Research Article

Yolov4 High-Speed Train Wheelset Tread Defect Detection System Based on Multiscale Feature Fusion

Changfan Zhang, Xinliang Hu, Jing He, and Na Hou

1School of Electrical and Information Engineering, Hunan University of Technology, Zhuzhou 412007, Hunan, China
2School of Railway Transportation, Hunan University of Technology, Zhuzhou 412007, Hunan, China

Correspondence should be addressed to Jing He; hejing@263.net

Received 24 September 2021; Revised 20 January 2022; Accepted 28 January 2022; Published 27 March 2022

Academic Editor: Seyed Ali Ghahari

Copyright © 2022 Changfan Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The Yolov4 detection algorithm does not sufficiently extract local semantic and location information. This study aims to solve this problem by proposing a Yolov4-based multiscale feature fusion detection system for high-speed train wheel tread defects. First, multiscale feature maps are obtained from a feature extraction backbone network. The proposed multiscale feature fusion network then fuses the underlying features of the original three scales. These fused features contain more defect semantic information and location details. Based on the fused features, a path aggregation network is used to fuse feature maps at different resolutions, with an improved loss function that speeds up the convergence of the network. Experimental results show that the proposed method is effective at detecting defects in the wheel treads of high-speed trains.

1. Introduction

High-speed train fault diagnosis is crucial for the safe operation and maintenance of high-speed trains. As an important support and running component of high-speed trains, wheelsets incur tread wear, scratches, and other damages caused by rolling contact between the wheel and rail. The deterioration can easily cause serious damage, such as wheelset tread fractures. Therefore, it is essential to diagnose wheelset tread damage to ensure the safety of high-speed trains.

Current defect detection methods include magnetic particle detection, ultrasonic detection, and machine vision detection [1]. In recent years, many scholars have studied machine vision methods [2–5], owing to their wide application range and high precision. Such methods are efficient at inspecting the damage and unaffected by the contour of the inspection part. Moreover, machine vision inspection can proceed automatically. Traditional defect detection methods based on machine vision for railway components begin from the perspective of image processing. On the one hand, they use low-level grayscale features [6], textures, colors, frequencies, and other features to detect defects, yet such artificial features must be selected manually and require rich expert knowledge. On the other hand, the damage is automatically located after image enhancement [7–9], although the images are susceptible to noise. Given the complex operating scenarios of high-speed rail, changing working conditions, and the nonlinearity of the sensor itself, the signals collected by the sensors often contain foreground interference, noisy backgrounds, and nonlinear characteristics such as corrosion, stains, uneven reflection, a low signal-to-noise ratio, excessive illumination, and uneven illumination. It is difficult for traditional machine vision defect detection methods to effectively extract small fault features under foreground interference and noise. Furthermore, the differences between wheelset tread defects are not obvious under these two conditions.

1.1. Defect Detection Based on a Deep Convolutional Neural Network (DCNN). Considering that deep learning has exhibited superiority in feature extraction and pattern recognition, an increasing number of scholars have
attempted to apply deep learning methods to defect detection [10]. Deep learning based on convolutional neural networks is widely used to detect railway component damage [11–13]. Faghih-Roohi et al. [14] proposed a DCNN with multiple structures and activation properties for rail damage detection. High-speed train wheelset tread defect detection requires scene analysis at the regional level. Current regional-level target detection methods based on DCNNs are generally divided into two types. One type is two-stage detection based on the region, such as the Regions with CNN (R-CNN) features [15] and Faster R-CNN [16]. Liu et al. [12] integrated the feature extraction module isoelectric line network (ILNET) in the Faster R-CNN and used the segmentation method intersecting the cortical model/maximization of the posterior marginal based on a Markov random field to locate and divide loose strands of isoelectric lines. He et al. [17] proposed an end-to-end two-step defect detection method for steel rolling defects and explored the trade-off between detection speed and accuracy for different numbers of regions. However, the detection speed of two-step algorithms is still slower than that of single-step detection algorithms, and it is difficult to deal with the problem of the short maintenance operation time for China’s high-speed railways. The other type is a series of single-step detection algorithms, including the you only look once (YOLO) series [18], which includes single-shot multitask detector (SSD) [19] and RetinaNet [20]. Kou et al. [21] introduced the DenseNet module in Yolov3 for strip defects and proposed a strip defect detection method based on Yolov3. Based on RetinaNet, Cheng et al. [22] proposed a retinal network (DEA RetinaNet) defect detection model based on differential channel attention and adaptive spatial feature fusion. Cui et al. [23] proposed an SSDNet for defect detection, which solved the problem of large texturing and small size defect detection by introducing feature retention blocks and skip dense connection modules. Considering that the YOLO series of target detection algorithms use fast detection tasks, resulting in low detection accuracy. To address this limitation, we propose a multiscale feature fusion module. The proposed module provides rich semantic information and location details after the feature extraction stage. Low-level features are integrated into high-level features, and the fused features effectively improve the classification and localization capabilities of the detection network.

1.3. Transfer Learning. Deep neural networks require a large number of datasets as a drive. For wheelset treads, building a large target detection dataset is very difficult. If the dataset is too small, then the performance of the deep neural network will be limited. Therefore, pretraining the network or transfer learning is commonly used for small samples. Yang et al. [30], Zhang et al. [31], Badmos et al. [32], and Sun et al. [33] used transfer learning to detect Mura defects in LCD panels, PCB defects, electrode defects in lithium batteries, and surface defects in metal parts. Kim et al. [34] compared the effects of fine-tuning-based transfer learning and training the network from scratch on the DAGM defect dataset. They demonstrated that transfer learning outperforms training the network from scratch.

The contributions of this study are as follows:

1. This study proposes an MFF-Yolov4 high-speed train wheelset tread defect detection algorithm. Yolov4 uses only three high-level features to perform detection tasks, resulting in low detection accuracy. To address this limitation, we propose a multiscale feature fusion module. The proposed module provides rich semantic information and location details after the feature extraction stage. Low-level features are integrated into high-level features, and the fused features effectively improve the classification and localization capabilities of the detection network.

2. A dataset that contains 277 high-speed train wheelset images is built by collecting data to fine-tune our pretrained model. In the case of small samples, the proposed MFF-Yolov4 achieves competitive performance on this dataset.

3. Based on the two-class detection problem—and considering that noisy background candidate frames that are not related to wheelset tread defects contribute to most of the loss—the classification loss function is optimized, and the adjustment factor α is added, thereby achieving a better anti-interference performance of the improved network.

1.2. Multiscale Feature Fusion (MFF). Some scholars have attempted using high-quality feature learning for networks by combining the features of different scales. Yang et al. [27] designed a multiscale channel compression deep surface defect detection algorithm that generates multiscale features through the convolutional layers of different sizes of cores to address messy backgrounds and defects of various scales. The added convolutional layer is compressed to increase the speed of the network. Hu et al. [28] proposed a spatio-temporal segmentation model with a hybrid multidimensional feature fusion structure for automatic thermal imaging defect detection. An attention module was designed that encourages local interaction between adjacent pixels and calibrates the feature map self-adaptively to lighten the model. Gao et al. [29] used feature acquisition and a compression network for multiscale feature fusion of IBD defect detection and used Gaussian weighted pooling instead of ROI pooling. Their method provides more accurate defect location information.

2. Construction of Defect Detection Model for Wheelset Treads

The proposed MFF-Yolov4 algorithm is described in this section (see Figure 1). A single wheelset tread image of any size is processed by CSPDarknet53 and SPP for feature
extraction, and a convolutional feature map at each stage is generated. Four feature maps are extracted and then merged into three-dimensional feature outputs through MFF. In this way, MFF contains both the bottom-level location information of the wheelset tread image and the high-level classification semantics. Subsequently, a path aggregation network (PANet) [35] is used to perform a secondary fusion of the three fusion features with underlying location information. Next, the three fusion features at different scales are divided into grids to predict the bounding box for each grid and then return to the basic truth box. Each scale feature map stores the defect categories (“classes”), regression parameters of prior boxes (“location”), and confidence scores (“confidence”) that correspond to the three prior boxes of each grid.

2.1. Backbone Network. Preprocessing the model on the VOC2007 dataset can improve the performance of the deep neural network. The preprocessing model can be fine-tuned on a smaller wheelset tread defect dataset. Given that MFF-Yolov4 is based on the Yolov4 target detection algorithm, the use of CSPDarknet53 and SPP as the backbone has the following characteristics:

(1) According to research by the author of Yolov4, CSPDarknet53 has the following advantages over the CSPResnext50 network: a higher input network size, which is conducive to the detection of small objects; a larger receptive field that covers a larger input network; more parameters to improve the model with single images; and the ability to detect multiple objects of different sizes.

(2) SPP can significantly increase the receptive field and isolate important context features. Furthermore, it will not increase the operating speed of the network.

In this study, CSPDarknet53 and SPP are selected as the backbone. The detailed structure of the network is shown in Table 1 and the output feature of the last layer of each CSP module is expressed as {F1, F2, F3, F4, F5, F6}.

2.2. Multiscale Feature Fusion. Yolov4 uses only three high-level scale features for feature extraction. The evolution of wheelset tread damage is a coupled development process. The network proceeds from shallow to deep, features are mapped to high-dimensional space, and the overall semantic information is gradually strengthened in abstraction. However, the hidden positioning information and local semantic features are gradually weakened layer by layer. In order to strengthen the detection ability of Yolov4

<table>
<thead>
<tr>
<th>Block name</th>
<th>Type</th>
<th>Filter size</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBM</td>
<td>Conv1</td>
<td>3 × 3, 32, stride = 1</td>
</tr>
<tr>
<td>Csp1(F1)</td>
<td>RS1</td>
<td>1 × 1, 128, 3 × 3, 64, stride = 2</td>
</tr>
<tr>
<td>Csp2(F2)</td>
<td>Conv3</td>
<td>3 × 3, 128, stride = 2</td>
</tr>
<tr>
<td>Csp3(F3)</td>
<td>RS2</td>
<td>1 × 1, 128, 3 × 3, 64, stride = 2</td>
</tr>
<tr>
<td>Csp4(F4)</td>
<td>Conv4</td>
<td>3 × 3, 128, stride = 2</td>
</tr>
<tr>
<td>Csp5(F5)</td>
<td>RS3</td>
<td>1 × 1, 128, 3 × 3, 64, stride = 2</td>
</tr>
<tr>
<td>CBL</td>
<td>Conv5</td>
<td>3 × 3, 512, stride = 2</td>
</tr>
<tr>
<td>SPP</td>
<td>MaxPool</td>
<td>1 × 1, 128, 3 × 3, 512, stride = 2</td>
</tr>
<tr>
<td>CBL(F6)</td>
<td>Conv6</td>
<td>3 × 3, 1024, stride = 2</td>
</tr>
<tr>
<td>CBL(F7)</td>
<td>Conv7</td>
<td>1 × 1, 1512, 3 × 3, 1024, stride = 2</td>
</tr>
<tr>
<td>CBL(F8)</td>
<td>Conv8</td>
<td>1 × 1, 1512, 3 × 3, 1024, stride = 2</td>
</tr>
</tbody>
</table>

Figure 1: MFF-Yolov4 framework.
to detect wheel tread damage, we need to extend each single-scale feature to a dual-scale fusion feature. The method we use here is to fuse the high-resolution underlying features with high-level features through convolution transformation. Two basic conditions must be met: the features must be nonadjacent, because adjacent features must have high similarity [36], and the features of each scale generated by the backbone network should be taken into account.

From Table 1, we know that when the image flows through the feature extraction backbone network, \{F1, F2, F3, F4, F5, F6\} multig1 scale features are generated. The original Yolov4 network uses only three high-level feature maps, \{F3, F4, F6\}. In order to integrate multiple scale features to obtain more comprehensive semantic information and location details, the proposed MFF module performs a fusion strategy on the underlying feature F2. In particular, F2 is connected to the same F3, F4, and F6 after L2 normalization. By modifying the number of filters in the 1 × 1 convolution, most MFFs reduce the required parameters. This operation may affect the accuracy but will prevent overfitting in the case of insufficient training data. Finally, by stitching the features together, multiscale fusion features \{FF1, FF2, FF3\} are generated, as shown in Algorithm 1. Its structure is shown in Figure 2.

2.3. Path Aggregation Network (PANet). For FF1, with more comprehensive feature information, the three fusion feature layers of the input are stacked through PANet's step-by-step upsamling and downsampling. A second fusion is performed to obtain three effective feature layers, namely, P1, P2, and P3, as shown in Figure 3.

2.4. Yolo Head. In particular, Yolo head is integrated through a 3 × 3 convolution, and then, a 1×1 convolution is used to obtain an \( S \times S \times 3(S + 1 + K) \) tensor, where the four coordinates of the bounding box, namely, \( b_x, b_y, b_w, \) and \( b_h \), are stored in “4,” the confidence of the detected object is stored in “1,” “K” is the detected object category, and the calculation formula is presented as follows:

\[
\begin{align*}
    b_x &= \sigma(t_x) + c_x, \\
    b_y &= \sigma(t_y) + c_y, \\
    b_w &= p_w e^{w}, \\
    b_h &= p_h e^{h},
\end{align*}
\]

Confidence = \( P(\text{object}) \times \text{IoU}_{\text{truth}} \), \( P(\text{object}) \in [0, 1] \).

The coordinate system is established with the upper-left corner of the sample as the origin; \( c_x \) and \( c_y \) represent the coordinates from the upper-left corner of the bounding box to the origin; \( p_w \) and \( p_h \) are the width and height of the anchor box, respectively; \( t_x \) and \( t_y \) are the distance from the center point of the bounding box to the origin; \( w \) and \( h \) are the width and height of the bounding box, respectively; \( b_x \) and \( b_y \) are the coordinates of the center point of the prediction box; \( w \) and \( h \) are the width and height of the prediction box, respectively; and \( \sigma(\cdot) \) is the sigmoid activation function. In equation (2), \( P(\text{object}) \) represents the probability that the prediction frame contains the target object, and \( \text{IoU}_{\text{truth}} \) is the intersection and union ratio of the truth frame and the prediction frame. Given that the Yolo head can extract thousands of prediction boxes, greedy nonmaximum suppression (NMS) is often used to eliminate areas with high overlap. The threshold of NMS is set to 0.5, and bounding boxes below 0.5 are discarded. After NMS, the remaining prediction boxes are used to fine-tune our MFF-Yolov4 network.

2.5. Improved Loss Function. The MFF-Yolov4 loss function can be divided into three parts, as shown in Figure 4. Hence, the loss function is calculated as follows:

\[
L = E_{\text{ciou}} + E_{\text{coord}} + E_{\text{cls}},
\]

where \( E_{\text{ciou}} \) is the location error, \( E_{\text{coord}} \) is the confidence error, and \( E_{\text{cls}} \) is the classification error.

The proposed algorithm is designed to detect defects in the tread surface of high-speed train wheels. In the actual operating environment of high-speed trains, however, the wheelsets are exposed to natural light, stains, rust, and other interference in long-term operation under various working conditions, and the background unrelated to the detected target contributes more classification loss \( E_{\text{cls}} \) in the total loss. As such, we draw on the idea of focal loss [20] and add an index adjustment factor \( \alpha \) to the classification loss to improve its ability to distinguish the foreground and background. The improved classification loss is expressed as follows:
\[
E_{cls} = \sum_{i=0}^{S \times S} \sum_{j=0}^{B} I_{ij}^{\text{defect}} \sum_{c} \left[ p_i(c) \log(p_i(c)) + (1 - p_i(c)) \alpha \log(1 - p_i(c)) \right],
\]

where \(S \times S\) represents the number of grid cells divided by the input image; \(B\) represents the number of bounding boxes generated by the cell; and \(I_{ij}^{\text{defect}}\) is the locked activation parameter item. To judge whether the \(j\)-th prior box in the \(i\)-
th grid is responsible for predicting the target, $I_{ij}^{\text{defect}} = 1$ when the wheelset tread damage target falls to the $j$-th bounding box generated by the $i$-th mesh; otherwise, $I_{ij}^{\text{defect}} = 0$. Therefore, only the grid responsible for predicting the target needs to punish the classification error. $p_i(c)$ represents the true probability of category $c$, and $\hat{p}_i(c)$ represents the predicted probability of category $c$.

### 3. Experiments

MFF-Yolov4 was evaluated on our self-built dataset, called WT-DET. The implementation results show that the model we designed is feasible and effective.

#### 3.1. Wheelset Tread Defect Dataset

#### 3.1.1. Dataset Collection

The image samples needed in the experiment were collected from the wheel axle workshop of CRRC Group Co., Ltd., Zhuzhou City, Hunan Province, China. The collection equipment included a wheel delivery track, CCD area camera, magnetic steel sensor, and computer for image storage (Figure 5). When the wheelset was sent into the acquisition area by the wheel delivery track, the camera waited for the hardware control signal, the locomotive wheelset triggered the magnetic steel sensor, and the camera began collecting images. The camera collected one or more wheelset images each time. The computer was used to save the collected images and generate the dataset.

#### 3.1.2. Dataset Production

We collected wheelset tread defect images according to actual needs, including defects and normal samples, with 204 images of defect samples and 74 normal samples. An example of a defect is shown in Figure 6. Note that a single image may have multiple defects. Our dataset thus contains 278 samples, of which 204 are wheelset tread defect samples, 74 are normal samples, and the number of wheelset tread defects is 218, as shown in Figure 7.

#### 3.2. Defect Detection on WT-DET

For the pretrained Yolov4, the MFF module is new. Thus, we trained Yolov4 and MFF to share the same convolutional features. The backbone is essentially a feature extraction network that generates a single multiscale feature $F_i$. The multiscale features generated by MFF can be fed into the Yolo head after the second fusion of PANet. Therefore, the pretrained backbone network was jointly trained with MFF and PANet for end-to-end training. In particular, the training model was divided into two stages. In the first stage, the shared convolutional layer (backbone) was frozen, and on this basis, the nonshared layer (others) was trained. In the second stage, the shared convolutional layer was unfrozen and the network was globally trained.

A defect detection experiment on the WT-DET dataset was conducted. The GPU used in the experiment was RTX 2080Ti, the Python version was 3.6, and it was carried out in Keras 2.1.5 and TensorFlow 1.13.2 environments. The improved Yolov4 model was used in this study. In the formal training, the training and the test sets were divided according to a 7:3 ratio. Among them, 193 were training sets and 84 were test sets. To ensure the reliability of model training, a ten-fold cross-validation method was used, and the 193 wheelset tread defects in the training set in one epoch were randomly divided into ten parts: nine parts as the training set, and one part as the verification set, to avoid model overfitting caused by an unreasonable data division in the case of too few samples. For MFF-Yolov4, the image input was adjusted to a uniform size of $416 \times 416 \times 3$. We used the Adam optimizer and adopted the freezing-based training procedure described above. The total number of epochs was 100. In the first 50 epochs, the backbone network training fusion network part (MFF and PANet) was frozen. At this stage, the batch size was set to eight, and the learning rate was set to 0.001. In the last 50 epochs, the entire network was trained. At this stage, the batch size was set to two, and the learning rate was set to 0.0001.

We used the above-divided dataset to fine-tune MFF-Yolov4. The MFF-Yolov4 model obtained after the improvement of Yolov4 was trained and tested, and an ablation experiment was carried out for each step of the improvement. Details of this are given in the following sections. The training results of the model are given, as shown in Figure 8.

As the epochs of the model increase, the loss value gradually decreases and finally reaches convergence, indicating that the model is effectively fitting the data. The loss is large at the beginning of model training. Thus, the first iteration is ignored when drawing the loss curve. Figure 8 shows that the loss value after 20 epochs before the start of training significantly drops, and when the training reaches a certain stage, the curve tends to stabilize. After 50 epochs, the loss value is maintained at about 1.3. The neural network learning effect is ideal, and the hyperparameter settings are reasonable in the training phase.

A comparative experiment was carried out on the self-built dataset with the current mainstream single-step and two-step target detection models (Table 1). In addition, unlike defect classification, in the case of defect detection, only the F1-score is not a suitable performance metric. Therefore, the accuracy, recall rate, and average precision (AP) were used to evaluate the results of the detection experiment. These indicators are defined as follows:
precision = \frac{TP}{TP + FP},
\text{recall} = \frac{TP}{TP + FN},
\text{AP} = \frac{\text{precision} + \text{recall}}{2},
F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}},

where TP, FP, and FN represent the number of true positives, false positives, and false negatives, respectively. The mean average precision (mAP) was also calculated to evaluate the overall performance. Table 2 shows the experimental results of defect detection. Under the same conditions, all aspects of the Yolov4 model’s data are different from those of other models with our self-built dataset. However, Yolov4 with the embedded MFF module has higher recall, mAP, and F1-score than the other models. The results comprehensively show that the wheelset tread features extracted from the multiscale features have more comprehensive semantic features and location details. Yolov4 itself is a multiscale feature detector, but the multiscale features fused by our method have a more comprehensive feature representation, as discussed in detail below. The detection example of WT-DET is shown in Figure 9.
The above ablation experiment and detection method comparison experiment show that MFF can effectively improve the defect detection mAP of Yolov4. However, it remains to be shown that the mAP improvement of MFF-Yolov4 benefits from the location information contained in the multiscale fusion features extracted by MFF. The positioning accuracy performance of MFF is evaluated in the next section.

3.3. MFF Semantic Analysis. To evaluate the impact of the MFF module on classification, the results of defect classification are first reported to show that our method offers improved accuracy compared to the competition. Table 3 shows the results of the ablation experiments before and after the optimization of Yolov4. According to F1-score in Table 3, we can draw the following conclusions: compared with the original single-scale algorithm, MFF has better classification capabilities. Thus, multiscale fusion features still have strong semantic capabilities. When the improved loss function $\alpha = 1.1$, the performance index of the proposed method is further improved on the basis of MFF-Yolov4. In Table 4, we added a comparative experiment, which replaced the MFF module part in MFF-Yolov4 with two fusion modules, FPN [40] and ASFF [41], and conducted two sets of comparative experiments to demonstrate the performance of the proposed method.

3.4. MFF Positioning Analysis. To verify that MFF improves the positioning accuracy, Yolov4 and MFF-Yolov4 were compared. If multiscale fusion features have more location details, MFF-Yolov4 should have a higher recall rate under the same IoU. Based on this, different IoU thresholds were used to evaluate the recall rate on the self-built dataset, WT-DET. IoU represents the ratio of the intersection and union of the prediction box and the underlying true value. Figure 10 shows the defect recall rate of Yolov4 with different IoU thresholds with and without the MFF module. The higher the IoU threshold, the higher the quality of the prediction box. As expected, Yolov4 with MFF is better than the original Yolov4. When $0 < \text{IoU} < 0.87$, the recall rate of Yolov4 is significantly lower than that of MFF-Yolov4. The original Yolov4 only regressed three high-level features of different scales, and the position information of wheelset tread defects was filtered by the layers in front of the network, which reduced the quality of the features. Our MFF selectively combines the previous layers with less interference information and rich location information, as discussed in Section 4. This gives MFF-Yolov4 stronger positioning capabilities.

4. Discussion

In this part, to prove the effectiveness of MFF, several hidden factors that affect the proposed wheelset tread defect detection module are discussed.

4.1. Determining the Connection Layer of MFF. MFF combines features from different levels into multiscale features, and this effectively improves detection. In Section 3.2, the kinds of layers that should be combined are briefly discussed. In MFF-Yolov4, two layers belong to the bottom-layer features, namely, the last layer of F1 and F2. We discuss how to integrate these two features into the three

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall (%)</th>
<th>Precision (%)</th>
<th>mAP (%)</th>
<th>Fps</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yolov4</td>
<td>66.67</td>
<td>88.00</td>
<td>76.06</td>
<td>39.09</td>
<td>0.76</td>
</tr>
<tr>
<td>MFF-Yolov4</td>
<td>77.27</td>
<td>94.44</td>
<td>86.21</td>
<td>36.38</td>
<td>0.85</td>
</tr>
<tr>
<td>Our method ($\alpha = 1.1$)</td>
<td>78.79</td>
<td>94.55</td>
<td>86.25</td>
<td>37.05</td>
<td>0.86</td>
</tr>
</tbody>
</table>

Figure 9: A visual example of WT-NET’s test results. The red box is the bounding box indicating its location, and the black word is its category score.
single-scale advanced features: the last layer of F3, F4, and F6 and whether other combinations of these five layers will lead to better performance. Therefore, on the WT-DET dataset, MFF-Yolov4 is trained in three different combinations, as shown in Figure 11, where ⊕ denotes a splicing operation.

As shown in Table 5, integrating the second layer into the other three high-level feature layers is significantly better than other methods, thereby showing that multiscale fusion features are effective at improving detection accuracy.

At the same time, the features of the lower level are integrated into F1. MFF should have stronger detection performance, but the quantitative indicators show that the fusion of low-level features in F1 leads to a decline in the model mAP. In Figure 12, we visualize the feature map F1 (Figure 12(a)) and F2(Figure 12(b)) flowing in the network in Table 5 and analyze the reasons that lead to the degradation of the model’s performance. Although the F1 feature has more locational details than the F2 feature, the F1 feature has more noise and interference because our image comes from an actual industrial environment, which causes difficulty in the network learning defect details.

MFF unifies the features of different levels of resolution and channels through $3 \times 3$ convolution and $1 \times 1$ convolution, respectively. To maintain consistency in the number of channels, a simple method uses $1 \times 1$ convolution to increase and decrease the number of channels. This $1 \times 1$ convolution method can be conducted in two ways. A placement strategy is selected by comparing the two ways of connecting the multiscale features: before and after placing the $1 \times 1$ convolution, and before and after multiscale feature fusion. Front mode refers to placing $1 \times 1$ convolution before multiscale feature connections, and rear mode refers to placing $1 \times 1$ convolution after multiscale feature fusion. We adopted postlocation. That is, after each two-scale feature is spliced, the channel is adjusted through a $1 \times 1$ convolution. Although the use of rear mode increases the number of parameters, it merely results in a slight drop in network detection speed (2 FPS), whereas the rear-mode mAP is higher, as shown in Table 6. Multiple $3 \times 3$ convolutional cascaded downsampling forms are used to deal with the fusion between the underlying F2 feature and the three different scale features, namely, F3, F4, and F6. Compared with direct downsampling, step-by-step downsampling can retain more image details.

### Table 4: Results of different fusion modules in Yolov4.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall (%)</th>
<th>Precision (%)</th>
<th>mAP (%)</th>
<th>Fps</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPN-Yolov4</td>
<td>72.73</td>
<td>96.00</td>
<td>80.63</td>
<td>35.85</td>
<td>0.83</td>
</tr>
<tr>
<td>ASFF-Yolov4</td>
<td>43.94</td>
<td>72.50</td>
<td>59.87</td>
<td>36.37</td>
<td>0.55</td>
</tr>
<tr>
<td>Our method (α = 1.1)</td>
<td>78.79</td>
<td>94.55</td>
<td>86.25</td>
<td>37.05</td>
<td>0.86</td>
</tr>
</tbody>
</table>

4.2. Case Analysis of Missed Inspections. Although the improved model is generally better than other methods on the self-built dataset, there are a few cases of missed detections. As shown in Figure 13, we analyze some failure cases and the reasons for the failure of detection. On the one hand, it is difficult to correctly identify damage in the initial stage of wheelset tread abrasion using MFF-Yolov4, mainly because of the lack of sample data and insufficient examples of wheelset tread defects. Thus, the network cannot fully learn the characteristics of wheelset tread damage. As shown in Figure 13(a), wheelset tread damage is easily confused with the complex background environment, and even experienced people cannot accurately distinguish them from the background. On the other hand, in a complex environment, the wheelset tread is
accompanied by oil, sand, and rust during the operation of the wheelset. These interferences are attached to the surface of the wheelset tread in blocks, and the dataset is incorrectly marked as a wheelset tread defect, as shown in Figure 13(b). Although the neural network does not predict the defect as a defect after learning the characteristics of the wheelset tread defect, in the quantitative statistics, the missed detection rate of each model is classified as a classification error, which affects the detection accuracy. Here, we compare the false detection rate of our model with the other models.

The log-average miss rate of the proposed model is 19%, which is lower than that of the six other models, as shown in Figure 14.
Table 6: Parameters and mAP in two different placement modes, and parameters and accuracy in two different placement modes.

<table>
<thead>
<tr>
<th>Parameters (KB)</th>
<th>Front mode</th>
<th>Rear mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,861,248</td>
<td>6,822,144</td>
<td></td>
</tr>
<tr>
<td>76.23</td>
<td></td>
<td>86.21</td>
</tr>
</tbody>
</table>

Figure 13: Missing case. (a) Missed detection. (b) Wrong label.
5. Conclusions

It is difficult for wheelset tread defect detection algorithms based on Yolov4 to consider local semantics and location details while ensuring real-time performance. Therefore, this study proposed an improved Yolov4 high-speed train wheelset tread defect detection algorithm based on multiscale feature fusion. To obtain more wheelset tread defect categories, and semantic and location details, we embedded a multiscale feature fusion module in the Yolov4 model that improves its detection accuracy. In addition, a valuable wheelset tread defect detection dataset, WT-DET, was constructed. The wheelset tread detection performance of the proposed algorithm was compared to that of current single-step and two-step detection algorithms on the self-built dataset. The results showed that the F1-scores of this algorithm for wheelset tread defect classification reached 86%, and the mAP of wheelset tread defect detection reached 86.25%. Moreover, the model reached speeds of 37.05 FPS. Given that regional-level detection can only obtain the approximate area of the wheelset tread, it cannot reflect the contour of the damage. In future research, we will study wheelset tread defect segmentation technology based on deep learning to obtain finer contour boundaries of wheelset tread defects.

Data Availability

The data used to support the findings of this study are not applicable because the data interface cannot provide external access temporarily.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of China (52172403 and 62173137) and the Project of the Hunan Provincial Department of Education (19A137).

References


