Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7240129, 10 pages
https://doi.org/10.1155/2020/7240129

Research Article

Concrete Cracks Detection Using Convolutional Neural Network Based on Transfer Learning

Chao Su and Wenjun Wang

College of Water Conservancy and Hydropower Engineering, Hohai University, Nanjing 210098, China

Correspondence should be addressed to Wenjun Wang; wenjunwang@hhu.edu.cn

Received 5 August 2020; Revised 21 September 2020; Accepted 26 September 2020; Published 17 October 2020

Academic Editor: Zheng-zheng Wang

Crack plays a critical role in the field of evaluating the quality of concrete structures, which affects the safety, applicability, and durability of the structure. Due to its excellent performance in image processing, the convolutional neural network is becoming the mainstream choice to replace manual crack detection. In this paper, we improve the EfficientNetB0 to realize the detection of concrete surface cracks using the transfer learning method. The model is designed by neural architecture search technology. The weights are pretrained on the ImageNet. Supervised learning uses Adam optimizer to update network parameters. In the testing process, crack images from different locations were used to further test the generalization capability of the model. By comparing the detection results with the MobileNetV2, DenseNet201, and InceptionV3 models, the results show that our model greatly reduces the number of parameters while achieving high accuracy (0.9911) and has good generalization capability. Our model is an efficient detection model, which provides a new option for crack detection in areas with limited computing resources.

1. Introduction

In the current infrastructure, the concrete structure accounts for the largest proportion. For the concrete structure, cracks are a frequently encountered disease. With the increase in service time, the number and width of cracks show a gradual increasing trend, which seriously affects the safety, applicability, and durability of the structure. Therefore, it is of great significance to detect cracks regularly and takes corresponding maintenance measures for the safety of the concrete structures [1–3]. The traditional crack detection method is mainly based on the direct detection of professionals with related instruments. This detection method is not only labor-consuming but also time wasting. In addition, it also brings great hidden dangers to the safety of people.

In order to find an efficient and safe crack detection method and overcome various shortcomings of manual detection, people turned their attention to image processing technologies (IPTs) [4]. In the past decades, the computer vision community has been working on the automated detection of images and proposed a series of image processing techniques, including thresholding [5, 6], edge detection [7], wavelet transforms [8, 9], and machine learning [10] and so on. Image thresholding divides the crack in the image at the pixel level according to the features of different pixel values, which makes the image simple and facilitates further processing. Differential operators used for edge detection mainly contain Roberts operator, Sobel operator, and Laplace operator [11]. The basic idea of wavelet transform is using a set of wavelet functions or basis functions to represent a function or signal, such as an image signal. Machine learning extracts feature vectors of crack from the training dataset and combines certain algorithms to make prediction. These methods solve the problem of crack detection in engineering effectively. However, due to the unevenness of cracks, the diversity of surface textures, and the complexity of the background, this field of research is still active.

As a new technology in the research of machine learning algorithms, deep learning is motivated by the establishment and simulation of a neural network for analyzing problems and learning like the human brain. Through typical feature

learning [12], each layer of the network takes the output of the previous layer as its own input, learns a deep nonlinear network structure, and transforms the specific raw data into a more abstract expression model. The rapid expansion of effective datasets, the realization of high-performance computing hardware, and the continuous improvement of training methods are driving the rapid development of deep learning. In 2012, AlexNet [13] won the championship with an overwhelming advantage in ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). Convolutional neural network (CNN) begins to attract the attention of many researchers since then. CNN is a feed-forward neural network, and the connection between neurons is inspired by the animal visual cortex. It has the characteristics of local connectivity and parameter sharing and has excellent performance in large-scale image processing [14]. In recent years, researchers have begun to use convolutional neural networks to detect road defects automatically. The following is a brief overview of the application of CNN in crack detection.

Zhang et al. [15] proposed a crack detection method based on deep learning, which seems to be one of the earliest works applying CNN to road crack detection. The pavement pictures are taken by smartphones, and the network model is built on the Caffe DeepLearning (DL) framework. By comparing with traditional machine learning classifiers such as support vector machine (SVM) and boosting methods, the author proved the effectiveness of deep learning methods. Pauly et al. [16] studied the influence of CNN depth and the position change between training dataset and test dataset on pavement crack detection accuracy. The results show that increasing the network depth can improve the network performance, but when the image position changes, the detection accuracy will be greatly reduced. Maeda et al. [17] created a large-scale road damage dataset and marked the location and type of road damage in each picture. Finally, an end-to-end object detection method based on deep learning was used to train the damage detection model. Maeda et al. also transplanted their model into a mobile phone application to facilitate road damage detection in areas lacking experts and financial resources. Xu et al. [18] established an end-to-end bridge crack detection model to realize automatic bridge crack detection. The use of depthwise separable convolution reduces the number of parameters effectively. The atrous spatial pyramid pooling (ASPP) module extracts information at multiple scales. The model achieves a detection accuracy of 96.37% without pretraining. Li et al. [19] proposed the YOLOv3-Lite method, which greatly improves the crack detection speed without reducing the detection accuracy. Tong et al. [20] used convolutional neural networks (CNNs) to detect, locate, and measure ground penetrating radar images automatically and finally reconstruct concealed cracks in three dimensions, realizing a low-cost damage characterization method. Yang et al. [21] realized the pixel-level detection of cracks based on a fully convolutional neural network. The fully convolutional neural network is composed of upsampling and downsampling and can detect objects at different scales. In terms of crack segmentation, the accuracy, precision, recall, and F1-score are 97.96%, 81.73%, 78.97%, and 79.95%, respectively. Zhu and Song [22] used the transfer learning method to improve VGG16 and realized the accurate classification of surface defects on concrete bridges. The training of convolutional neural networks usually requires a large number of data, but in many cases, it is more expensive to obtain large-scale data. The pretrained model can be transferred to the task of crack detection by means of transfer learning. The results show that the model can effectively extract defect features and provide a new idea for surface defect detection. Deng et al. [23] added a region-based deformable module to Faster R-CNN [24], R-FCN [25], and FPN-based Faster R-CNN [26] to improve the evaluation accuracy of crack detection.

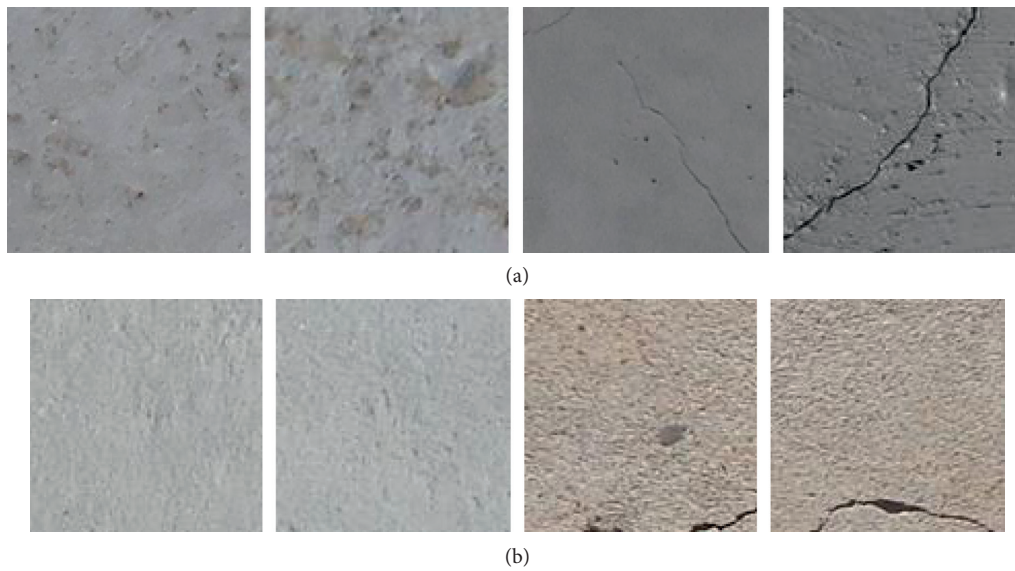
In this paper, we use the transfer learning method to build a model for concrete surface crack detection. Compared with existing models, our model achieves a good balance among accuracy, model size, and training speed. Due to the use of transfer learning, the model becomes easier to train and faster to converge and has better generalization capability.

The remaining of this paper is structured as follows: Section 2 describes the dataset and image preprocessing method; Section 3 presents the overall model architecture and training details; Section 4 shows our experimental results; and Section 5 delivers the conclusion of this paper.

2. Dataset and Data Augmentation

2.1. Building Dataset. In this study, we use the dataset collected by Li and Zhao [27]. The photos in this dataset were obtained by a smartphone with a resolution of 4160×3120 pixels from the surface of a pylon and anchor room of a suspension bridge in Dalian, Liaoning, China. Then, images are cropped to 256×256 pixel resolutions. After cropping, the images are manually divided into two categories: with cracks and without cracks. In this study, we only use 12,000 photos of the dataset, and the number of crack and noncrack images are set to equal. These images include crack features and background features under various conditions. The 12,000 selected images are divided into the training set, validation set, and test set at the ratio of $6:2:2$. The number of crack and noncrack images in the three datasets are set to equal. In addition, we also select 1,000 concrete bridge images with cracks and 1,000 images without cracks from the SDNET2018 [28] dataset. This will introduce various changes, such as changes in lighting conditions and the features of cracks and crack surface texture to further test the generalization capability of the model and make a more comprehensive evaluation of the model. Figure 1 shows several crack and noncrack images in the two datasets.

2.2. Data Augmentation. The generalization capability of the neural network model is closely related to the number of training datasets. But in reality, the amount of data is limited. One way to solve this problem is to create fake data and add it to the training set—data augmentation [29]. By allowing limited data to generate more equivalent data to artificially expand the training dataset, data augmentation can also



(a)

(b)

Figure 1: (a) Sample noncrack and crack images from the original dataset; (b) sample noncrack and crack images from the SDNET2018 dataset.

effectively overcome the overfitting phenomenon. Currently, it is widely used in various fields of deep learning. At present, the commonly used methods of data augmentation in the field of computer vision mainly include data augmentation based on image processing techniques and data augmentation based on deep learning.

In this paper, we use the built-in ImageDataGenerator interface of Tensorflow2.0 to enhance the input image data, including image flip, rotation, shift, and other operations. Figure 2 shows several crack images after data augmentation.

3. Model Construction and Training

3.1. The CNN. Convolutional neural network is a special kind of neural network. Its main feature is convolution operation, which has excellent performance for large-scale image processing. Generally speaking, a convolutional neural network is a hierarchical model that extracts the original data (such as RGB images) from the input layer through a series of operations such as convolution, pooling, and nonlinear activation function mapping. Abstract layer by layer, extract feature information, and finally make predictions. Deep convolutional neural networks have become popular since 2012, and now, they have become a pivotal research topic in the field of artificial intelligence. Classical convolutional neural network includes AlexNet [13], VGG [30], GoogLeNet [31], and ResNet [32]. A layer is the basic calculation unit of a CNN. CNN is mainly composed of input layer, convolution layer, activation function, pooling layer, fully connected layer, and Softmax layer.

3.2. Swish Activation Function. Ramachandran et al. [33] proposed a Swish activation function using a combination of exhaustive and reinforcement learning-based search. The effectiveness of this activation function has been verified in some large neural networks. The EfficientNet model used in this article uses the Swish activation function. The definition of Swish is defined as follows:

$$f(x) = x \cdot \sigma(\beta x), \tag{1}$$

where $\sigma(t) = (1 + \exp(-t)^{-1})$ and β is either a constant or a trainable parameter.

Figure 3 shows the graph of Swish for different values of β.

3.3. Architecture Description. EfficientNet [34] was proposed by Google in 2019 which has great capability of feature extraction. Compared with other classic convolutional neural networks, it has fewer parameters and higher accuracy. The baseline network of EfficientNet is designed using multiobjective neural architecture search, and then, the baseline network is scaled in terms of depth, width, and resolution to achieve a balance among them. The compound scaling method is defined as follows:

$$\text{depth} = \alpha^{\phi}, \tag{2}$$

$$\text{width} = \beta^{\phi}, \tag{3}$$

$$\text{resolution} = \gamma^{\phi}, \tag{4}$$

$$\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2, \tag{5}$$

$$\begin{aligned} \alpha &\gg 1, \\ \beta &\gg 1, \\ \gamma &\gg 1, \end{aligned} \tag{6}$$

where α, β, and γ can be calculated by a small grid search.

Firstly, EfficientNetB0 performs a 3×3 convolution operation on the input image, and then, the next 16 mobile

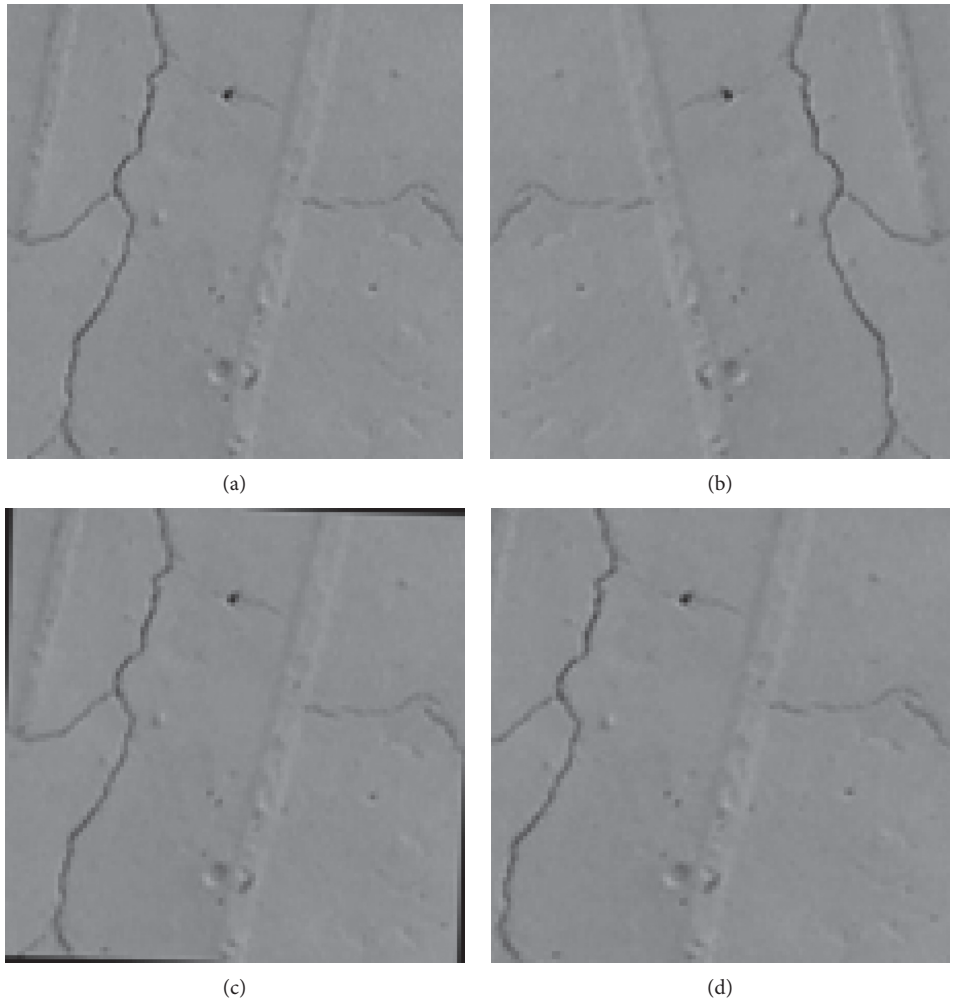


Figure 2: Images after data augmentation: (a) original image; (b) horizontal flip; (c) rotation; (d) horizontal shift.

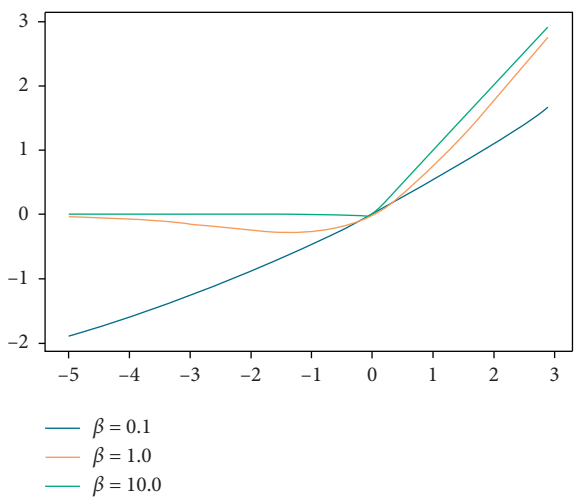


Figure 3: The Swish activation function.

inverted bottleneck convolution modules are used to further extract image features. Finally, after 1×1 convolution and global average pooling operations, the classification results can be obtained in the fully connected layer. After each convolution operation in the network, batch normalization is performed. The activation function used in this network is Swish. The overall architecture of EfficientNetB0 is shown in Figure 4.

The core component of the network is a mobile inverted bottleneck convolution module (MBConv). Figure 5 shows the framework of this module. The design of this module is inspired by inverted residual and residual structure. Before performing on 3×3 or 5×5 convolution, the dimension of images is increased via 1×1 convolution in order to extract more feature information. The Squeeze-and-Excitation (SE) [35] model is added after 3×3 or 5×5 convolution operation to further improve performance. Finally, 1×1 convolution operation is used to reduce the dimension, and a residual connection is added.

The Squeeze-and-Excitation block compresses the feature map, performs a global average pooling operation in the direction of the channel dimension, and performs an excitation operation on the global feature to learn the relationship in each channel. Then, it obtains the weights of different channels through the sigmoid activation function. Finally, the weights multiply the original feature map to get

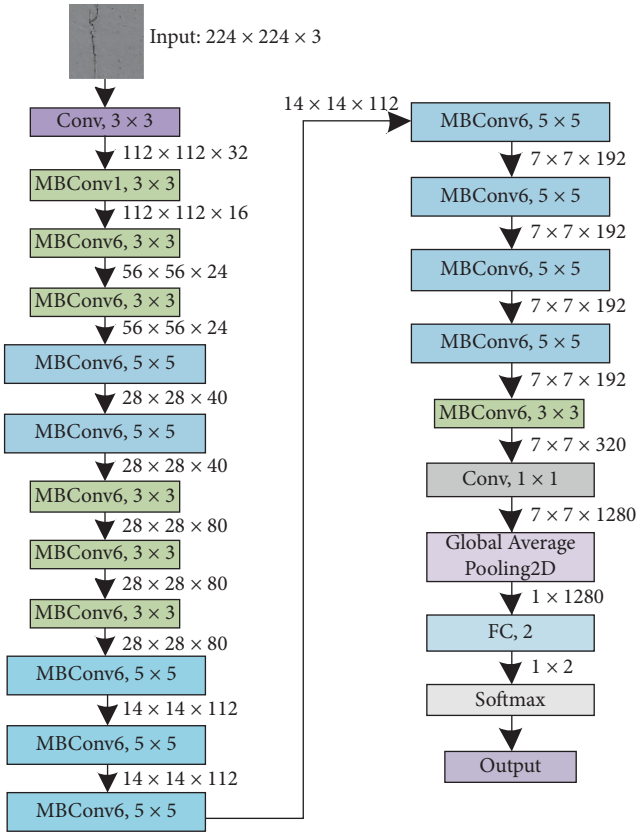


Figure 4: The EfficientNetB0 network architecture.

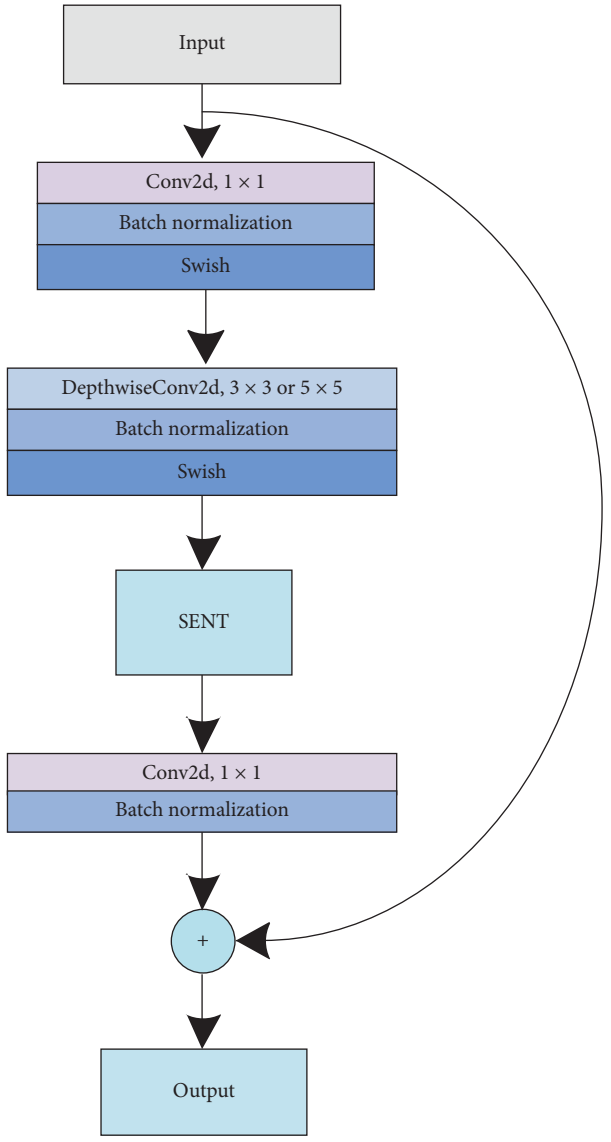


Figure 5: Illustration of MBConv's framework.

the final feature. The block allows the model to pay more attention to the channel features which has the most information, while suppressing those that are not important. Figure 6 illustrates the detail of the Squeeze-and-Excitation block.

3.4. Loss Function and Adam

3.4.1. Loss Function. The loss function is mainly used to evaluate the effect of the model. For a large amount of information, the machine discovers the laws through autonomous learning and makes predictions. The loss function is used to measure the degree of deviation between the predicted result and the actual value. During the network training process, the function is continuously updated until the best fitting result is found to reduce the error.

The cross-entropy loss function, also known as the Softmax function, is widely used to measure the gap between the predicted value and the actual value when the convolutional neural network deals with classification problems. The Softmax function is defined as follows:

$$L_{\text{softmax loss}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{a_k}}{\sum_j e^{a_j}}\right), \tag{7}$$

where N represents the number of neurons in the output layer. a_k is the input signal.

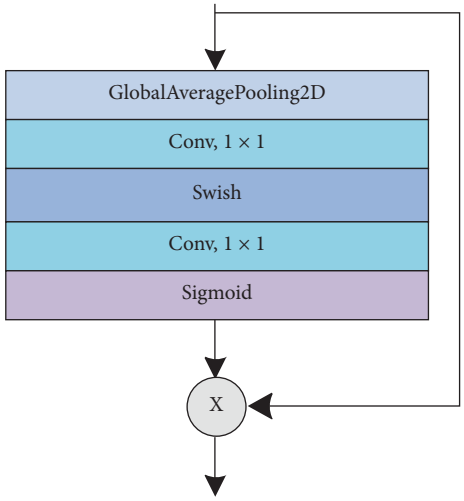


Figure 6: Squeeze-and-excitation block.

3.4.2. Adam. When training the model effectively and making accurate predictions, the internal parameters of the model play a significant role. This is why we ought to choose a suitable optimizer to update network parameters to approximate or reach the optimal value. The optimization algorithm helps to minimize the loss function, update the model parameters, and finally reach converge. In this article, the Adam algorithm is utilized to update weights.

Kingma [36] utilizes exponentially moving averages to estimate the moments:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \cdot g_t, \tag{8}$$

$$\nu_t = \beta_2 \nu_{t-1} + (1 - \beta_2) \cdot g_t^2, \tag{9}$$

where m and ν are moving averages, g_t is gradient, and β_1 and β_2 are the decay rates of moment estimate (setting to 0.9 and 0.999, respectively).

Then, we do bias correction:

$$\widehat{m_t} = \frac{m_t}{1 - \beta_1^t}, \tag{10}$$

$$\widehat{\nu_t} = \frac{\nu_t}{1 - \beta_2^t}. \tag{11}$$

The final formula for weight update is

$$\omega_{t+1} = \omega_t - lr \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{\nu_t}} + \epsilon}, \tag{12}$$

where lr is the learning rate and ϵ is the hyperparameter (setting to $1e-3$).

3.5. Training. The convolutional neural network models used in this article use transfer learning methods to detect concrete surface crack. Specifically, the weight of the model is trained on the ImageNet and saved and then transferred to our model. Therefore, our model has a higher starting point, which greatly saves training time and obtains better performance.

All the experiments in this paper are performed on TensorFlow in Windows system: hardware settings: CPU: Intel (R) Core (TM) i7CPU@3.20 GHz, RAM: 16G, and GPU: NVIDIA GTX1080Ti.

Pretrained model is a model that has been trained and saved in advance on a large dataset. In order to realize the detection of concrete surface cracks, we need to retrain the pretrained convolutional neural network model. In addition, the last fully connected layer of the original model is replaced by a new fully connected layer. The specific experimental steps are as follows:

Step 1: data loading

Importing concrete surface crack images. A batch of data is randomly loaded from the training set (batch size: 16) for subsequent data processing.

Step 2: image preprocessing

We use built-in function in TensorFlow to adjust the size of input image to the fixed size of the model. Then, do data augmentation via image flip, translation, rotation, and other operations. It is worth noting that, due to the use of TensorFlow's built-in function, a large number of pictures generated by data augmentation will not be saved to the local computer. All pictures are online.

Step 3: define the structure of the crack detection model

The pretrained model (such as EfficientNetB0) is loaded and fine-tuned. We remove the last fully connected layer and replace it with a custom layer. In this article, the number of classifications in the custom layer is set to 2. The value of weights in other layers will not change.

Step 4: compile the model and start training

Before training the model, it is necessary to specify the hyperparameters related to the network structure and select the appropriate optimization strategy. In this experiment, a random batch training method is used to train the neural network. The dataset is randomly shuffled before each epoch of training to ensure that the same batch of data in each epoch of model training is different, which can increase the rate of model convergence. The learning rate plays a significant role in the training of model. Choosing an appropriate learning rate can speed up the model's convergence speed; on the contrary, it may cause the loss value of objective function to explode. Since the transfer learning method is used, the network has converged on the original dataset, so the initial learning rate in this experiment is set to $5e-7$. When the validation loss does not decrease for two consecutive times, the model learning rate will be reduced to half. We select the adaptive learning rate optimization algorithm Adam and use the cross-entropy loss function to guide the model training. The initialization method of weights is as follows: initialize the weights of newly added fully connected layer randomly and initialize the remaining weights with the pretrained weights. Figure 7 shows the process of model adjustment and weights initialization.

In order to solve the problem of how long to train the model, we adopt an early stopping strategy to avoid overtraining the network. During the training process, we monitor the validation loss. Once the validation loss drops less than $1e-3$ for 30 epochs, the model will stop training.

Step 5: test the performance of the model.

Test the performance of the model on the test dataset. In addition, we also select 1000 crack and 1000 nocrack images of concrete bridge decks from the SDNET2018 dataset to construct a completely different dataset so that we can further evaluate the detection performance of the model.

4. Experimental Results and Analyses

4.1. Experimental Results and Evaluation Index. When doing image classification tasks, in order to evaluate the

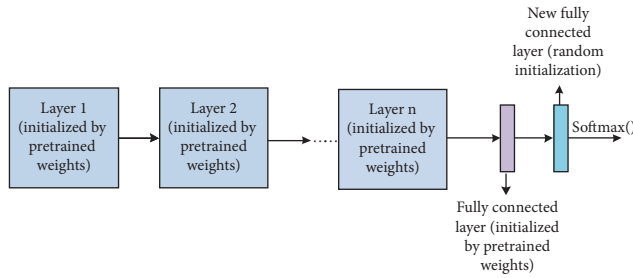


Figure 7: Transfer learning.

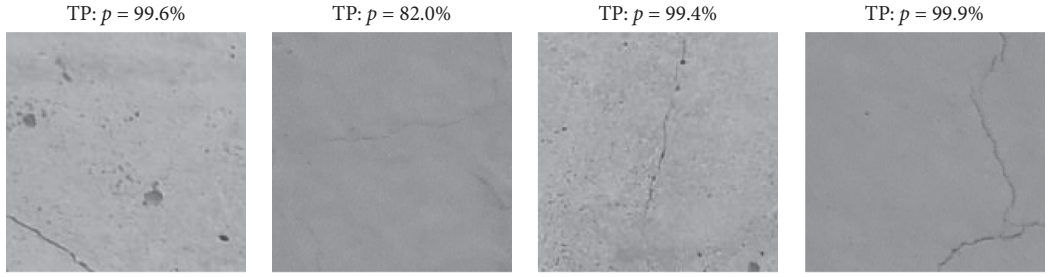


Figure 8: Detection of crack. TP denotes true positive.

performance of different algorithms, some evaluation metrics need to be selected. Accuracy refers to the ratio of number of correctly predicted crack and noncrack images to the total number of input images. Precision can be understood as the number of correctly predicted crack images divided by the number of crack images predicted by the classifier. Recall is the percentage of the number of correctly predicted crack images to the total number of cracked images. $F_{1-\text{score}}$ is the harmonic mean of precision and recall. Accuracy, Precision, Recall, and $F_{1-\text{score}}$ are defined as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \tag{13}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{14}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{15}$$

$$F_{1-\text{score}} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{16}$$

where TP and TN mean images with crack and without crack, which are correctly classified. FP and FN mean images with crack and without crack which are wrongly classified.

Figures 8 and 9 show the images, together with the respective probability of correct classification.

4.2. Comparative Experiments. In order to verify the performance of the model, we compare the proposed model with other classic convolutional neural networks on the same dataset. Figure 10 shows the change in loss and accuracy during the training and validation process. The number of parameters of each model can be seen in Table 1. The results of different methods are compared in Table 2. Table 3 compares the training time of four models. Table 4 shows the size of four models. We can see that, in contrast to the other three models, MobileNetV2 [37] has the smallest number of parameters in the task of detecting concrete surface cracks, but its test accuracy is obviously lower than the other three models. Although the accuracy of EfficientNetB0 on the test dataset is slightly lower than DenseNet201 [38] (0.21%), its model size is 3.5x smaller and 2.5x faster (average training time of each epoch). Its parameters are reduced by 77.89% at the same time. EfficientNetB0 achieves a good balance among accuracy, model size, and training speed. In terms of crack detection task, the model is quite efficient. It can also be seen from Table 2 that when tested on a dataset which is quite different from the training dataset, the performance of the network drops a little. This drop is mainly caused by the changes in background conditions of the images in the dataset, and some image features have not been well learned by the network.

It is worth noting that, in Figure 10, the accuracy of validation during the training and the validation process is slightly higher than that of the training. Two reasons can account for this phenomenon. On the one hand, we use the transfer learning method to train the model. The network was initialized with pretrained weights (pretrained on ImageNet). Therefore, the model has a better feature extraction ability and features are more effective. On the other hand, this phenomenon results from the use of dropout since its behavior during training and validation is different. Dropout forces the neural network to become a very large set

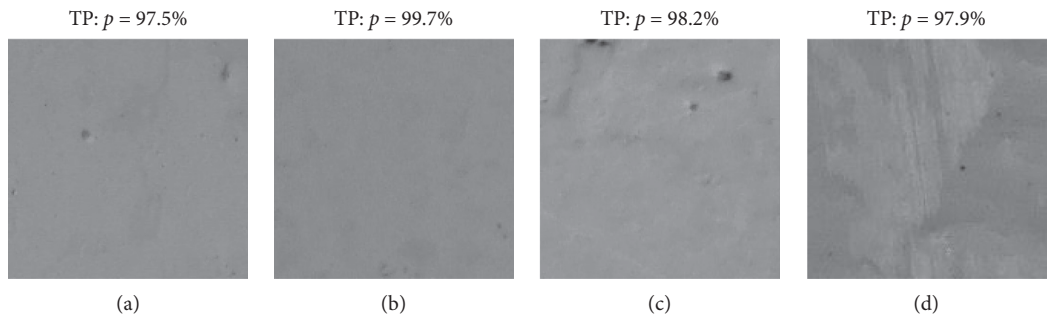


Figure 9: Detection of noncrack. TN denotes true negative.

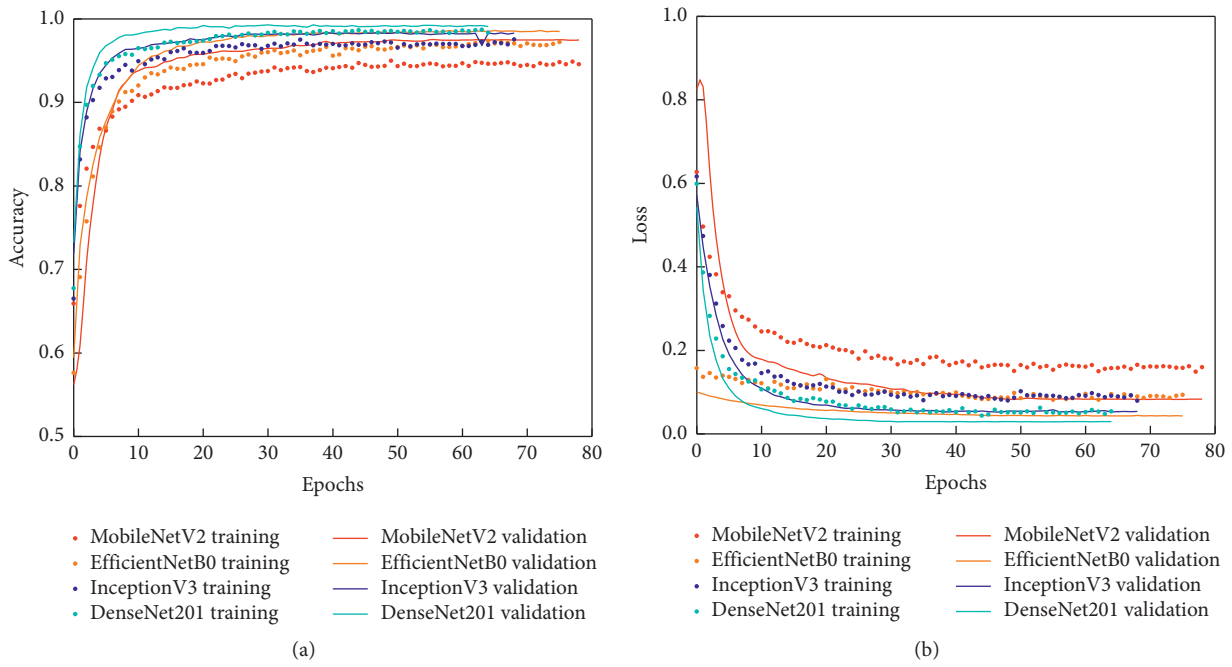


Figure 10: Accuracy (a) and loss (b) during training and validation.

Table 1: Parameters of different models.

Recognition model	Number of parameters
MobileNetV2	2,260,546
EfficientNetB0	4,052,126
DenseNet201	18,325,826
InceptionV3	21,806,882

Table 2: Comparison of four models' crack detection results.

Recognition model	Accuracy	Precision	Recall	F_1 score	Accuracy (tested on different datasets)
MobileNetV2	0.9782	0.9821	0.9740	0.9781	0.9655
EfficientNetB0	0.9911	0.9878	0.9945	0.9912	0.9737
DenseNet201	0.9932	0.9892	0.9973	0.9932	0.9749
InceptionV3	0.9898	0.9891	0.9904	0.9898	0.9715

of weak classifiers, which means that a single classifier does not have too high classification accuracy, and only when we connect them together will they become more powerful. During training, dropout cuts off the random set of these classifiers, so the training accuracy will be affected. During validation, dropout will automatically turn off and allow all

Table 3: Comparison of four models' training time.

Model	Training epochs	Total training time	Average training time of each epoch (s)
MobileNetV2	79	2 h 1 m 47 s	92.49
EfficientNetB0	76	3 h 14 m 32 s	153.58
DenseNet201	65	9 h 47 m 8 s	541.97
InceptionV3	69	3 h 46 m 14 s	196.72

Table 4: Size of four models.

Model	Size (M)
MobileNetV2	8.8
EfficientNetB0	15.7
DenseNet201	70.6
InceptionV3	83.5

weak classifiers in the neural network to be used, so the validation accuracy is improved.

5. Conclusions

In this paper, a concrete surface crack detection model based on transfer learning and convolutional neural network is proposed. EfficieneNetB0 is a highly effective convolutional neural network. The last fully connected layer is replaced by a new fully connected layer with a classification number of 2. The newly added fully connected layer is initialized randomly, and the remaining weights are initialized with pretrained weights. Finally, by comparing with other models, the results show that our model achieves a good balance among accuracy, model size, and training speed. In addition, when tested on crack images taken from other places, our model also shows good performance and generalization capability. Our model is an efficient crack detection model, which is a good choice for areas with limited computing resources. Traditional crack detection mostly pays attention to how to identify the cracks in the image. In addition, it is also important to characterize the severity of the cracks, which is an area that is often overlooked. We will be devoted to this work in future research.

Data Availability

The codes used in this paper are available from the author upon request.

Conflicts of Interest

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

Acknowledgments

This study was funded by the National Natural Science Foundation of China (grant no. 51579089).

References

[1] D. G. Aggelis, N. Alver, and H. K. Chai, "Health monitoring of civil infrastructure and materials," *Scientific World Journal*, vol. 2014, Article ID 435238, 2 pages, 2014.

[2] I.-H. Kim, H. Jeon, S.-C. Baek, W.-H. Hong, and H.-J. Jung, "Application of crack identification techniques for an aging concrete bridge inspection using an unmanned aerial vehicle," *Sensors*, vol. 18, no. 6, p. 1881, 2018.

[3] T. Liu, H. Huang, and Y. Yang, "Crack detection of reinforced concrete member using rayleigh-based distributed optic fiber strain sensing system," *Advances in Civil Engineering*, vol. 2020, Article ID 8312487, 11 pages, 2020.

[4] T. Yamaguchi, S. Nakamura, R. Saegusa, and S. Hashimoto, "Image-based crack detection for real concrete surfaces," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 3, no. 1, pp. 128–135, 2008.

[5] Y.-C. Tsai, V. Kaul, and R. M. Mersereau, "Critical assessment of pavement distress segmentation methods," *Journal of Transportation Engineering*, vol. 136, no. 1, pp. 11–19, 2010.

[6] D. Zhang, Q. Li, Y. Chen, M. Cao, L. He, and B. Zhang, "An efficient and reliable coarse-to-fine approach for asphalt pavement crack detection," *Image and Vision Computing*, vol. 57, pp. 130–146, 2017.

[7] A. Ayenu-Prah and N. Attoh-Okine, "Evaluating pavement cracks with bidimensional empirical mode decomposition," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. 1–7, 2008.

[8] P. Subirats, J. Dumoulin, V. Legeay, and D. Barba, "Automation of pavement surface crack detection using the continuous wavelet transform," in *Proceedings of the International Conference on Image Processing*, pp. 3037–3040, IEEE, Atlanta, GA, USA, October 2006.

[9] L. Ying and E. Salari, "Beamlet transform-based technique for pavement crack detection and classification," *Computer-Aided Civil and Infrastructure Engineering*, vol. 25, no. 8, pp. 572–580, 2010.

[10] A. Hizukuri and T. Nagata, "Development of a classification method for a crack on a pavement surface images using machine learning," in *Proceedings of the Thirteenth International Conference on Quality Control by Artificial Vision*, Tokyo, Japan, May 2017.

[11] P. P. Acharjya, R. Das, and D. Ghoshal, "Study and comparison of different edge detectors for image segmentation," *Global Journal of Computer Science and Technology*, vol. 12, no. 13, 2012.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105, Lake Tahoe, NV, USA, December 2012.

[14] M. Valueva, N. Nagornov, P. Lyakhov, G. Valuev, and N. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Mathematics and Computers in Simulation*, vol. 177, 2020.

[15] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in

Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3708–3712, IEEE, Phoenix, AZ, USA, September 2016.

[16] L. Pauly, D. Hogg, R. Fuentes, and H. Peel, "Deeper networks for pavement crack detection," in *Proceedings of the 34th ISARC*, IAARC, Taipei, Taiwan, pp. 479–485, June 2017.

[17] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, and H. Omata, "Road damage detection using deep neural networks with images captured through a smartphone," 2018, https://arxiv.org/abs/1801.09454.

[18] H. Xu, X. Su, Y. Wang, H. Cai, K. Cui, and X. Chen, "Automatic bridge crack detection using a convolutional neural network," *Applied Sciences*, vol. 9, no. 14, p. 2867, 2019.

[19] Y. Li, Z. Han, H. Xu, L. Liu, X. Li, and K. Zhang, "YOLOv3-lite: a lightweight crack detection network for aircraft structure based on depthwise separable convolutions," *Applied Sciences*, vol. 9, no. 18, p. 3781, 2019.

[20] Z. Tong, J. Gao, and H. Zhang, "Recognition, location, measurement, and 3D reconstruction of concealed cracks using convolutional neural networks," *Construction and Building Materials*, vol. 146, pp. 775–787, 2017.

[21] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, and X. Yang, "Automatic pixel-level crack detection and measurement using fully convolutional network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 12, pp. 1090–1109, 2018.

[22] J. Zhu and J. Song, "An intelligent classification model for surface defects on cement concrete bridges," *Applied Sciences*, vol. 10, no. 3, p. 972, 2020.

[23] L. Deng, H.-H. Chu, P. Shi, W. Wang, and X. Kong, "Region-based CNN method with deformable modules for visually classifying concrete cracks," *Applied Sciences*, vol. 10, no. 7, p. 2528, 2020.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 91–99, 2015.

[25] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: object detection via region-based fully convolutional networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 379–387, Barcelona, Spain, December 2016.

[26] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, Honolulu, HI, USA, July 2017.

[27] S. Li and X. Zhao, "Image-based concrete crack detection using convolutional neural network and exhaustive search technique," *Advances in Civil Engineering*, vol. 2019, 12 pages, 2019.

[28] S. Dorafshan, R. J. Thomas, and M. Maguire, "SDNET2018: an annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks," *Data in Brief*, vol. 21, pp. 1664–1668, 2018.

[29] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017, https://arxiv.org/abs/1712.04621.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[31] C. Szegedy, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Boston, MA, USA, June 2016.

[33] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, https://arxiv.org/abs/1710.05941.

[34] M. Tan and Q. V. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," 2019, https://arxiv.org/abs/1905.11946.

[35] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Salt Lake City, UT, USA, June 2018.

[36] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/pdf/1412.6980.pdf.

[37] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Salt Lake City, UT, USA, June 2018.

[38] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Honolulu, HI, USA, July 2017.