symmetry

MDPI

Article

A Real-Time Detection Method for Concrete Surface Cracks Based on Improved YOLOv4

Gang Yao [1,2], Yujia Sun [1,2], Mingpu Wong [1,2] and Xiaoning Lv [3,*]

[1] School of Civil Engineering, Chongqing University, Chongqing 400044, China; yaogang@cqu.edu.cn (G.Y.); sunyujia@cqu.edu.cn (Y.S.); wongmingpu@cqu.edu.cn (M.W.)
[2] Key Laboratory of New Technology for Construction of Cities in Mountain Area, Ministry of Education, Chongqing 400044, China
[3] Science & Technology on Integrated Information System Laboratory, Institute of Software Chinese Academy of Sciences, Beijing 100190, China
* Correspondence: xiaoning@iscas.ac.cn

Abstract: AbstractMany structures in civil engineering are symmetrical. Crack detection is a critical task in the monitoring and inspection of civil engineering structures. This study implements a lightweight neural network based on the YOLOv4 algorithm to detect concrete surface cracks. In the extraction of backbone and the design of neck and head, the symmetry concept is adopted. The model modules are improved to reduce the depth and complexity of the overall network structure. Meanwhile, the separable convolution is used to realize spatial convolution, and the SPP and PANet modules are improved to reduce the model parameters. The convolutional layer and batch normalization layer are merged to improve the model inference speed. In addition, using the focal loss function for reference, the loss function of object detection network is improved to balance the proportion of the cracks and the background samples. To comprehensively evaluate the performance of the improved method, 10,000 images (256 × 256 pixels in size) of cracks on concrete surfaces are collected to build the database. The improved YOLOv4 model achieves an mAP of 94.09% with 8.04 M and 0.64 GMacs. The results show that the improved model is satisfactory in mAP, and the model size and calculation amount are greatly reduced. This performs better in terms of real-time detection on concrete surface cracks.

Keywords: structural health monitoring; deep learning; crack detection; improved YOLOv4; concrete surface

check for updates

Citation: Yao, G.; Sun, Y.; Wong, M.; Lv, X. A Real-Time Detection Method for Concrete Surface Cracks Based on Improved YOLOv4. *Symmetry* **2021**, *13*, 1716. https://doi.org/10.3390/sym13091716

Academic Editors: Yang Yang, Ying Lei, Xiaolin Meng and Jun Li

Received: 1 September 2021
Accepted: 14 September 2021
Published: 16 September 2021

1. Introduction

As one of the most common materials in civil engineering, concrete is widely used in dams, buildings, tunnels, bridges, and other infrastructure. Owing to the influence of internal and external factors (such as temperature change, foundation deformation, shrinkage, etc.), cracks often appear on concrete surfaces. As a common defect in civil engineering, cracks not only affect the health of structures, but also lead to other problems [1]. Therefore, crack detection is an essential part of structural health monitoring.

The traditional detection method relies on human vision, which has high cost but low detection efficiency, and the detection results depend on subjective human judgment. To solve these problems, researchers have proposed many methods of automatically detecting concrete surface defects [2,3]. However, these methods usually have heavy workloads and low precision, and thus adequately cannot meet the demand. With the continuous development of computer vision technology, image processing technology (IPT) has been widely used to detect defects on various structural surfaces [4–6]. Yeum et al. [7] used IPT combined with sliding window technology to detect cracks, which clearly shows the potential of IPT. Due to the existence of background noise, the differences between the database, the limited number of features, and the diversity of application scenarios, traditional image processing methods may produce undesirable results. The adaptability of the model is also limited [8]. IPTs are usually used to help inspectors detect defects, but the final results are still obtained relying on manual judgment [9].

At present, with the improvement of computing capabilities and image acquisition equipment, many machine learning algorithms (such as deep learning) are being used to recognize objects with acceptable results [10–14]. Deep learning can automatically extract high-level semantic information from original images, which provides a new method for the automatic crack detection on concrete surfaces. The convolutional neural network (CNN), which has a strong capacity to learn depth features directly from training data, greatly improves the efficiency and accuracy of detecting concrete surface defects. In addition, it has been emphasized in image classification and object detection [15]. Researchers have developed many methods based on deep learning to detect pavement cracks [16–18], concrete cracks [19–22], concrete bug holes [23–25], and other defects [26–31]. CNN-based crack detection methods generally have problems, such as excessive training parameters and complex network structures. To overcome these problems, the object detection algorithms are further explored.

There are two categories in object detection models: one-stage models and two-stage models. The SSD [32] and YOLO series represent one-stage models. They regard object detection as a regression problem. The Faster-RCNN [15], SPP-NET [12], etc. represent two-stage models. In the two-stage model training process, the network for object regions detection is trained after the region proposal network (RPN) is trained. Consequently, the two-stage model has high precision but slow speed. For the purpose of completing the whole detection process without using RPN and realizing the end-to-end object detection, the initial anchors are used in the one-stage model to predict the category and locate the object area. Correspondingly, the one-stage model has fast speed but low precision. The accuracy and reasoning speed of object detection algorithms are the critical problems in object detection. Balancing the efficiency and accuracy of the detection is a crucial technical problem. YOLOv4 has good processing speed and performance, which just meets the requirement. However, it is difficult to use in embedded devices, which cannot meet the needs of accurate real-time detection.

In this study, we choose YOLOv4 [33] (a one-stage model) as the basic model, and improvements are made to achieve accurate real-time crack detection on concrete surfaces. We make the following major contributions. (1) The model framework is improved. The SwishBlock bottleneck module is established and replaces the ResBlock_body as the main framework of the YOLOv4 model. Meanwhile, the number of original YOLOv4 model modules is changed, and the number of layers in the entire network is reduced, in order to compress the YOLOv4 model. (2) The SPP structure and PANet module are improved. With the aim of maintaining channel separation, the common convolution is replaced by separable convolution, which reduces model parameters and further compresses the network model. (3) Model reasoning speed is improved. In the SwishBlock bottleneck module, the parameters in the batch normalization layer and the convolutional layer are merged to improve the forward reasoning speed of the model. (4) The loss function is improved. Using the focal loss function of the RetinaNet network for reference, a modulation coefficient α is added to the cross-entropy loss function of the object detection to balance the proportion of foreground and background data samples. (5) The trained model is deployed to the Jetson Xavier NX embedded platform for testing to verify that it meets the requirements for accurate real-time detection of concrete surface cracks, which can provide support for the development of mobile monitoring systems.

2. Lightweight Model for Concrete Crack Detection

2.1. The Principles of YOLOv4

The backbone network CSPDarknet53 of YOLOv4 is the core of the algorithm and is used to extract the target features. CSPNet can maintain accuracy and reduce computing bottlenecks and memory costs while being simplified. Drawing from the experience of

CSPNet, YOLOv4 adds CSP to each large residual block of Darknet53. It divides the feature mapping of the base layer into two parts, and then merges them through a cross-stage hierarchical structure to reduce the amount of calculations while ensuring accuracy. The base layer of CSPDarknet53 uses the Mish function as the activation function, and the feature extraction layer network uses the Leaky_relu function. Experiments have shown that the above activation function setting makes the object detection more accurate. Unlike the YOLOv3 algorithm, which uses FPN for upsampling, YOLOv4 draws on the idea of information circulation in the PANet network. The semantic information of the layer features is propagated to the low-level network by upsampling and is then fused with the high-resolution information of the underlying features to improve the small target detection effect. Next, the information transmission path from the bottom to the top is increased, and the feature pyramid is enhanced through downsampling. Finally, the feature maps of different layers are fused to make predictions. The specific network framework is shown in Figure 1. The ResBlock_body is the residual block of CSPDarknet53, which can extract the target features of the image and reduce the computational bottleneck and memory cost, as shown in Figure 2.

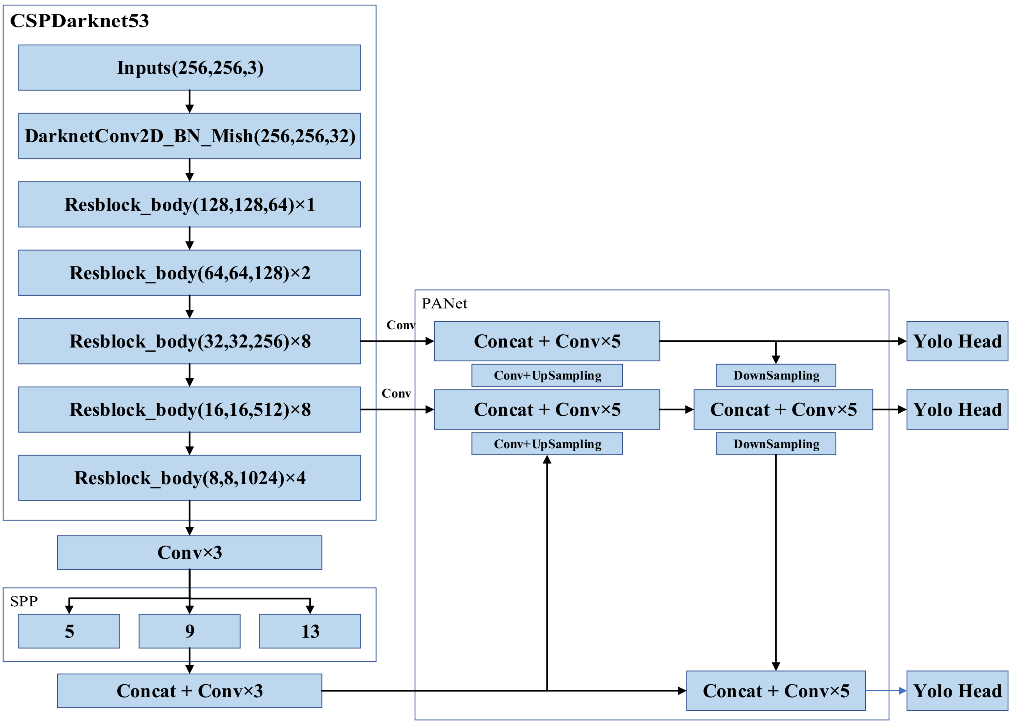


Figure 1. YOLOv4 network architecture [33].

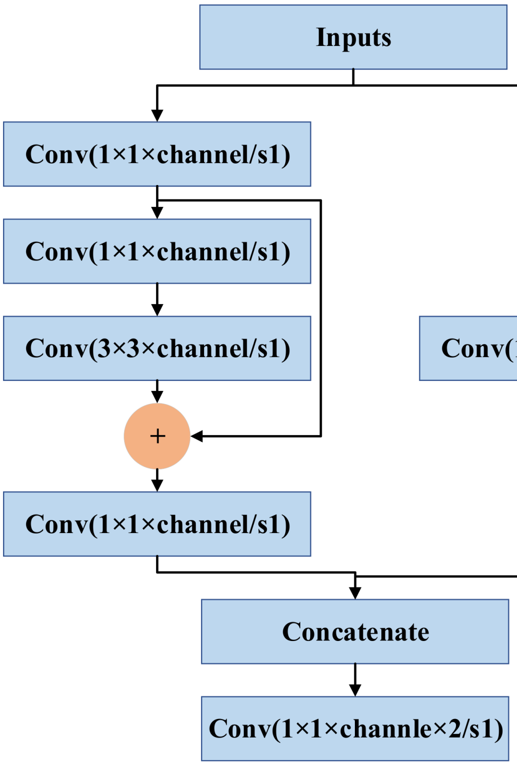


Figure 2. ResBlock module structure.

2.2. Improvements to Model Framework

Three criteria need to be met during the construction of the neural network model [34]. First, the residual neural network is used to increase the depth of the network, and the feature extraction is achieved by the deeper neural network. Second, the number of feature layers extracted from each layer is changed to achieve more feature extraction layers, get more features, and increase the width. Third, by increasing the resolution of the input picture, the features of network learning and expression can be enriched, which is conducive to improving the accuracy. The above criteria are followed in the YOLOv4 model compression. The SwishBlock bottleneck module is established based on the depthwise separable convolution and the construction concept of reverse residual structure. The characteristics of the network are expanded from three aspects at the same time, and the ResBlock_body is replaced as the overall design concept of the main YOLOv4 framework. Meanwhile, the SENet Channel Attention idea is used for reference into the network structure, and different weights are assigned to the extracted feature maps to extract more critical feature information without increasing the model calculation and storage costs.

The actual architecture of the SwishBlock bottleneck module is an inverted residual structure. In a residual structure, there are fewer feature map channels in the middle and more feature map channels on both sides, while in reverse residual structure, there are more feature map channels in the middle and fewer feature map channels on both sides. When processing deep separable convolution, the method of first raising and then reducing the dimension of the feature map greatly improves the feature extraction ability of the network, and also speeds up the calculation. The Shortcut connection used in the module ensures that the gradient will not be affected during the propagation of the deep convolutional neural network, and the reverse residual structure has been shown to improve the memory utilization efficiency. Before the 3 × 3 separable convolution structure, the 1 × 1 convolution is used to increase the dimension to improve the feature extraction of the image. At the same time, a channel attention mechanism structure is added after the 3 × 3 network structure. First, the global pooling is performed, and the neural network is used to train the weight value of each channel to extract more important feature information. Then, a large residual edge is added after a 1 × 1 convolution dimensionality reduction to avoid the gradient disappearance of the network. The SwishBlock module structure is shown in Figure 3.

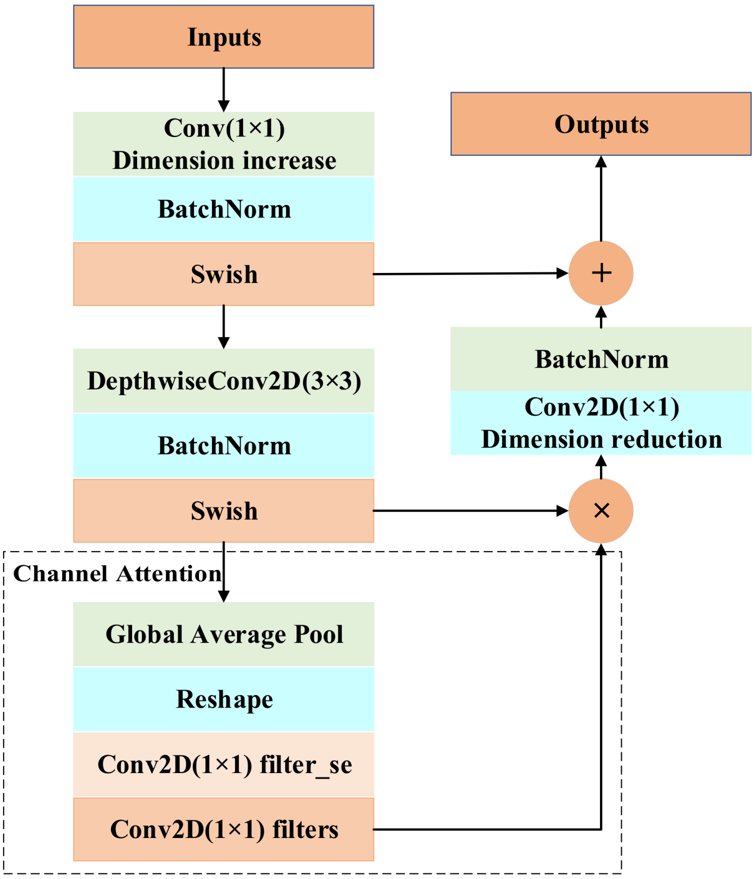


Figure 3. SwishBlock module structure.

In this study, using the advantages of the basic SPP and PANet architecture in the YOLOv4 module, the SwishBlock bottleneck is used to replace the ResBlock_body structure in the main YOLOv4 framework. The neural network model is constructed using the SwishBlock bottleneck structure. Meanwhile, the number of original YOLOv4 network model modules is changed and the number of layers in the whole network is reduced, such that the YOLOv4 network model can be compressed. The compressed network architecture of the YOLOv4 network model is shown in Figure 4.

The improved YOLOv4 prediction network still predicts three feature maps with different sizes to generate the location and target category of the detection box. Considering the imbalance between the cracks and background pixels in the image, the input image size of the model is 256 × 256, and the dimensions of the last three layers are 8 × 8, 16 × 16 and 32 × 32. Among them, the 8 × 8 feature map constructed by combining the two downsamplings of the shallow network with the deep network is mainly used to detect large objects, the once-upsampled 16 × 16 feature map spliced with the middle layer feature map of the backbone network is mainly used to detect medium objects, and the twice-upsampled 32 × 32 feature map spliced with the middle layer feature map of the backbone network is mainly used to detect small targets. The feature fusion and multi-scale detection method can make full use of different scales to extract different layers of semantic information, enhance the feature expression ability of the network, and improve the accuracy of object detection.

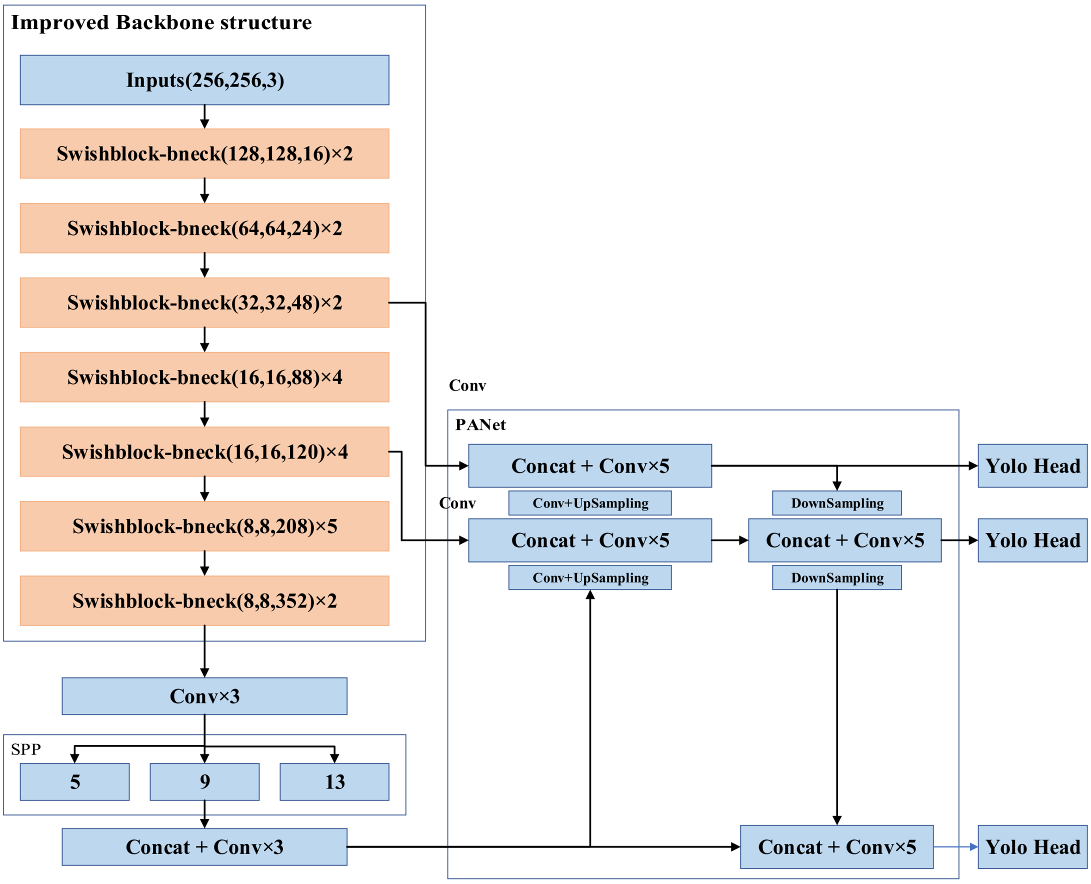

Improved Backbone structure
Inputs(256,256,3)
Swishblock-bneck(128,128,16)×2
Swishblock-bneck(64,64,24)×2
Swishblock-bneck(32,32,48)×2
Swishblock-bneck(16,16,88)×4
Swishblock-bneck(16,16,120)×4
Swishblock-bneck(8,8,208)×5
Swishblock-bneck(8,8,352)×2
Conv×3
SPP
5
9
13
Concat + Conv×3
Conv
PANet
Concat + Conv×5
Conv
Conv+UpSampling
DownSampling
Concat + Conv×5
Concat + Conv×5
Conv+UpSampling
DownSampling
Concat + Conv×5
Yolo Head
Yolo Head
Yolo Head

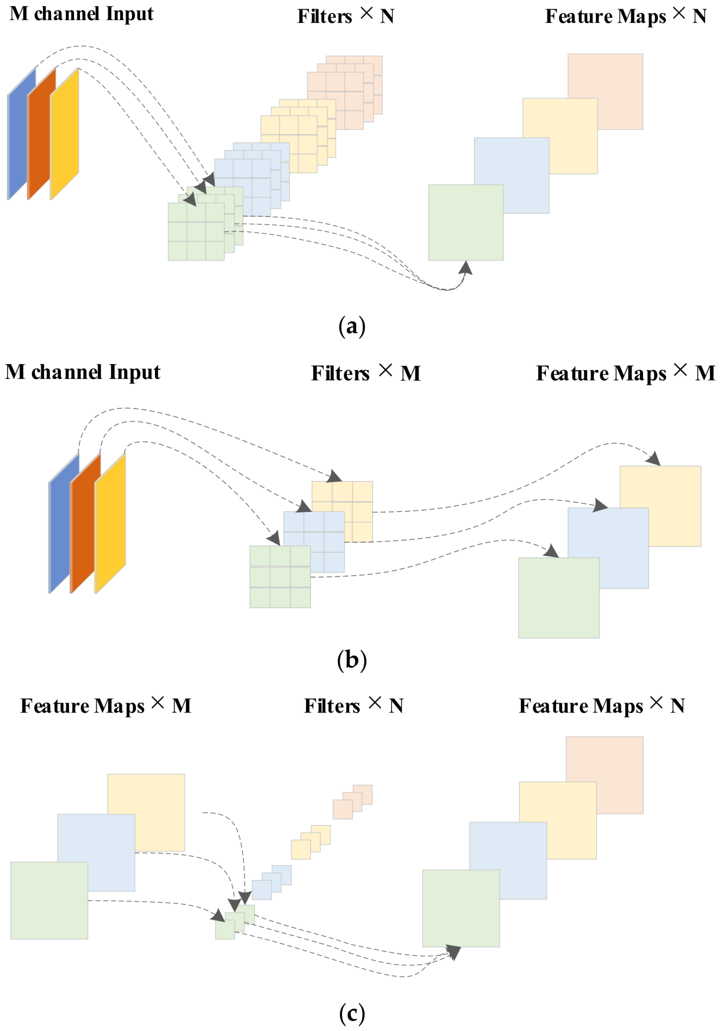
Figure 4. Compressed network architecture of the YOLOv4 network model.

2.3. Improvements to SPP Structure and PANet Module

The SPP structure and PANet module in the YOLOv4 network model play a role in enhancing the network feature extraction. By analyzing the structure of the SPP and PANet modules, it is found that the network structure contains a large number of 3 × 3 convolution layers and continuous quintic convolution structures that greatly increase the calculation amount of the model.

Figure 5a shows the process of extracting image detail features by conventional convolution, where M is the dimension of the input image, N is the number of channels filtered by the convolution kernel, and the size of the convolution kernel is 3 × 3. The dimension of the feature layer then extracted by the conventional convolution is N, and the number of parameters of the conventional convolution is $M \times N \times 3 \times 3$. The depthwise separable convolution is composed of depth convolution and 1 × 1 point convolution, as shown in Figure 5b,c. The depthwise separable convolution adopts the strategy of channel-by-channel convolution, and the extracted feature map adopts the form of point-by-point convolution to obtain a feature map with dimension N. The dimension of the input image is M, and the size of the separable convolution kernel is 3 × 3. Therefore, the number of parameters for the separable convolution operation is $M \times 3 \times 3 + 1 \times 1 \times M \times N$. Compared with the conventional convolution operation, the separable convolution is used to extract the texture features of the image, and the number of parameters is reduced, as shown in Equation (1).

$$\frac{M \times 3 \times 3 + 1 \times 1 \times M \times N}{M \times N \times 3 \times 3} = \frac{1}{N} + \frac{1}{9} \tag{1}$$



(**a**)

(**b**)

(**c**)

Figure 5. Depthwise separable convolution versus ordinary convolution; (**a**) general convolution filtering; (**b**) depthwise separable convolution; (**c**) point convolution operation.

The PANet module and the SPP structure contain continuous quintic convolution and continuous cubic convolution to enhance the process of image feature extraction. Equation (1) shows that the separable convolution operation greatly reduces the number of parameters compared with the conventional convolution operation. In order to further compress the network model, with the aim of maintaining channel separation, spatial convolution is realized based on separable convolution. The ordinary convolution of the SPP structure and PANet module is replaced in order to reduce the number of model parameters and memory dependence, as shown in Figure 6.

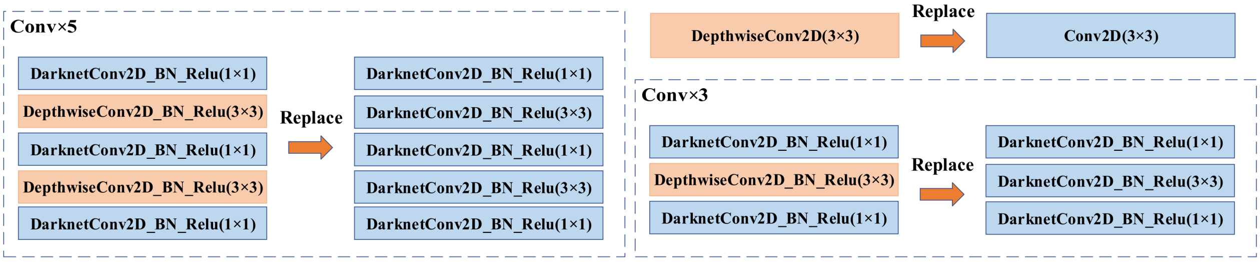

Conv×5
DarknetConv2D_BN_Relu(1×1)
DepthwiseConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)
DepthwiseConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)
Replace
DarknetConv2D_BN_Relu(1×1)
DarknetConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)
DarknetConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)
DepthwiseConv2D(3×3)
Replace
Conv2D(3×3)
Conv×3
DarknetConv2D_BN_Relu(1×1)
DepthwiseConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)
Replace
DarknetConv2D_BN_Relu(1×1)
DarknetConv2D_BN_Relu(3×3)
DarknetConv2D_BN_Relu(1×1)

Figure 6. Depthwise separable convolution replaces ordinary convolution.

2.4. Model Inference Speed Improvements

The SwishBlock bottleneck module in the previous section uses the convolution layer and Batch Normalization layer for forward operation. The batch normalization layer can accelerate the network's convergence and control its overfitting. Although it plays an active role during training, the forward operation is added in the network forward reasoning, which takes up more memory space and affects the processing speed. Therefore, in this section, the parameters in the batch normalization layer of the SwishBlock bottleneck module and the convolutional layer are merged to improve the forward reasoning speed of the model. The original batch normalization layer is represented as follows:

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2}$$

$$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{3}$$

$$\overset{\wedge}{x_i} \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{4}$$

$$y_i \leftarrow \gamma \overset{\wedge}{x_i} + \beta \tag{5}$$

where μ_B is the mean value of the dataset, and σ_B^2 is the variance of the dataset. $\hat{x}_i$ is the normalization of the dataset, and y_i is the output result after translation and scaling through the batch normalization layer. The convolution layer is calculated as shown in Equation (6), and the output of the BN layer through the convolution layer is shown in Equation (7). Equation (6) is substituted into Equation (7) and expanded to calculate the new weight and bias term of the convolution layer, as shown in Equation (8). Thus, the new weight and bias are shown in Equations (9) and (10), respectively. The new weight and bias term are used to perform the convolution layer calculation of the SwishBlock bottleneck module, and the result is the same as that of the original convolution layer plus the batch normalization layer, while reducing the forward reasoning speed of the model.

$$out = \sum_{i=1}^{k} w_i x_i + b \tag{6}$$

$$BN = \frac{\gamma(out - \mu_B)}{\sqrt{\sigma_B^2 + \varepsilon}} + \beta \tag{7}$$

$$BN = \frac{\gamma(\sum\limits_{i=1}^{k} w_i x_i + b - \mu_B)}{\sqrt{\sigma_B^2 + \varepsilon}} + \beta = \frac{\gamma\sum\limits_{i=1}^{k} w_i x_i + \gamma(b - \mu_B)}{\sqrt{\sigma_B^2 + \varepsilon}} + \beta \tag{8}$$

$$w_{new} = \frac{\gamma\sum\limits_{i=1}^{k} w_i}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{9}$$

$$b_{new} = \frac{\gamma(b - \mu_B)}{\sqrt{\sigma_B^2 + \varepsilon}} + \beta \tag{10}$$

2.5. Improvements of Loss Function

The loss function of the YOLOv4 network during training consists of three parts: boundary box regression loss L_{ciou}, confidence loss L_{conf} and classification loss L_{class}, as shown in Equation (11).

$$\begin{cases} L_{conf} = & -\sum\limits_{i=0}^{S^2}\sum\limits_{j=0}^{B} I_{i,j}^{obj}[C_i^j \log(C_i^j) + (1 - C_i^j)\log(1 - C_i^j)] \\ & -\lambda_{noobj}\sum\limits_{i=0}^{S^2}\sum\limits_{j=0}^{B} I_{i,j}^{noobj}[C_i^j \log(C_i^j) + (1 - C_i^j)\log(1 - C_i^j)] \\ L_{class} = & -\sum\limits_{i=0}^{S^2} I_{i,j}^{obj} \sum\limits_{c\in classes} [P_i^j \log(P_i^j) + (1 - P_i^j)\log(1 - P_i^j)] \\ Loss = & L_{ciou} + L_{conf} + L_{class} \end{cases} \tag{11}$$

where S^2 and B are the feature map scale and a priori box, and λ_{noobj} is the weight coefficient. $I_{i,j}^{obj}$ and $I_{i,j}^{noobj}$ represent the target and no target at the j-th a priori box of the i-th grid, respectively. c is the diagonal distance between the predicted box and the closure area of the actual box. b, w, and h represent the center coordinates, width, and height of the prediction box, while b^{gt}, w^{gt}, and h^{gt} represent the center coordinates, width, and height of the actual box, respectively. C_i^j and $\overline{C}_i^j$ represent the confidence of the prediction box and the labeled box, and P_i^j and $\overline{P}_i^j$ represent the class probability of the prediction box and the labeled box, respectively.

When the model is used to detect cracks, the size of the cracks themselves is small, and the objects occupy a small proportion of the background. To balance the proportion of the foreground and background data samples, a modulation coefficient α is added to the cross-entropy loss function of object detection and classification using the focal loss function of the RetinaNet network for reference, as shown in Equation (12).

$$L_{class} = -\sum_{i=0}^{S^2} I_{i,j}^{obj} \sum_{c\in classes} [P_i^j \log(P_i^j) + (1 - P_i^j)^{\alpha} \log(1 - P_i^j)] \tag{12}$$

3. Experiments

A program of experimental studies is shown in Figure 7.

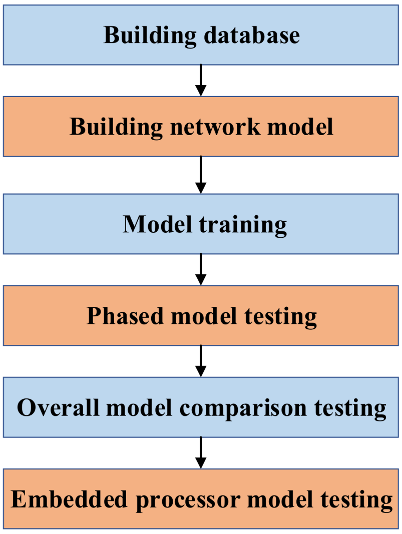


Figure 7. A program of experimental studies.

3.1. Image Database Creation

A smartphone is used for image acquisition. For the purpose of collecting images of small cracks on a concrete surface, all images are taken with a distance of 0.1 m between the smartphone and the concrete surface. Two-thousand original images with sizes of 3024 × 3024 pixels are extracted from the surfaces of concrete buildings. Each original image can be cropped to generate 139 images with sizes of 256 × 256 pixels. However, several cropped images do not include cracks. As a result, the images with cracks are meticulously selected from the cropped image set. Finally, 10,000 images conforming to the requirements are selected to create the database.

To assess the generalization ability of the improved model, the 10,000 images are divided into five parts according to the fivefold cross-validation principle, of which 80% are used to train and validate the model and the remaining 20% are used to test. More precisely, 8000 images are randomly selected from the 10,000 images, among which 7000 images are used to generate a training set and 1000 images are used to create a validation set. The remaining 2000 images not selected for training or validation are used to build a testing set.

3.2. Model Initialization

In the process of network training, in order to improve efficiency and better save computing resources and time, this paper adopts the training strategy of freezing certain layers. The entire training process is divided into two stages. In the first stage, only the backbone network structure is trained; in the second stage, the overall network structure is trained. In the training process, the Cosine Annealing learning rate strategy is adopted, and the hyperparameters are optimized according to the genetic algorithm. The initial parameter settings of the first stage and the second stage are shown in Tables 1 and 2, respectively.

Table 1. Initial parameters of the first stage of the training process.

Parameter	Value
Base_LR	10^{-3}
Batch_Size	16
Train_Epoch	100
Weight_decay	5×10^{-4}
Lr_scheduler_Max_iterations	5
Lr_scheduler_Minimum_lr	10^{-5}

Table 2. Initial parameters of the second stage of the training process.

Parameter	Value
Base_LR	10^{-4}
Batch_Size	16
Train_Epoch	500
Weight_decay	5×10^{-4}
Lr_scheduler_Max_iterations	5
Lr_scheduler_Minimum_lr	10^{-5}

3.3. Evaluation Metrics of Accuracy

Crack detection based on deep learning is quantitatively measured by objective evaluation metrics, which can measure many aspects of the quality of a restoration algorithm. There are many objective evaluation metrics commonly used in object detection, such as intersection over union (IoU), precision, recall and mean average precision (mAP). IOU is the ratio of the intersection and union between the bounding box predicted by the model and the real bounding box, which is also called the Jaccard index.

mAP is a common index used to evaluate the accuracy of algorithms in the field of object detection. In this paper, the objective evaluation index mAP is used for calculation, as shown in Equation (13), where AP is the average precision. Taking recall as the horizontal axis and precision as the vertical axis, the P-R curve can be obtained, and the AP value can then be calculated. Simply, this averages the precision values on the P-R curve. The definition of AP is shown in Equation (14).

$$mAP = \frac{1}{|Q_R|}\sum_{q=1}^{Q_R} AP(q) \tag{13}$$

$$AP = \int_0^1 p(r)dr \tag{14}$$

The construction of the P-R curve is drawn by the precision and the recall. The precision refers to the number of correct recognitions of all samples predicted to be positive. The recall reflects the missed detection rate of the model. Precision and recall are defined in Equations (15) and (16), respectively. True positive (TP) indicates that the detection category is positive and predicted to be positive, while false positive (FP) indicates that the detection category is negative and predicted to be negative. False negative (FN) indicates that the detection category is positive and predicted to be negative, and P is the number of positive samples in the testing set. The precision and recall are independent of each other. High precision means that the false detection rate is low, which can lead to a high missed detection rate.

In this paper, in addition to the mAP, the model size and computational complexity FLOPs are used to evaluate the model compression algorithm. The model's size is closely related to its parameters, which can be used to measure the simplification of the YOLOv4 model. FLOPs reflect the calculation amount of the algorithm. The unit of FLOPs is GMacs, which is short for Giga multiply–accumulation operations per second. It represents the floating-point operations per second, which can reflect the algorithm's calculation performance.

$$Precision = \frac{TP}{TP+FP} \tag{15}$$

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

4. Results and Discussion

The training process was implemented on a server with a high-performance GPU (NVIDIA GeForce RTX 2080 Super), 64 GB DDR4 memory, and an Intel i9-10900K CPU. The training process is based on the deep learning framework Pytorch 1.14.

4.1. Test Comparison Results of Sub-Modules of the Improved Algorithm

In order to fully verify the rationality of the various improvements to the YOLOv4 network model in this paper, a step-by-step verification experiment is performed. Objective evaluation indexes are used to qualitatively measure the quality and speed of the restoration algorithm. The commonly used objective metrics for evaluating object detection performance are mAP, parameters, and calculation amount (FLOPs).

First, the main framework of the model is improved, and the specific results are compared as shown in Table 3. It can be seen that when only the main framework of the model is improved, mAP is 96.43% of the original YOLOv4 model, and the amounts of the model parameters and algorithm calculations are reduced by 83.89% and 97.12%, respectively.

Table 3. Improved performance comparison of main model framework.

Method	mAP (%)	Parameters (M)	FLOPs (GMacs)
YOLOv4	95.50	64.00	63.92
Improved backbone	92.09	10.31	1.84

Based on the depth separable convolution structure, the ordinary convolution of the SPP structure and the PANet module is replaced by spatial convolution. Table 4 shows the performance comparison of the improved main model framework, PANet and SPP structure. It can be seen that after replacing the ordinary convolution of the SPP and PANet structure with separable convolution, the parameters are reduced from 10.31 M to 8.22 M, and the amount of calculations is reduced from 1.84 GMacs to 0.69 GMacs. The amount of model parameters and calculations are both greatly reduced.

Table 4. Performance comparison of improved main model framework, PANet and SPP structure.

Method	mAP (%)	Parameters (M)	FLOPs (GMacs)
YOLOv4	95.50	64.00	63.92
Improved backbone	92.09	10.31	1.84
Improved backbone + PANet + SPP structure	91.88	8.22	0.69

To improve the forward reasoning speed of the model, the parameters in the convolutional layer and the batch normalization (BN) layer are merged. Frames per second (FPS) is used to consider the effectiveness of the forward inference speed, and the processing speed is compared with that of the original YOLOv4 network model. The specific index FPS are shown in Table 5. It can be seen that compared with the original YOLOv4 network model; the FPS processed by the improved network model is greatly improved. After the parameters in the convolution layer and the BN layer are merged, the processing speed of a 256 × 256 pixel image is increased from 101 FPS to 112 FPS.

Table 5. Performance comparison of merge convolution layer and BN layer.

Method	FPS
	256 × 256
YOLOv4	31
Improved backbone + PANet+ SPP structure	101
Improved backbone + PANet + SPP structure + Merging convolution layer and BN layer	112

To balance the proportion of the foreground and background data samples, the specific mAP, model parameters and calculation amount of the improved loss function are shown in Table 6. It can be seen that after adding the modulation factor to the loss function of the model, the mAP of the algorithm in the test database increases from 91.89% to 94.09%.

Through a complete comparison between the original YOLOv4 model and the improved YOLOv4 model, the tested mAP is shown in Figure 8. It can be seen that when the self-made database is used for detection, the mAP of the original YOLOv4 model is 95.50%, the mAP of the improved model is 94.09% and the detection performance decreases slightly. It can be concluded from Table 6 that the image detection performance mAP of the improved Yolov4 model is 98.52% of that of the original model, and the model parameters and calculation amount are reduced by 87.43% and 99.00%, respectively.

Table 6. Comparison of performance of different algorithms using RSOD dataset for object detection.

Method	mAP (%)	Parameters (M)	FLOPs (GMacs)
YOLOv4	95.50	64.00	63.92
Improved backbone	92.09	10.31	1.84
Improved backbone + PANet + SPP structure	91.88	8.22	0.69
Improved backbone + PANet + SPP structure + Merging convolution layer and BN layer	91.89	8.04	0.64
Improved backbone + PANet + SPP structure + Merging convolution layer and BN layer + loss function	94.09	8.04	0.64

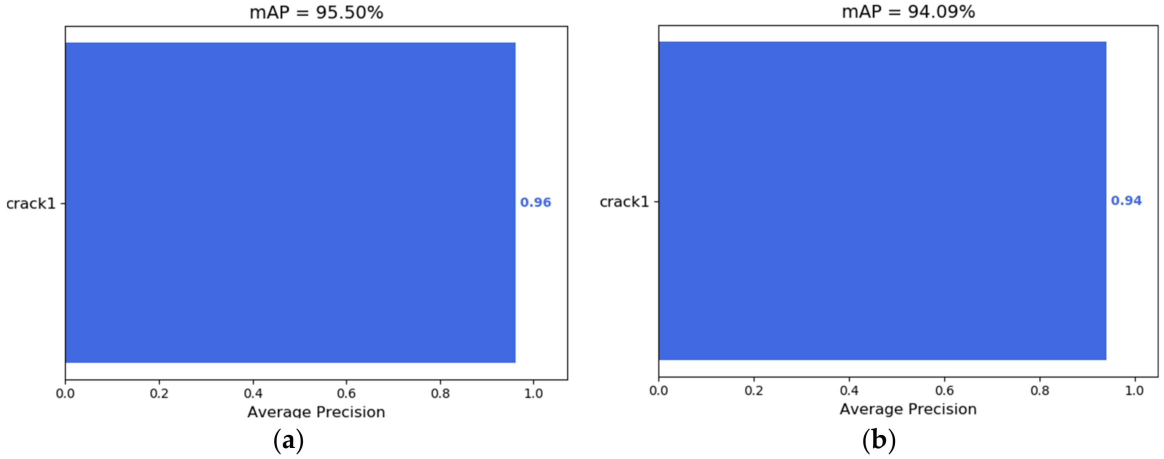


Figure 8. mAP of the original YOLOv4 model and the improved model. (**a**) mAP of the original YOLOv4 model; (**b**) mAP of the improved model.

In this study, the model modules are improved to reduce the depth and complexity of the overall network structure. Meanwhile, the separable convolution is used to realize spatial convolution, and the SPP and PANet modules are improved to reduce the model parameters. The convolutional layer and batch normalization layer are merged to improve the model inference speed. In addition, using the focal loss function for reference, the loss function of object detection network is improved to balance the proportion of the cracks and the background samples. The detection performance of the improved model is satisfactory in mAP, and the model size and calculation amount are greatly reduced.

4.2. Comparative Results of Frontier Algorithm Tests

In this paper, the self-made database is used for training and testing, and the frontier network models in the field of object detection are used for comparison. The comparison results are shown in Table 7. It can be concluded that the improved model exhibits almost no loss in mAP compared to the high-performance algorithms, but the model size and calculation amount are greatly reduced. Through comparison with the faster lightweight network models, it can be seen that the model sizes are close, but the calculation amount FLOPs are reduced, and the detection performance mAP is higher than that of the classic lightweight network models. To demonstrate the detection performance of the improved model more intuitively, images shown in Figure 9 were randomly selected from the database for testing.

Table 7. Comparison of object detection performance of different algorithms.

Method	Backbone	mAP (%)	Parameters (M)	FLOPs (GMacs)
YOLOv4	CSPDarknet	95.50	64.00	63.92
YOLOv5m	CSPDarknet	85.58	21.40	51.30
SSD	VGG-16	89.64	26.29	127.50
CenterNet	ResNet-50	92.35	32.67	35.79
YOLOv4-tiny	CSPDarknet	72.22	5.90	4.31
MobileNet-SSD	MobileNet-v1	84.28	8.85	12.40
Ours	CSPDarknet	94.09	8.04	0.64

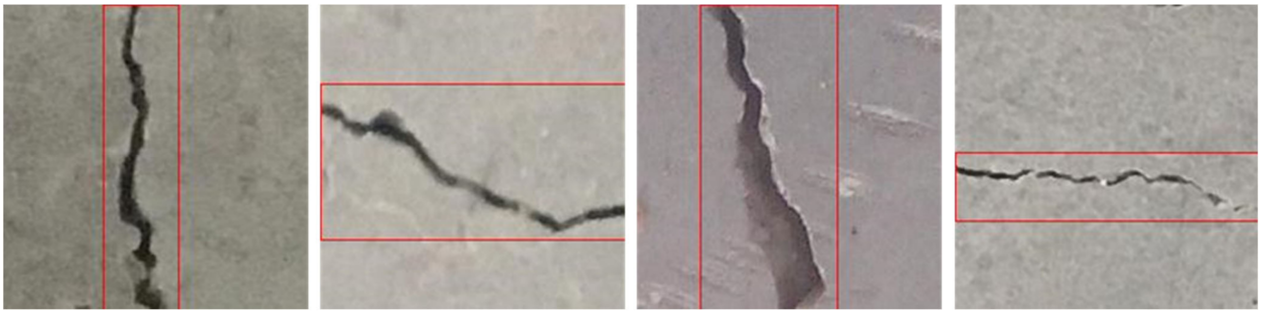


Figure 9. Detection results of the concrete surface cracks.

4.3. Experimental Results of Embedded Platform

To further verify the processing capability of the improved model in mobile devices, the trained model is deployed to the Jetson Xavier NX embedded platform for verification. The processor is small in size, low in power consumption, and strong in computing performance. The performances of the YOLOv4 network model, the YOLOv4-tiny network model and the improved model in this paper are compared in terms of the objective evaluation indicators mAP and FPS respectively, as shown in Table 8. It can be concluded that for the Jetson Xavier NX embedded platform, the input image is 256 × 256 pixels, and the YOLOv4 network model can process 16 FPS due to its complex structure, which cannot meet the needs of mobile devices for real-time crack detection. The YOLOv4-tiny network model and the improved model in this paper can process 56 and 44 FPS, which can meet the needs of real-time detection. However, the mAP of the YOLOv4-tiny network model is 72.22%, and the recognition rate is low. Compared with the YOLOv4-tiny network model, the improved YOLOv4 network model has higher accuracy and faster processing speed, which meets the requirements of accurate real-time object detection.

Table 8. Comparison of object detection performance of different algorithms.

Method	mAP (%)	FPS
YOLOv4	95.50	16
YOLOv4-tiny	72.22	56
Ours	94.09	44

5. Conclusions

A real-time concrete surface crack detection method based on the improved YOLOv4 is proposed. The improved model for concrete crack detection adopts the symmetry concept in the extraction of backbone and the design of neck and head. It is described in detail in Section 2. A smartphone is used to collect 2000 raw 3024 × 3024 pixel images from the surfaces of concrete buildings. To reduce the computation of the training process, the collected images are cropped to 256 × 256 pixels. Sets of 7000, 1000, and 2000 images are used for training, validation, and testing, respectively. The improved YOLOv4 model achieved an mAP of 94.09%, which is 98.52% of the original YOLOv4 model. The crack detection performance decreased slightly, but the parameters and calculation amount of the model are reduced by 87.43% and 99.00%, respectively. Compared with the results of the high-performance network models in object detection (such as YOLOv4, YOLOv5m, SSD, and CenterNet), it can be concluded that the improved model has almost no loss in mAP, but the model size and calculation amount are greatly reduced. In addition, compared with the detection results of the lightweight network models (such as YOLOv4-tiny and MobileNet-SSD), the model sizes are close, but the calculation amount FLOPs are reduced, and the detection performance mAP is higher. When the improved model was deployed to the Jetson Xavier NX embedded platform for testing, it achieved an mAP of 94.06% with 44 FPS. The size, accuracy, and processing speed of the model can meet the requirements of accurate real-time object detection, which can provide support for the development of mobile monitoring system. As a result, it can achieve real-time automatic vision-based crack detection on concrete surface without other equipment.

Although the improved YOLOv4 model shows good performance, there is still a long way to go before it is suitable for engineering applications. First, in the implementation of the improved method, there are many artificially adjusted hyperparameters derived from the training and verification set. Many experiments need to be conducted to explore the influence of these hyperparameters on the performance of the model. Second, a real-time mobile crack detection system (including APPs and a website) should be developed to monitor the concrete surface cracks for timely repair and protection. Lastly, we will collect more types of defect images to expand the database, such that the proposed method has greater accuracy and robustness.

Author Contributions: Conceptualization, G.Y. and Y.S.; methodology, Y.S. and X.L.; software and formal analysis, X.L.; writing—original draft preparation, Y.S.; review and editing, M.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities (no. 2020CDJQY-A067), the National Key Research and Development Project (no. 2019YFD1101005) and the National Natural Science Foundation of China (no. 51608074).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Please contact the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gavilan, M.; Balcones, D.; Marcos, O.; Llorca, D.F.; Sotelo, M.A.; Parra, I.; Ocana, M.; Aliseda, P.; Yarza, P.; Amirola, A. Adaptive road crack detection system by pavement classification. *Sensors* **2011**, *11*, 9628–9657. [CrossRef] [PubMed]
2. Cabaleiro, M.; Lindenbergh, R.; Gard, W.F.; Arias, P.; van de Kuilen, J.W.G. Algorithm for automatic detection and analysis of cracks in timber beams from LiDAR data. *Constr. Build. Mater.* **2017**, *130*, 41–53. [CrossRef]
3. Li, S.; Yuan, C.; Liu, D.; Cai, H. Integrated Processing of Image and GPR Data for Automated Pothole Detection. *J. Comput. Civ. Eng.* **2016**, *30*, 04016015. [CrossRef]
4. Li, Q.; Zou, Q.; Zhang, D.; Mao, Q. FoSA: F* Seed-growing Approach for crack-line detection from pavement images. *Image Vis. Comput.* **2011**, *29*, 861–872. [CrossRef]
5. Zhou, Y.; Wang, F.; Meghanathan, N.; Huang, Y. Seed-Based Approach for Automated Crack Detection from Pavement Images. *Transp. Res. Rec.* **2016**, *2589*, 162–171. [CrossRef]
6. Choi, J.-I.; Lee, Y.; Kim, Y.Y.; Lee, B.Y. Image-processing technique to detect carbonation regions of concrete sprayed with a phenolphthalein solution. *Constr. Build. Mater.* **2017**, *154*, 451–461. [CrossRef]
7. Yeum, C.M.; Dyke, S.J. Vision-Based Automated Crack Detection for Bridge Inspection. *Comput. Aided Civ. Inf. Eng.* **2015**, *30*, 759–770. [CrossRef]

8. Dorafshan, S.; Thomas, R.J.; Maguire, M. Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete. *Constr. Build. Mater.* **2018**, *186*, 1031–1045. [CrossRef]
9. Oh, J.-K.; Jang, G.; Oh, S.; Lee, J.H.; Yi, B.-J.; Moon, Y.S.; Lee, J.S.; Choi, Y. Bridge inspection robot system with machine vision. *Automat. Constr.* **2009**, *18*, 929–941. [CrossRef]
10. Ciresan, D.; Meier, U.; Masci, J.; Schmidhuber, J. Multi-column deep neural network for traffic sign classification. *Neural Netw.* **2012**, *32*, 333–338. [CrossRef]
11. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal.* **2015**, *37*, 1904–1916. [CrossRef]
13. Chen, F.-C.; Jahanshahi, M.R. NB-CNN: Deep Learning-Based Crack Detection Using Convolutional Neural Network and Naïve Bayes Data Fusion. *IEEE Trans. Ind. Electron.* **2018**, *65*, 4392–4400. [CrossRef]
14. Bang, S.; Park, S.; Kim, H.; Kim, H. Encoder–decoder network for pixel-level road crack detection in black-box images. *Comput. Aided Civ. Inf. Eng.* **2019**, *34*, 713–727. [CrossRef]
15. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal.* **2017**, *39*, 1137–1149. [CrossRef]
16. Mei, Q.; Gül, M. A cost effective solution for pavement crack inspection using cameras and deep neural networks. *Constr. Build. Mater.* **2020**, *256*, 119397. [CrossRef]
17. Yang, Y.; Xiang, C.; Jiang, M.; Li, W.; Kuang, Y. Bridge damage identification method considering road surface roughness by using indirect measurement technique. *China J. Highw. Transp.* **2019**, *32*, 99–106.
18. Fei, Y.; Wang, K.C.P.; Zhang, A.; Chen, C.; Li, J.Q.; Liu, Y.; Yang, G.; Li, B. Pixel-Level Cracking Detection on 3D Asphalt Pavement Images Through Deep-Learning- Based CrackNet-V. *IEEE Trans. Intell. Transp.* **2020**, *21*, 273–284. [CrossRef]
19. Dung, C.V.; Anh, L.D. Autonomous concrete crack detection using deep fully convolutional neural network. *Automat. Constr.* **2019**, *99*, 52–58. [CrossRef]
20. Yang, Y.; Liang, J.; Yuan, A.; Lu, H.; Luo, K.; Shen, X.; Wan, Q. Bridge element bending stiffness damage identification based on new indirect measurement method. *China J. Highw. Transp.* **2021**, *34*, 188–198.
21. Lee, D.; Kim, J.; Lee, D. Robust Concrete Crack Detection Using Deep Learning-Based Semantic Segmentation. *Int. J. Aeronaut. Space* **2019**, *20*, 287–299. [CrossRef]
22. Sun, Y.J.; Yang, Y.; Yao, G.; Wei, F.J.; Wong, M.P. Autonomous Crack and Bughole Detection for Concrete Surface Image Based on Deep Learning. *IEEE Access.* **2021**, *9*, 85709–85720. [CrossRef]
23. Wei, F.J.; Yao, G.; Yang, Y.; Sun, Y.J. Instance-level recognition and quantification for concrete surface bughole based on deep learning. *Autom. Construction* **2019**, *107*, 102920. [CrossRef]
24. Yao, G.; Wei, F.J.; Yang, Y.; Sun, Y.J. Deep-Learning-Based Bughole Detection for Concrete Surface Image. *Adv. Civ. Eng.* **2019**, *2019*, 1–12. [CrossRef]
25. Wei, W.; Ding, L.Y.; Luo, H.B.; Li, C.; Li, G.W. Automated bughole detection and quality performance assessment of concrete using image processing and deep convolutional neural networks. *Constr. Build. Mater.* **2021**, *281*, 122576. [CrossRef]
26. Luo, R.F.; Zhang, L. Intelligent Detection Method for Internal Cracks in Aircraft Landing Gear Images under Multimedia Processing. *Symmetry* **2021**, *13*, 778. [CrossRef]
27. Yang, Y.; Cheng, Q.; Zhu, Y.; Wang, L.; Jin, R. Feasibility study of tractor-test vehicle technique for practical structural condition assessment of beam-like bridge deck. *Remote Sens.* **2020**, *12*, 114. [CrossRef]
28. Zhang, Y.X.; Lei, Y. Data Anomaly Detection of Bridge Structures Using Convolutional Neural Network Based on Structural Vibration Signals. *Symmetry* **2021**, *13*, 1186. [CrossRef]
29. Yang, Y.; Li, J.L.; Zhou, C.H.; Law, S.S.; Lv, L. Damage detection of structures with parametric uncertainties based on fusion of statistical moments. *J. Sound Vib.* **2019**, *442*, 200–219. [CrossRef]
30. Liu, B.; Zhang, Y.; He, D.J.; Li, Y.X. Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks. *Symmetry* **2017**, *10*, 11. [CrossRef]
31. Yang, Y.; Li, C.; Ling, Y.; Tan, X.; Luo, K. Research on new damage detection method of frame structures based on generalized pattern search algorithm. *China J. Sci. Instrum.* **2021**, *42*, 123–131.
32. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the 2016 14th European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–10 October 2016; pp. 21–37.
33. Bochkovskiy, A.; Wang, C.Y.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
34. Tan, M.X.; Pang, R.M.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.