International Journal of Innovative
Computing, Information and Control
Volume **17**, Number **5**, October **2021**

ICIC International ©2021 ISSN 1349-4198
pp. **1763–1774**

A ROAD SURFACE DAMAGE DETECTION METHOD USING YOLOV4 WITH PID OPTIMIZER

Li Guo[1,2,3], Runze Li[1] and Bin Jiang[2,*]

[1]College of Information and Engineering
Hubei Minzu University
No. 39, Xueyuan Road, Enshi 445000, P. R. China
{ gl-hbmzu; Lrz-hbmzu }@hbmzu.edu.cn

[2]College of Automation Engineering
Nanjing University of Aeronautics and Astronautics
No. 29, Jiangjun Road, Nanjing 211106, P. R. China
*Corresponding author: binjiang@nuaa.edu.cn

[3]College of Electronics and Information Engineering
Sichuan University
No. 24, Yihuan Road, Chengdu 610065, P. R. China

Received April 2021; revised July 2021

Abstract. *Road damage is a type of direct and obvious indicator for structural durability and maintainability. Manual visual inspection is the usually used method, but it is generally considered as unsafe, cost-consuming and unreliable. In recent decades, deep learning (DL)-based road surface damage detection has been paid more attention and achieved superior detection performance. In this study, a rapid detection method YOLOv4 is adopted in our road surface damage detection framework, which is improved with a proportional-integral-differential (PID) optimization. The presented framework can intelligently learn inherent features of different road surface damages and avoid overshoot problem usually existing in deep learning. Experimental results show that the mean average precision (MAP) of the proposed method is 47.35% on Dataset 1 and 62.62% on Dataset 2, which is 5.29% and 5.22% higher than original YOLOv4, respectively. Furthermore, it significantly achieves up to 50% acceleration on popular used DL-based YOLOv4 with competitive accuracy. All of experiments prove that the presented method in this study can achieve superior performance in training time and detection accuracy.*
Keywords: Road damage detection, Deep learning, YOLOv4, Proportional-integral-differential (PID), Optimization

1. **Introduction.** With rapid economic and transportation development in China, more and more civil infrastructures such as roads, bridges, and tunnels were constructed extensively. Concrete cracks often appear in the surface of those infrastructures due to temperature variation, heavy load, deformation, etc. Human visual inspection is regularly adopted in road damage detection on the surface of concrete structures. However, it has low efficiency, and is time-consuming and unsafe. With the widespread application of computer technology, automatic road damage detection methods based on machine learning have received much attention in the past several decades [1-8]. Among them, DL-based methods are the hot and mainstream methods for road damage detection application in recent years [9-15]. Among those methods, YOLO is the widely used represented one-stage object detection model [16-20], which is more efficient than two-stage detection model such as spatial pyramid pooling in deep convolutional networks (SPP-NET) [21], faster

DOI: 10.24507/ijicic.17.05.1763

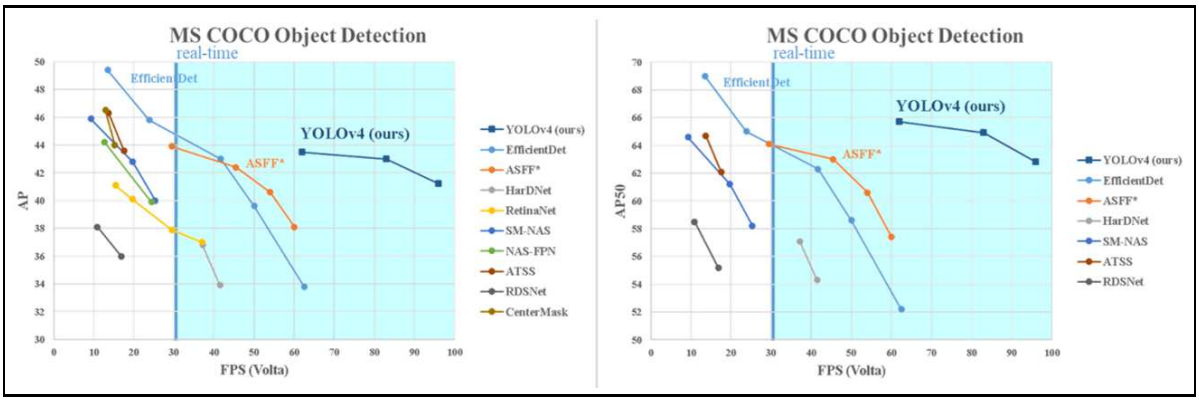
region – convolution neural network (Faster-RCNN) [22], and region proposal network (RPN) [23]. However, there still exist some drawbacks in deep learning, such as overshoot problem, complicated structure and time-consuming training.

In order to achieve accurate detection and improve training efficiency, this paper adopts YOLOv4 [24] as the basic model, and uses PID optimization in YOLOv4 model to replace previous optimization. It exploits change of gradients and fast update parameters of YOLOv4 model, so as to improve training speed while achieving accurate road damage detection. The major contributions of this paper are summarized as follows: 1) We present an automatic road surface damage detection framework based on improved YOLOv4; 2) We add PID optimizer into YOLOv4 model instead of previous SGD or SGD-Momentum optimizer, and this improvement will overcome overshoot problem usually happening in deep learning model; 3) We practically evaluate the proposed method on two public road surface damage datasets, and extensive experimental results demonstrate the superior performance of the presented framework.

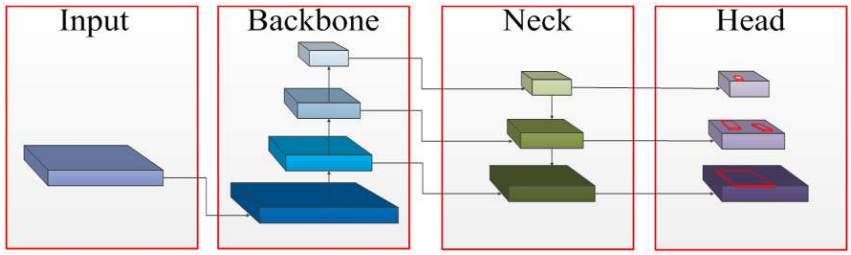
The rest of this paper is organized as follows. Section 2 briefly describes the related work and gives whole framework of the proposed methodology applied on road surface damage detection. Section 3 demonstrates our experimental results and discussions. The conclusion and future scope are given in Section 4.

2. Related Work and the Proposed Methodology.

2.1. Structure of YOLOv4 detector. YOLOv4 is firstly proposed by Bochkovskiy et al. in CVPR 2020. As mentioned in [24], evaluation of AP and FPS of YOLOv4 has increased by 10% and 12% compared with YOLOv3, respectively. In addition, YOLOv4 is twice faster than EfficientDet with acceptable performance shown in Figure 1(a) (it is Figure 8 in [24]). Compared to 107 layers in YOLOv3 and 24 layers in YOLOv3-tiny,



(a) AP/AP50-FPS with Volta GPU provided by Bochkovskiy et al. (Figure 8 in [24])



(b) Architecture of YOLOv4 detector24

FIGURE 1. (color online) Performance and architecture of YOLOv4

YOLOv4 has 162 layers. The main goal of YOLOv4 is to perform a fast object detection that can be applied in the real environment, and it can be optimized in parallel computing and be easily trained and used. As long as the computer is equipped with a conventional GPU such as GTX-2080ti or Titan-XP, it can train YOLOv4 and achieve a good real-time detection. The simple diagram of main architecture for YOLOv4 is illustrated in Figure 1(b). It consists of four main parts described as follows.

Input: Input of the YOLOv4, it includes a whole image, a patch or an image pyramid.

Backbone: Image feature extraction module. Because the low-level feature is similar in images, such as edge, color, and texture, CSPDarknet53 [25] is used in this module.

Neck: Image feature enhancement module. Some low-level feature is obtained in backbone, this part processes and enhances the low-level feature, so as to obtain the important we-wanted feature. Spatial pyramid pooling (SPP) [21] and path aggregation network (PAN) [26] are adopted in this part.

Head: It is connected with convolution to output detection result (boundary box). This part is similar to YOLOv3 [27].

According to this structure, the advantage of YOLOv4 is detailed as: 1) A simple and efficient detection algorithm was researched and designed, which lowers the training threshold so that ordinary people can train a faster and accurate target detector when they just have an ordinary GPU; 2) During the training process, the effect of some tricks (Bag-of-Freebies and Bag-of-Specials) on YOLOv4 was verified; 3) Simplify and optimize some newly proposed algorithms, so that YOLOv4 can be trained on a GPU.

2.2. **PID controller and the correlation of deep learning.** PID controller is the most widely used feedback control method in automatic control. It considers the present, past and future information of prediction error, and continuously calculates an error $e(t)$ to adjust a feedback system [28]. As mentioned in [29], here $e(t)$ is the difference between the desired optimal output and a measured real value of the system. A correction $u(t)$ is constructed by the proportional (P), integral (I), and derivative (D) terms of $e(t)$, and it is formulated as

$$u(t) = K_p e(t) + K_i \int_0^t e(t)d(t) + K_d \frac{d}{dt} e(t) \tag{1}$$

where K_p, K_i and K_d are the coefficients, respectively. The algorithm for the next update of the machine based on the "error" is the controller.

On the other hand, the error between the model output and the expected value is defined as the loss function in deep learning. The loss function affects the weight by performing the "back propagation of error" to obtain the gradient. One can see that gradient used in deep learning optimization has the same spirit as the error $e(t)$ in PID controller [29].

For deep learning optimization, the parameter of stochastic gradient descent (SGD-Moment) from time t to $t+1$ is updated by

$$\begin{cases} V_{t+1} = \alpha V_t - r\partial L_t/\partial \theta_t \\ \theta_{t+1} = \theta_t + V_{t+1} \end{cases} \tag{2}$$

where θ_t represents parameters of deep network, V_t is the history gradient accumulation, $\alpha \in (0, 1)$ is the decay rate, r is the learning rate, and $\partial L_t/\partial \theta_t$ is viewed as the gradient. With some mathematical tricks, Equation (2) can be rewritten as:

$$\theta_{t+1} = \theta_t - r\partial L_t/\partial \theta_t - r\left(\sum_{i=0}^{t-1} \left(\partial L_i/\partial \theta_i \alpha^{t-i}\right)\right) \tag{3}$$

It can be seen that the parameters updating relies on both of the present gradient $(\partial L_t/\partial \theta_t)$ and the integral of past gradient $r\left(\sum_{i=0}^{t-1}(\partial L_i/\partial \theta_i \alpha^{t-i})\right)$. For this formulation, SGD-Momentum optimizer can be viewed as a PI controller [29].

An et al. regarded the "gradient" in deep learning as the "error" in PID controller [29], They explored the connection between the control process and the optimization process of deep learning. They proved SGD-Momentum can be considered as a PI controller [29]. Usually the D item in the PID control algorithm is specifically used to reduce the overshoot problem, because D is the changing trend of the gradient. Based on this discussion, D term can be put into SGD-Momentum optimizer to reduce the overshoot problem existing in deep learning model.

2.3. **The framework of our methodology.** In road surface damage type detection, each road damage can be viewed as a single-object detection. In most of superior DL-based object detection model such as ResNet [30] and DenseNet [31], SGD-Momentum and its variant form are widespread used optimization algorithms in those models, so does it in YOLOv4 detection framework proposed in [24]. SGD-Momentum optimization uses both the past and the present gradient to update network parameters. In this paper, we change the SGD-Momentum optimizer in original YOLOv4 to be PID optimizer. SGD-Momentum optimizer has been proved to be a PI controller [29], and it can be faster converged by adding D term. Inspired by this work, we present a road surface damage detection methodology using improved YOLOv4 with PID optimizer, the pipeline is given in Figure 2, all of the blue rectangles represent some detected road damage. In this framework, a good model is obtained in training stage. After that, it is applied on a new test image to showing the damage location and giving the damage type.

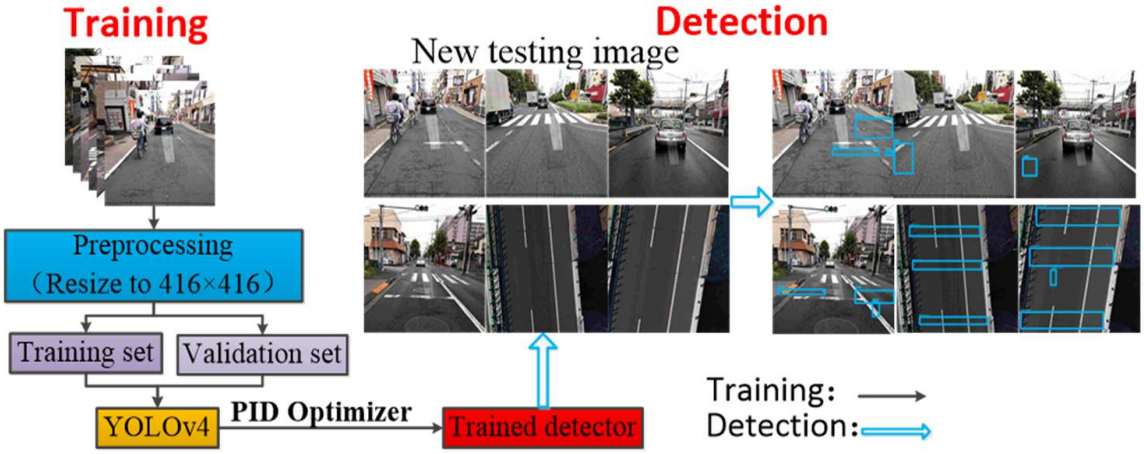


FIGURE 2. (color online) Framework of the proposed method for road surface damage detection

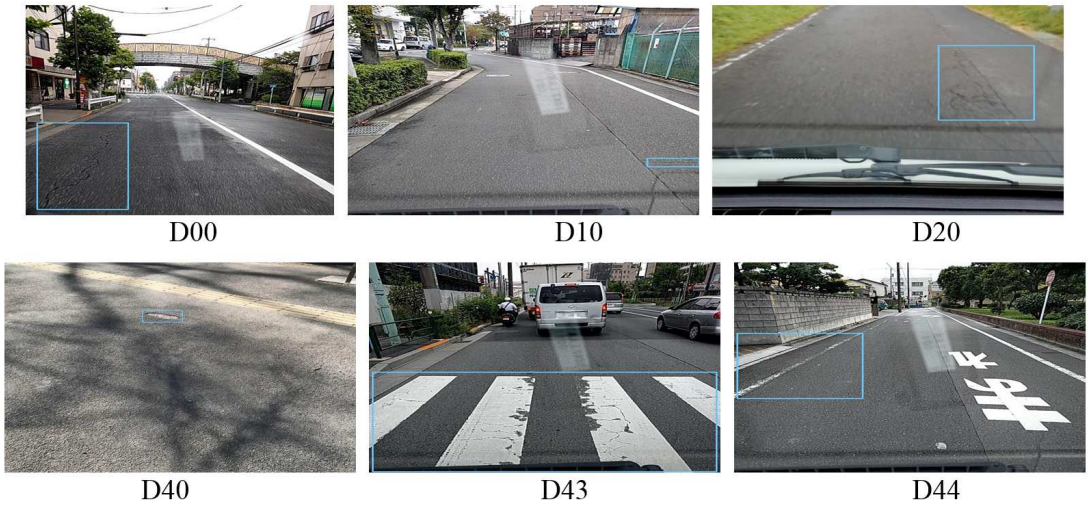
Followed by the work of [29], we introduce future gradient by adding a derivative term into YOLOv4 instead of previous SGD-Momentum optimizer. A simple PID optimizer is built by adding a derivative (change of gradient) term into SGD-Momentum, and it is described as:

$$PID = Momentum + K_d\left(\partial f(x)/\partial x_c - \partial f(x)/\partial x_{c-1}\right) \tag{4}$$

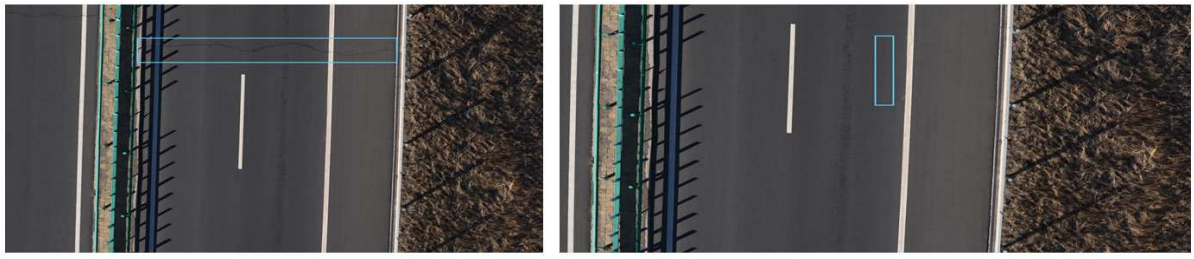
where c is the current iteration number for x, and the coefficient K_d determines the contribution of future errors to the current correction [29]. This PID optimizer exploits more "future" error (change of gradient), and largely reduces the overshoot problem existing in deep learning, so as to achieve fast convergence of the model training.

3. **Experiment and Discussion.** In order to validate the presented detection method using improved YOLOv4 with PID optimizer, it is compared with the related detection framework (such as YOLOv3 and YOLOv4). All experiments are conducted on a PC equipped with an ordinary GPU (NVIDIA GeForce GTX 1060), Intel Core i7-8750H CPU @ 2.20GHZ and 32GB RAM, and operation system is Linux.

3.1. **Datasets description.** Two public road damage datasets are adopted in this paper for performance validation, and some examples are shown in Figures 3(a) and 3(b).



(a) Some examples in Dataset 1



(b) Some examples in Dataset 2 (the left is horizontal crack, and the right is longitudinal crack)

FIGURE 3. Some data examples in two used datasets

Dataset 1 is named Road Damage Dataset 2019, which is collected from various roads in Japan and provided by Maeda et al. in 2019 [32]. Although Maeda et al. stated that the images is 13,135 in the dataset, it actually contains 13,376 road damage images in the file provided by Maeda et al.

For this dataset, since the number of images is not completely consistent to the number of corresponding label files, some label files are absent. In order to ensure the objectivity evaluation, we delete some images without label files, and finally 10,976 images are adopted in our experiments. Among those images, 6,914 images are used for training, 769 images are used for validation and the rest 3,293 images are used for testing. Table 1 lists six categories and their definition of road damage on Dataset 1. Each damage type is represented with a class name such as D00, D10, and D20.

Dataset 2 adopted in our experiments was provided by Wang [33]. The damage types in this dataset are divided into two categories: horizontal crack and longitudinal crack. It is acquired by aerial imagery and consists of 3,525 images. In this dataset, 1,163 images

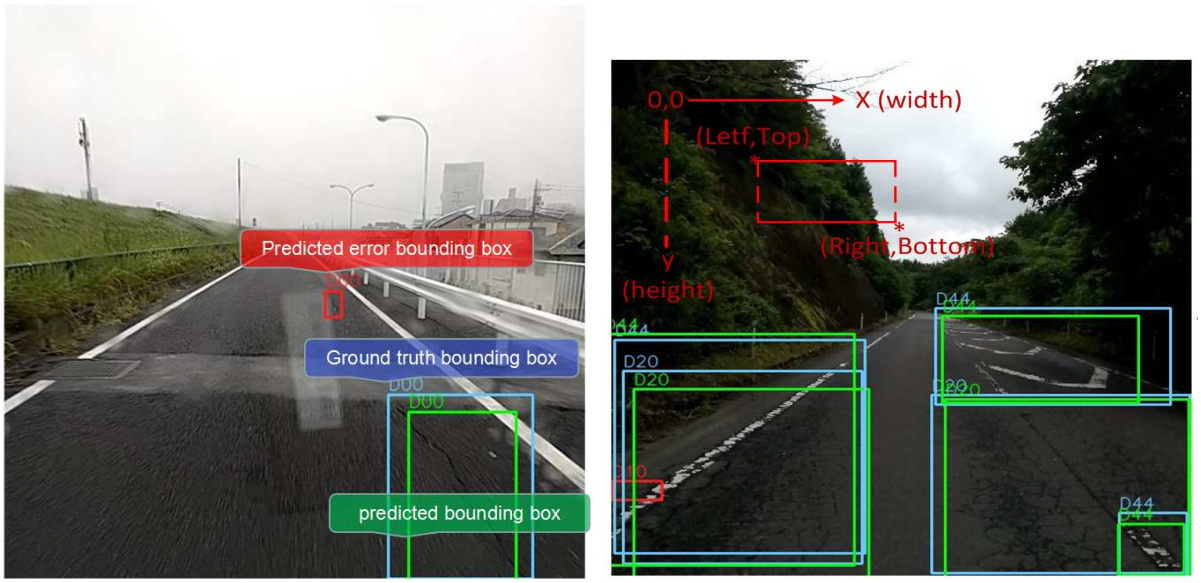
TABLE 1. Road damage types on Dataset 1

Type	Description
D00	Longitudinal linear crack, wheel mark part
D10	Lateral linear crack, equal interval
D20	Alligator crack
D40	Bump, rutting, separation, pothole
D43	Cross walk blur
D44	White line blur

are used for training, 1,163 images used for validating and the rest 1,199 images are used for testing.

3.2. **Description of experiments and discussions.** In our experiment, all of the input images are resized to 416×416 pixels for saving computation. For hyperparameter setting, the learning rate is initiated with 0.001 and learning rate decay is 0.0001 on two datasets. To better have an idea how a good road damage detection is supposed to be looked like, there is a sample illustration of groundtruth box vs predicted box and error bounding box in Figure 4(a). Table 2 shows the coordinates of groundtruth (blue marked) and detected bounding boxes of the testing image (green marked and red marked, respectively. Green means correct detection while red means error detection), which are corresponded to the bounding boxes themselves shown in Figure 4(b). We can see detected bounding boxes look random and there is non-correlation between the groundtruth (blue rectangles) and predicted objects (green rectangles), an error detection is also provided here (as red rectangle indicated). Threshold was set on 0.1 to achieve those detection, and YOLOv4 displays objects detected with a confidence of 0.7 by default.

All of two datasets are conducted on the presented methodology, and experimental results are demonstrated in Figures 5(a) and 5(b). The blue rectangle indicates the



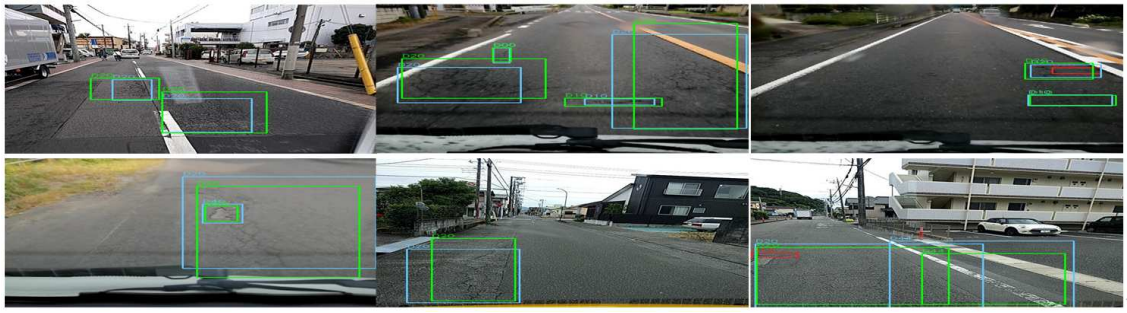
(a) Detection results

(b) Sample of groundtruth box (blue) vs predicted boxes (green)

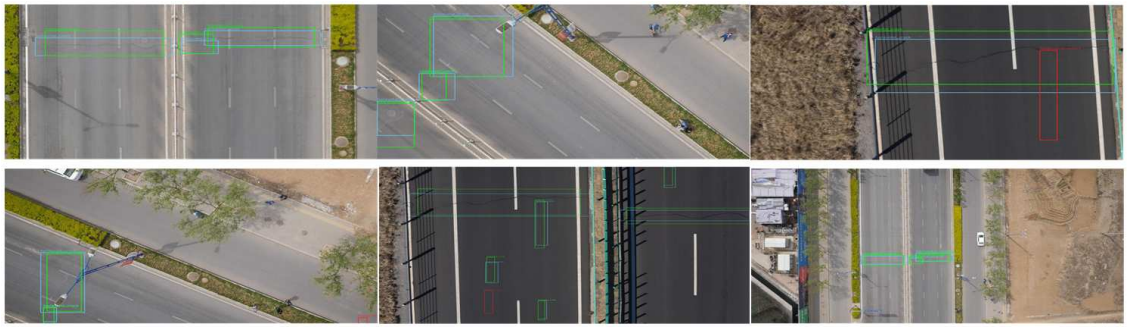
FIGURE 4. (color online) Illustration of groundtruth box vs predicted box and error bounding box

TABLE 2. Groundtruth and detected bounding boxes coordinates in Figure 4(b)

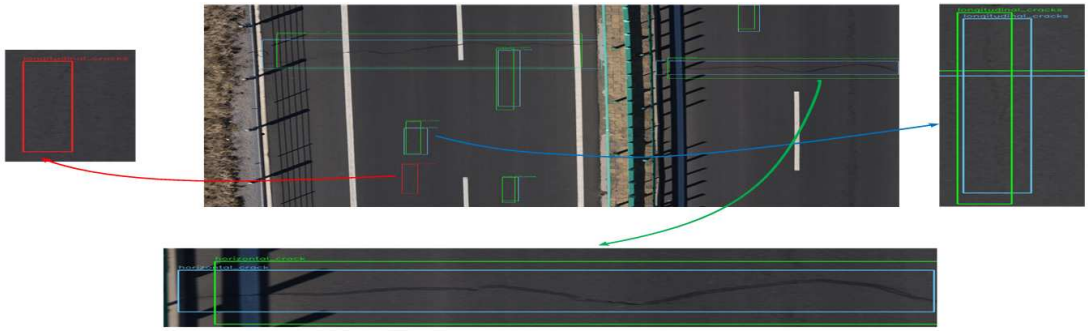
Groundtruth/detection	Class name	Confidence	Left	Top	Right	Bottom
Groundtruth	D20	–	332	390	599	597
Groundtruth	D44	–	4	326	263	574
Groundtruth	D44	–	526	527	596	600
Groundtruth	D44	–	336	289	579	401
Groundtruth	D20	–	13	362	260	586
Correct detection	D20	0.98	346	395	598	603
Correct detection	D20	0.95	24	383	267	605
Correct detection	D44	0.55	−7	319	252	588
Correct detection	D44	0.42	525	540	593	599
Correct detection	D44	0.35	343	298	546	399
Error detection	D10	0.11	1	490	53	512



(a) Some detection results on Dataset 1



(b) Some detection results on Dataset 2



(c) Some local magnified parts of the detection result in (b) for clear displaying

FIGURE 5. (color online) Some detection results on two datasets and local magnified parts for clear displaying

groundtruth box, the green rectangle indicates the prediction box, and the red indicates the error detection. Because some detection results are too small to see clearly, some local small parts are magnified in Figure 5(c) for clear displaying. In addition, we also provide some comparative results in detection accuracy and training time for the proposed detection method and other mainstream DL-based detection methods (such as YOLOv3 [27] and YOLOv4 [24]), and the comparative results are provided in Table 3 and Table 4.

TABLE 3. Comparative results of our method and other YOLO models on Dataset 1

Methods	D00 (AP)	D10 (AP)	D20 (AP)	D40 (AP)	D43 (AP)	D44 (AP)	MAP	Train time (s)
YOLOv3	18.29%	11.07%	38.79%	38.65%	48.03%	48.68%	33.91%	–
YOLOv4	29.19%	23.22%	47.93%	40.80%	56.61%	54.61%	42.06%	56282.36
Our method	**35.65%**	**27.53%**	**53.16%**	**45.52%**	**65.23%**	**57.00%**	**47.35%**	**28144.66**

TABLE 4. Comparative results of our method and other YOLO models on Dataset 2

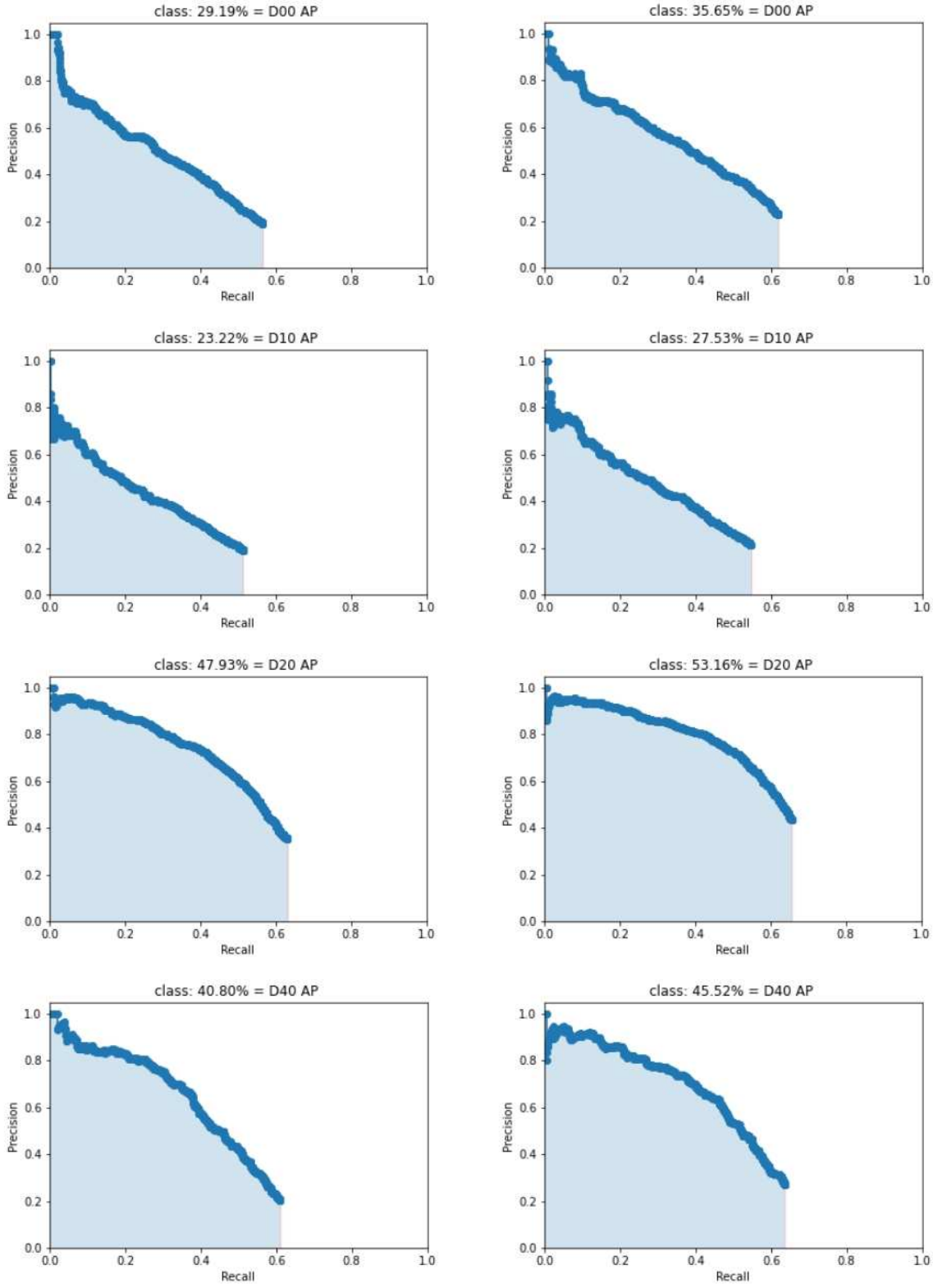
Methods	Horizontal crack (AP)	Longitudinal cracks (AP)	MAP	Train time (s)
YOLOv3	73.50%	7.21%	40.35%	–
YOLOv4	84.77%	30.03%	57.40%	17370.63
Our method	**86.89%**	**38.36%**	**62.62%**	**8878.23**

3.3. **Comparative validation of PID optimizer and basic YOLOv4.** In this part, detection performance of the proposed method and original YOLOv4 is compared by the objective metric of average precision (AP), respectively. Generally speaking, AP is the average value of the precision value on the Precision-Recall curve of detection. For the Precision-Recall curve, we use integral to calculate the AP metric as follows.

$$AP = \int_0^1 p(r)dr \tag{5}$$

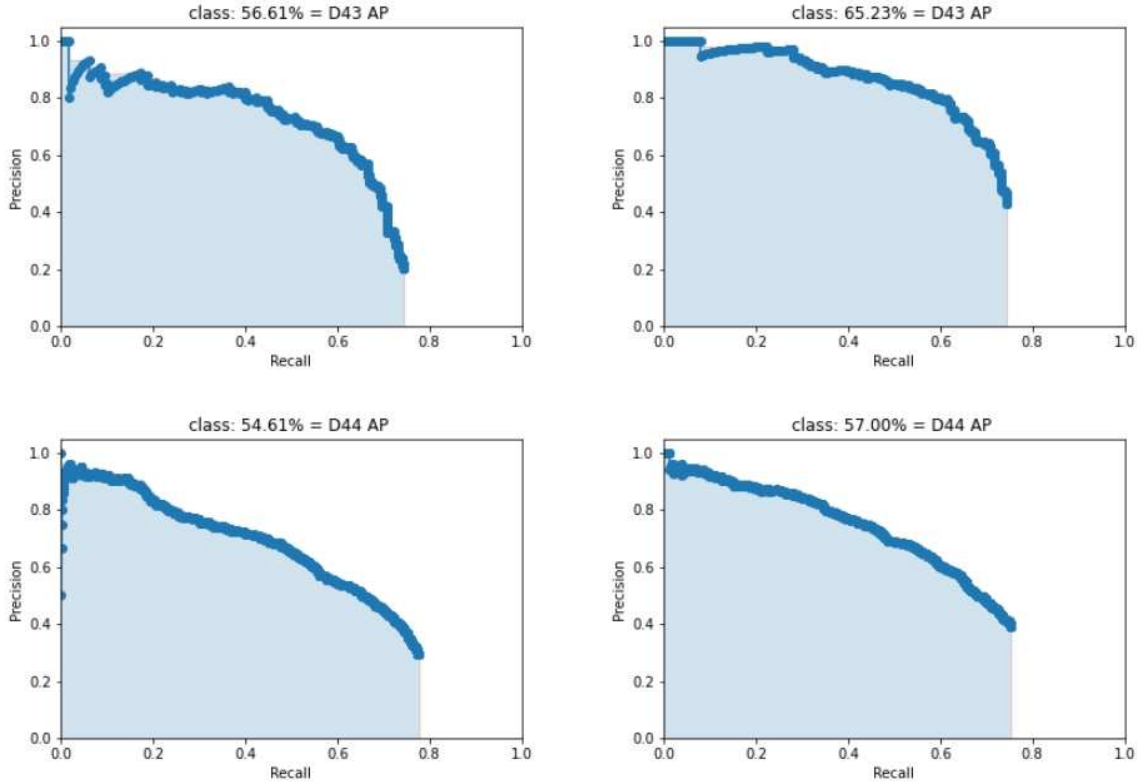
where $p(r)$ represents the precision value of the rth point in the Precision-Recall curve. The comparative results of AP value are shown in Figure 6. It clearly shows that the AP value of the presented method is higher than original YOLOv4 on two datasets. The improved value is ranged from 2.39% to 8.62% on Dataset 1, and 2.12% to 8.33% on Dataset 2, respectively.

3.4. **Comparative results with some YOLO methods.** The presented method is further compared with other versions of YOLO (YOLOv3 [27] and YOLOv4 [24]), and these YOLO models adopt SGD optimizer. The learning rates of both YOLOv3 and YOLOv4 are 0.001. This value of parameter will reduce the learning rate decay on Dataset 1 and Dataset 2 to 0.003, and 0.001. The comparative experiment results are provided in Table 3 and Table 4, which indicate that the performance of our developed method is better than popular traditional versions of YOLO in MAP (improve 13.4% and 22.27% compared with YOLOv3, and 5.25% and 5.22% compared with YOLOv4, on two datasets, respectively). Efficiency of the presented method achieves 50% and 38% faster than original YOLOv4 on two datasets, respectively.

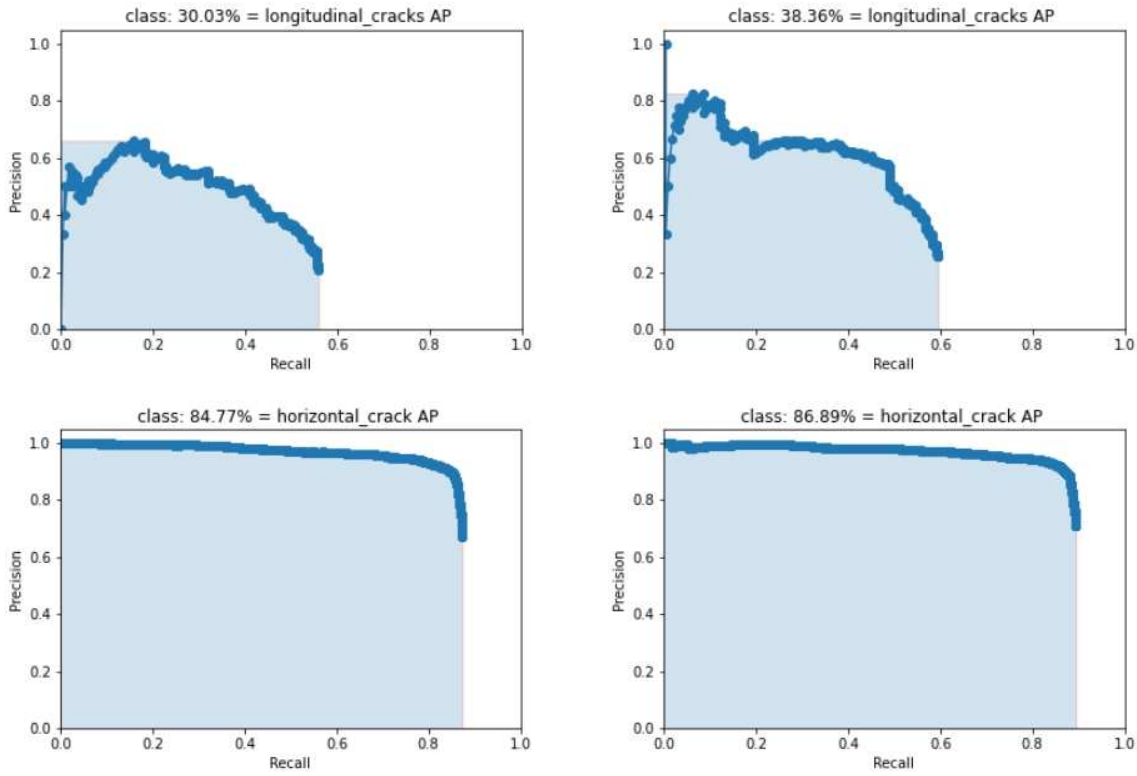

class: 29.19% = D00 AP
class: 35.65% = D00 AP
class: 23.22% = D10 AP
class: 27.53% = D10 AP
class: 47.93% = D20 AP
class: 53.16% = D20 AP
class: 40.80% = D40 AP
class: 45.52% = D40 AP
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

(*Continued*)

(*Continued*)



(a) AP comparison of YOLOv4 and the proposed method on Dataset 1 (six categories)



(b) AP comparison of YOLOv4 and the proposed method on Dataset 2 (two categories)

FIGURE 6. AP value comparison of YOLOv4 (in the left) and the proposed method (in the right) on two datasets

4. **Conclusions.** An automatic road surface damage detection method using improved YOLOv4 with PID optimizer was presented in this study. It used PID optimization instead of SGD or SGD-Momentum usually used in DL-based object detection model, which significantly achieves up to twice faster than popular used DL-based YOLOv4 with competitive improved accuracy. In prospective work, we will investigate how to adopt PID optimizer to other network architectures, and exploit several possible extensions for improving detection accuracy in the natural scenes.

Acknowledgment. This work was supported by the National Natural Science Foundation of China (Grant. 61663008), Science & Technology Foundation of Enshi (Grant. D201 80013) and the PhD Foundation of Hubei Minzu University (Grant. MD2019B006).

REFERENCES

[1] A. Cord and S. Chambon, Automatic road defect detection by textural pattern recognition based on AdaBoost, *Computer-Aided Civil and Infrastructure Engineering*, vol.27, no.4, pp.244-259, 2012.

[2] P. Chun, K. Hashimoto, N. Kataoka, N. Kuramoto and M. Ohga, Asphalt pavement crack detection using image processing and Naive Bayes based machine learning approach, *Journal of Japan Society of Civil Engineers, Ser. E1 (Pavement Engineering)*, vol.70, no.3, 2015.

[3] Y. Shi, L. Cui, Z. Qi, F. Meng and Z. Chen, Automatic road crack detection using random structured forests, *IEEE Trans. Intelligent Transportation Systems*, vol.17, no.12, pp.3434-3445, 2016.

[4] Z. Zhang, O. Hamata, T. Akiduki, T. Mashimo, T. Saito and K. Hayashi, Cracks in bridge floor detected by 2-dimensional complex discrete wavelet packet transform, *International Journal of Innovative Computing, Information and Control*, vol.16, no.6, pp.2007-2019, 2020.

[5] A. Zhang, K. C. P. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu and J. Li, 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, pp.805-819, 2017.

[6] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary and A. Agrawala, Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection, *Construction and Building Materials*, vol.157, pp.322-330, 2017.

[7] L. Guo, R. Li, B. Jiang and X. Shen, Automatic crack distress classification from concrete surface images using a novel deep-width network architecture, *Neurocomputing*, vol.397, pp.383-392, 2020.

[8] B.-C. Yeo, W.-S. Lim and H.-S. Lim, Lane detection in the absence of lane markings for roadway surveillance with thermal vision, *International Journal of Innovative Computing, Information and Control*, vol.12, no.3, pp.677-688, 2016.

[9] Q. Mei, M. Gül and M. R. Azim, Densely connected deep neural network considering connectivity of pixels for automatic crack detection, *Automation in Construction*, vol.110, 2020.

[10] Y. J. Cha, W. Choi and O. Büyüköztürk, Deep learning-based crack damage detection using convolutional neural networks, *Computer-Aided Civil and Infrastructure Engineering*, vol.32, no.5, pp.361-378, 2017.

[11] L. Zhang, F. Yang, Y. D. Zhang and Y. J. Zhu, Road crack detection using deep convolutional neural network, *2016 IEEE International Conference on Image Processing (ICIP)*, pp.3708-3712, 2016.

[12] H. Nhat-Duc, Q. L. Nguyen and V. D. Tran, Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network, *Automation in Construction*, vol.94, pp.203-213, 2018.

[13] F. C. Chen and M. R. Jahanshahi, NB-CNN: Deep learning-based crack detection using convolutional neural network and Naïve Bayes data fusion, *IEEE Trans. Industrial Electronics*, vol.65, no.5, pp.4392-4400, 2017.

[14] M. Kouzehgar, Y. K. Tamilselvam, M. V. Heredia and M. R. Elara, Self-reconfigurable façade-cleaning robot equipped with deep-learning-based crack detection based on convolutional neural networks, *Automation in Construction*, vol.108, 2019.

[15] Y. Shin, M. Kim, K.-W. Pak and D. Kim, Practical methods of image data preprocessing for enhancing the performance of deep learning based road crack detection, *ICIC Express Letters, Part B: Applications*, vol.11, no.4, pp.373-379, 2020.

[16] Z. W. Yu, Y. G. Shen and C. K. Shen, A real-time detection approach for bridge cracks based on YOLOv4-FPM, *Automation in Construction*, vol.122, 2021.

[17] J. A. Sarmiento, Pavement distress detection and segmentation using YOLOv4 and DeepLabv3 on pavements in the Philippines, *arXiv.org*, arXiv: 2103.06467, 2021.
[18] Z. Chen and J. Juang, Attention-based YOLOv4 algorithm in non-destructive radio-graphic testing for civic aviation maintenance, *Preprint.org*, DOI: 10.20944/preprints202104.0653.v1, 2021.
[19] Y. F. Cai, T. Y. Luan, H. B. Gao et al., YOLOv4-5D: Effective and efficient object detector for autonomous driving, *IEEE Trans. Instrumentation and Measurement*, vol.70, 2021.
[20] L. Tan, X. Lv, X. Lian and G. Wang, YOLOv4_Drone: UAV image target detection based on an improved YOLOv4 algorithm, *Computers & Electrical Engineering*, vol.93, pp.107261-107269, 2021.
[21] K. He, X. Zhang, S. Ren and J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.37, no.9, pp.1904-1916, 2015.
[22] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.39, no.6, pp.1137-1149, 2016.
[23] J. Bowles, An assessment of RPN prioritization in a failure modes effects and criticality analysis, *Journal of the IEST*, vol.47, no.1, pp.51-56, 2004.
[24] A. Bochkovskiy, C. Y. Wang and H. Y. M. Liao, YOLOv4: Optimal speed and accuracy of object detection, *arXiv.org*, arXiv:2004.10934, 2020.
[25] C. Y. Wang, H. Y. M. Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh and I. H. Yeh, CSPNet: A new backbone that can enhance learning capability of CNN, *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp.390-391, 2016.
[26] S. Liu, L. Qi, H. Qin, J. Shi and J. Jia, Path aggregation network for instance segmentation, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.8759-8768, 2018.
[27] J. Redmon and A. Farhadi, YOLOv3: An incremental improvement, *arXiv.org*, arXiv: 1804.02767, 2018.
[28] K. H. Ang, G. Chong and Y. Li, PID control system analysis, design, and technology, *IEEE Trans. Control Systems Technology*, vol.13, no.4, pp.559-576, 2005.
[29] W. An, H. Wang, Q. Sun, J. Xu, Q. Dai and L. Zhang, A PID controller approach for stochastic optimization of deep networks, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.8522-8531, 2018.
[30] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.770-778, 2016.
[31] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, Densely connected convolutional networks, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.4700-4708, 2017.
[32] H. Maeda, T. Kashiyama, Y. Sekimoto, T. Seto and H. Omata, Generative adversarial network for road damage detection, *Computer-Aided Civil and Infrastructure Engineering*, vol.36, no.1, pp.47-60, 2021.
[33] B. Wang, *AerialCrackDataset: Towards Object Detection with Dataset*, Key Laboratory of Optoelectronic Imaging Technology and System, Ministry of Education, School of Optoelectronics, Beijing Institute of Technology, 2017.