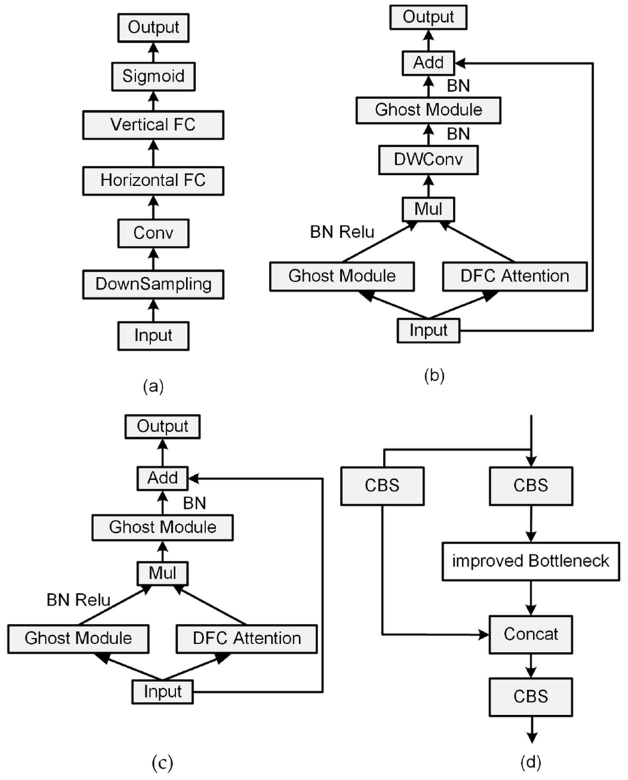

Fig. 6 The composition of the GhostNetV2-C3 module: **a** DFC attention mechanism; **b** Bottleneck structure with step size of 2; **c** bottleneck structure with step size of 1; **d** GhostNetV2-C3 module

4.2 Design of weighted feature fusion network

The feature fusion network is essentially a fusion of deep features and shallow features. With the deepening of the neural network, the original object will lose part of the relationship with the whole. The deep features in the network get high-level semantic features, while the shallow features get positioning features. To ensure the effective fusion of features, the feature fusion network used by YOLOv5 has two paths, transmitting strong semantic features from top to bottom, and transmitting strong positioning features from bottom to top. However, this approach fails to account for differences in features across various levels, and direct fusion of features may limit model performance. To improve object detection accuracy and optimize feature maps at different levels, this paper proposes a redesign of the feature fusion network, including skip connections between feature maps of the same scale and assigning different weights to various feature maps. The network structure is shown in Fig. 7. Among them, P3, P4, and P5 are feature maps of different sizes output by the feature extraction network. The first feature fusion is completed by horizontal connection and downsampling, and then the second feature fusion is completed by jump connection, horizontal connection and upsampling layer between the same levels. Finally, the feature map of cross-scale fusion is output. This simple processing can more comprehensively integrate shallow and deep information, thereby improving the feature expression ability of the model.

If the obtained two or three feature maps are directly fused, it may be difficult to effectively use the information in the feature map. Therefore, it is necessary to establish an adaptive module for assigning weight coefficients. By introducing learnable variables into the network, an untrainable type Tensor is transformed into a trainable type Parameter, which is automatically optimized in the process of parameter backpropagation. Using the function shown in Eq. (1) to dynamically allocate weights will ensure that the weight coefficient is within [0,1]. Because the Softmax operation is not used, the speed will be very fast.

$$\Lambda = \sum_i \frac{\rho_i}{\tau + \sum_j \rho_j} x_i, \tag{1}$$

where τ is taken as 0.0001 to avoid numerical instability, ρ denotes the learned weights, and x_i represents the input feature map in the network structure.

Using the feature fusion network shown in Fig. 7, here take module A and module B as examples to provide specific fusion formulas. Equation (2) shows the fusion formula for module A, and Eq. (3) is the fusion formula for module B.

$$P_4^{\text{mid}} = \text{Conv}\left(\frac{\rho_1 \cdot P_4^{\text{in}} + \rho_2 \cdot Resize(P_5^{\text{mid}})}{\rho_1 + \rho_2 + \tau}\right), \tag{2}$$

$$P_5^{\text{out}} = \text{Conv}\left(\frac{\rho_1' \cdot P_4^{\text{in}} + \rho_2' \cdot \text{Resize}(P_4^{\text{mid}}) + \rho_3' \cdot P_3^{\text{out}}}{\rho_1' + \rho_2' + \rho_3' + \tau}\right), \tag{3}$$

where *Resize* represents an upsampling or downsampling operation and *Conv* represents a convolutional layer for performing convolutional operations on features.

5 Experimental results and analysis

5.1 Construction of the dataset

Due to the limited quantity and single type of defects in existing publicly available CPLID defect datasets, a dataset named DTLFD was constructed. The data used in this paper are provided by a power company in China, and the image data are taken by the inspection UAV. 4300 images were selected from inspection data to form the dataset used in this paper, including damage, dirt, drop, bird's nest, and damper. The types, labels, and quantities of various defects in the dataset are shown in Table 1. The ratio of the training set and validation set is 8:2. 1000 images from actual inspections

Table 1 Detailed information on the dataset

Defect type	Label name	Number of images	Number of labels
Damage	Posun	1023	2352
Dirt	Wushan	870	3045
Drop	Diaochuan	839	1867
Bird's nest	Niaowo	1156	1360
Damper	Fangzhenchui	1358	4586

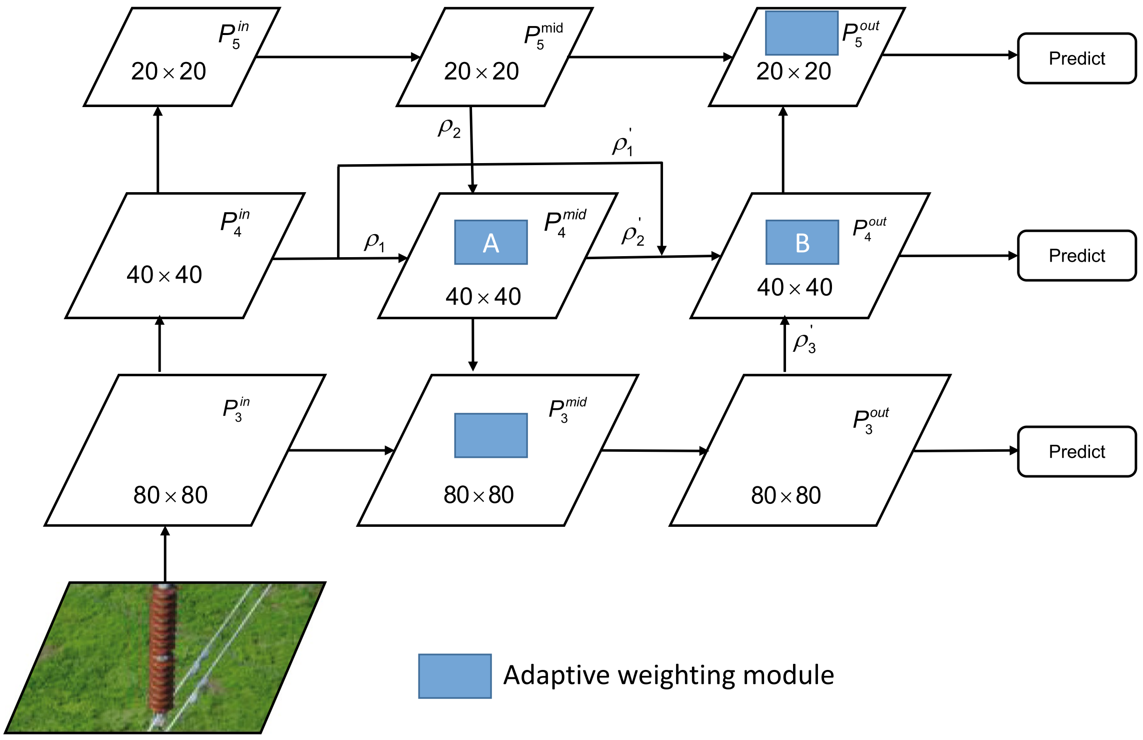

Fig. 7 The structure of weighted feature fusion network

are used as the test set. Image labels are annotated using the Labelimg tool.

In the actual inspection, the UAV is fast-moving relative to the target, which will inevitably lead to blurred images. The use of motion-blurred image data may lead to a decrease in the accuracy of the target detector, because the blurred image may blur the edges and details of the object, making it difficult for the target detector to accurately identify the position and shape of the target. In addition, using motion-blurred images may also introduce errors. For example, when shooting fast-moving objects, motion blur may cause the shape of the target to change, resulting in misjudgment. Therefore, the image deblurring algorithm is used to deblur 4300 images. DeblurGANv2 [21] is used to solve the problem of image blurring, thus improving the quality of the dataset.

5.2 Parameter settings and evaluation metrics

5.2.1 Implementation details

This study employs the NVIDIA Jetson Xavier NX as the edge intelligence device. The development environment for this device utilizes Ubuntu 16.04, PyTorch 1.8, and Python 3.7. For cloud-based detection, a PC machine with an Intel Core i5-12400F CPU and an Nvidia GeForce RTX 3060 GPU with 12 GB of memory is used. The development environment for this setup consists of Windows 11, PyTorch 1.13, Python 3.9, and CUDA 11.7. Training, validation, and testing were performed under the same hyperparameters. The hyperparameter settings used in the experiment are shown in Table 2.

5.2.2 Evaluation metrics

We use AP, mAP@0.5, model size, GFLOPs, inference time, and FPS to evaluate the performance of the model. AP refers to the average accuracy of the model in a certain category. mAP@0.5 represents mAP at the intersection over union (IoU) threshold of 0.5. mAP refers to the average AP of the model in all categories. GFLOPs represent floating-point operations. The inference time refers to the time spent by the model on the edge device to detect the image on the edge device. FPS refers to how many images can be detected per second. The AP equation (Eq. 4) and mAP equation (Eq. 5) are as follows:

$$\mathrm{AP} = \int_0^1 P(R)\mathrm{d}R, \tag{4}$$

$$m\mathrm{AP} = \frac{\sum_{i=1}^{n} AP(i)}{n}, \tag{5}$$

where P represents the precision rate, R represents the recall rate, i denotes the detected category, AP(i) denotes the average precision of the category, and n is the number of detected categories.

Table 2 Hyperparameter setting

Hyperparameter	Value
Image size	640× 640
Optimizer	SGD
Batch size	16
Epochs	300
Learning rate	0.01
Momentum	0.937
Weight decay	0.0005

5.3 Comparison of YOLO-inspection with other detection algorithms

In order to verify the effectiveness of YOLO-inspection, four classical object detection models are constructed and compared with YOLO-inspection, including FasterRCNN, SSD, EfficientDet [22], and YOLOv5. However, these four algorithms are not lightweight network models, and their detection speed is not superior to YOLO-inspection. Therefore, the lightweight network models MobileNetV3 [23], ShuffleNetV2 [24], and GhostNet [25] were selected to compare with YOLO-inspection to prove the advantages of YOLO-inspection. Among them, MobileNetV3 and ShuffleNetV2 are only feature extraction networks, so the feature extraction network DarkNet53 of YOLOv5 is replaced by them to construct a lightweight network model. At the same time, considering the characteristics of the GhostNet model, the C3 module and the ordinary convolution module in the YOLOv5 network are replaced by the GhostC3 module and the GhostConv module respectively, so that it is similar to the improved method in this paper. Training and testing are completed on the cloud server, and the test results are shown in Table 3.

Table 3 Performance indicators of different detection models

Model	$\mathrm{mAP}_{@0.5}$/%	Size/MB	FPS
SSD	76.3	84.3	19
EfficientDet	68.1	25.6	22
FasterRCNN	91.2	110	7
YOLOv5s	94.1	13.8	99
YOLOv5s-MobileNetV3	85.7	7.1	115
YOLOv5s-ShuffleNetV2	81.3	6.99	113
YOLOv5s-GhostC3	92.4	28.9	102
YOLO-inspection	94.3	11.6	111

According to the experimental results in Table 3, the YOLO-inspection algorithm achieves a detection accuracy of 94.3 % and a detection speed of 111 pictures per second. The YOLO-inspection algorithm at mAP@0.5 is not only 0.2% higher than YOLOv5s, but also detects 12 more images per second than YOLOv5s. In comparison to the mainstream object detection models, FasterRCNN, SSD, and EfficientDet, YOLO-inspection exhibits the best detection accuracy and speed. Compared with the lightweight models YOLOv5s-MobileNetV3, YOLOv5-ShuffleNetV2, and YOLOv5s-GhostC3, YOLO-inspection has the highest detection accuracy, but FPS is slightly lower than YOLOv5s-MobileNetV3 and YOLOv5-ShuffleNetV2. The main reason is that the skip connection on the feature fusion network makes the detection speed slightly slower. The above results show that the design of YOLO-inspection can well balance the detection speed and detection accuracy, and is suitable for the defect detection task of UAV inspection transmission lines.

5.4 Ablation experiment

To further explore the influence of the improvement points proposed in this algorithm on the performance of the network model, an ablation experiment was designed. On the basis of the original model, the improvement measures are added one by one, and the experiment is carried out. The experiment is carried out on the cloud server side. The specific results of the ablation experiment are shown in Table 4.

Through the experiments carried out in Table 4, it can be found that each step of YOLO-inspection optimization has an impact on the YOLOv5s model. Among them, the lightweight GhostNetv2C3 design reduced the mAP@0.5 of the model by 0.6%, but the FPS of the model increased by 16. It is normal to bring a decrease in accuracy after lightweight, but 16 more images can be detected per second. After the improvement of the weighted feature fusion network, the mAP@0.5 of the model is increased by 0.8%, and its calculation amount is slightly increased, which is acceptable for this model. Combining them for experiments, the model's inference speed is also significantly improved when the detection accuracy is improved by 0.2%. Compared with YOLOv5s, the proposed method is more suitable for defect detection of transmission lines.

To observe the detection effect of YOLO-inspection more intuitively, five images in the validation set are randomly selected for detection and compared with the detection effect of the YOLOv5s model. Figure 8 shows the detection results of YOLOv5s and YOLO-inspection. In the detection results of the YOLOv5s model, a bent piece of wire in the third image was misdetected as a damper; however, YOLO-inspection did not show any misdetection. Taken together, the confidence scores of defective targets detected by YOLO-inspection are on average higher than those of the YOLOv5s model, and the application of an adaptively weighted feature fusion network can reduce the occurrence of target detector misdetection.

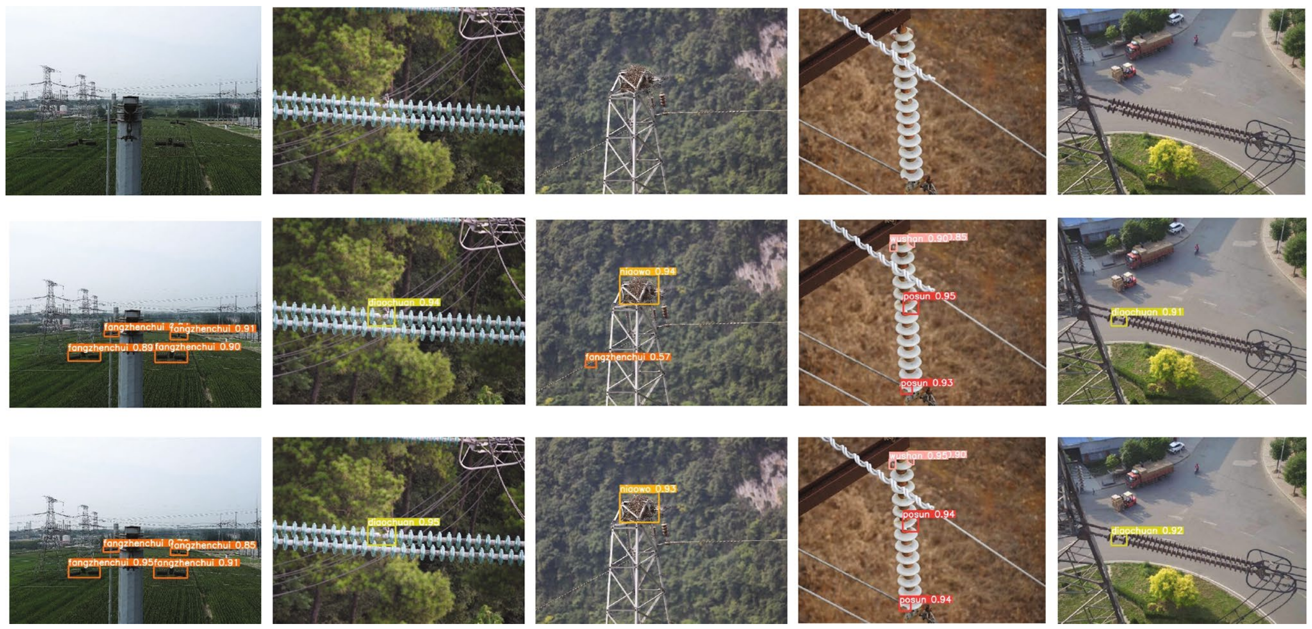

Fig. 8 Detection results of different models. The first row is the raw data, the second row is the image after YOLOv5s detection, and the third row is the image after YOLO-inspection

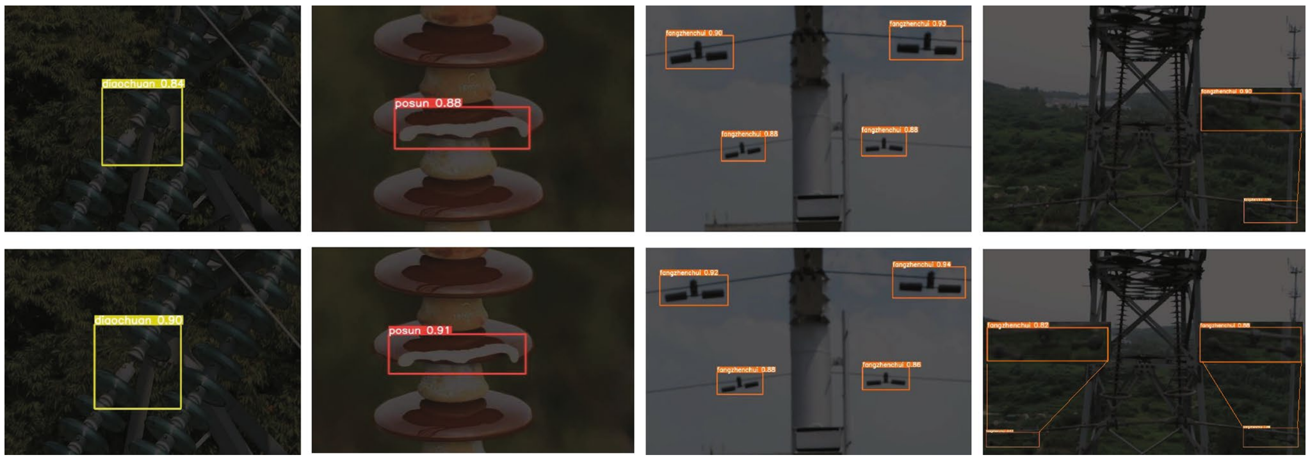

Fig. 9 Low illumination image detection results: **a** Insulator drop; **b** Insulator damage; **c** Damper; **d** Damper and bird's nest

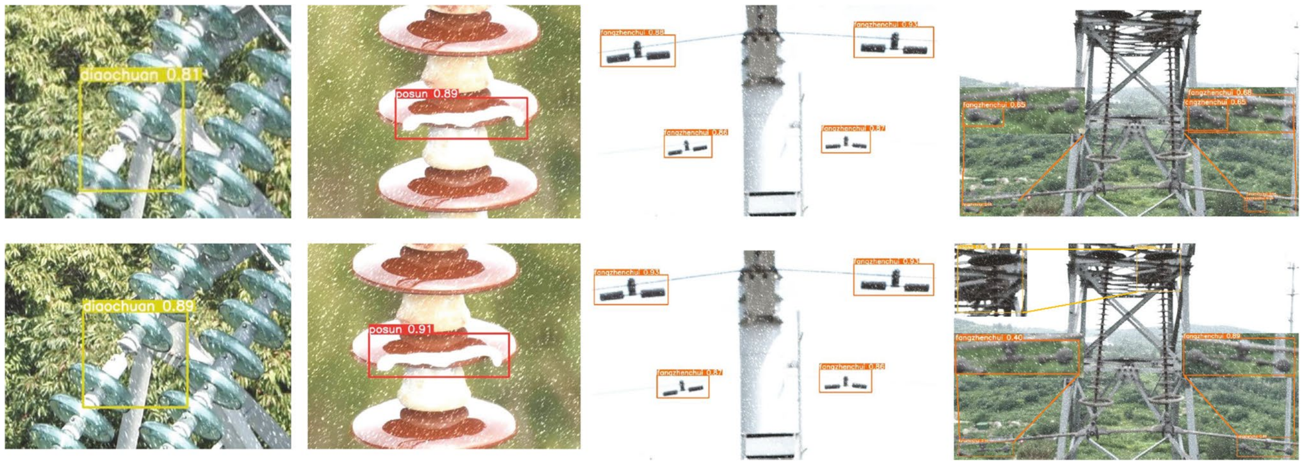

Fig. 10 Detection results in snowy scenes: **a** Insulator drop; **b** Insulator damage; **c** Damper; **d** Bird's nest

Table 4 The performance index obtained by ablation experiment on YOLO-inspection

Model	mAP$_{@0.5}$/%	GFLOPs	FPS
YOLOv5s	94.1	16.0	99
+GhostNetv2C3	93.5	11.7	115
+Improved Neck	94.9	17.4	92
YOLO-inspection	94.3	11.8	111

Table 5 Experimental performance index of optimal reasoning for edge devices

Model	mAP$_{@0.5}$/%	FPS
YOLOv5s	94.1	21
YOLOv5s-Tensor RT	94.1	40
YOLO-inspection	94.3	30
YOLO-inspection-Tensor RT	94.3	63

5.5 Inference experiment of edge device

The computing power of cloud servers and edge devices differs greatly. Therefore, after the training is completed, the TensorRT is used to further optimize and accelerate the model, so as to improve the inference speed of the model. We not only compare the detection efficiency of the server-side model on edge devices, but also verify that TensorRT has a positive effect on the improvement of model inference speed. The experimental results are shown in Table 5.

Through experiments, it can be found that the reasoning speed of the model will be greatly improved after TensorRT optimization. After algorithm improvement and TensorRT optimization, the detection efficiency of this method is increased by 300% compared to the YOLOv5s model without using TensorRT. Making full use of the computing

Table 6 Detection accuracy of each category

Model	Damage/%	Dirt/%	Drop/%	Bird's nest/%	Damper/%
YOLOv5s	99.4	99.1	99.5	86	86.6
YOLO-inspection	99.4	99.5	99.6	86.4	86.6

power of hardware devices to optimize the model can greatly improve the efficiency of the model. The recognition accuracy of YOLOv5 and YOLO-inspection for five defect data is shown in Table 6. It can be seen that the accuracy of each category of the lightweight YOLO-inspection model is better.

5.6 Generalization ability verification

To evaluate the detection performance of the YOLO-inspection algorithm in different environments, we purposely simulated images in low-light and snowy weather. To achieve the low-light effect, we adjusted the image with the help of the OpenCV library to give it a dimmer appearance. In addition, the iaa.imgcorruptlike.Snow function in the image enhancement library imgaug can be used to simulate the snowflake effect on the image. After optimizing and accelerating the model through TensorRT, the images of the two scenarios are detected on the edge device and the results are shown in Figs. 9 and 10. In the fourth image of Fig. 9, the YOLOv5s fails to detect the damper at the bottom of the figure and the bird's nest obscured by the pole tower. Although YOLO-inspection failed to detect the bird's nest, it successfully identified two dampers. In the other three images, each of the samples can be detected correctly and YOLO-inspection detected all of them with higher confidence than YOLOv5s. In the fourth image of Fig. 10, YOLOv5s fails to detect the bird's nest, the damper is poorly recognized, and the bounding box fails to completely enclose the damper. However, YOLO-inspection both detects all samples correctly and has a higher confidence level than YOLOv5s. In conclusion, the YOLO-inspection model shows excellent generalization ability and is more suitable than YOLOv5s for inspection tasks in bad weather scenarios.

6 Conclusion

In this paper, we introduce an integrated UAV inspection system based on "edge-cloud-end" collaboration, and innovatively propose a YOLO-inspection algorithm suitable for defect detection on edge devices. First, this paper proposes an edge intelligent analysis system for identifying transmission line defects. Whenever an image frame containing defect information is captured, the system immediately writes the currently detected frame into the database along with the defect type, following the principle of sending only image frames containing defect information. It solves the high latency problem of transmitting a large number of images to the cloud server for detection and improves the operational efficiency of the whole detection system. Secondly, a detection algorithm named YOLO-inspection is proposed for UAVs transmission line inspection. A DTLFD transmission line defect dataset is constructed and an image deblurring algorithm is used to solve the problem of image blurring caused by the relative motion of UAVs. By reconstructing the lightweight C3 module and feature fusion network in YOLOv5s, the accuracy and inference speed of the model are improved. By making full use of the computing power of edge computing devices, the models obtained from training are optimized and accelerated on TensorRT. Finally, the experiment results indicate that an image can be reasoned in 15.87 ms on the NVIDIA Jetson Xavier NX platform. Compared with YOLOv5s, mAP@0.5 increased by 0.2%, and FPS increased by 42. Through simple design and improvement, both speed and accuracy are well balanced. YOLO-inspection realizes the high-precision and fast detection of UAVs inspection transmission lines on the edge computing platform, which has certain practical value for the application of UAVs inspection transmission lines.

Acknowledgements This work is supported by the the Shandong Natural Science Foundation under Grant ZR2019MEE054, and the National Natural Science Foundation of China under Grant no. 51405262.

Data availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest This paper does not contain any studies with human or animal subjects and all authors declare that they have no conflict of interest.

References

1. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., Berg, A.: Ssd: single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37. Springer, New York (2016). https://doi.org/10.1007/978-3-319-46448-0_2
2. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.:You only look once: Unifed, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779-788 (2016). https://doi.org/10.48550/Arxiv.1506.02640
3. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271(2017). https://doi.org/10.1109/CVPR.2017.690
4. Redmon, J., Farhadi, A.: Yolov3: An Incremental Improvement. arXiv preprint, (2018). Arxiv:1804.02767

5. Bochkovskiy, A., Wang, C., Liao, H.: Yolov4: Optimal Speed and Accuracy of Object Detection. arxiv preprint, (2020). Arxiv:2004.10934
6. Glenn, J.: yolov5. Git code, (2020). https://github.com/ultralytics/yolov5
7. Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., Wei, X.: YOLOv6: A Single-stage Object Detection Framework for Industrial Applications. arxiv preprint, (2022). arxiv:2209.02976
8. Wang, C., Bochkovskiy, A., Liao, H.: YOLOv7: Trainable Bag-of-freebies Sets New State-of-the-art for Real-time Objectdetectors. arxiv preprint, (2022). Arxiv:2207.02696
9. Glenn, J.: yolov8. Git code, (2023). https://github.com/ultralytics/ultralytics/tree/main/ultralytics/models/v8
10. Cai, Z., Vasconcelos, N.: Cascade r-cnn: delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6154–6162 (2018). arXiv:1712.00726
11. He, K., Gkioxari, G., Dollar, P., Girshick, R.:Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969(2017). arXiv:1703.06870
12. 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**, 1137–1149 (2017). https://doi.org/10.1109/TPAMI.2016.2577031
13. Zhang, W., Liu, X., Yuan, J., Xu, L., Sun, H., Zhou, J., Liu, X.: RCNN-based foreign object detection for securing power transmission lines (RCNN4SPTL). Proc. Comput. Sci. **147**, 331–337 (2019). https://doi.org/10.1016/j.procs.2019.01.232
14. Li, J., Yan, D., Luan, K., Li, K., Li, Z., Liang, H.: Deep learning-based bird's nest detection on transmission lines using UAV imagery. Appl. Sci. **10**(18), 6147 (2020). https://doi.org/10.3390/app10186147
15. Bao, W., Du, X., Wang, N., Yuan, M., Yang, X.: A defect detection method based on BC-YOLO for transmission line components in UAV remote sensing images. Remote Sens. **14**(20), 5176 (2022). https://doi.org/10.3390/rs14205176
16. Liu, X., Li, Y., Shuang, F., Gao, F., Zhou, X., Chen, X.: ISSD: improved SSD for insulator and spacer online detection based on UAV system. Sensors **20**(23), 6961 (2020). https://doi.org/10.3390/s20236961
17. Huang, Y., Jiang, L., Han, T., Xu, S., Liu, Y., Fu, J.: High-accuracy insulator defect detection for overhead transmission lines based on improved YOLOv5. Appl. Sci. **12**(24), 12682 (2022). https://doi.org/10.3390/app122412682
18. Zhang, Y., Gong, X., Sun, J., Tao, Y., Su, W.:Research on Transmission Line Foreign Object Detection Based on Edge Calculation. In: Proceedings of the 2022 International Conference on Computational Infrastructure and Urban Planning. 22–25 (2022). https://doi.org/10.1145/3546632.3546876
19. Shen, H., Fan, P., Wei, Z., Zhao, C., Zhou, S., Wu, Q.: Research on transmission equipment defect detection based on edge intelligent analysis. J. Phys. Conf. Ser. 1828, (2021)
20. Tang, Y., Guo, J., Xu, C., Xu, C., Wang, Y.: GhostNetV2: Enhance Cheap Operation with Long-Range Attention. arxiv preprint, (2022). Arxiv:2211.12905
21. Kupyn, O., Martyniuk, T., Wu, J., Wang, Z.: DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better. In: 2019 IEEE/CVF International Conference on Computer Vision, pp. 8877–8886(2019). arXiv:1908.03826
22. Tan, M., Pang, R., Le, Q.: EfficientDet: Scalable and Efficient Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10778–10787 (2020)
23. Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L., Tan, M., Chu, G., Vasudevan, V., Zhu, Y., Pang, R., Adam, H., Le, Q.: Searching for MobileNetV3. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1314–1324(2019)
24. Ma, N., Zhang, X., Zheng, H., Sun, J.: Shufenet v2: practical guidelines for efficient cnn architecture design. In: Proceedings of the European conference on computer vision (ECCV), pp. 116–131(2018). https://doi.org/10.1007/978-3-030-01264-9_8
25. Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: Ghostnet: more features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1580–1589 (2020). https://doi.org/10.1109/CVPR42600.2020.00165