Received February 17, 2022, accepted April 4, 2022, date of publication April 8, 2022, date of current version April 18, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3165835

An Automatic Fault Detection Method of Freight Train Images Based on BD-YOLO

LONGXIN ZHANG[1], MIAO WANG[1], KE LIU[1], MANSHENG XIAO[1], ZHIHUA WEN[1], AND JUNFENG MAN[2]
[1]School of Computer Science, Hunan University of Technology, Zhuzhou 412007, China
[2]School of Computer Science, Hunan First Normal University, Changsha 410205, China

Corresponding authors: Mansheng Xiao (xiaomansheng@hut.edu.cn) and Junfeng Man (mjfok@qq.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1003401; in part by the National Natural Science Foundation of China under Grant 61702178 and Grant 62072172; in part by the Natural Science Foundation of Hunan Province under Grant 2019JJ50123, Grant 2020JJ4275, Grant 2021JJ50049, and Grant 2019JJ60054; in part by the Research Foundation of Education Bureau of Hunan Province under Grant 20C0625 and Grant 18C0528; and in part by the China Scholarship Council under Grant 201808430297.

ABSTRACT The automatic fault detection of freight train image is difficult to locate, and its average detection accuracy is low. In this paper, an object detection model Bidirectional You Only Look Once (BD-YOLO), is proposed. The process of BD-YOLO is divided into four steps. First, the feature extraction network extracts and separates image features. Second, multiscale features are fused to aggregate features of different scales. Third, prediction across scale modules is used to forecast the feature layers obtained after aggregation. Fourth, the final prediction result is obtained by decoding the prediction feature layer. The BD-YOLO model is trained using the mosaic data enhancement method and K-means clustering algorithm to improve detection accuracy and speed. Experimental results show that the mean average precision of BD-YOLO model is improved by 17.57%, on average, compared with the state-of-the-art object detection models on four types of data sets. The BD-YOLO model can immediately detect three typical faults, namely, movement of upper lever, offset of locking plate, and closing of truncated plug door handle. It has high detection rate, low false detection rate, and good robustness.

INDEX TERMS Convolutional neural network, fault detection, freight trains, object detection.

I. INTRODUCTION

With the rapid development of the economy and technological progress in the field of railway transportation, railway freight trains begin to change into heavy-load and high-density operation mode. The high efficiency and accuracy of train fault detection are critical for the safe operation of trains. Traditional manual detection methods are difficult to adapt to the high-speed, high-frequency operation requirements of freight trains. The National Railway Department of China has promoted the Trouble of Running Freight Train Detection System (TFDS) to promote the intelligent construction of the railway industry and satisfy the requirements for the rapid, accurate fault detection of freight trains [1]. TFDS uses the special camera beside the railway track to capture the image of the bottom of the freight train, transmits it, and stores it in the system through the network. The existing TFDS system is still in the manmachine combination mode and cannot realize automatic fault image detection. The traditional detection method mainly discriminates the suspected fault images manually by inspectors in the TFDS system. However, the complicated outdoor weather and changes in illumination affect the image capture and negatively affect the image detection. TDFS produces numerous images of freight trains daily. Many types of freight train fault images are available. These factors greatly affect the judgment of the inspectors. In addition, longtime indoor inspection work causes fatigue to the inspectors, easily leading to missed and false inspection events.

In recent years, more researchers work on automatic fault detection models for TDFS systems. Sun *et al.* [1] designed an automatic fault recognition system (AFRS) based on the convolutional neural network (CNN) model for the fault image recognition of side frame keys and shaft bolts. AFRS is a two-stage system, which contains object region detection and fault recognition. AFRS has high accuracy and

The associate editor coordinating the review of this manuscript and approving it for publication was Yongming Li.

good robustness. Liu *et al.* [2] proposed an automated visual inspection system (VIS) for checking missing keys of bogie blocks in freight cars. VIS uses gradient coded co-occurrence matrix feature and support vector machine for feature extraction and classification. VIS has good robustness in illumination. However, recognizing various freight train faulty images is difficult for VIS. Fu *et al.* [3] proposed a two-stage detection method to identify rolling bearing faults, which cascaded a bearing positioning stage and a fault classification stage. Fu *et al.* also designed an attention aware network APP-UNet16, to identify the defect areas of small oil spots in bearings. APP-UNet16 achieves a high-quality inspection and increases detection speed. Most researchers focus on fault detection of freight train bearings. AFRS, VIS, and other two-stage methods perform well, but they are designed for some specific freight component. This fact proves that a certain fault determination algorithm can be used to solve a certain type of fault detection, but it is not so suitable for other faults and multiple types of fault detection. Moreover, few studies are about the typical fault detection of other freight train parts, such as upper lever, truncated plug door handle, and locking plate. The traditional manmachine combination mode has low efficiency and is prone to the problem of missing and misdetection of train fault images. An object detection model named Bidirection You Only Look Once (BD-YOLO) is proposed to improve the automatic detection efficiency of train fault images and solve the automatic detection problem of typical train fault detections.

The main contributions of this paper are summarized as follows:

- An efficient one-stage object detection model, namely, BD-YOLO, is proposed. BD-YOLO model contains feature extraction network (FEN), feature integrated network (FIN), multiscale feature fusion (MFF), prediction across scales (PAS), and decoding of five modules. BD-YOLO can maintain a high detection precision while reducing the number of MFF module layers.
- The MFF module, which realizes the function of cross-scale information fusion and greatly improves the accuracy of the BD-YOLO model, is devised. MFF uses a horizontal skipping connection and two additional convolution layers to enhance the efficiency of multiscale feature information sharing.
- The mosaic data enhancement method is used to increase the sample data sets. K-means clustering algorithm is also used to generate the prior bounding box clustering center of the dataset with good fitting effect. More fitting data set clustering centers are used to improve the detection accuracy of BD-YOLO.
- Data sets of three freight train parts, namely, truncated plug door handle, upper lever, and locking plate, are annotated. Moreover, the BD-YOLO model is applied to the fault detection of truncated plug door handle closing, upper lever jumping out, and offset of locking plate. The experimental results show that the BD-YOLO model has good detection accuracy and robustness.

The remainder of this paper is organized as follows. Section II introduces related works. Section III describes the proposed BD-YOLO framework. Section IV presents the data set, experimental parameters, results, and performance analysis. Section V summarizes this study and plans for the future.

II. RELATED WORK

Deep learning models have powerful characterization and modeling capabilities. Through supervised or unsupervised training methods, the feature representation of the object can be learned layer by layer automatically, and the hierarchical abstraction and description of the object can be realized. The development of deep learning technology has greatly promoted the research of object detection and computer vision. The latest methods for common object detection problems can be simply divided into two categories, namely, anchor-based and anchor-free methods. Anchor-based methods include two types, namely, one-stage method and two-stage method.

One-stage methods include You Only Look Once (YOLO) [4], YOLOv2 [5], YOLOv3 [6], YOLOv4 [7], single-shot multi-box detector [8], RetinaNet [9], EfficientDet [10], and feature-selective anchor-free [11]. One-stage method research mainly adopts the idea of regression. For example, Redmon *et al.* [4] designed an anchor-free object detection model called YOLO. YOLO is a unified, real-time object detection framework, though its detection accuracy and speed are not as good as those of two-stage method, which provides a new idea for object detection. Redmon and Farhadi [5] proposed an improved YOLO model, denoted as YOLOv2. YOLOv2 uses multiple strategies such as multi-scale training, batch normalization, and high-resolution classifiers to improve recall and positioning accuracy while maintaining classification accuracy. Redmon and Farhadi [6] also improved backbone network depth and loss function based on YOLOv2 to obtain YOLOv3. Bochkovskiy *et al.* [7] proposed the YOLOv4 object detection model. Compared with the YOLOv3 model, the backbone network of YOLOv4 is changed from Darknet53 to CSPDarknet53. The feature feature fusion module of YOLOv4 is spatial pyramid pooling and path aggregation network (PAN) [12], and the prediction classification regression layer is still YOLO-Head. In the YOLO series of models, YOLOv4 has higher detection accuracy and faster detection speed than others. Liu *et al.* [9] designed a multiobject detection model, RetinaNet, and proposed focal loss to solve the problem of category imbalance in the training of the one-stage object detector. Compared with other models, RetinaNet can achieve end-to-end training, and does not reduce detection speed while improving detection accuracy. Typical MFF modules include feature pyramid network (FPN) [9], neural architecture search FPN [13], bi-directional feature pyramid network (Bi-FPN) [14] and PAN to achieve rapid feature fusion.

The two-stage method usually refers to the object detection method based on region proposal, mainly including region-based convolutional neural networks (R-CNN) series methods, such as Fast R-CNN [15], Faster R-CNN [16], Mask R-CNN [17], T-CNN [18], Cascade R-CNN [19], Libra R-CNN [20] and other frameworks. Ren *et al.* proposed a region proposal network (RPN), a merged RPN, and a Fast R-CNN [15] into a unified network called Faster R-CNN [16]. Faster R-CNN reduces the computational cost in the region proposal stage by sharing the convolution feature. Compared with previous frameworks, the number of candidate regions extracted by Faster R-CNN is remarkably reduced, and the quality of the candidate regions is improved simultaneously. Thus, the performance of the entire object detection network is improved, and real-time detection is almost realized. He *et al.* [17] proposed the Mask R-CNN instance segmentation framework, which can realize object detection and object instance segmentation. Cai and Vasconcelos [19] proposed Cascade R-CNN framework, a multi-stage framework, where each stage is assigned a different Intersection over Union (IoU) threshold. Cascade R-CNN can remove fake samples that are close to real samples stage by stage and prevent training from overfitting, but stage-by-stage calculation also leads to a slow detection speed.

In addition, many researchers have carried out many studies on anchor-free object detection. Huang *et al.* [21] designed an end-to-end object detection framework, DenseBox, which uses a fully connected neural network to detect multiple different objects accurately. However, the detection accuracy of DenseBox is not as high as other object detectors using anchor boxes. Law and Deng [22] proposed the Cornernet framework, which uses a corner pooling strategy to establish the relationship between the point and the object position. Cornernet uses two key points to replace the anchor mechanism, which can substantially reduce the number of parameters and calculation, resulting in fast detection speed. However, its detection accuracy is not as good as that of object detectors using the anchor mechanism. Zhou *et al.* [23] proposed the CenterNet framework, which uses key points to find the center point and regress to other object attributes, such as size, dimension, 3D extent, orientation, and position. CenterNet increases detection speed without sacrificing accuracy. Kong *et al.* [24] designed a completely anchor-free object detection framework named Fovea Box. Fovea Box does not require the use of anchors as a reference and can detect objects with any aspect ratio and shape. Tian *et al.* [25] proposed a pixel-by-pixel object detector named FCOS base on a fully connected layer. FCOS realizes anchor-free and proposal-free solutions and puts forward the idea of center-ness.

Liu *et al.* [26] developed a composite backbone network architecture, CBNet, which uses more than one backbone extraction feature. CBNet combines multiple backbone networks to extract more effective features, but leads to decrease detection speed. Dong *et al.* [27] designed a framework that could be embedded into multiple detection models, which obtain better accuracy and recall rate through the iteration of training process samples from simple to complex. Cheng *et al.* [28] proposed an online instance classifier refinement (OICR) framework to solve the problem of weakly supervised object detection and proposal generation and selection. OICR combines selective search and grad-weighted class activation mapping (Grad-CAM) based on Grad-CAM technology to generate proposals with higher IoU than the greedy search strategy. OICR selects more positive samples and ignores negative samples by enhancing the loss weight. Tang *et al.* [29] proposed Weakly Supervised Object Detection Deep Network (WSOD), which learns and improves the classifier in the iterative process by generating proposal clusters. The application of convolutional neural network based on deep learning in the field of object detection brings new ideas for TFDS typical fault detection.

Inspired by the YOLO model and Bi-FPN network, this work proposes a bidirectional YOLO (BD-YOLO) object detection model for the three types of typical train fault detection, namely, upper lever jumping out, offset of locking plate, and truncated plug door handle closing, owing to the rapid development of object detection models.

Fig. 1 shows the detection of BD-YOLO. First, gray bars are added to the original input image. Second, the image is divided into three grids with different scales (13×13, 26×26, 52×52), which are used to predict large, medium, and small objects, respectively. When the center of a certain target A falls in grid G, target A is finally predicted by grid G. Each bounding box contains five predicted values, namely, x, y, h, w, and confidence. The (x, y) coordinates represent the center point of the box; h and w indicate the height and width of the object to be predicted respectively. Each grid predicts B bounding boxes, and B is the number of objects. These boxes will be assigned a confidence level. The confidence score is equal to the probability of the target object times the IoU of the prediction bounding box and the ground truth bounding box, and the range is [0, 1]. When the grid has no object, the confidence score is 0. Otherwise, the confidence score is equal to the IoU of the predicted bounding box and the ground truth bounding box. Finally, the prediction bounding box with the highest confidence is obtained through nonmaximum suppression (NMS), and the detection result is obtained.

III. PROPOSED FRAMEWORKS

In this section, the overall structure of the BD-YOLO model is initially introduced. Then, FEN, FIN, MFF, loss function of classification regression, and decoding of classification regression are presented. Finally, the freight train fault image detection based on BD-YOLO is demonstrated.

A. BD-YOLO

Fig. 2 shows the overall architecture of BD-YOLO, including FEN, FIN, and MFF, PAS, and decoding five modules.

The first module is FEN, which contains 29 convolutional layers. The model inputs the original image of 416×416 pixels into FEN to obtain three effective feature layers. Each convolutional layer is defined by the structure

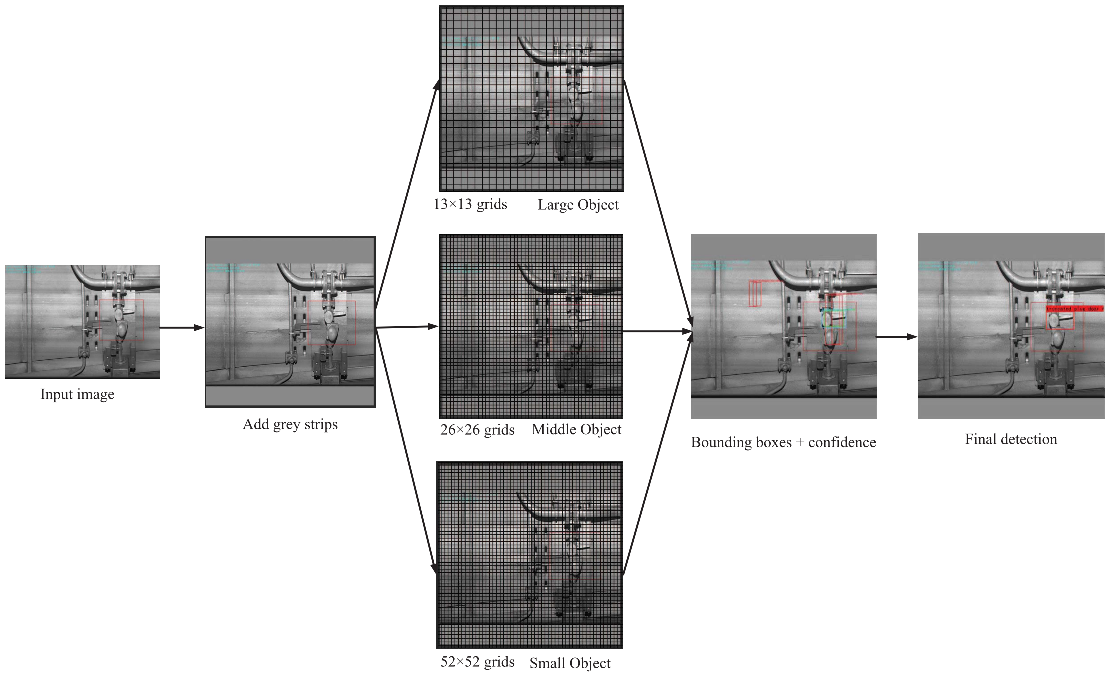


FIGURE 1. BD-YOLO model detection.

of shared parameters (weight vectors and bias) and has the characteristics of translation invariance. The input and output of each convolutional layer are 3D arrays, called feature layers, with a size of $H \times W \times C$, where H, W, and C are the height, width, and number of feature channels, respectively. The three feature layers of FEN output to FIN and MFF modules are P_3 (52, 52, 256), P_4 (26, 26, 512), and P_5 (13, 13, 1024). FEN consists of five feature layers, namely, P_1, P_2, P_3, P_4, and P_5. FEN input is the original image of 416×416 pixels. The three feature layers of PEN output to FIN and MFF modules are P_3 (52, 52, 256), P_4 (26, 26, 512), and P_5 (13, 13, 1024) respectively.

The second module is FIN. BD-YOLO performs maximum pooling of four different sizes for the feature layer P_5 of the last FEN layer to integrate the feature information of P_5.

The third module is MFF. By adding bottomup, topdown, and horizontal paths, MFF fuses FEN and FIN modules to output feature information of feature layers P_3, P_4 and P_5 of three different sizes. MFF outputs feature layer P'_3 (52, 52, 128), P'_4 (26, 26, 256), and P'_5 (13, 13, 512).

The fourth module is PAS. PAS predicts feature layers P'_3, P'_4, and P'_5 of MFF output. PAS outputs three feature layers P''_3 (BS, 52, 52, 27), P''_4 (BS, 26, 26, 27), and P''_5 (BS, 13, 13, 27) containing the predicted results. BS indicates batch size. PAS generates three prior bounding boxes in each feature layer. Each prior bounding box contains the offset of x, the offset of y, h, w, and category confidence. The dataset has four classification labels. Therefore, the third dimension of P''_3, P''_4, and P''_5 feature layer is $3 \times (5 + 4) = 27$.

The fifth module is decoding. The working process of the decoding module includes three stages: calculation of the five prediction parameters of the prediction bounding box, adjustment of the position of the prior bounding box, and NMS. The decoding module calculates the five predicted parameters of feature layers P''_3, P''_4, and P''_5 of PAS output. The decoding module adjusts the center point, width, and height of the prior bounding box according to the prediction parameters. The position of the detection bounding box is obtained by calculating the offset between the prediction bounding box and the prior bounding box. The decoding module removes redundant detection bounding boxes through the NMS and obtains the detection bounding box with the highest score.

1) FEATURE EXTRACTION NETWORK

Fig. 3 shows that, FEN consists of one Conv2D-BN-Mish layer and five Resblock-body layers. Resblock-body consists of a stack of one down sampling and multiple residual structures. Convolution block Conv2D-BN-Mish is composed of Conv2D, Batch Normalization and *Mish* functions. *Mish* is the activation function and defined as follows:

$$Mish = x \times \tanh\left(\ln(1 + e^x)\right). \tag{1}$$

Conv2D is the basic convolution block. Batch Normalization can speed up network convergence, prevent gradient explosion and gradient disappearance, and reduce the occurrence of overfitting.

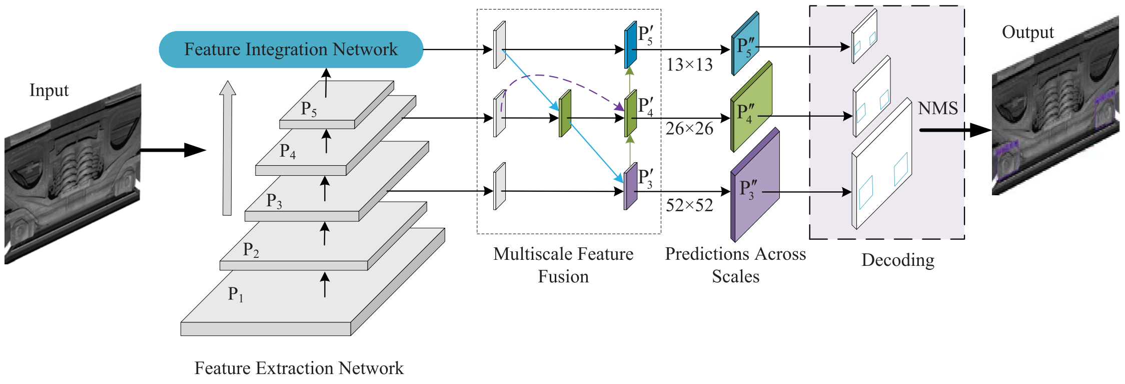

Input
Feature Integration Network
P5
P4
P3
P2
P1
Feature Extraction Network
P′5
P′4
P′3
Multiscale Feature Fusion
13×13
26×26
52×52
P″5
P″4
P″3
Predictions Across Scales
NMS
Decoding
Output

FIGURE 2. **Overall framework.**

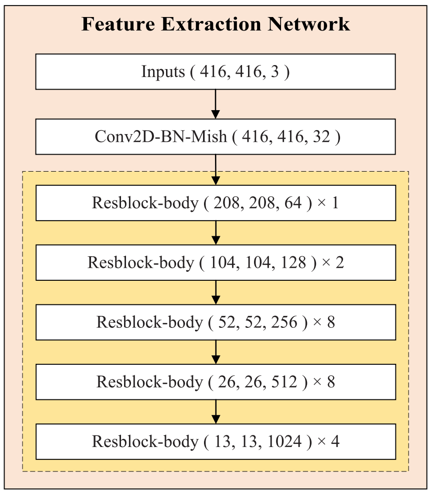

Feature Extraction Network
Inputs (416, 416, 3)
Conv2D-BN-Mish (416, 416, 32)
Resblock-body (208, 208, 64) × 1
Resblock-body (104, 104, 128) × 2
Resblock-body (52, 52, 256) × 8
Resblock-body (26, 26, 512) × 8
Resblock-body (13, 13, 1024) × 4

FIGURE 3. **Feature extraction network.**

2) FEATURE INTEGRATION NETWORK

Fig. 4 shows the feature integration module. FIN can greatly increase the receptive field and separate the most remarkable contextual features. Regardless of input size, FIN can produce a fixed size output. When the FIN module has completely performed three Conv2D-BN-Leaky convolutions on the last feature layer of FEN, maximum pooling is then conducted on four different pooling cores with different sizes of 13 × 13, 9 × 9, 5 × 5, and 1 × 1.

3) MULTISCALE FEATURE FUSION

The MFF module not only adds topdown, bottomup, and horizontal paths in each layer but also places an additional horizontal path in the middle feature layer P_4. The P_4 feature layer with dual paths can more effectively aggregate features with different resolutions.

Fig. 5 shows that the feature layers P_3, P_4, and P_5 with three scales of 13 × 13, 26 × 26, and 52× 52, respectively, are used to predict large, medium, and small objects.

The output feature P_3^{out} of feature layer P_3 is defined as:

$$P_3^{out} = Conv\left(P_3^{in} + Resize\left(P_4^{td}\right)\right), \tag{2}$$

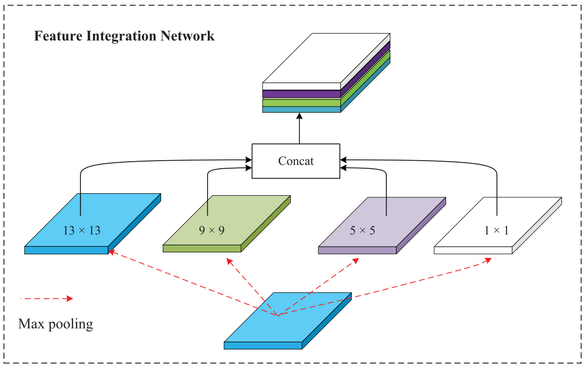


FIGURE 4. **Feature integration network.**

where, P_3^{in} is the input feature layer of P_3, *Conv* is a convolution operation for adjusting the number of channels on the feature layer, and *Resize* usually represents an increase or decrease in resolution, that is, upsampling and downsampling. P_4^{td} is the intermediate feature layer of P_4, which is calculated as follows:

$$P_4^{td} = Conv\left(P_4^{in} + Resize\left(P_5^{in}\right)\right). \tag{3}$$

P_4^{out} is the output feature layer of a topdown path, and is expressed as follows:

$$P_4^{out} = Conv\left(Conv\left(P_4^{in}\right) + Conv\left(P_4^{td}\right) + Resize\left(P_3^{out}\right)\right), \tag{4}$$

where P_4^{in} is the input feature layer of the P_4. Two additional convolutions are added in P_4^{out} to adjust the number of channels.

The output feature P_5^{out} of the feature layer P_5 is expressed as follows:

$$P_5^{out} = Conv\left(P_5^{in} + Resize(P_4^{out})\right), \tag{5}$$

where P_5^{in} represents the input feature layer of P_5.

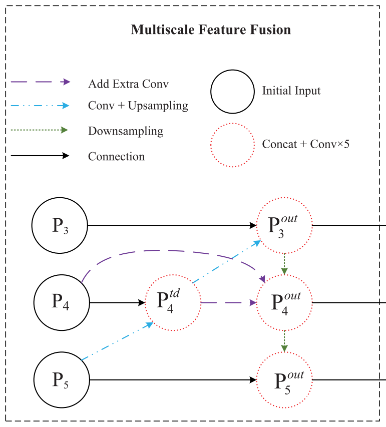


FIGURE 5. **Multiscale feature fusion module.**

4) LOSS FUNCTION

CIoU comprehensively considers the IoU, scale, penalty term and distance between the object and the anchor. Thus, it has good convergence and avoids divergence problem in training. IoU (pre, gt) is the ratio of the intersection and union of the predicted bounding box and the ground truth bounding box. The function IoU (pre, gt) is expressed as follows:

$$\text{IoU}\,(pre, gt) = \frac{\text{Intersection}(pre, gt)}{\text{Union}(pre, gt)}, \tag{6}$$

where Intersection(C, D) is the intersection of C and D; Union(C, D) is the union of C and D; gt and pre are the ground truth bounding boxes and predicted bounding boxes of the object to be detected, respectively.

CIoU can be formulated as follows:

$$\text{CIoU} = \text{IoU}\left(b, b^{gt}\right) - \frac{\rho^2\left(b, b^{gt}\right)}{c^2} - \alpha\upsilon, \tag{7}$$

where $\rho^2\left(b, b^{gt}\right)$ is the Euclidean distance between the center point of the predicted bounding box and the ground truth bounding box, b is the position of the predicted bounding box, b^{gt} is the position of the ground truth bounding box, and c is the diagonal distance of the minimum bounding rectangle that can contain both the predicted bounding box and ground truth bounding box.

The calculation of positive trade-off parameter α and parameter υ measuring the consistency of the aspect ratio are shown in Eqs. (8) and (9), respectively.

$$\alpha = \frac{\upsilon}{1 - \text{IoU}\left(b, b^{gt}\right) + \upsilon}, \tag{8}$$

$$\upsilon = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2, \tag{9}$$

where w^{gt} and h^{gt} are the width and height of the ground truth bounding box, respectively; w and h are the width and height of the prediction bounding boxes, respectively.

Loss function $loss_{\text{CIoU}}$ is calculated as follows:

$$\text{loss}_{\text{CIoU}} = 1 - \text{IoU}\left(b, b^{gt}\right) + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + \alpha\upsilon. \tag{10}$$

5) DECODING

Decoding is divided into three steps. Step 1 obtains the prediction results of three feature layers from PAS. Step 2 calculates the center point coordinates, width, and height of the detection bounding box. Step 3 inputs the position and type confidence score of the detection bounding box for NMS and outputs the detection bounding box with the highest category confidence score.

The center point (b_x, b_y), width b_w, and height b_h of the detection bounding box are calculated as follows:

$$b_x = \sigma(t_x) + c_x, \tag{11}$$

$$b_y = \sigma(t_y) + c_y, \tag{12}$$

$$b_w = p_w e^{t_w}, \tag{13}$$

$$b_h = p_h e^{t_h}. \tag{14}$$

where t_x and t_y are the horizontal and vertical coordinates of the center point of the predicted bounding box, respectively; t_w and t_h are the width and height of the predicted bounding box, respectively; p_w and p_h are the width and height of the prior bounding box, respectively; c_x and c_y are the offset of the distance between the prior bounding box and the upper-left cell of the image, respectively; $\sigma(x)$ is the $sigmoid(x)$ function.

According to Eq. (15), the NMS algorithm filters all the detection bounding boxes to be detected and selects the detection bounding box with the highest score. The input of NMS algorithm includes all detection bounding box set $B = \{b_1, \ldots, b_i, \ldots, b_N\}$, type confidence score set $S = \{S_1 \ldots S_i, \ldots, S_N\}$ and preset score threshold N_t. NMS initially traverses sets B and S to find the detection bounding box M with the highest score. Then, NMS calculates the IoU values of M and other detection bounding box b_i. If IoU(M, b_i) is greater than N_t, then b_i has a score of 0. If IoU(M, b_i) is less than N_t, then the score S_i of b_i is maintained and placed into an empty set D until all detection bounding boxes are calculated. NMS finally outputs sets D and S.

$$S_f, = \begin{cases} S_i, & \text{IoU}(M, b_i) < N_t \\ 0, & \text{IoU}(M, b_i) \geq N_t \end{cases}, \tag{15}$$

where b_i is the i^{th} boxes to be detected, S_i is the original score of b_i, S_f is the final score of b_i, IoU(M, b_i) is the combined ratio of M and b_i, and N_t is the preset score threshold.

B. FAULT DETERMINATION

Fig. 6 shows the object detection of the faulty image of the freight train. Step 1 inputs the freight train image, extracts the image features and fusion features through the BD-YOLO model, and predicts the fault location. Step 2 obtains the normal images and images with faults by classification. Fault types are divided into upper lever, locking plate, and truncated plug door handle.

IV. EXPERIMENTS AND RESULTS

In this section, the data set in the experiment is initially described, and then the software and hardware configuration of the experimental environment, parameter setting of

BD-YOLO, and evaluation performance metrics are introduced. Finally, the proposed model is compared and tested for robustness, and the experimental results are analyzed.

A. EXPERIMENT DATA SET

A total of 11,769 images are randomly selected from the images collected by the TFDS system to develop a data set. The dataset uses images of freight trains in the Zhuzhou Rolling Stock Depot, which are captured by the TFDS system of China Railway Guangzhou Group Company Limited with an initial pixel size of 1400 × 1024. LabelImg software is used to annotate the position of the object bounding box in the data set to generate XML files, and the data set is set to PASCAL VOC format. The labels of the data set images include truncated plug door handle, upper lever, locking plate, and normal. In this experiment, the training data set and test data set are randomly divided at a ratio of 9:1. The training dataset contains 10,589 images, covering all categories of fault and normal images. The test dataset which named Test1180 contains 1,180 images.

Test1176 contains 1,176 images, covering three categories of fault and normal images. In this experiment, the training data set and test data set are randomly divided at a ratio of 1:1. In addition, 300 fault images of three types and 100 normal images, namely, upper lever, locking plate, and truncated plug door handle, are randomly selected to generate a test dataset, Test400.

In actual application scenarios, images obtained from TFDS are susceptible to image distortion due to the influence of weather and illumination changes. Images in Test400 are flipped or added with noise to simulate the poor outdoor environment. Test400 is processed by adding Gaussian noise, random noise, salt-and-pepper noise, horizontal flip, vertical flip, and horizontal vertical flip as well as generates Test400AddGN, Test400AddRN, Test400AddSPN, Test400AddFH, Test400AddFV, and Test400AddFHV test data sets, respectively. The robust test results and analyses of the BD-YOLO model are introduced in Section IV-E.

B. PARAMETER SETTINGS

The BD-YOLO model is implemented based on Pytorch framework. All experiments are conducted on a platform with GTX1080Ti GPU and 128 GB memory. The software configurations of the experiment platform are as follows: Ubuntu 20.0.0 OS, Python 3.6, Torch 1.2.0, Torchvision 0.4.0, Cuda 10.0, and Cudnn 7.4.1.5.

In BD-YOLO model training, parameters are set as follows: learning rate is 0.001, iteration is 100, smooth label is 0.01, and confidence score threshold is 0.5. The BD-YOLO model uses cosine decay with warm up. The BD-YOLO model uses freezing training to prevent the weights from being destroyed at the initial stage of training. BD-YOLO does not have to start training again even fault occurs.

PAS uses k-means clustering algorithm [30] to aggregate the same type of initial prior bounding boxes. Different initial prior bounding boxes affect the results of detection bounding boxes. Table 1 shows that three groups of different initial prior bounding boxes of freight train data sets are used in the experiment. The first type is generated by the YOLOv4 model based on COCO format. The second type is generated by the YOLOv4 model based on VOC format. The third type is generated by BD-YOLO based on VOC format.

TABLE 1. Parameters of initial prior bounding boxes cluster center.

Model	Dataset format	Cluster center
YOLOv4 [7]	COCO	16, 89, 19, 43, 53, 50, 61, 88, 67, 93, 67, 88, 68, 85, 71, 72, 75, 87
YOLOv4 [7]	VOC	15, 97, 18, 44, 22, 85, 54, 49, 62, 89, 67, 94, 68, 90, 68, 86, 72, 73
BD-YOLO	VOC	22, 88, 38, 55, 49, 80, 116, 81, 122, 85, 118, 86, 91, 93, 95, 114

C. PERFORMANCE METRICS

Seven performance metrics, namely, recall, precision, average precision (AP), mean average precision (mAP), F1, log average miss rate (LAMR), and frames per second (FPS) are used to evaluate the performance of the BD-YOLO model. When the confidence score of the sample is greater than the preset threshold, the sample is judged to be a positive sample. Otherwise, it is judged to be a negative sample. When a positive sample s_p is predicted to be a positive sample, the prediction result is accurate, and the result is true positive (TP). When a negative sample s_p is predicted to be a negative sample, the prediction result is accurate, and the result is true negative (TN). When a negative sample s_p is predicted to be a positive sample, the prediction result is inaccurate, and the result is false positive (FP). When a positive sample s_p is predicted as negative sample, the prediction result is inaccurate, and the result is false negative (FN).

1) RECALL

Recall is the ratio of positive samples with accurate predictions to all positive samples with accurate predictions. Recall is expressed as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{FN} + \text{TP}}. \tag{16}$$

2) PRECISION

Precision is the ratio between the predicted positive samples and the actual positive samples. Precision is calculated as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{FP} + \text{TP}}. \tag{17}$$

3) F1

A contradiction is found between precision and recall. Therefore, precision and recall results need to be combined. F1 is a compromise between precision and recall. A higher F1 value indicates a better detection performance of the model. F1 can

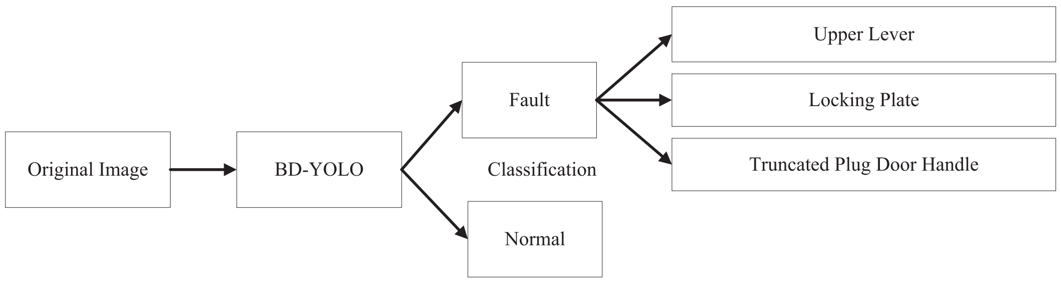


FIGURE 6. Fault image object detection.

be formulated as follows:

$$\mathrm{F1} = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precsion} + \mathrm{Recall}}. \tag{18}$$

4) AP

AP is the average of the precision of a certain category. A higher AP value indicates a better detection effect of a certain category. AP can be expressed as follows:

$$\mathrm{AP} = \int_0^1 \mathrm{P}(r)dr, \tag{19}$$

where P is precision, r is recall, and $\mathrm{P}(r)$ is the maximum precision corresponding to recall.

5) mAP

mAP is the average of the sum of APs of all categories in the data set. A higher the mAP indicates a higher average accuracy of all categories of the model. mAP can be formulated as follows:

$$\mathrm{mAP} = \sum_{s=0}^{class} \mathrm{AP}_s \Big/ class, \tag{20}$$

where s is AP of the class, and $class$ is the number of sample categories in the data set.

6) LAMR

LAMR represents the logarithmic average false detection rate. A smaller LAMR indicates a smaller number of false detections and a better model detection effect. LAMR is calculated as follows:

$$\mathrm{LAMR} = exp\left[\frac{1}{n}\sum_{h=1}^{n} \ln a_h\right], \tag{21}$$

where $a_1, a_2, \ldots, a_h$ is the miss rate of n false positive per image points uniformly distributed between [0.01, 1].

7) FPS

A higher FPS indicates more frames transmitted in a second, and a faster detection speed of the model. FPS can be computed as follows:

$$\mathrm{FPS} = \frac{frameNum}{elapsedTime}, \tag{22}$$

where *frameNum* represents the number of frames, *elapsedTime* represents a fixed time, and the interval is 1 s. This experiment calculates FPS by recording the number of frames within a fixed time.

D. EXPERIMENT RESULTS AND ANALYSIS

Many types of freight train components are available. This paper mainly focuses on the detection of three typical fault images of freight train components, namely, truncated plug door handle, locking plate, and upper lever. In the data set, the closed truncated plug door handle images are shown in Figs. 7(a) and 7(b). The normal images of the truncated plug door handle are shown in Figs. 7(c) and 7(d). The offset of the locking plate images is shown in Figs. 7(e) and 7(f). The normal images of the locking plate are shown in Figs. 7(g), and 7(h). The upper lever jumping out images are shown in Figs. 7(i), 7(j) and 7(k). The normal images of the upper lever are shown in Figs. 7(l) and 7(m). There are many types of image freight train components.

Fig. 8 shows the prediction results obtained through the BD-YOLO model detection. The truncated plug door handle in Fig. 8(a) is the closing mark of the truncated plug door handle. The locking plate in Fig. 8(b) shows the mark of the offset of the locking plate. The upper lever in Fig. 8(c) is the mark of the upper lever jumping out. The normal images of the truncated plug door handle, the locking plate, and the upper lever are regarded as the normal category and the normal marks in Figs. 8(d), 8(e), and 8(f). The values next to all identifiers are category confidence scores.

In this experiment, the mosaic data enhancement method is used to expand the number of samples during training. It enables a single image to contain semantic information from four images. Its specific steps are shown in Fig. 9. Step 1 randomly imports four images, as shown in Fig. 9(a). The four imported images are flipped, zoomed, and color gamut changed, and the size of the image is randomly cropped in Step 2. The cropped image is placed in four directions, namely, top left, top right, bottom left, and bottom right, as shown in Fig. 9(b). Step 3 splices the placed image into one image, as shown in Fig. 9(c).

Table 2 shows the test results of the BD-YOLO model in the Test1180. NORMAL indicates the normal category,

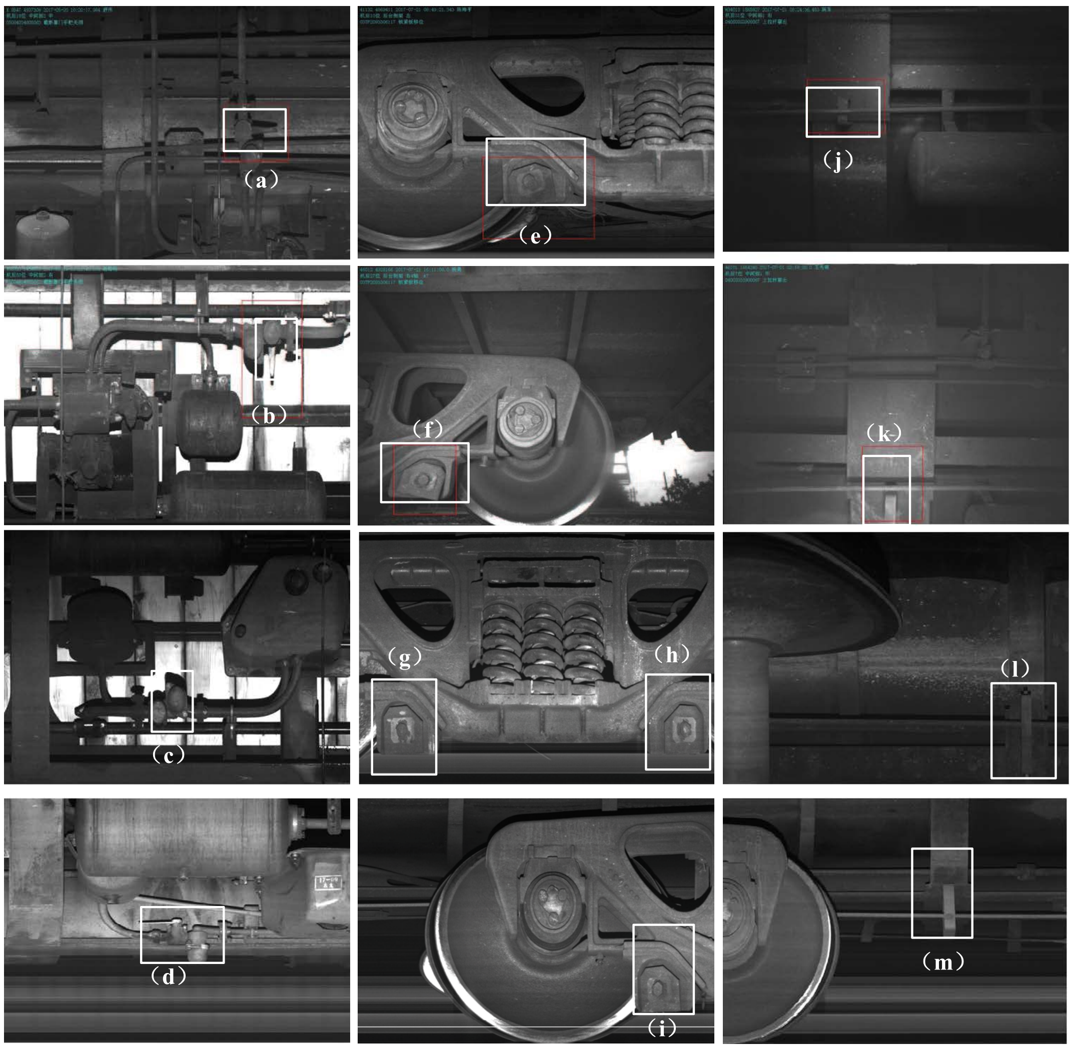


FIGURE 7. Faulty and normal images.

UL indicates the upper lever jumping out category, LP indicates the locking plate deviation category, and TPDH indicates the closed truncated plug door handle category. The experimental results show that the BD-YOLO model performs effectively in the detection of all categories. Specifically, the AP of TPDH is 100%, the F1 of UL is 96%, and the LAMR of TPDH and UL is 0.

When the ratio of training set to test set is 1:1, as shown in Table 3, the experimental results of the BD-YOLO model on Test1176 show a slight decrease in detection accuracy, while reaching the same accuracy. This finding also highlights BD-YOLO's ability to achieve equally accurate results when training with a small scale of images. The feature information of TPDH is more evident than that of other categories, which is why AP and F1 values of TPDH are the highest in Table 3.

To verify the performance of the BD-YOLO model, the detection results of test dataset with the well-known YOLOv4, CenterNet, EfficientDet, RetinaNet, and Faster R-CNN models are shown in Table 4. Specially, AP_{NOAML} is the AP of the normal category, AP_{UL} is the AP of the upper lever fault category, AP_{LP} is the AP of the locking plate fault category, and AP_{TPDH} is the AP of the truncated door plug handle fault category. The BD-YOLO model improves mAP by 22.41%, 19.99%, 17.21%, 12.25% and 9.54%, compared with YOLOv4, CenterNet, EfficientDet, RetinaNet, and Faster R-CNN, respectively. The FPS of BD-YOLO is 32, and the FPS of CenterNet is 46. Although the FPS of BD-YOLO is not as good as that of CenterNet, it is faster than those of the other object detection models. The reason is that CenterNet is an object detection model based on the anchor-free method without the calculating anchor. Thus, the FPS of CenterNet is higher than that of BD-YOLO.

After obtaining the initial prior boxes in Table 1, the results of the ablation experiment of K-means clustering algorithm are shown in Table 5. CenterNet is based on the anchor-free mechanism. Thus, only anchor-based YOLOv4 and BD-YOLO models are selected for comparison. The experimental results show that the mAP and FPS of the

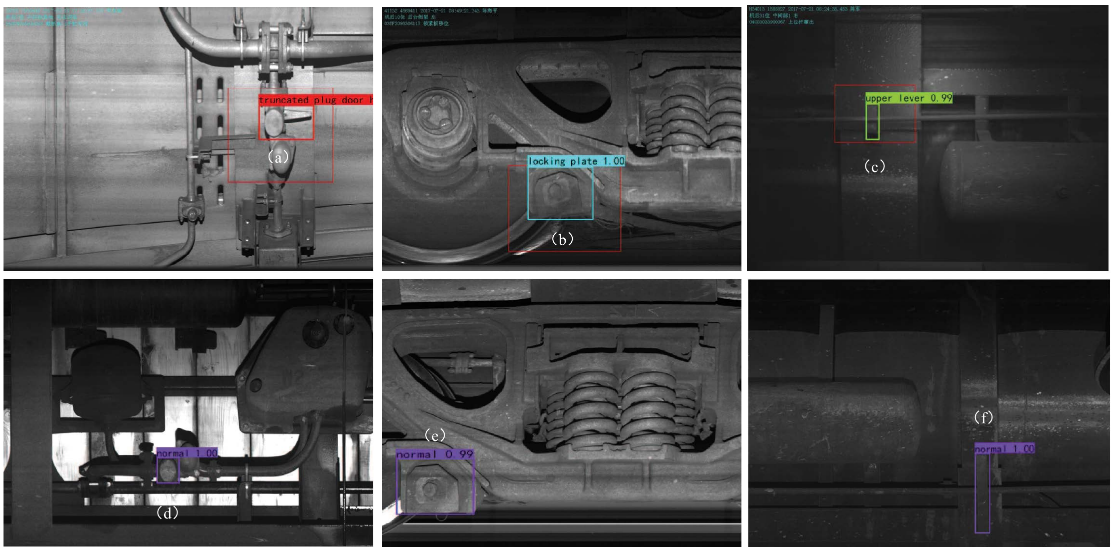


FIGURE 8. Detection result under the BD-YOLO model.

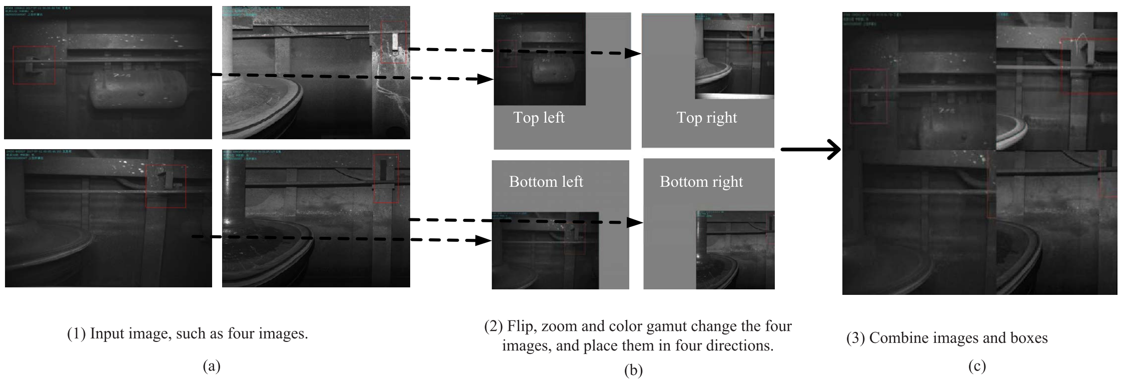

Top left
Top right
Bottom left
Bottom right
(1) Input image, such as four images.
(a)
(2) Flip, zoom and color gamut change the four images, and place them in four directions.
(b)
(3) Combine images and boxes
(c)

FIGURE 9. Mosaic data enhancement example.

TABLE 2. Results for Test1180 on BD-YOLO.

Metric	Type name			
	NOAML	UL	LP	TPDH
AP	90.68%	96.21%	91.29%	100.00%
F1	90.12%	96.53%	91.34%	93.21%
Precision	87.10%	72.97%	91.67%	87.50%
Recall	93.91%	100.00%	96.43%	100.00%
LAMR	19.56%	15.23%	0.00%	0.00%

prior bounding box generated by the YOLOv4 model without K-means clustering algorithm are 70.95% and 27, respectively. Compared with a prior bounding box generated by the YOLOv4 model using the K-means clustering algorithm, the mAP of prior bounding box generated by the K-means clustering algorithm is increased by 13.27%, and FPS is increased by 5. The mAP and FPS of the prior bounding box generated by the BD-YOLO model without K-means clustering algorithm are 93.36% and 32, respectively. The priori bounding box generated by BD-YOLO using K-means clustering algorithm is improved by 1.19% compared with the priori bounding box generated without K-means, and FPS is improved by 2. The experimental results show that the mAP and FPS of the model can be improved using K-means clustering algorithm.

E. ROBUSTNESS TEST

1) ROBUSTNESS TEST FOR ADDING NOISE

Table 6 shows the experimental results of the BD-YOLO model on Test400. Evidently, BD-YOLO performs well in the

TABLE 3. Results for Test1176 dataset on BD-YOLO.

Metric	Type name			
	NOAML	UL	LP	TPDH
AP	88.81%	94.75%	89.90%	100.00%
F1	88.23%	86.12%	79.19%	93.37%
Precision	86.55%	90.34%	68.42%	87.50%
Recall	89.57%	81.82%	92.86%	100.00%
LAMR	22.21%	0.00%	19.14%	0.00%

TABLE 4. Classical model comparison experiment results.

Model	Performance					
	mAP	AP_{NOAML}	AP_{UL}	AP_{LP}	AP_{TPDH}	FPS
YOLOv4 [7]	70.95%	86.00%	56.00%	73.00%	69.00%	27
CenterNet [23]	73.37%	85.00%	65.00%	72.00%	71.00%	**46**
EfficientDet [10]	76.15%	78.00%	49.00%	78.00%	**100.00%**	18
RetinaNet [9]	80.61%	**89.92%**	82.27%	76.91%	73.33%	12
Faster R-CNN [16]	83.82%	79.70%	83.84%	82.23%	78.57%	13
BD-YOLO	**93.36%**	89.00%	**95.00%**	**90.00%**	**100.00%**	32

TABLE 5. Results of k-means ablation experiment for different models.

Model	Method	Metric					
		mAP	AP_{NOAML}	AP_{UL}	AP_{LP}	AP_{TPDH}	FPS
YOLOv4 [7]	None	70.95%	86.00%	56.00%	73.00%	69.00%	27
YOLOv4 [7]	K-means	84.22%	**92.82%**	83.86%	74.52%	85.66%	32
BD-YOLO	None	93.36%	89.00%	95.00%	90.00%	**100.00%**	32
BD-YOLO	K-means	**94.55%**	91.00%	**96.00%**	**91.00%**	**100.00%**	**34**

TABLE 6. Results for Test400 on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	89.26%	92.31%	90.42%	91.00%	18.00%
UL	73.13%	89.09%	83.48%	80.00%	30.00%
LP	64.71%	90.41%	82.95%	75.00%	29.00%
TPDH	100.00%	100.00%	100.00%	100.00%	0.00%

four categories of the train images. The AP of TPDH is 100%, F1 is 100%, and LAMR is 0%. The experimental results show that the detection effect of BD-YOLO on Test400 on TPDH is better than that of the other types of train images.

Table 7 shows the experimental results of the BD-YOLO model on Test400AddGN. Compared with Test400, the AP of NORMAL with Gaussian noise under BD-YOLO increases by 0.14%, F1 is decreased by 1%, and LAMR does not change. This finding shows that Gaussian noise has a minimal influence on NORMAL images. Compared with the UL of Test400, the AP of UL with Gaussian noise is decreased by 52.80%, F1 is reduced by 47%, and LAMR is increased by 42%. The BD-YOLO model has a substantial drop in the detection effect of UL with Gaussian noise because Gaussian noise is densely distributed on the image, covering the position of the upper lever. The finding evidently shows that the addition of Gaussian noise also has a great influence on the detection effect of the BD-YOLO model in detecting the closing of the truncated plug door handle.

TABLE 7. Results for Test400AddGN with Gaussian noise on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	88.52%	92.31%	90.59%	90.00%	18.00%
UL	100.00%	20.00%	30.68%	33.00%	72.00%
LP	70.89%	76.71%	73.05%	74.00%	43.00%
TPDH	100.00%	75.00%	82.82%	86.00%	19.00%

TABLE 8. Results for Test400AddRN with random noise on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	89.26%	92.31%	90.46%	91.00%	18.00%
UL	76.19%	87.27%	87.13%	81.00%	26.00%
LP	65.00%	89.04%	82.75%	75.00%	29.00%
TPDH	100.00%	100.00%	100.00%	100.00%	0.00%

Table 8 shows the test results of the BD-YOLO model on Test400AddRN. Compared with the UL of Test400, the AP of UL with random noise is increased by 3.65%, F1 is increased by 1%, and LAMR is reduced by 1%. Compared with the TPDL of Test400, the AP, F1, and LAMR of TPDH with random noise do not change. In general, the BD-YOLO model slightly improves the fault image detection effect after adding random noise.

TABLE 9. Results for Test400AddSPN with salt-and-pepper noise on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	87.80%	92.31%	89.69%	90.00%	20.00%
UL	79.07%	61.82%	65.16%	69.00%	44.00%
LP	58.24%	72.60%	63.97%	65.00%	55.00%
TPDH	100.00%	89.58%	93.45%	90.00%	8.00%

Table 9 shows the test results of the BD-YOLO model on Test400AddSPN. Compared with the UL of Test400, the AP of UL with salt-and-pepper noise is decreased by 18.32%, F1 is reduced by 21%, and LAMR is increased by 14%. Compared with the LP of Test400,the AP of LP with salt-and-pepper noise is decreased by 18.98%, F1 is reduced by 10%, and LAMR is increased by 26%. Although the AP of UL, TPDH, and LP are decreased after adding salt-and-pepper noise, the AP of NORMAL is increased by 0.73%.

2) ROBUSTNESS TEST FOR FLIPPING

After flipping the image horizontally, the left and right positions of the target object are mirrored. Table 10 shows the experimental results of the BD-YOLO model on the test set Test400AddFH. Compared with the NORMAL of the Test400 experiment result, AP of NORMAL after the horizontal flip is reduced by 2.51%, F1 is reduced by 1%, and LAMR is increased by 2%. The experiment proves that the horizontal flip has a small effect on the detection effect of NORMAL. Compared with the UL after the horizontal flip of the Test400 experiment result, AP of UL is reduced by 30.29%, F1 is reduced by 19%, and LAMR is increased by 29%. The experiment proves that the horizontal flip has a great influence on the detection effect of UL. Compared with the LP of the Test400 experiment result, AP of LP after the horizontal flip is increased by 3.72%, the F1 is increased by 1%, and the LAMR is decreased by 3%. The experiment proves that the detection effect of LP is improved after the horizontal flip. Compared with the TPDH of Test400 experiment results, AP of TPDH after the horizontal flip is reduced by 8.3%, F1 is reduced by 7%, and LAMR is increased by 11%. Table 10 clearly shows that the horizontal flipping of the image has a great influence on the detection effect of UL and TPDH.

TABLE 10. Results for Test400AddFH with flip horizontally on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	87.80%	91.53%	87.91%	90.00%	22.00%
UL	60.00%	61.02%	53.19%	61.00%	59.00%
LP	66.00%	90.41%	86.67%	76.00%	26.00%
TPDH	97.92%	88.68%	91.70%	93.00%	11.00%

Table 11 shows the experimental results of the BD-YOLO model on Test400AddFV. Compared with the NORMAL of Test400, the AP value of NORMAL after vertical flipping decreases by 5.77 %, F1 is reduced by 3%, and the LAMR is increased by 9%. Compared with the LP of Test400, the AP value of the LP after the vertical flipping is reduced by 56.17%, F1 is decreased by 40%, and LAMR is increased by 49%. Table 11 shows that image vertical flipping has a certain negative effect on UL, TPDH, and LP detection. In general, the mAP of the BD model is in an acceptable range in this case.

TABLE 11. Results for Test400AddFV with flip vertically on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	87.70%	87.70%	84.65%	88.00%	27.00%
UL	53.23%	55.93%	45.43%	55.00%	65.00%
LP	48.78%	27.78%	26.78%	35.00%	78.00%
TPDH	97.87%	86.79%	93.43%	92.00%	10.00%

The horizontal and vertical flipping of the image indicates that the image is flipped horizontally initially and then vertically. Compared with the original image, the object changes up, down, left, and right in the flipped image. Table 12 shows the experimental results of the BD-YOLO model on Test400AddFHV. Compared with the NORMAL of Test400, AP of NORMAL after the horizontal and vertical flipping is reduced by 3.06%, F1 is decreased by 2%, and LAMR is increased by 9%. Compared with the LP of Test400, AP of LP after the horizontal and vertical flipping is reduced by 58.57%, F1 is decreased by 46%, and LAMR is increased by 48%.

TABLE 12. Results for Test400AddFHV with flip horizontally and vertically on BD-YOLO.

Type name	Precision	Recall	AP	F1	LAMR
NORMAL	86.99%	91.45%	87.36%	89.00%	27.00%
UL	66.18%	76.27%	69.14%	71.00%	65.00%
LP	47.50%	26.39%	24.38%	34.00%	78.00%
TPDH	94.00%	88.68%	92.22%	91.00%	10.00%

F. RESULTS ANALYSIS

Table 13 shows the experimental results of the robustness test. The mAP of the BD-YOLO model in Test400 is 89.21%. The mAPs of the BD-YOLO model using the test sets Test400AddGN and Test400AddSPN are 69.23% and 78.07%, respectively. Compared with the mAP of BD-YOLO in Test400, their mAPs are reduced by 19.98% and 11.14%. The mAP of the BD-YOLO model in Test400AddRN is 90.09%. The mAP of the BD-YOLO model using the test set Test400AddRN is increased by 0.88% compared with that in Test400.

The mAPs of the BD-YOLO model in Test400AddGN and Test400AddSPN are 19.98% and 11.14% lower than that in Test400, respectively, due to the dense distribution of Gaussian noise and salt-and-pepper noise, which seriously affect the image quality. The random noise distribution is sparse, which almost does not affect image quality. The random noise distribution is sparse, which almost does not affect the image quality. The mAP of BD-YOLO is slightly increased in Test400AddRN. The experimental results show that BD-YOLO model has good noise disturbance.

The mAPs of the BD-YOLO model in Test400AddFH, Test400AddFV, and Test400AddFHV are 79.87%, 62.57% and 68.28%, respectively. Their maps are 9.34%, 26.64%, and 20.93% lower than those on Test400.

Under the influence of image flipping, the mAP of the BD-YOLO model is decreased in all data sets after flipping, and that in Test400AddFV is decreased the most. When the image is flipped horizontally, the left and right positions of the object are only mirror changes, without causing a large change in the position of the object in the image. Therefore, it has a minimal influence on the mAP of the BD-YOLO model.

However, when the image is vertically flipped, the upper and lower positions of the object in the image change greatly, thus considerably affecting the mAP of the BD-YOLO model. In general, the average mAP of the BD-YOLO model in the test sets after flipping is 70.24%, indicating that the BD-YOLO model can resist the interference of image flipping.

TABLE 13. Experimental results of robustness tests on BD-YOLO.

Dataset	Data enhancement method	mAP
Test400	None	89.21%
Test400AddGN	Gaussian Noise	69.28%
Test400AddRN	Random Noise	**90.09**%
Test400AddSPN	Salt-and-pepper noise	78.07%
Test400AddFH	Flip horizontally	79.87%
Test400AddFV	Flip vertically	62.57%
Test400AddFHV	Flip horizontally and vertically	68.28%

V. CONCLUSION

An object detection model for the image fault detection of freight trains, BD-YOLO, is proposed in this paper. The proposed BD-YOLO model consists of five modules, namely, FEN, FIN, MFF, PAS, and decoding module. FEN is used to extract the features of the object. FIN and MFF module are responsible for fusing the extracted features to obtain more detailed feature information. PAS is responsible for the output of three different scale feature layers containing the predicted results. The feature layer obtains the position of the detection bounding box through decoding. The BD-YOLO model uses the mosaic data enhancement method to increase the number of data sets. It also uses K-means clustering algorithm to generate better prior bounding boxes to improve the precision of the model. The experimental results show that the maximum mAP of BD-YOLO in the test data set is 94.55%, and the FPS is 34. Compared with the well-known YOLOv4, CenterNet, RetinaNet, EfficientDet, and Faster R-CNN object detection models, the mAP of BD-YOLO model is improved by 23.6%, 21.18%, 13.94%, 18.4%, and 10.73%, respectively. The BD-YOLO model has better detection accuracy and speed, and enhanced robustness in complex environments.

BD-YOLO can improve the precision of the fault image detection of freight trains and realize multifault object detection. Our next plan is to research on improving detection speed without reducing detection accuracy.

ACKNOWLEDGMENT

The authors would like to thank the three anonymous reviewers for their constructive comments.

REFERENCES

[1] J. Sun, Z. Xiao, and Y. Xie, "Automatic multi-fault recognition in TFDS based on convolutional neural network," *Neurocomputing*, vol. 222, pp. 127–136, Jan. 2017.

[2] L. Liu, F. Zhou, and Y. He, "Automated visual inspection system for bogie block key under complex freight train environment," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 1, pp. 2–14, Jan. 2016.

[3] X. Fu, K. Li, J. Liu, K. Li, Z. Zeng, and C. Chen, "A two-stage attention aware method for train bearing shed oil inspection based on convolutional neural networks," *Neurocomputing*, vol. 380, pp. 212–224, Mar. 2020.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[5] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

[6] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[7] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[10] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10781–10790.

[11] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 840–849.

[12] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[13] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7036–7045.

[14] L. Zhu, Z. Deng, X. Hu, C.-W. Fu, X. Xu, J. Qin, and P.-A. Heng, "Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 121–136.

[15] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 91–99.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[18] K. Kang, K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, and W. Ouyang, "T-CNN: Tubelets with convolutional neural networks for object detection from videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2896–2907, Oct. 2018.

[19] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.

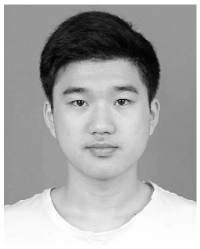
[20] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards balanced learning for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 821–830.

[21] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "DenseBox: Unifying landmark localization with end to end object detection," 2015, *arXiv:1509.04874*.

[22] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 734–750.
[23] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6569–6578.
[24] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, "FoveaBox: Beyound anchor-based object detection," *IEEE Trans. Image Process.*, vol. 29, pp. 7389–7398, 2020.
[25] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: A simple and strong anchor-free object detector," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 1922–1933, Apr. 2020.
[26] Y. Liu, Y. Wang, S. Wang, T. Liang, Q. Zhao, Z. Tang, and H. Ling, "CBNet: A novel composite backbone network architecture for object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 11653–11660.
[27] X. Dong, L. Zheng, F. Ma, Y. Yang, and D. Meng, "Few-example object detection with model communication," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1641–1654, Jul. 2019.
[28] G. Cheng, J. Yang, D. Gao, L. Guo, and J. Han, "High-quality proposals for weakly supervised object detection," *IEEE Trans. Image Process.*, vol. 29, pp. 5794–5804, 2020.
[29] P. Tang, X. Wang, S. Bai, W. Shen, X. Bai, W. Liu, and A. Yuille, "PCL: Proposal cluster learning for weakly supervised object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 1, pp. 176–191, Jan. 2020.
[30] T. Zhang, F. Ma, D. Yue, C. Peng, and G. M. P. O'Hare, "Interval type-2 fuzzy local enhancement based rough K-means clustering considering imbalanced clusters," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 1925–1939, Sep. 2020.

LONGXIN ZHANG received the Ph.D. degree in computer science from Hunan University, China, in 2015. He was also a Visiting Scholar with the University of Florida, from 2019 to 2020. He is currently an Associate Professor in computer science with the Hunan University of Technology. His research interests include modeling and scheduling for distributed computing systems, distributed system reliability, parallel algorithms, cloud computing, and deep learning. He has published more than 20 technique papers in international journals and conferences, such as IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Sustainable Computing, *Information Sciences*, *Peer-to-Peer Networking and Applications*, and ICA3PP. He is a Reviewer of IEEE Transactions on Parallel and Distributed Systems, *ACM TIST*, IEEE Transactions on Industrial Informatics, IEEE Internet of Things Journal, *INS*, and *ASC*.

MIAO WANG is currently pursuing the master's degree in computer science with the Hunan University of technology. Her research interests include object detection and deep learning.



KE LIU is currently pursuing the master's degree in computer science with the Hunan University of Technology. His research interests include high-performance computing and big data analysis.

MANSHENG XIAO received the M.S. degree in computer science and technology from Xi'an Jiaotong University, China, in 2005. He is currently a Professor in computer science and technology with the Hunan University of Technology. His research interests include intelligence computation, intelligent information processing, pattern recognition, and image processing.

ZHIHUA WEN is currently pursuing the Ph.D. degree in informatics of TCM with the Hunan University of Traditional Chinese Medicine. He is also a Lecturer in computer science with the Hunan University of Technology. His research interests include big data, natural language processing, graph neural networks, and intelligent diagnosis.

JUNFENG MAN received the Ph.D. degree from the School of Information Science and Engineering, Central South University, China, in 2010. He is currently a Professor in computer science with Hunan First Normal University. His research interests include networked software, big data technology, machine learning, knowledge representation, and artificial intelligence.