TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
1–14

DOI: 10.1177/0361981211019034
journals.sagepub.com/home/trr
SAGE

Automatic Rail Surface Defects Inspection Based on Mask R-CNN

Feng Guo[1], Yu Qian[1], Dimitris Rizos[1], Zhi Suo[2], and Xiaobin Chen[3]

Abstract

Rail surface defects have negative impacts on riding comfort and track safety, and could even lead to accidents. Based on the safety database (2020) of the Federal Railroad Administration (FRA), rail surface defects have been among the main factors causing derailments. During the past decades, there have been many efforts to detect such rail surface defects. However, the applications of earlier methods are limited by the high requirements of specialized equipment and personnel training. To date, rail surface defect inspection is still a very labor-intensive and time-consuming process, which hardly satisfies the field maintenance expectations. Therefore, a cost-effective and user-friendly automatic system that can inspect the rail surface defects with high accuracy is urgently needed. To address this issue, this study proposes a computer vision-based instance segmentation framework for rail surface defect inspection. A rail surface database including 1,040 images (260 source images and 780 augmented images) has been built. The classic instance segmentation model, Mask R-CNN, has been re-trained and fine-tuned for inspecting rail surface defects with the customized dataset. The influences of different backbones and learning rates are investigated and discussed. Experimental results indicate the ResNet101 backbone reaches better inspection capability. With a learning rate of 0.005, the re-trained Mask R-CNN model can achieve the best performance on the bounding box and mask predictions. Sixteen images are used to test the inspection performance of the fine-tuned model. The results are promising and indicate potential field applications in the future.

The health condition of the railroad infrastructure plays an important role in train operation and track safety. Reducing the accident risks resulting from unfavorable infrastructure conditions is of great interest to the public, railroads, and government. According to the FRA report and prior studies, rail defects have been one of the main factors causing serious train accidents, as well as one of the major causes that are responsible for over 70% train derailments on freight mainlines (*1–3*). On the one hand, rail surface defects can directly affect riding comfort and normal train operations because of the additional excitations caused by surface irregularities. On the other hand, it introduces potential risks for rail breakage that could result in catastrophic track failures. However, current inspection approaches often cannot provide very reliable results and the inspection equipment is expensive and difficult to operate (*4*). Thus, a considerable amount of inspection works heavily relies on visual inspections, which are labor-intensive, inefficient, and subjective. As freight shipment increases, current track inspection speed and accuracy cannot satisfy the rapidly growing track inspection demand. Therefore, a cost-effective, highly robust, reliable, and accurate inspection system is urgently needed.

Over the past decades, many efforts have focused on the inspection of rail surface defects and most of these works rely on rail texture classification, identification, and analysis. Mandriota et al. utilized and compared three feature extractors, that is, Gabor, wavelet, and Gabor wavelet filters, to extract rail defect textures (*5*). Their results showed that the Gabor filter outperforms the other two filters on rail defects identification. However, one of the main problems of their study is that these three filters require many feature images, which are typically difficult to obtain. Jie et al. proposed a rail head surface defect detection framework which included image pre-processing, defect locating, defect identifying, post-processing, and a geometrical defect locating method (*6*). Test results suggest that their defect locating method is

[1]Department of Civil and Environmental Engineering, University of South Carolina, Columbia, SC
[2]Department of Civil and Environmental Engineering, Beijing University of Civil Engineering and Architecture, Beijing, China
[3]Department of Civil Engineering, Central South University, Changsha, China

Corresponding Author:
Yu Qian, yuqian@sc.edu

more effective and robust compared with the pixel-level feature extraction method, and that the framework has high precision and could meet real-time inspection requirements of field applications. However, noise in the images has significant negative effects in defect detection, while the proposed way to remove image noise requires extensive validation and needs to be investigated further. Li and Ren proposed an intelligent vision detection system (VDS) for inspecting rail surface defects, and addressed two issues of improving image quality and automatic thresholding by using the local Michelson-like contrast (MLC) and the proportion emphasized maximum entropy (PEME) thresholding algorithm (*7*). The results show that high recall values on Type I and Type II defects are achieved. Although the proposed approach can effectively address the issue of inspecting rail surface defects, the problem of distinguishing the rusted rail area from the area of the real defect needs to be improved. Li and Ren developed a real-time visual inspection system (VIS) for the detection of discrete rail surface defects (*8*). VIS aims to solve four difficult problems related to: (*a*) limited features for recognition; (*b*) illumination inequality; (*c*) variation of rail surface reflection; and (*d*) the requirements of high-speed detection. The proposed local normalization (LN) method and the defect localization based on projection profile (DLBP) algorithm are used to address these four challenges. The results show that VIS has a recall rate of 93.1% and 80.41% on Type I and Type II defects, respectively. They also mentioned the detection speed can be improved, and the LN method needs to be more robust for field applications.

It is worth noting that above approaches are focusing on traditional hand-crafted feature learning to identify and classify rail surface defects, requiring technicians to have rich experience in feature selection and training parameter adjustment, and a large amount of training data. Compared to those methods, deep learning approaches are more flexible and can automatically extract and learn problem-specific features from the original data without subjectively defining any hand-crafted features. Leveraging the fast development of convolutional neural networks (CNN), many recent developments have been successfully applied to civil engineering, such as pavement crack detection, concrete crack detection, structural health monitoring, and so forth (*9–17*). However, few studies used deep learning models to identify and characterize rail surface defects. Faghih-Roohi et al. proposed using deep convolutional neural networks (DCNN) to learn features of rail surface defects (*18*). Three neural network structures, including small, medium, and large DCNN, are trained with two kinds of activation functions, that is, Tanh and ReLU. Their experimental results indicate that DCNN can detect rail surface cracks with high accuracy, while the large DCNN with ReLU performs better than other models. Also, they mentioned that the larger DCNN model required longer training time. Yanan et al. proposed using YOLOv3 to detect rail surface flaws (*19*). YOLOv3 is a one-stage detector with real-time speed for detection. It only detects the target objects with bounding boxes while it does not characterize the defect shape nor quantify the sizes of the detected rail surface defects.

The current challenges of identifying rail surface defects are: 1) The rail surface defects are not of any regular shape; 2) The features are difficult to extract with a hand-crafted feature design; and 3) The inspection work does not produce reliable results. To address the above issues, this study proposes to train the computer vision-based instance segmentation model, Mask R-CNN (*20*), for the identification of rail surface defects. The contributions of this study are: (i) The development of a customized rail surface defect image database which includes 260 source images and 780 augmented images for deep learning model training, evaluation, and testing; (ii) Fine-tuning the Mask R-CNN model with two backbones (ResNet50 and ResNet101) and three learning rates (LRs) (0.02, 0.01, and 0.005) for better inspection and identification performance on rail surface defects; (iii) The inspection performance evaluation of Mask R-CNN model on rail surface defects with different severity levels and different orientations of rails; (iv) Performance comparison between Mask R-CNN and Otsu's method on rail surface defects inspection; and (v) The impact of different light conditions on rail surface defects inspection with Mask R-CNN.

Methodology and Mask R-CNN Model

Methodology Overview

In this study, to accurately inspect the rail surface defects, the instance segmentation model, Mask R-CNN with two different backbones, and three different learning rates, is trained and evaluated. Figure 1 shows the overall methodology of this study.

This study has three tasks: (i) Data preparation; (ii) Model training and evaluation; and (iii) Analysis and comparison of inspection results of the rail and its surface defects. The data preparation tasks include data collection, image augmentation, and data labeling with the generation of annotation files. The popular labeling tool, *labelme*, is used in the labeling process (*21*). For the model training and evaluation, two backbones and three learning rates are selected and used in this study, aiming to find the optimal parameters for the inspection. Finally, concerning the inspection performance, the original images containing rail surface defects are used to test and evaluate the robustness of the fine-tuned model.

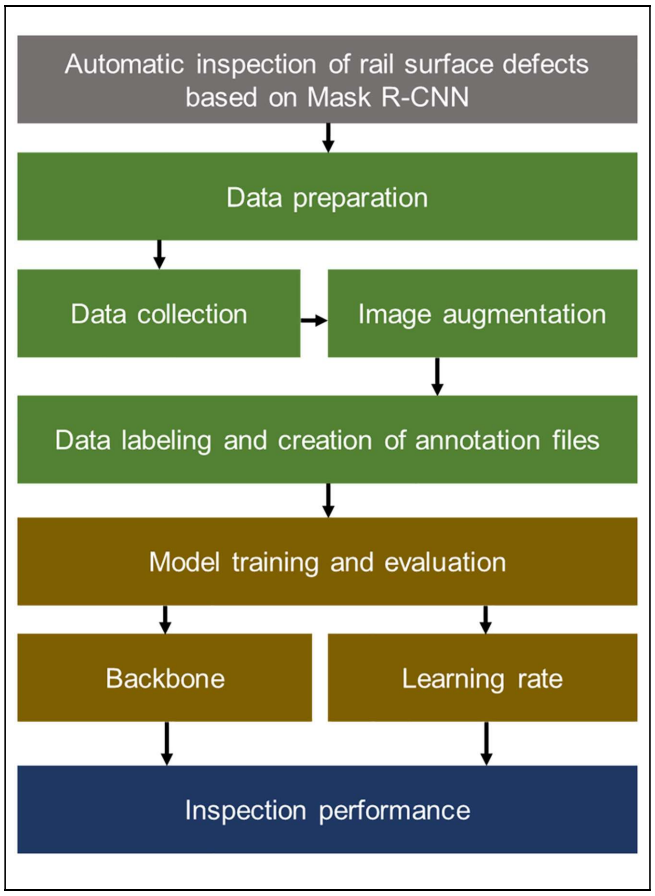

Figure 1. The methodology of this study.

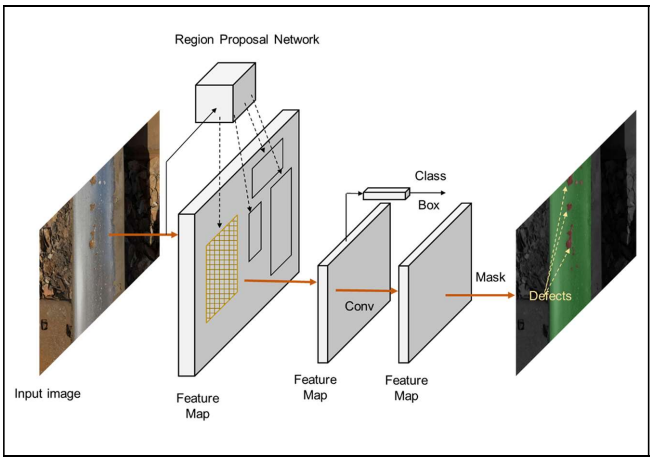

Figure 2. The overview of Mask R-CNN architecture.

Mask R-CNN Architecture

Mask R-CNN is an instance segmentation model which is developed by He et al. (*20*). Compared to the original two-stage detector Faster R-CNN, it adds a parallel branch for recognizing the mask for each instance (*22*). It is worth noting that the semantic segmentation model can distinguish multiple objects of the same class as a single entity, while the target function of Mask R-CNN is more challenging, since it segments each instance based on semantic segmentation. Similar to Faster R-CNN, it adopts the two-stage procedure consisting of the region proposal network (RPN) in the first stage and parallelized class, box and mask detection in the second stage. The loss L in Mask R-CNN is defined by the three loss functions of classification loss L_{cls}, bounding box loss L_{box}, and mask loss L_{mask}. The sum of the three loss functions is the total loss, that is, $L = L_{cls} + L_{box} + L_{mask}$. Mask R-CNN network architecture, shown in Figure 2, includes two parts: 1) The backbone for feature extraction over the input image to generate feature maps; and 2) The prediction head for bounding box recognition (including classification and regression) and mask generation. In practice, many CNN frameworks have been used as the backbone for feature extraction, and Mask R-CNN demonstrates that adding feature pyramid network (FPN) architecture in the backbone can achieve better accuracy (*20*, *23*).

Some of the existing object detectors perform worse on small objects than on large objects because lower layers provide accurate location information and less semantic information (*24*, *25*). As the layers increase, the feature semantic information becomes abundant, but the object location information is not accurate. The proposed backbone structure of Mask R-CNN, depicted in Figure 3, addresses this issue with its network design. Specifically, the Mask R-CNN backbone consists of ResNet (*26*) with 50 or 101 layers as the bottom-up pathway, FPN as the top-down pathway, and lateral connections. Specifically, all of them are depicted in Figure 3. The bottom-up pathway in ResNet facilitates feature extraction. Including the FPN structure into the backbone can help maintain strong semantic features at different scales, since it constructs higher-resolution layers from the top layers. The lateral connections function as bridges connecting the feature maps and the reconstruction layers for better predictions of object locations. Concerning the prediction head, Mask R-CNN extends the box head of the previously developed Faster R-CNN for classification and regression and adds a parallel mask branch for the mask prediction.

Data Preparation

A dataset with rail surface defect images (260 original images and 780 augmented images) has been built. Two object classes are included in this study, that is, the "rail" class and the "defect" class. To keep the labeling clean and neat in the prediction images, "surface defect" is labeled as "defect." The images have been taken by an iPhone8 smartphone from two rail sections in different

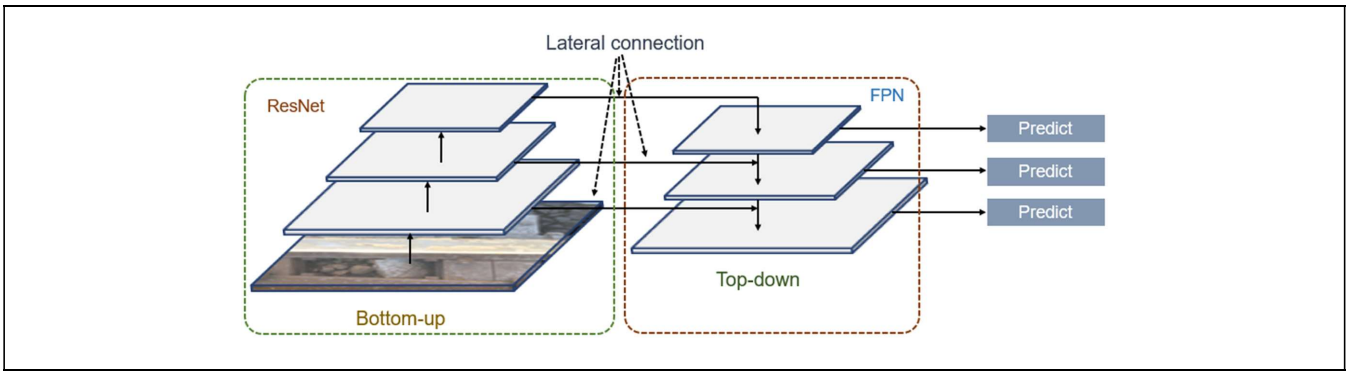

Figure 3. The overview of the backbone structure of Mask R-CNN.
Note: FPN = feature pyramid network.

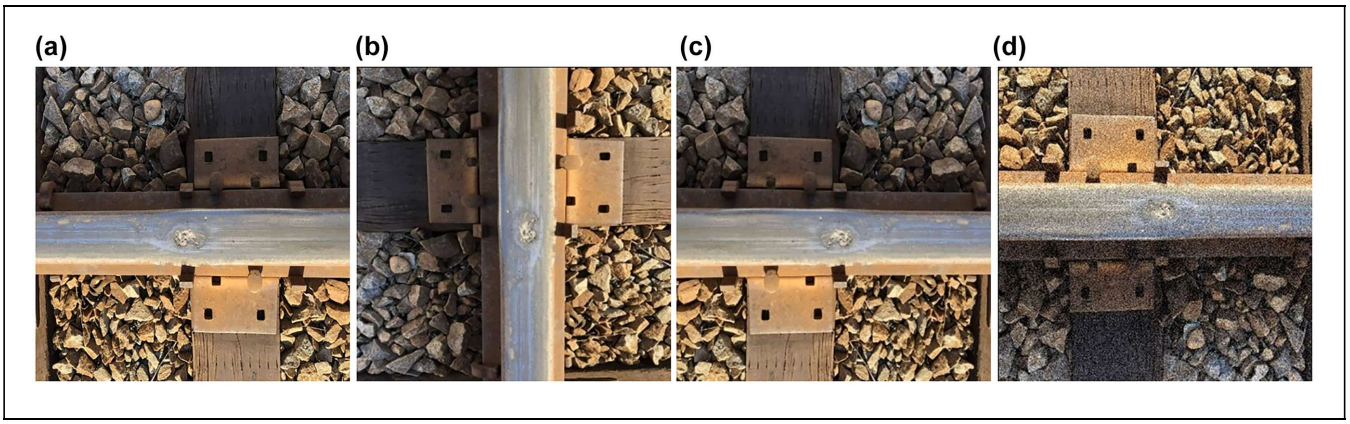

Figure 4. Source image and augmented image: (*a*) Source image, (*b*) 90° rotation, (*c*) mirroring, and (*d*) 180° rotation and Gaussian noise.

time near the campus of the University of South Carolina, U.S. One is a 50 m rail section close to Whaley St and the other one is a 100 m rail section close to Assembly St, Columbia, South Carolina. The height between the camera and the rail surface is 20 ± 5 cm. The original image resolution is 1920 × 1080 pixel^2. Because of the resolution requirements of the training process, the original images are converted to 512 × 512 pixel^2 resolution. Overfitting refers to the trained model, and it only performs well on the training dataset but loses its accuracy with any other dataset, which is typically because of the insufficient useful information of the training dataset. In this study, to improve the detection performance and reduce possible overfitting, image augmentations, including image rotation, mirroring, and Gaussian noise, are conducted on the source images. On the one hand, image augmentation can generate similar images to enrich the dataset. On the other hand, the model would not be overfitted with more data. Examples of the original and augmented images can be seen in Figure 4.

Before the training procedure, the prepared image data needs to be manually labeled first. A popular image labeling tool, *labelme*, is used for data labeling. A total of 1,040 images after the augmentation are labeled as two classes, which are rail and defect. It is worth noting that the output file is in a JSON format which contains the location information of each manually labeled shape. All labeled images and JSON files are fed into the neural network for training and validation. The ratio of the training set and validation set is 3:1, which means there are 780 images for training and 260 images for validation. It needs to be mentioned that the test set is the same as the validation set and the images in each set are randomly selected. To better show the features contained in the JSON files, an example JSON file is converted to the source image, generated mask image, and its corresponding visualization image, and is shown in Figure 5.

Model Training and Evaluation

Model Training

In this study, two different backbones, ResNet50 (50 convolutional layers) and ResNet101 (101 convolutional layers), and three different learning rates (0.005, 0.01, and 0.02) are used to train and evaluate the models.

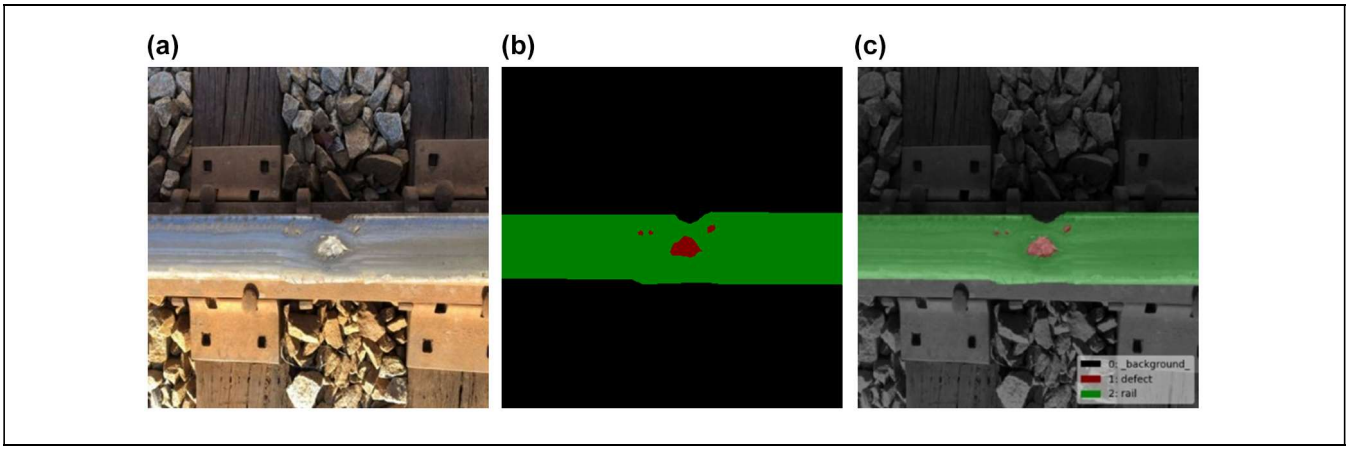

Figure 5. The converted results of a JSON file: (*a*) Source image, (*b*) Mask file, and (*c*) Visualization of mask file.

Table 1. Hyperparameters of each Training

Backbone	Learning rate	Input size	Epoch	Batch size	Momentum	Decay
ResNet50	0.02	512 × 512	12	8	0.9	0.0005
	0.01	512 × 512	12	8	0.9	0.0005
	0.005	512 × 512	12	8	0.9	0.0005
ResNet101	0.02	512 × 512	12	8	0.9	0.0005
	0.01	512 × 512	12	8	0.9	0.0005
	0.005	512 × 512	12	8	0.9	0.0005

MMDetection, an open-source object detection toolbox that is based on the PyTorch library, is used to train Mask R-CNN models in different settings (*27*). All the training, validation, and testing tasks are completed in a workstation equipped with four NVIDIA 2080 Ti GPUs. It is worth noting that only one single GPU is called during all training, validation, and testing processes because the data size is small, and it does not need multiple GPUs for computing. The operating system is Ubuntu 18.04, and the NVIDIA driver version is 440.64. To leverage the powerful computation capability of the graphic card, CUDA (version 10.2) and cuDNN (version 7.6.5) are used in the training. Specifically, CUDA is the NVIDIA's language for expediting computation applications, and cuDNN is a library that provides highly tuned implementations for different layers in deep neural networks. The PyTorch version is 1.5.0 (for Linux) and the Python version is 3.7 in this study.

Table 1 shows the hyperparameters of each training. Note that the input size is the height and weight of an image. One epoch indicates one cycle that the entire dataset passes through the neural network with updated weight once. In this study, each model is trained with 12 epochs. The batch size is the number of images fed into the neural network in each training iteration. Momentum is used to accelerate the speed of converging and it is set to 0.9. Decay is used to prevent weights from growing too large to introduce overfitting. The value of decay is set to 0.0005. The learning rate is an important parameter to control the update rate of the training weights. If the learning rate is set too high, the model would not converge and may cause unfavorable divergent behavior. If the learning rate is set too low, the training progress would be very slow and may lead to marginal updates in the neural network. In this study, three learning rates are tested on the two different backbones, aiming to explore an optimal hyperparameter combination for the customized dataset on the Mask R-CNN framework.

Figure 6 shows a representative loss graph depicting the loss changes over training iterations. As mentioned in the previous section, three losses in the Mask R-CNN model are bounding box loss, mask loss, and regression loss. It is evident that the mask loss has the highest loss values and the class loss has the lowest loss values. It is also interesting to observe that, in the initial stage, the bounding box loss has a low value, but with the training progressing, the loss values grow bigger. At the end of the training, the values of the class loss, bounding box loss, and mask loss are 0.109, 0.205, and 0.204, respectively. Therefore, the total loss can be 0.518, which is the summation of all three losses. Note that, typically, the total loss should be less than one if there are good training and configuration.

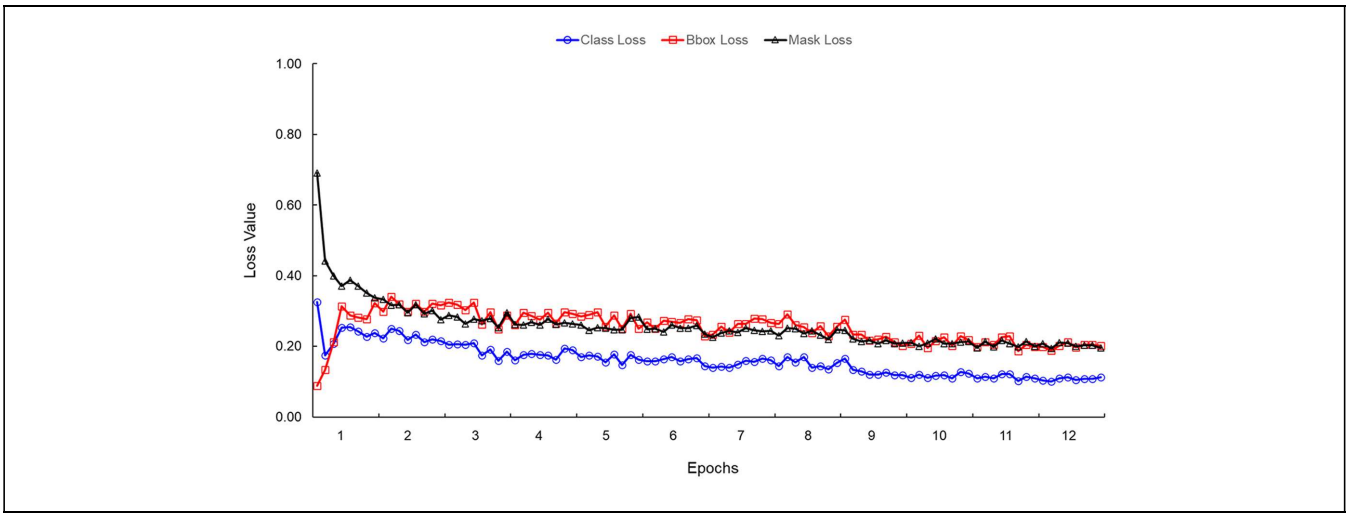

Figure 6. Representative training losses over epochs.

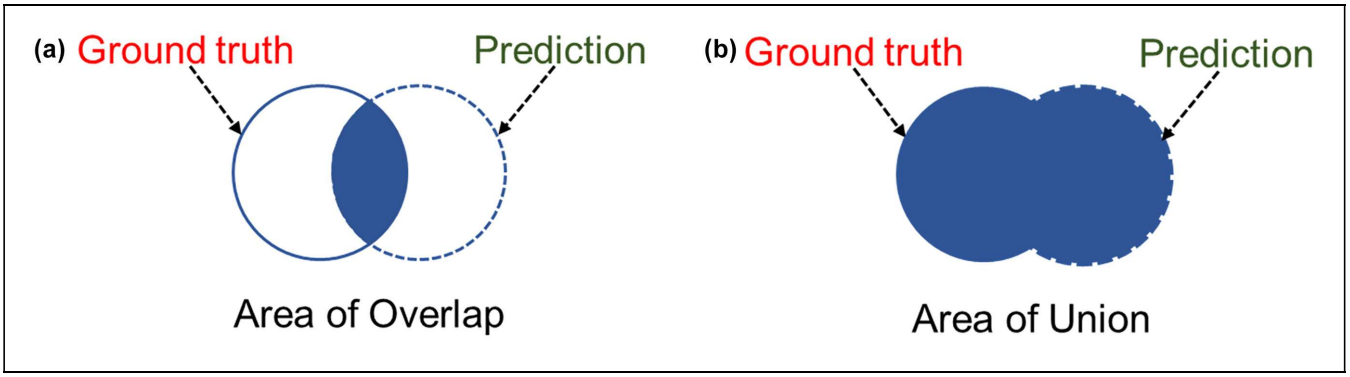

Figure 7. The definition of overlap and union: (*a*) Area of overlap and (*b*) Area of union.

Model Evaluation

To evaluate the prediction performance in different settings, the parameters of mean average precision (mAP), AP_{50}, and AP_{75} are used for comparison. Precision is the percentage of correct positive predictions for overall predictions. Specifically, the mAP is the mean value (MV) of average precision (AP) for each object class. The intersection over the union (IoU) is defined in Equation 1. As shown in Figure 7, IoU measures the overlap between the ground truth and the prediction result for either a bounding box or a mask, and distinguishes the positive case and negative case in the training and testing. Specifically, if the IoU has a ground-truth box above and equal to 0.5 it is a positive case, and negative otherwise. AP_{50} refers to the AP value when the IoU threshold is 50%. Similarly, AP_{75} indicates the AP value when the IoU threshold is 75%. If the prediction is perfect, the IoU would equal 1. In contrast, if the prediction is missed, the IoU is 0. Generally, the IoU above 50% is considered as a good prediction for object detection.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{1}$$

Figure 8 and Table 2 present the mAP, AP_{50}, and AP_{75} results with different backbones and learning rates on the prediction of the bounding box and mask. Specifically, three repeated tests have been performed on each configuration, aiming to accurately evaluate each configuration's performance in a statistical manner. The MV and standard deviation (SD) have been calculated after each parallel test set. It is worth noting that a high indicator value on MV and a low indicator value on SD mean a good prediction performance on either bounding box or mask. Typically, the mAP value is the smallest because it includes each object class. AP_{50} value is higher than AP_{75} value since the lower IoU threshold will introduce more positive cases in the experiments and generates the higher indicator value.

Figure 8, *a* and *b*, depict the bounding box prediction results with the backbone of ResNet101 and ResNet50, respectively. For the bounding box prediction results

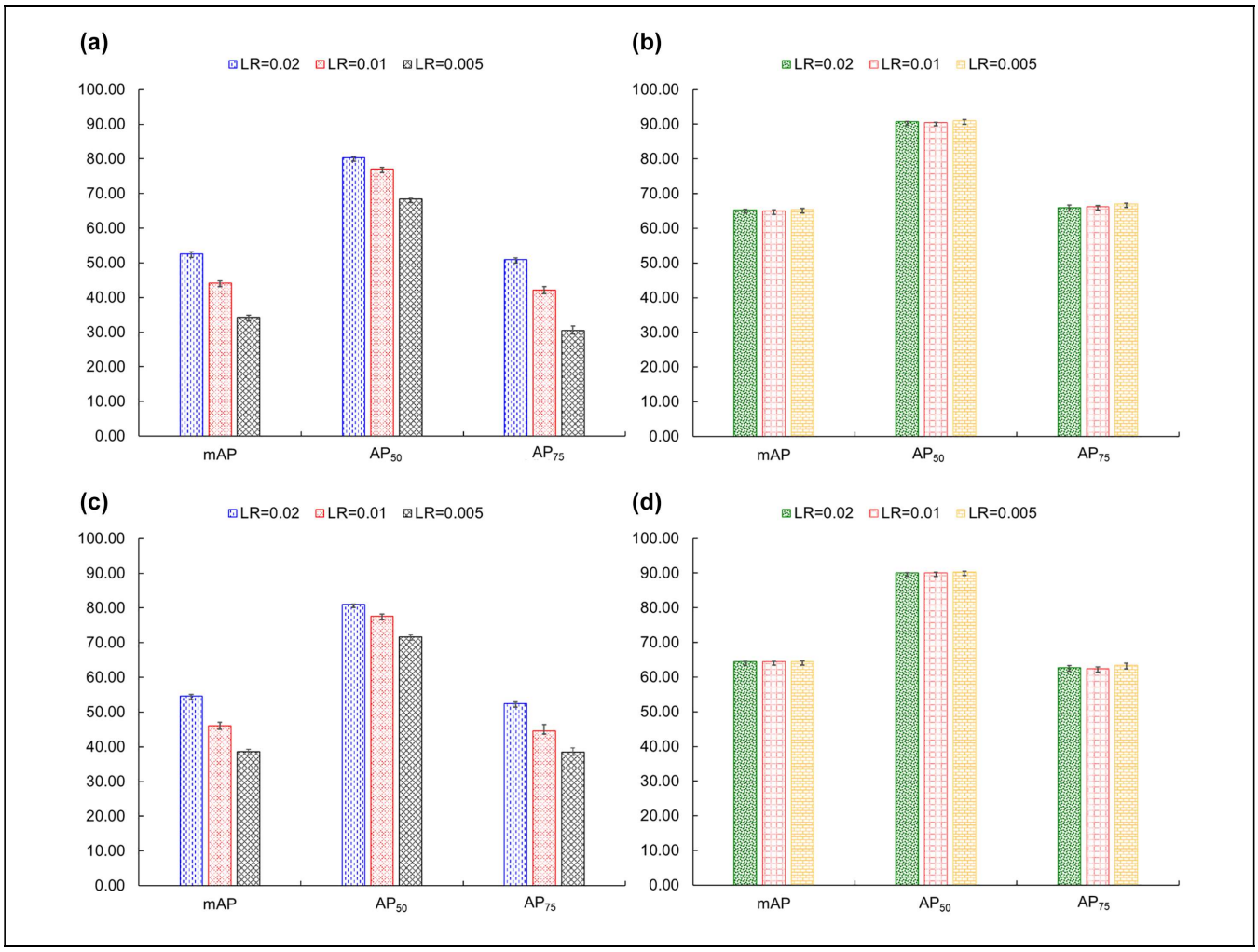

Figure 8. AP results of Mask R-CNN models with different backbones and learning rates. (*a*) Bounding box results of ResNet101, (*b*) Bounding box results of ResNet50, (*c*) Mask results of ResNet101, and (*d*) Mask results of ResNet50.
Note: AP = average precision; LR = learning rate; mAP = mean average precision.

shown in Table 2, ResNet101 (LR = 0.005) has the maximum MV on mAP which is 65.43% (SD = 0.25), 12.86% higher than the maximum MV on mAP with ResNet50. Concerning the prediction results with ResNet50, it is clear that with the increase of learning rate on ResNet50, the Mask R-CNN's bounding box prediction performance improves. For example, with ResNet50, when the learning rate is 0.02, there are the maximum MVs on mAP, AP_{50} and AP_{75}, which are 52.57, 80.33, and 50.90, respectively. The configuration of Mask R-CNN with ResNet50 and LR = 0.005 has the minimum MVs on mAP, AP_{50} and AP_{75}, which are 34.30% (SD = 0.59), 68.37% (SD = 0.26), and 30.47% (SD = 1.32), respectively. In other words, a low learning rate with ResNet50 produces worse performance on bounding box prediction. Interestingly, there is little difference on the bounding box prediction results with different configurations on ResNet101. For the results shown in Table 2, the indicator values are pretty close with different learning rate values on ResNet50. Overall, the Mask R-CNN model with the backbone of ResNet101 performs best on bounding box prediction on the dataset and different learning rate values will not introduce a large performance gap on bounding box predication.

Figure 8, *c* and *d*, show the mask prediction results with the backbones of ResNet50 and ResNet101, respectively. Similar to the bounding box prediction results, the configuration of Mask R-CNN with ResNet101 (learning rate = 0.005) achieves the best MV results on mAP, AP_{50}, and AP_{75}, which are 64.50% (SD = 0.22), 90.37% (SD = 0.12), and 63.37% (SD = 0.66), respectively. As for the configuration with ResNet50, when the learning rate is 0.02, Mask R-CNN achieves the best MV result on mAP, AP_{50}, and AP_{75}, which are 54.53% (SD = 0.56), 81.07% (SD = 0.05), and 52.37% (SD = 0.59),

Table 2. Parallel Test Results of Mask R-CNN Models with Different Backbones and Learning Rates

Backbone	Learning rate (LR)	Bbox			Mask		
		mAP	AP_{50}	AP_{75}	mAP	AP_{50}	AP_{75}
ResNet101	LR = 0.02	65.50	90.60	66.80	64.60	89.90	62.50
		65.10	90.60	66.10	64.30	90.20	63.60
		65.40	90.90	65.10	64.30	90.00	61.80
	Mean value (MV)	65.33	90.70	66.00	64.40	90.03	62.63
	Standard deviation (SD)	0.17	0.14	0.70	0.14	0.12	0.74
	LR = 0.01	65.40	90.50	66.60	64.30	90.30	62.00
		64.90	90.50	66.20	64.60	90.00	63.10
		64.70	90.60	66.00	64.30	89.70	62.10
	MV	65.00	90.53	66.27	64.40	90.00	62.40
	SD	0.29	0.05	0.25	0.14	0.24	0.50
	LR = 0.005	65.70	90.50	67.10	64.70	90.20	64.10
		65.10	91.30	66.90	64.20	90.40	62.50
		65.50	91.10	67.20	64.60	90.50	63.50
	MV	*65.43*	*90.97*	*67.07*	*64.50*	*90.37*	*63.37*
	SD	0.25	0.34	0.12	0.22	0.12	0.66
ResNet50	LR = 0.02	53.20	80.50	51.40	55.30	81.10	53.20
		52.70	80.70	50.30	54.30	81.10	52.00
		51.80	79.80	51.00	54.00	81.00	51.90
	MV	*52.57*	*80.33*	*50.90*	*54.53*	*81.07*	*52.37*
	SD	0.58	0.39	0.45	0.56	0.05	0.59
	LR = 0.01	44.50	77.40	41.90	46.80	78.20	45.10
		44.70	77.50	43.40	46.70	78.00	46.60
		43.20	76.40	40.90	44.80	76.70	42.20
	MV	44.13	77.10	42.07	46.10	77.63	44.63
	SD	0.66	0.50	1.03	0.92	0.66	1.83
	LR = 0.005	34.90	68.50	31.50	38.80	71.60	39.00
		33.50	68.00	28.60	37.80	71.10	37.00
		34.50	68.60	31.30	39.30	72.40	39.60
	MV	34.30	68.37	30.47	38.63	71.70	38.53
	SD	0.59	0.26	1.32	0.62	0.54	1.11

Note: AP = average precision; mAP = mean average precision Except for standard deviation (SD), the parameters' unit is percentage (%).

respectively. It is worth noting that, under the configuration of ResNet50 backbone, with the increase of the learning rate, the indicator values of mask prediction results increase. Concerning the mask prediction performance under the configuration of Mask R-CNN with ResNet101 and different learning rate values, it is easy to find there are similar performances with each configuration. In detail, the gaps between the maximum MVs and minimum MVs on the mAP, AP_{50}, and AP_{75} are 0.1%, 0.37%, and 0.97%, respectively. Similar to the bounding box prediction results, it shows different learning rates would not introduce much difference in mask prediction under the model configuration with ResNet101. In short, it can be concluded that the Mask R-CNN setting of ResNet101 (learning rate = 0.005) has the best performance on the prediction of the bounding box and mask on this customized rail surface defects dataset.

Inspection Performance

To evaluate the prediction performance of Mask R-CNN on rail surface defects, the parameter setting of the ResNet101 backbone with the learning rate of 0.005, which achieves the best performance as discussed in the last section, is used to inspect the rail surface defects. A total of 16 testing images are tested in this section. Figures 9 and 10 show the testing images and their corresponding prediction results with different orientations and different defect severities. Figure 11 presents the comparison results of Mask R-CNN and Otsu's method (*28*). Figure 12 depicts the inspection performance under different light conditions.

Influence of Rail Orientation and Different Defect Severities

In Figure 9*a*, the rail has a vertical orientation. While another horizontal section of rail is shown in Figure 9*b*. The reason to have different orientations is to investigate if the orientation has an impact on the prediction performance. In Figure 10*a*, there are relatively mild defect conditions. In Figure 10*b*, there are relatively more severe defect conditions processed by the enlarged defect parts in the images. The reason behind this is to explore the

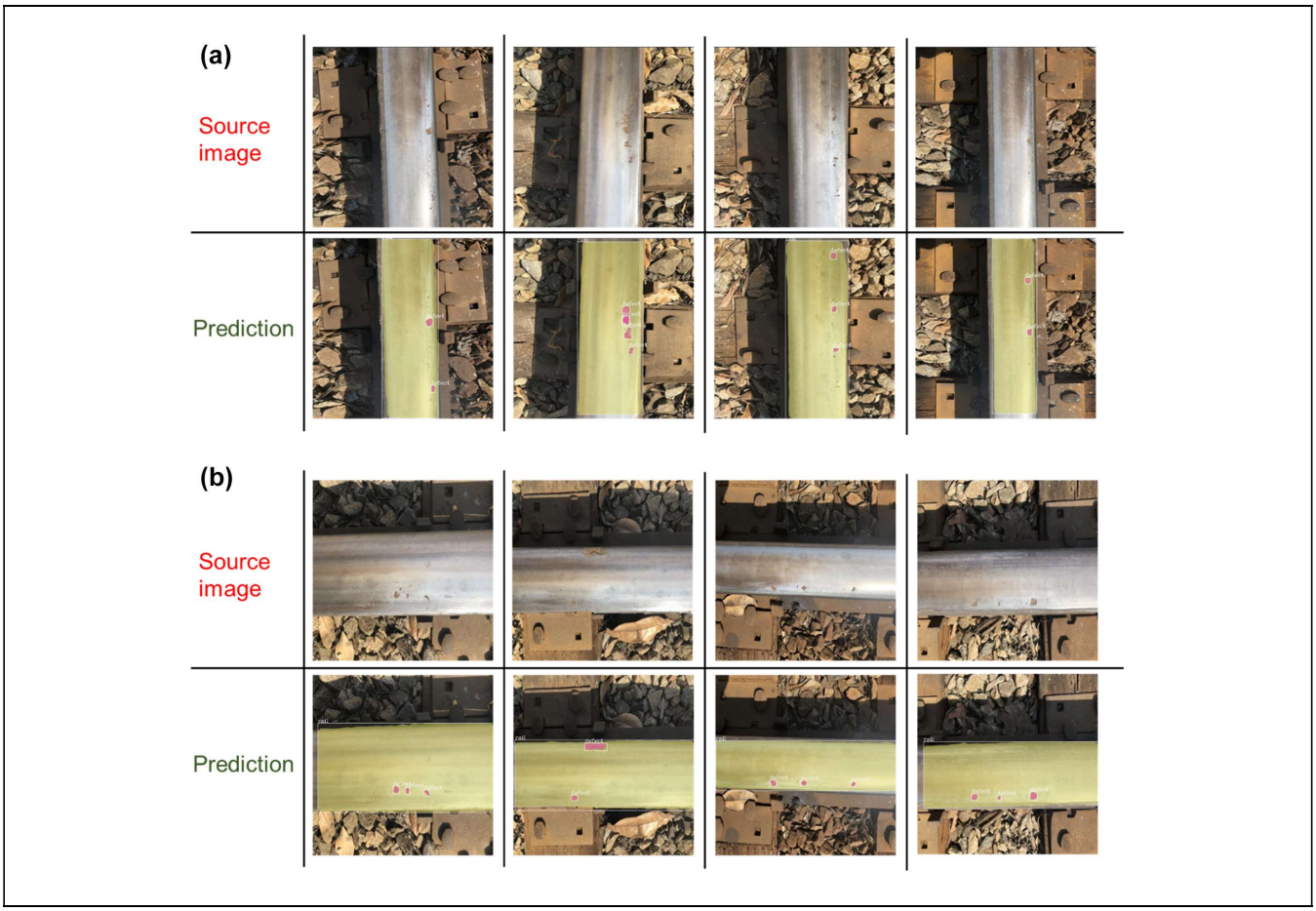

Figure 9. Inspection performance of Mask R-CNN on the rail surface defect with different orientations: (*a*) Images with vertical orientation and (*b*) Images with horizontal orientation.

inspection performance on different defect severities. In Figure 9*a*, some obvious defects can be seen, and the developed model predicts those defects well, especially in the second source image which has a continuously defected area. However, as the evaluation metrics in the previous section indicate, the performance on small defects detection could be further improved as suggested by the last two pictures of Figure 9*a*. However, those very tiny defects may not affect the track performance very much unless they grow larger, in which case they would be detected by the proposed model. In Figure 9*b*, it clearly shows that almost all defects are successfully detected by Mask R-CNN, indicating promising inspection results. In Figure 10*a*, with the relatively mild defect conditions, it can be found that the trained Mask R-CNN model can perform well. There are at least seven defects on each original image shown in Figure 10*a* and the re-trained model has detected the defects' shapes and locations in an accurate manner, indicating its promising performance on the mild defect conditions. As for Figure 10*b*, to further investigate the model's performance on the relatively more severe conditions, which are common in the field, four images with dense and packed defects have been utilized for testing. Each image has at least eight dense defects with small or large shapes. Obviously, it can be found that the re-trained model can detect dense rail surface defects very well on either their shape or locations. Overall, the orientation of the rail has no impact on the detection of the rail surface defect, suggesting a camera could be mounted at an arbitrary orientation for inspection. The re-trained model can detect different defect severities on the rail surface. The results also indicate Mask R-CNN has promising performance for rail surface defect detection and the potential for field applications, but further improvements can be done for very small defects detection, and the implementation of the model with a computing board needs to be developed in the near future.

Comparison of Inspection Performance between Mask R-CNN and Otsu's Method

The next step is to evaluate the inspection performance and segmentation effect between Mask R-CNN and the

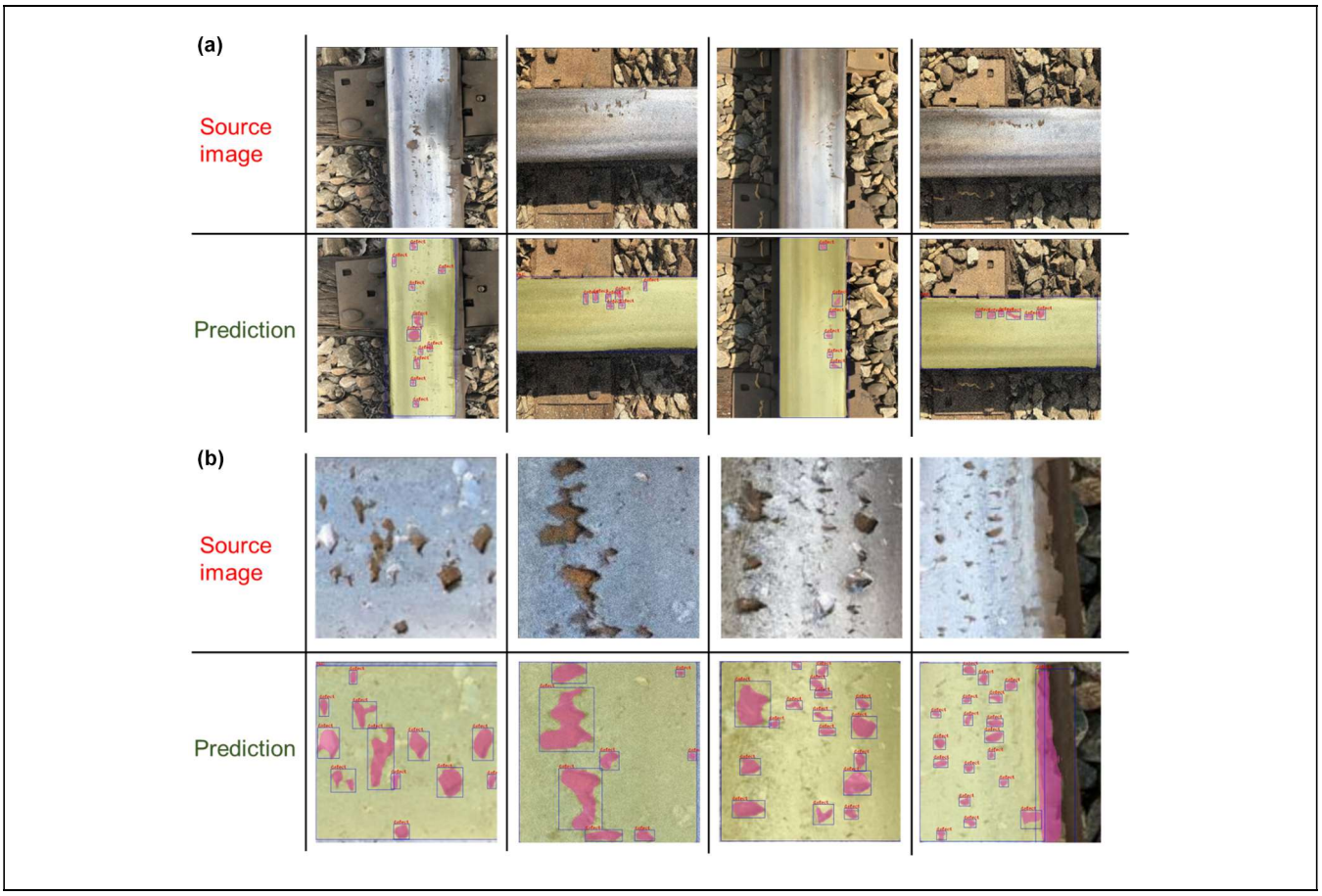

Figure 10. Inspection performance of Mask R-CNN on the rail surface defect with different defect severities: (*a*) Images with relatively mild defect conditions and (*b*) Images with relatively severer defect conditions (enlarged to show details).

traditional image-processing algorithm, Otsu's method, which is commonly used for obtaining segmentation results with simple thresholding to separate foreground and background. It needs to be mentioned that the threshold is the average of mean levels of foreground and background divided by it (*28*). The reasons to choose Otsu's method are twofold: 1) It is a representative method which is simple and widely studied in other engineering domains; and 2) It has similar functionality with Mask R-CNN, and both of them can be used for image segmentation, which is useful for defect size analysis in future study. Figure 11 presents the comparison results between Mask R-CNN and Otsu's method.

In Figure 11, compared with Otsu's method, Mask R-CNN has better performance on both dense defect conditions (columns 1 and 2) and sparse defect conditions (columns 3 and 4). In columns 1 and 2, there are 13 defects and 8 defects identified by Mask RCNN. Meanwhile, although Otsu's method can separate the area of large defects, it still introduces a lot of noise on the testing images especially for the image in column 2. In columns 3 and 4, the sparse defect cases, it can be concluded that the fine-tuned Mask R-CNN model can also inspect and distinguish the rail and the defects well. Concerning the performance of Otsu's method in columns 3 and 4, it can be found that Otsu's method cannot separate the foreground and background (e.g., column 3) since their grayscale is similar and the noise is still significant in the sparse case. Besides, Otsu's method cannot deliver the type of object class in the inspection results. For example, taking a view on the inspection result using Otsu's method in column 4, if there is no label to mark it as surface defect, it is difficult to judge.

Inspection Performance under Different Light Conditions

In real practice, there are many environmental noises which can negatively affect the inspection results. Those factors that need to be considered include, but are not limited to, rust, shadow, mud-spot, over-exposure, and dust. Because of the limitation of the available testing data, many of those factors are not covered. Inspired by previous studies, the inspection results of rail surface

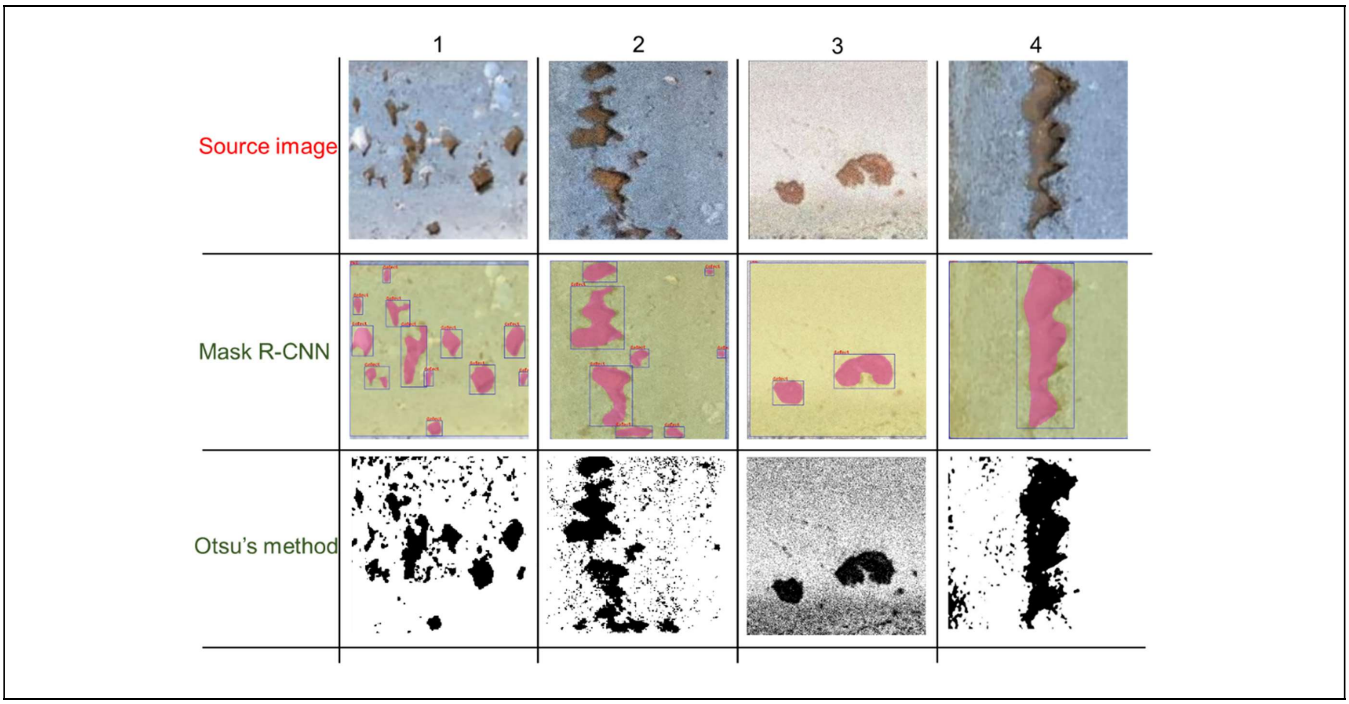

Figure 11. Performance comparison between Mask R-CNN and Otsu's method.

defects under different light conditions are discussed in this section (*29–31*).

Figure 12 shows the inspection performance of rail surface defects under three light conditions which are normal, over-exposure, and weak-light conditions, respectively. It can even be found that the fine-tuned Mask R-CNN can perform well under all three light conditions; the inspection results are still influenced by the light variation. Specifically, in Figure 12*b*, there are mislabeled defect results in column 1 and missed defect results in column 3. In Figure 12*c*, it is easy to find that each mask is a little bit larger than the results in Figure 12, *a* and *b*. It may be because the boundary of defects is not as obvious as in previous conditions and it will introduce errors in the defects size evaluation in future study.

To summarize, all testing results indicate Mask R-CNN has promising performance for rail surface defect detection and the potential for field applications, but further improvements can be done for defects size evaluation and grading, and the implementation of the model with a computing board needs to be developed in the near future.

Conclusions

In this study, an automatic inspection framework using Mask R-CNN to inspect rail surface defects is proposed, aiming to improve inspection accuracy and efficiency, save labor costs, and improve railroad safety. Two different backbones, ResNet50 and ResNet 101, are implemented into Mask R-CNN to test their feature-extraction capability on a customized rail surface defect dataset. Three different learning rates, which are 0.02, 0.01, and 0.005, are selected for testing the optimal learning speed on this customized dataset. A total of 1,040 images, including two object classes, rail and defect, are utilized for training and evaluation. Different parameter configurations are trained and evaluated based on the MMDetection toolbox. The evaluation metrics, including mAP, AP_{50}, and AP_{75}, are used for model evaluation. Parallel tests have been performed and the test results have been analyzed in a statistical manner using the parameters of MV and SD. Test results indicate that, under the configuration of the ResNet101 backbone and the learning rate of 0.005, Mask R-CNN can achieve the highest MVs on mAP with respect to the bounding box and mask predictions, which are 65.43 (SD = 0.25) and 64.50 (SD = 0.22), respectively. A total of 16 images with different defect severities have been tested on the optimal parameter settings of Mask R-CNN. A comparison between the fine-tuned Mask R-CNN model and Otsu's method has been conducted. Experimental results present a better performance than Otsu's method and show a promising performance on the rail surface defect inspection regardless of the rail orientation, different defect severities, and different light conditions.

To the authors' best knowledge, this is the first attempt using the instance segmentation model, Mask R-CNN, to inspect or predict rail surface defects. The results indicate a possible solution by using Mask

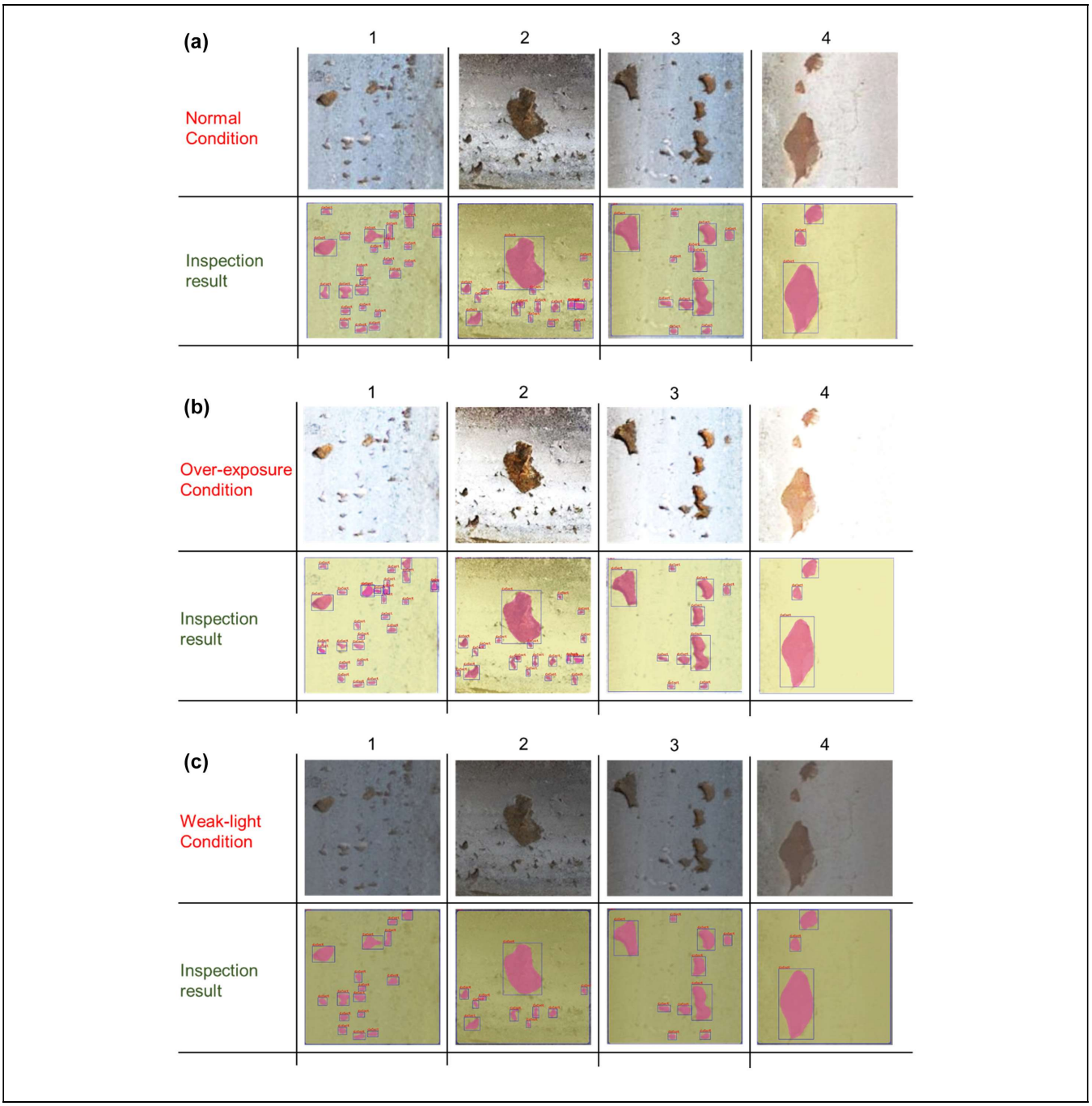

Figure 12. Inspection performance under different light conditions: (*a*) Normal condition, (*b*) Over-exposure condition, and (*c*) Weak-light condition.

R-CNN to inspect rail surface defects in future applications. However, the prediction performance of the developed model can be further improved as more training data are used.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: F. Guo, Y. Qian; data collection: F. Guo; analysis and interpretation of results: F. Guo, Y. Qian, D. Rizos, Z. Suo, X. Chen; draft manuscript preparation: F. Guo, Y. Qian. All authors reviewed the results and approved the final version of the manuscript.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research is partially funded by the new faculty start-up fund provided by the College of Engineering and Computing at the University of South Carolina.

ORCID iDs

Yu Qian https://orcid.org/0000-0001-8543-2774
Dimitris Rizos https://orcid.org/0000-0001-5764-7911
Zhi Suo https://orcid.org/0000-0002-7136-2776

References

1. FRA. Accident Data as Reported by Railroads. https://safetydata.fra.dot.gov/OfficeofSafety/publicsite/on_the_fly_download.aspx.
2. Liu, X., T. Turla, and Z. Zhang. Accident-Cause-Specific Risk Analysis of Rail Transport of Hazardous Materials. *Transportation Research Record: Journal of the Transportation Research Board*, 2018. 2672: 176–187.
3. Liu, X., C. P. Barkan, and M. R. Saat. Analysis of Derailments by Accident Cause: Evaluating Railroad Track Upgrades to Reduce Transportation Risk. *Transportation Research Record: Journal of the Transportation Research Board*, 2011. 2261: 178–185.
4. FRA. Rail Defect Detection. https://railroads.dot.gov/program-areas/track-and-structures/inspection-techniques.
5. Mandriota, C., M. Nitti, N. Ancona, E. Stella, and A. Distante. Filter-Based Feature Selection for Rail Defect Detection. *Machine Vision and Applications*, Vol. 15, No. 4, 2004, pp. 179–185.
6. Jie, L., L. Siwei, L. Qingyong, Z. Hanqing, and R. Shengwei. Real-Time Rail Head Surface Defect Detection: A Geometrical Approach. *Proc., 2009 IEEE International Symposium on Industrial Electronics*, Seoul, South Korea, IEEE, New York, 2009, pp. 769–774.
7. Li, Q., and S. Ren. A Visual Detection System for Rail Surface Defects. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 42, No. 6, 2012, pp. 1531–1542.
8. Li, Q., and S. Ren. A Real-Time Visual Inspection System for Discrete Surface Defects of Rail Heads. *IEEE Transactions on Instrumentation and Measurement*, Vol. 61, No. 8, 2012, pp. 2189–2199.
9. Zhang, A., K. C. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J. Q. Li, and C. Chen. Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 32, No. 10, 2017, pp. 805–819.
10. Yang, F., L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling. Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 21, No. 4, 2019, pp. 1525–1535.
11. Zhang, K., H. Cheng, and B. Zhang. Unified Approach to Pavement Crack and Sealed Crack Detection Using Preclassification Based on Transfer Learning. *Journal of Computing in Civil Engineering*, Vol. 32, No. 2, 2018, p. 04018001.
12. Zhang, X., D. Rajan, and B. Story. Concrete Crack Detection Using Context-Aware Deep Semantic Segmentation Network. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 34, No. 11, 2019, pp. 951–971.
13. Dung, C. V. Autonomous Concrete Crack Detection Using Deep Fully Convolutional Neural Network. *Automation in Construction*, Vol. 99, 2019, pp. 52–58.
14. Kim, B., and S. Cho. Automated Vision-Based Detection of Cracks on Concrete Surfaces Using a Deep Learning Technique. *Sensors*, Vol. 18, No. 10, 2018, p. 3452.
15. Bao, Y., Z. Tang, H. Li, and Y. Zhang. Computer Vision and Deep Learning–Based Data Anomaly Detection Method for Structural Health Monitoring. *Structural Health Monitoring*, Vol. 18, No. 2, 2019, pp. 401–421.
16. Kang, D., and Y. J. Cha. Autonomous UAVs for Structural Health Monitoring Using Deep Learning and an Ultrasonic Beacon System with Geo-Tagging. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 33, No. 10, 2018, pp. 885–902.
17. Azimi, M., and G. Pekcan. Structural Health Monitoring Using Extremely Compressed Data through Deep Learning. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 35, No. 6, 2020, pp. 597–614.
18. Faghih-Roohi, S., S. Hajizadeh, A. Núñez, R. Babuska, and B. De Schutter. Deep Convolutional Neural Networks for Detection of Rail Surface Defects. *Proc., 2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, IEEE, New York, 2016, pp. 2584–2589.
19. Yanan, S., Z. Hui, L. Li, and Z. Hang. Rail Surface Defect Detection Method Based on YOLOV3 Deep Learning Networks. *Proc., 2018 Chinese Automation Congress (CAC)*, Xi'an, China, IEEE, New York, 2018, pp. 1563–1568.
20. He, K., G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. *Proc., IEEE International Conference on Computer Vision*, Venice, Italy, IEEE, New York, 2017, pp. 2961–2969.
21. Wada, K. *Labelme: Image Polygonal Annotation with Python*. GitHub Repository, 2016.
22. Ren, S., K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems*, Vol. 28, 2015, pp. 91–99.
23. Lin, T. Y., P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, IEEE, New York, 2017, pp. 2117–2125.
24. Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single Shot Multibox Detector. In *Proc., Lecture Notes in Computer Science: European Conference on Computer Vision* (Leibe, B., J. Matas, N. Sebe, and M. Welling, eds.), Vol. 9905, Amsterdam, The Netherlands, October 8–16, 2016, Springer, Cham, pp. 21–37.
25. Redmon, J., S. Divvala, R. Girshick, and A. Farhadi. You only Look Once: Unified, Real-Time Object Detection.

Proc., IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, IEEE, New York, 2016, pp. 779–788.
26. He, K., X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, IEEE, New York, 2016, pp. 770–778.
27. Chen, K., J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, and J. Xu. MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv Preprint arXiv:1906.07155*, 2019.
28. Xu, X., S. Xu, L. Jin, and E. Song. Characteristic Analysis of Otsu Threshold and its Applications. *Pattern Recognition Letters*, Vol. 32, No. 7, 2011, pp. 956–961.
29. Guo, F., Y. Qian, Y. Wu, Z. Leng, and H. Yu. Automatic Railroad Track Components Inspection Using Real-Time Instance Segmentation. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 36, No. 3, 2021, pp. 362–377.
30. Guo, F., Y. Qian, and Y. Shi. Real-Time Railroad Track Components Inspection Based on the Improved YOLOv4 Framework. *Automation in Construction*, Vol. 125, 2021, p. 103596.
31. Wu, Y., Y. Qin, and L. Jia. Research on Rail Surface Defect Detection Method Based on UAV Images. *Proc., Prognostics and System Health Management Conference (PHM)*, Chongqing, China, IEEE, New York, 2018, pp. 553–558.

The opinions expressed in this article are solely those of the authors and do not represent the opinions of the funding agencies.