applied sciences

MDPI

Article

Accurate Classification of Tunnel Lining Cracks Using Lightweight ShuffleNetV2-1.0-SE Model with DCGAN-Based Data Augmentation and Transfer Learning

Ningyu Zhao [1,2], Yi Song [1,*], Ailin Yang [1], Kangping Lv [1], Haifei Jiang [1,2] and Chao Dong [1]

[1] School of Civil Engineering, Chongqing Jiaotong University, No. 66, Xuefu Road, Nan'an District, Chongqing 400074, China; zhny@cqjtu.edu.cn (N.Z.); nonhan@outlook.com (A.Y.); kangpl@mails.cqjtu.edu.cn (K.L.); jhfcivil@cqjtu.edu.cn (H.J.); dongchao@mails.cqjtu.edu.cn (C.D.)
[2] State Key Laboratory of Mountain Bridge and Tunnel Engineering, No. 66, Xuefu Road, Nan'an District, Chongqing 400074, China
* Correspondence: yisong@mails.cqjtu.edu.cn

Abstract: Cracks in tunnel lining surfaces directly threaten structural integrity; therefore, regular inspection of cracks is essential. Lightweight convolutional neural networks (LCNNs) have recently offered a promising alternative to conventional manual inspection. However, the effectiveness of LCNNs is still adversely affected by the lack of sufficient crack images, which limits the potential detection performance. In this paper, transfer learning was used to optimize deep convolutional generative adversarial networks (DCGANs) for crack image synthesis to significantly improve the accuracy of LCNNs. In addition, an improved LCNN model named ShuffleNetV2-1.0-SE was proposed, incorporating the squeeze–excitation (SE) attention mechanism into ShuffleNetV2-1.0 and realizing highly accurate classification results while maintaining lightness. The results show that the DCGAN-based data enhancement method can significantly improve the classification accuracy of ShuffleNetV2-1.0-SE for tunnel lining cracks. ShuffleNetV2-1.0-SE achieves an accuracy of 98.14% on the enhanced dataset, which is superior to multiple advanced LCNN models.

Keywords: deep learning; lightweight convolutional neural network; deep convolutional generative adversarial networks; tunnel lining; defect inspection; crack classification; shuffleNetV2

Citation: Zhao, N.; Song, Y.; Yang, A.; Lv, K.; Jiang, H.; Dong, C. Accurate Classification of Tunnel Lining Cracks Using Lightweight ShuffleNetV2-1.0-SE Model with DCGAN-Based Data Augmentation and Transfer Learning. *Appl. Sci.* **2024**, *14*, 4142. https://doi.org/10.3390/ app14104142

Academic Editor: Laurent Daudeville

Received: 17 April 2024
Revised: 8 May 2024
Accepted: 10 May 2024
Published: 13 May 2024

1. Introduction

Tunnels can improve the operational efficiency of transportation systems. However, most tunnels operate in harsh environments, which lead to the inevitable cracking of the tunnel lining surfaces [1]. As cracks in tunnel linings threaten structural integrity and pose safety risks, timely crack inspection is essential for maintaining the structural safety of tunnel linings [2]. However, conventional crack detection methods rely on manual inspection, which is not only time-consuming and laborious, but its accuracy is also limited by the subjective experience of the inspector [3]. In addition, the increasing number of tunnels in operation requires more automated and reliable inspection methodologies. Image recognition methodology has recently offered a promising alternative for structural crack monitoring, as it can potentially replace manual inspection and automate the realization of the crack detection process [4,5].

Image recognition methodology for crack detection can be categorized into model-driven and data-driven technologies [6,7]. A model-driven method is represented by traditional digital image methods, which design algorithms around a specific recognition target object to extract features and separate the target object from the background in the digital image. These methods typically involve smoothing the image with mean, median or Gaussian filters, and algorithms based on Otsu thresholding segmentation and edge detection enhancements are often used for crack detection [8–10]. However,

the above methods may suffer from low accuracy and generalization in tunnel scenes with interference backgrounds and variable illumination owing to the limitations of the designer's experiences.

Data-driven methods based on deep learning, especially convolutional neural networks (CNNs) [11–14], have outperformed model-driven methods in crack detection, as demonstrated by numerous studies [5,15,16]. Zhang et al. used CNNs to classify road crack image patches captured by smartphones, and the results revealed that CNNs' detection performance was better than the traditional manual feature extraction method [17]. Gao et al. applied transfer learning with VGGNet to various structural damages, revealing the potential of deep learning in structural damage detection [18]. Han et al. achieved a classification accuracy of up to 98.26% on a small sample dataset using transfer learning optimized AlexNet [19]. However, the above models require high computational power in practical applications, which is challenging to apply on mobile devices with limited computational resources.

Recently, researchers have made progress in developing lightweight convolutional neural networks (LCNNs); various LCNNs have achieved high accuracy and low computational cost in image classification. For example, Google proposed MobileNetV1 [20] in 2017, incorporating depthwise separable convolution for feature extraction to reduce computational resources. A subsequent study on MobileNetV2 [21] introduced an inverted residual block, and further enhancements were made in MobileNetV3 [22] by utilizing the hard-swish activation function and the squeeze–excitation (SE) attention mechanism. Zhang et al. proposed ShuffleNetV1 [23], which employed a channel shuffle for additional computational efficiency. Then, they proposed a lightweight network design guideline, which was utilized for developing ShuffleNetV2 [24]. EfficientNet [25], another impressive design by Tan et al., investigated the effects of image input resolution, network depth and network width rationalization on the classification results. Thanks to the development of LCNNs, they have emerged as promising alternatives for crack detection because of their potential use in mobile devices [26]. Hou et al. proposed MobileCrack, an LCNN model for pavement classification with only a quarter of the parameters of MobileNet [27]. Chen et al. combined MobileNetV3-large with the CBAM attention mechanism to achieve highly accurate localization and classification of different crack images [28]. All of the above research has fully demonstrated the potential of LCNNs in crack detection.

Crack image classification is a fundamental task in image recognition and is the basis for the subsequent implementation of crack target detection and semantic segmentation. However, the evaluation of LCNNs for image classification of tunnel lining cracks is still lacking. As a data-driven method, LCNNs based on deep learning depend on the quantity and quality of training data, and limited samples can lead to overfitting or low accuracy of LCNNs. Therefore, researchers have explored image augmentation methods. Recently, unsupervised learning-based methods for generating virtual crack images have been proven by several studies [29–31] to be a worthwhile method. For example, Pei et al. proposed a method combining variational auto-encode (VAE) and deep convolutional generative adversarial networks (DCGANs) to generate virtual pavement crack images, which outperformed traditional data augmentation methods [32]. Jin et al. used DCGAN to generate crack-labeled images and pixel-to-pixel models to generate corresponding virtual crack images, providing a new way to reduce the cost of crack labeling [33]. Zhou et al. proposed WGAN-RE model for generating images to address the problem of data scarcity in tunnel lining crack detection. However, the studies above ignored the interference background of the tunnel [34], and the extension of crack image generation under interference backgrounds remains underexplored.

Based on the literature review above, this paper aims to overcome the challenge in tunnel lining crack classification: the scarcity of datasets with tunnel lining crack images under various interference backgrounds. Meanwhile, an LCNN model named ShuffleNetV2-1.0-SE was proposed to classify tunnel lining cracks accurately.

The main contributions of this paper can be summarized as follows:

(1) A novel dataset for classification was constructed by manually taking field photographs of tunnel lining cracks under various interference backgrounds.
(2) A publicly available concrete dataset with a small sample size was used to pretrain DCGAN, resulting in higher quality synthetic images of tunnel lining cracks.
(3) The ShuffleNetV2-1.0 network was improved by incorporating a squeeze–excitation (SE) attention module, which improved its crack feature extraction and balanced its accuracy and speed.

The rest of this paper is organized as follows. Section 2 provides a method overview, including the overall steps and the specific theory for each step. Section 3 shows the experimental configuration and detail. Section 4 shows the results of our model and compares them with other LCNNs. Section 5 outlines the conclusions and outlook.

2. Materials and Methods

2.1. Dataset Production

2.1.1. Basic Dataset Production

A total of 225 images with tunnel lining cracks of four road tunnels were collected, and all images were 3648 × 2736 pixels. The collected images were cropped to 224 × 224 pixels. The images were manually screened, and a classification dataset was obtained, including 2600 background and 2600 crack images. The above production process and typical images are shown in Figure 1.

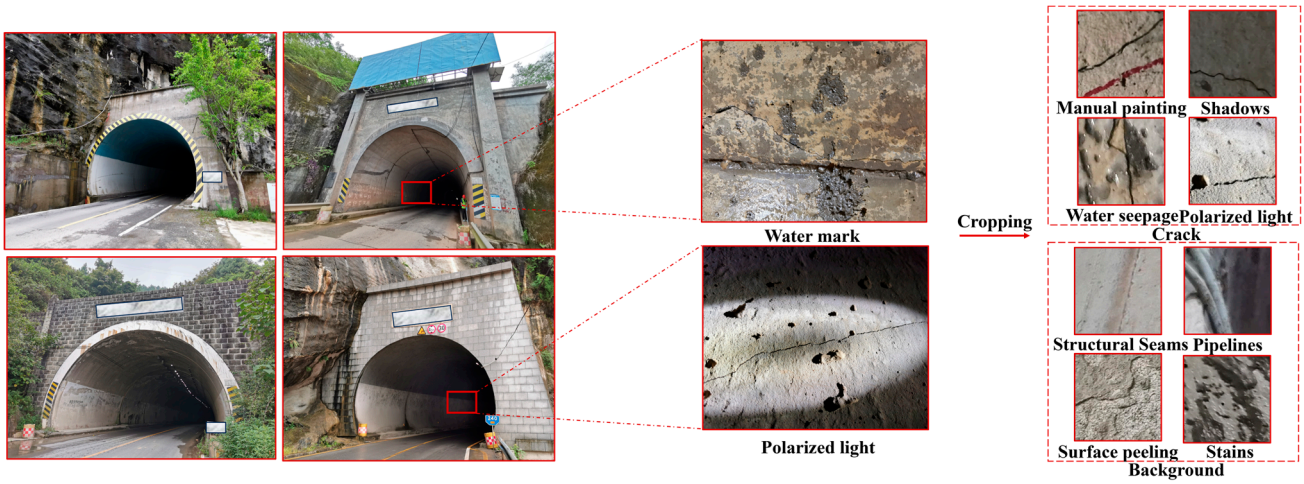


Figure 1. Images in the homemade dataset.

The tunnel lining is made of concrete but may have paint and marks from manual inspection or stains due to long-term water seepage, making image recognition more difficult. The interference background considered in this paper consists of two main aspects.

On the one hand, we chose crack images under interference backgrounds for cracks—for example, "manual painting", "water seepage", "shadows" and "polarized light", as shown in Figure 1. On the other hand, some background images similar to the "structural seams", "pipelines", "surface peeling" and "stains", as shown in Figure 1, were also used for training, considering that in the actual tunnel scenario, it is possible to encounter these features.

2.1.2. Data Augmentation Using IDG Methods

Data augmentation was implemented using the image data generator (IDG) [3] method in Keras to increase the diversity and size of our image dataset. Randomly rotated, flipped, shifted and cropped images were used to generate new samples. The process is shown in Figure 2, and after augmentation, 5200 images of cracks and backgrounds were obtained, respectively.

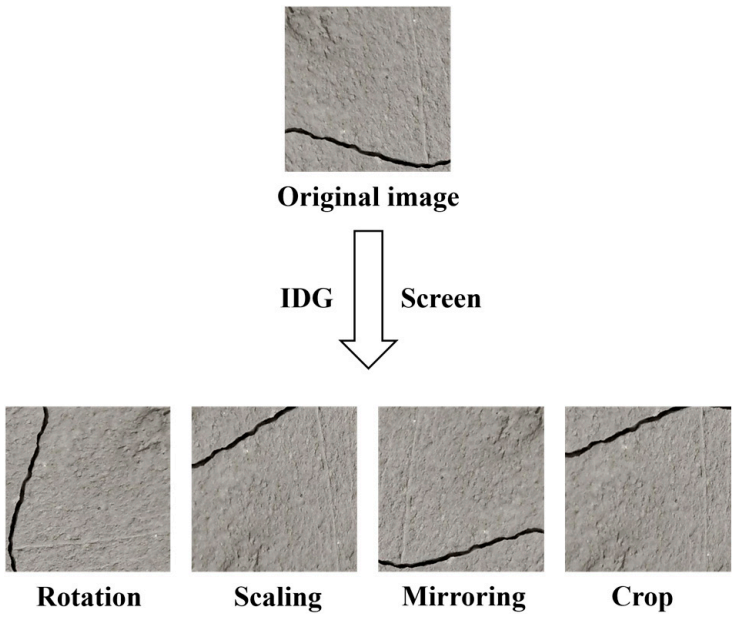
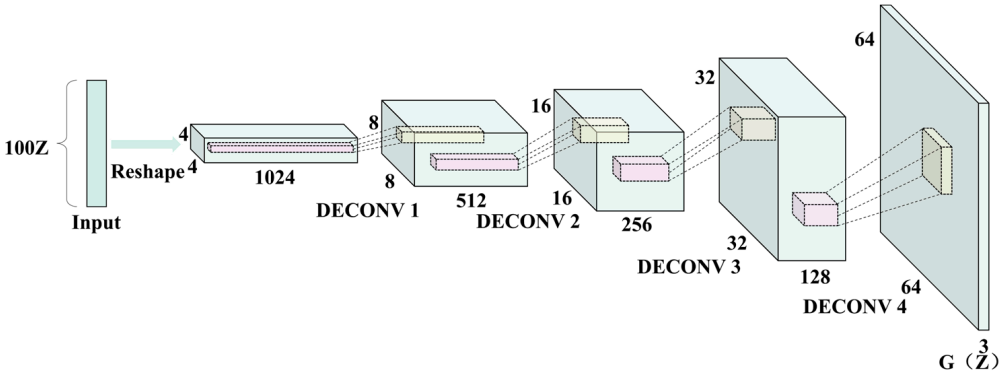


Figure 2. Schematic of IDG.

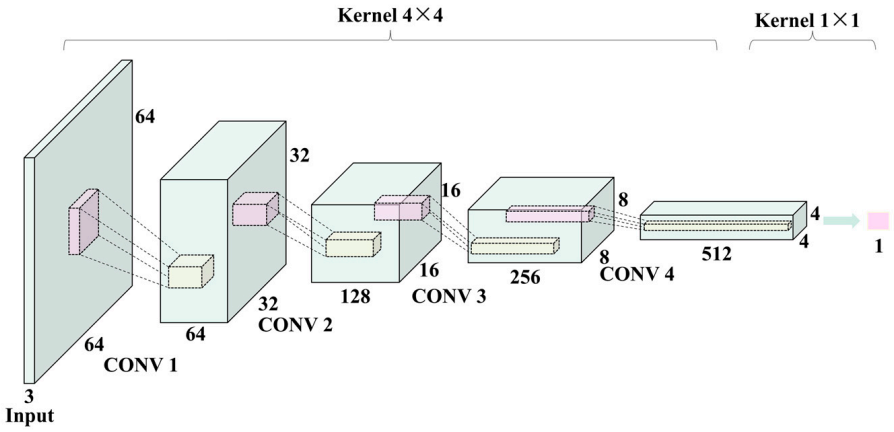
2.1.3. Data Augmentation Using DCGAN

DCGAN [35] is a variant of Generating Adversarial Networks (GANs) [36]. As Figure 3 shows, a generator and a discriminator form the structure of DCGAN. The generator transforms a random noise with a length of 100 into an image using a fully connected layer and a reshaping operation, followed by five transposed convolution layers and a Tanh activation function. The discriminator, whose architecture is similar to that of a CNN, downsamples the input image and extracts the features; then, it uses a fully connected layer to determine whether the image is an actual image of a crack. The generator updates the image until the discriminator cannot distinguish it from the images in the training dataset.



(**a**)

Figure 3. *Cont.*



(**b**)

Figure 3. DCGAN architecture [35]. (**a**) DCGAN generator architecture; (**b**) DCGAN discriminator architecture.

2.1.4. Dataset Overview

Four different datasets were used to evaluate different data augmentation methods, as summarized in Table 1. Dataset A was the original dataset. Datasets B1 and B2 were augmented from Dataset A using two data augmentation techniques, IDG and DCGAN, respectively. Dataset C was a larger dataset generated by DCGAN from Dataset A.

Table 1. Overview of the datasets.

Dataset	Numbers		Augmentation
	Crack	Background	
A	2600	2600	
B1	5200	5200	IDG
B2	5200	5200	DCGAN
C	7800	7800	DCGAN

2.2. ShuffleNetV2-1.0-SE Model

2.2.1. ShuffleNetV2 Architecture

ShuffleNetV2 [24] uses group convolution for feature extraction, similar to ShuffleNetV1 [23]. Unlike depthwise separable convolution, which applies a depthwise convolution followed by a 1×1 point convolution to stitch the results together, group convolution performs only one convolution and concatenates the results of different groups. This further reduces the computational complexity of the model and increase the semantic information exchange between different groups, which can improve the feature extraction ability of the network. In addition, ShuffeNetV2 employs a channel shuffle to enhance inter-channel communication and prevent information loss, similar to ShuffleNetV1, as shown in Figure 4.

As shown in Figure 5, the ShuflleNetV2 architecture utilizes a new type of stacking block, which contains two parts. The one shown in Figure 5a is the standard block, which is stacked repeatedly to form the overall framework of the model. In contrast, the block shown in Figure 5b is utilized for downsampling.

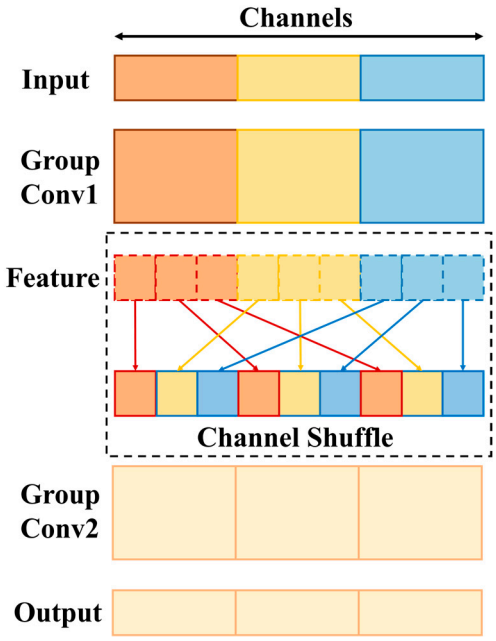


Figure 4. Schematic of channel shuffle [23].

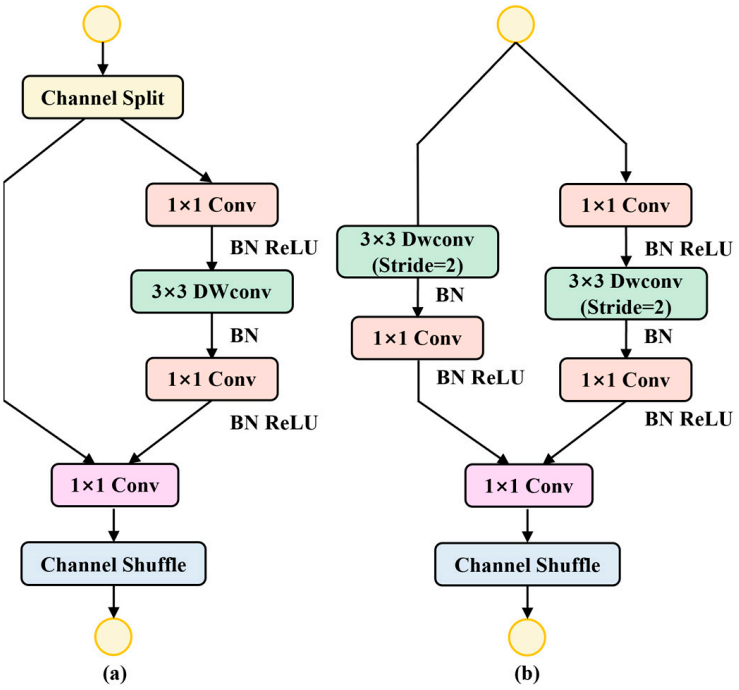


Figure 5. Schematic of (**a**) Standard block, (**b**) Block for downsampling [24].

2.2.2. SE Attention Mechanism

The SE attention mechanism [37] improved communication between the channel dimensions and adjusted important feature channel weights. Figure 6 shows the structure of the SE module. For an input feature map $U \in H \times W \times C$, where H, W and C are the height, width and number of channels, respectively, the SE module applies global average pooling (GAP) to reduce the channel dimension to $Z \in 1 \times 1 \times C$. Then, the excitation part learns the feature weights of each channel through two fully connected layers and assigns them to different channels accordingly.

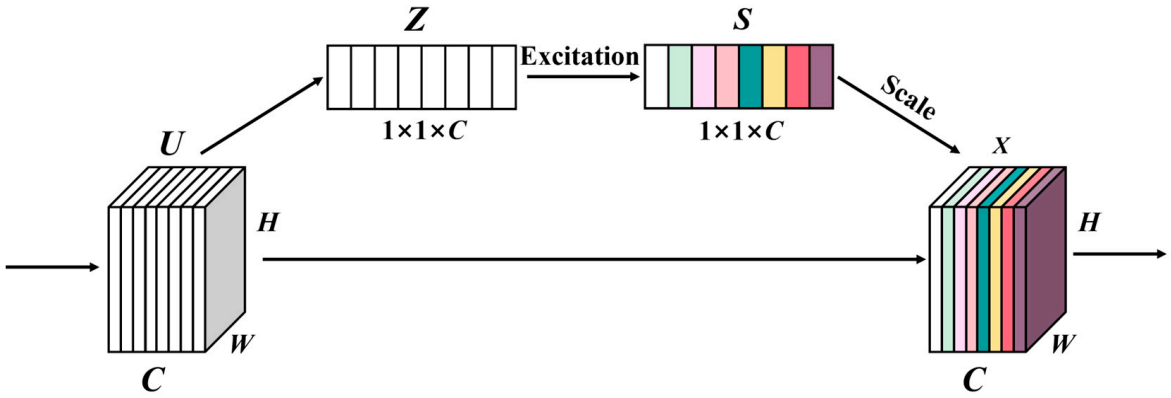


Figure 6. Schematic of SE attention mechanism [37].

2.2.3. ShuffleNetV2-1.0-SE Architecture

The ShuffleNetV2-1.0-SE model proposed in this paper is realized by introducing the SE attention mechanism to the ShuffleNetV2-1.0 architecture. As shown in Figure 7, we introduce the SE attention mechanism into the main branches in the standard block of ShuffleNetV2-1.0 and then realize the construction of the ShuffleNetV2-1.0-SE model via stacking, as shown in Table 2.

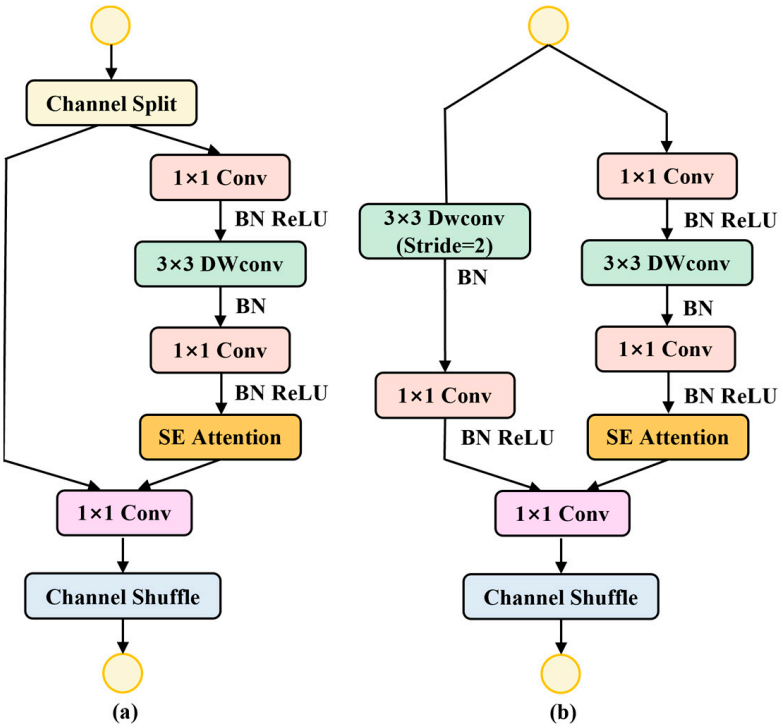


Figure 7. Schematic of the block of ShuffleNetV2-1.0-SE model. (**a**) standard block of ShuffleNetv2-1.0-SE model, (**b**) downsampled block of ShuffleNetv2-1.0-SE model.

Table 2. ShuffleNetV2-1.0-SE model architecture [24].

Layer	Output Size	KSize	Stride	Repeat	Output Channels
Image	224 × 224				3
Conv1	112 × 112	3 × 3	2	1	24
Maxpool	56 × 56	3 × 3	2		
Stage2	28 × 28		2	1	116
	28 × 28		1	3	
Stage3	14 × 14		2	1	232
	14 × 14		1	7	
Stage4	7 × 7		2	1	464
	7 × 7		1	3	
Conv5	7 × 7	1 × 1	1	1	1024
GlobalPool	1 × 1	7 × 7			
FC					2

Notes: KSize: Using the size of the convolutional kernel; Stride: The step size of the convolution operation. In the "Stage" phase, the downsampled blocks are used for the first operation and the standard blocks are repeated operation. Repeat: Number of times the operation is repeated at that stage.

2.3. Transfer Learning

Transfer learning was used to increase the accuracy of the networks when the number of samples was limited. In this paper, the DCGAN and the ShuffleNetV2-1.0-SE model were pretrained on the concrete dataset provided by Ref. [38], ensuring similarity between the datasets and providing suitable pretraining weights. Figure 8 shows the typical images of the concrete dataset from Ref. [38]. The dataset consists of 40,000 images of concrete cracks and backgrounds, each with 20,000 images. For DCGAN training, we did not split the dataset into training and validation sets. For LCNN training, the dataset was randomly split into training (80% of the data) and validation (20% of the data) sets, and training was performed for 200 epochs. The minimum loss value was used as the monitoring index. The best weights obtained during pretraining were transferred to the lining crack dataset.

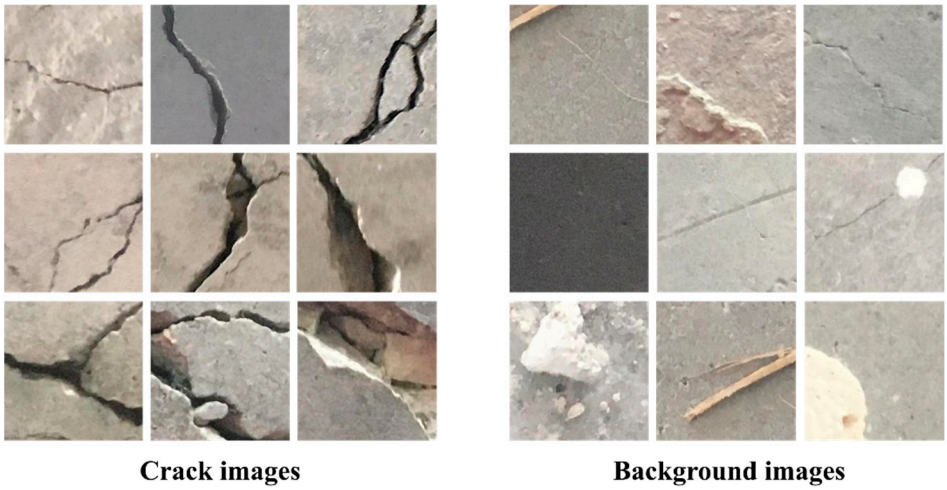


Figure 8. Images of concrete cracks.

3. Experiments and Configuration Detail

3.1. Experimental Environment for DCGAN Training

This paper used the DCGAN model to synthesize realistic tunnel crack images based on the features learned from actual data. First, the model was pretrained on a concrete dataset to obtain initial weights and then fine-tuned on a base dataset of tunnel crack images. The experiments were conducted on a computer with an Intel I5-9600KF CPU (Santa Clara, CA, USA), 16 GB of memory and an NVIDIA GeForce GTX3060 (12 GB) GPU

(Santa Clara, CA, USA) with CUDA support. Python 3.7 and Pytorch 1.7.1 were used as the programming environment and tools. The hyperparameters were set as follows: the number of epochs was 300 [29]; the initial learning rate was 0.002 [30] with adaptive decay; the batch size was 128 [32]; and the optimizer was Adam with a momentum of 0.9 [34].

3.2. Experimental Environment for LCNN Training

The cross-entropy loss function (L_{CE}) was used to measure the discrepancy between the predicted and actual labels, as defined in Equation (1). In this equation, y represents the actual label value, and $\hat{y}$ represents the predicted value.

$$L_{CE} = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \quad (1)$$

The Adam optimizer trained the networks [39] with an initial learning rate of 0.001, and the model with the lowest loss value was selected for testing. The He strategy initialized the weights of the convolutional layers [40], and the number of epochs was set to 200 for all models. The batch size was 32, and the images were sent to the networks with original resolution except Efficienetb1 and Efficienetb2.

The ShuffleNetv2-1.0-SE was compared with MobielNetV1 [20], MobilNetV2 [21], MobilNetV3 [22] and EfficieNet [25]. The alpha width factor was set to 1 for MobilNetv1, MobilNetv2 and MobilNetv3. We used 240 × 240 and 260 × 260 resolution inputs for Efficienetb1 and Efficienetb2, respectively. The experiments were run on a computer with an Intel I5-9600KF CPU, 16 GB of memory and an NVIDIA GeForce GTX3060 (12 GB) GPU with CUDA support. We used Python 3.7 and TensorFlow 2.4 as the programming environment and tools.

3.3. Evaluation Metrics

The accuracy and the F1 score were used as the primary metrics to evaluate the classification performance of the networks. These metrics are calculated based on the confusion matrix, as shown in Table 3.

In addition, the complexity of the models was measured in terms of the "Parameters", "Complexity" and "Weights". "Parameters" represents the total number of participants in the model. "Complexity" represents the model's computational complexity, commonly expressed in GFLOPs. "Weight" represents the size of the file where the model's weights are stored.

Table 3. Confusion matrix.

Image		Prediction	
		Crack	**Background**
Label	Crack	TP	FN
	Background	FP	TN

In this equation, true positive (TP) denotes the number of correctly classified crack images; false positive (FP) denotes the number of background images, which are misclassified as crack images; false negative (FN) denotes the number of crack images, which are misclassified as background images; and true negative (TN) denotes the number of correctly classified background images.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \quad (2)$$

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (5)$$

4. Results and Discussion

4.1. Dataset Augmentation Using DCGAN

Figure 9 shows the images synthesized by DCGAN at different stages of training. The initial images were influenced by the pretrained weights from the concrete dataset. It exhibited some concrete crack features in the synthetic images. After 150 epochs, DCGAN started to learn the interference features of lining cracks, but some images still lacked details or had errors. After 300 epochs, most of the images could synthesize realistic crack images. The images had high diversity and captured the main characteristics of the cracks, which could effectively increase the sample.

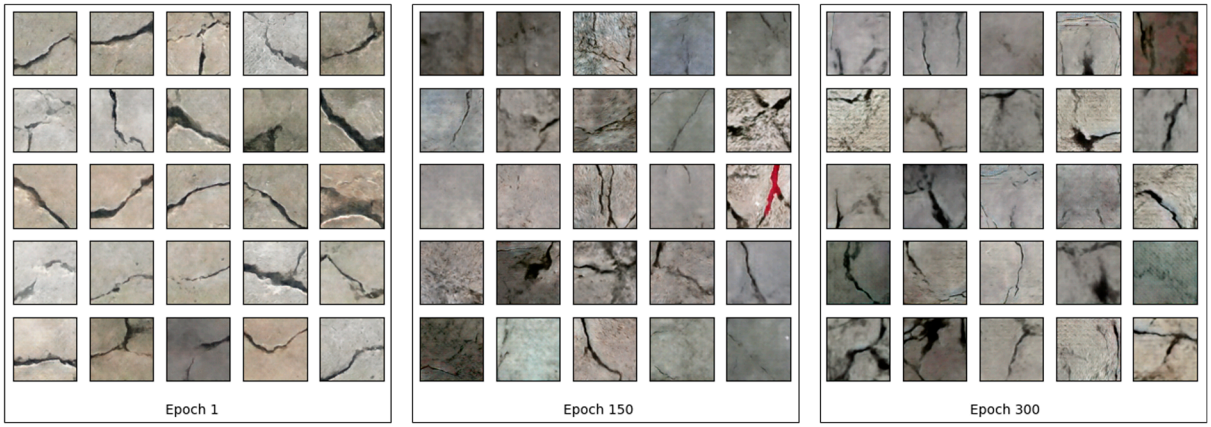


Figure 9. Images in the DCGAN augmentation dataset.

The loss curves of DCGAN under different training tactics are compared in Figure 10. It can be seen that generative adversarial image generation is an unsupervised learning technique. Suppose the transfer learning strategy is not used, the network will face a significant challenge in extracting the crack contour features, increasing the rising loss of the generator G and resulting in a continuous and significant fluctuation in the loss of the discriminator D throughout the training process.

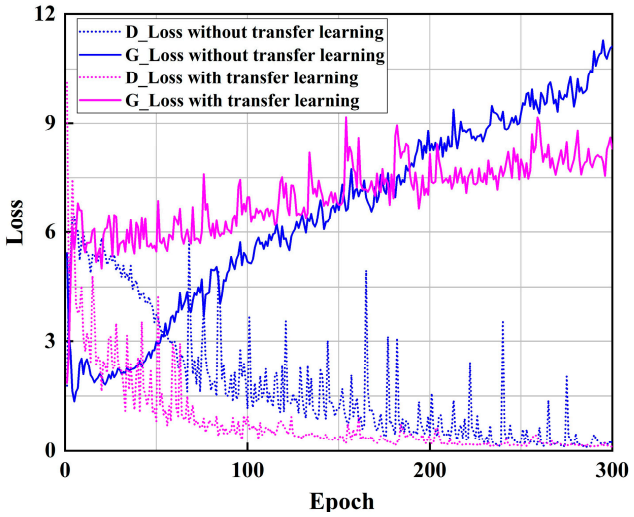


Figure 10. Comparison of DCGAN training loss curves under different training strategies.

4.2. LCNN Performance Comparison

Table 4 shows the results of different LCNNs in Dataset A with transfer learning. When all the LCNNs were trained on Dataset A, all LCNNs had a lower true positive rate for crack images (TP) than for background images, indicating that the crack features were more challenging to learn than the background images (TN).

All the LCNNs achieved an accuracy above 90% in the validation set. ShufflenetV2-0.5 had the lowest accuracy (91.73%) and complexity among all the LCNNs, and ShufflenetV2-1.5 had the highest accuracy (93.85%). However, ShuffleNetV2-1.0 only had 0.58% lower accuracy than ShuffleNetV2-1.5 despite having half the "Parameters", "Complexity" and "Weights" of ShuffleNetV2-1.5.

At the same time, ShuffleNetV2-1.0-SE increased the accuracy of ShuffleNetV2-1.0 by about 15% but still had a much lower complexity than ShuffleNetV2-1.5. ShuffleNetV2-1.0-SE achieved the highest accuracy of 94.23% with transfer learning, which was 0.96% higher than that achieved with ShuffleNetV2-1.0, suggesting that utilizing the SE attention mechanism for information interaction between the channels will help improve the accuracy of crack detection significantly.

Table 4. Performance validation results of different LCNNs trained on Dataset A.

Model	Transfer Learning	Parameters (Millions)	Complexity (GFLOPs)	Weight (MB)	Accuracy (%)	TP (%)	TN (%)
MobileNetV1	Yes	4.25	1.16	12.50	92.79	92.22	93.37
MobileNetV2	Yes	2.26	0.59	7.47	91.83	90.2	93.59
MobileNetV3-Small	Yes	1.53	0.12	6.32	92.31	90.74	94.23
MobileNetV3-Large	Yes	4.23	0.45	16.70	93.08	92.42	93.85
ShuffleNetV2-0.5	Yes	0.35	0.04	1.77	91.73	90.64	92.89
ShuffleNetV2-1.0	Yes	1.27	0.15	5.29	93.27	92.94	93.6
	No				91.06	90.51	91.62
ShuffleNetV2-1.0-SE	Yes	1.45	0.17	6.15	94.23	93.73	94.75
	No				92.12	91.32	92.94
ShuffleNetV2-1.5	Yes	2.5	0.30	10.00	93.85	93.35	94.36
ShuffleNetV2-2.0	Yes	5.38	0.59	20.90	93.17	93.09	93.26
EfficieNet-b0	Yes	4.05	0.80	15.90	92.31	90.89	94.02
EfficieNet-b1	Yes	6.58	1.29	25.70	93.27	91.82	94.82
EfficieNet-b2	Yes	7.77	1.79	30.30	93.37	92.63	94.13

Notes: "Transfer learning" represents pretraining the LCNNs on public crack dataset [38].

4.3. Comparison of Different Augmentation Methods

All LCNNs were trained on Datasets B1 (IGD-Augmentation) and B2 (DCGAN-Augmentation) simultaneously and evaluated the results, which are shown in Figure 11. Combining Table 4 and Figure 11, we can see that both augmentation methods improved the accuracy of all LCNNs. We can see that only the ShuffleNetV2 series models achieved higher accuracy after training on Dataset B2; therefore, we could not conclude that one enhancement method gave universally better results. However, we can confirm that data augmentation using DCGAN is as effective as IGD method.

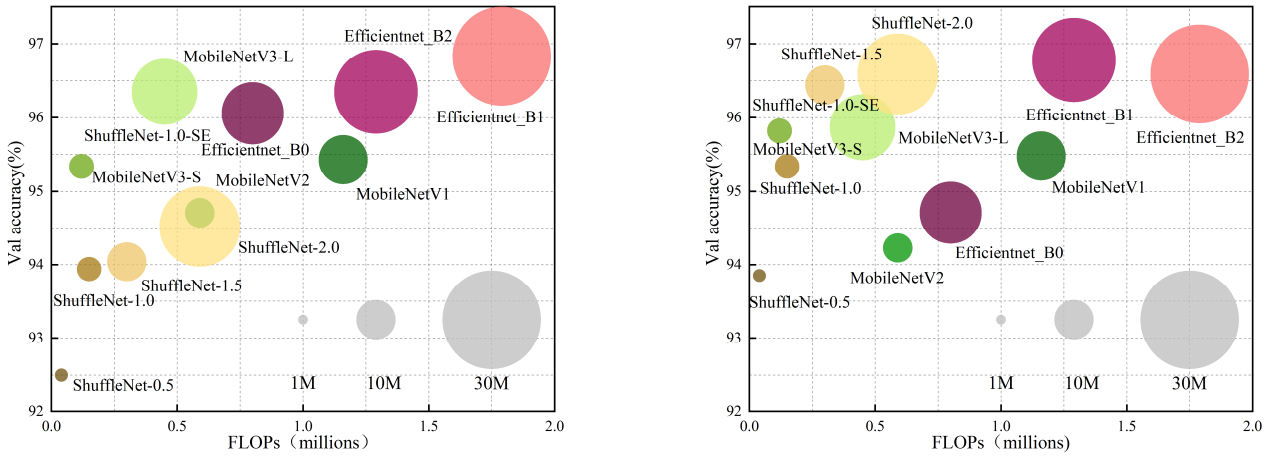


Figure 11. Validation results of different augmentation methods. (**a**) Experimental results for Dataset B1; (**b**) Experimental results for Dataset B2.

4.4. Experimental Results of DCGAN Augmentation

4.4.1. Performance Comparison of Different LCNNs

The results of the experiments using Dataset C are shown in Table 5. Compared to the accuracy improvement obtained by transferring the LCNNs from Dataset A to Dataset B2, it can be seen that the accuracy improvement obtained by transferring the LCNNs from Dataset B2 to Dataset C decreased. Furthermore, we can see that only MobileNetV1, MobileNetV2 and ShuffleNetV2-1.5 had a slightly smaller correct classification rate (TP) for crack images than for background images (TN), indicating that the DCGAN augmentation made the crack image features more evident in the dataset. Thus, the extraction of crack features was significantly improved for most LCNNs.

The experimental results show that the migration learning method based on small-scale datasets applies equally to all LCNNs trained on Dataset C. However, the effectiveness of enhancement through transfer learning diminishes as the number of crack samples increases. In addition, the SE module still improves the performance of ShuffleNetV2-1.0. ShuffleNetV2-1.0-SE achieves the highest maximum accuracy of 98.14% on Dataset C. At the same time, its "Parameters", "Complexity" and "Weights" are smaller than those of various LCNNs, which achieve similar accuracy. Its ability to correctly classify cracks is second only to EfficieNet_b2, while its ability to correctly classify backgrounds is the best.

Table 5. Performance validation results of different LCNNs trained on Dataset C.

Model	Transfer Learning	Parameters (Millions)	Complexity (GFLOPs)	Weight (MB)	Accuracy (%)	TP (%)	TN (%)
MobileNetV1	Yes	4.25	1.16	12.50	97.66	97.45	97.88
MobileNetV2	Yes	2.26	0.59	7.47	97.44	97.31	97.56
MobileNetV3_Small	Yes	1.53	0.12	6.32	97.44	97.50	97.38
MobileNetV3_Large	Yes	4.23	0.45	16.70	97.47	97.50	97.44
ShuffleNetV2_0.5	Yes	0.35	0.04	1.77	95.45	96.52	94.29
ShuffleNetV2_1.0	Yes	1.27	0.15	5.29	96.70	97.46	95.96
	No				95.61	97.21	94.11
ShuffleNetV2_1.0-SE	Yes	1.45	0.17	6.15	98.14	98.02	97.95
	No				97.24	97.49	97.00
ShuffleNetV2_1.5	Yes	2.5	0.30	10.00	97.85	97.76	97.94
ShuffleNetV2_2.0	Yes	5.38	0.59	20.90	97.95	98.01	97.51
EfficieNet_b0	Yes	4.05	0.80	15.90	96.79	97.22	96.38
EfficieNet_b1	Yes	6.58	1.29	25.70	97.79	97.82	97.76
EfficieNet_b2	Yes	7.77	1.79	30.30	97.98	**98.07**	97.89

Notes: "Transfer learning" represents pretraining the LCNNs on public crack dataset [38].

4.4.2. Comparison of Loss Curves for Different Datasets and Different Conditions

The training loss curves of ShuffleNetV2-1.0 with and without transfer learning and with and without the SE module were compared, as shown in Figure 12. Without transfer learning (red line), the final loss value was higher, and the fluctuations during training were more severe than with transfer learning (green line). ShuffleNetV2-1.0-SE had lower loss values than ShuffleNetV2-1.0 on both datasets with transfer learning, indicating a better fit and higher classification accuracy.

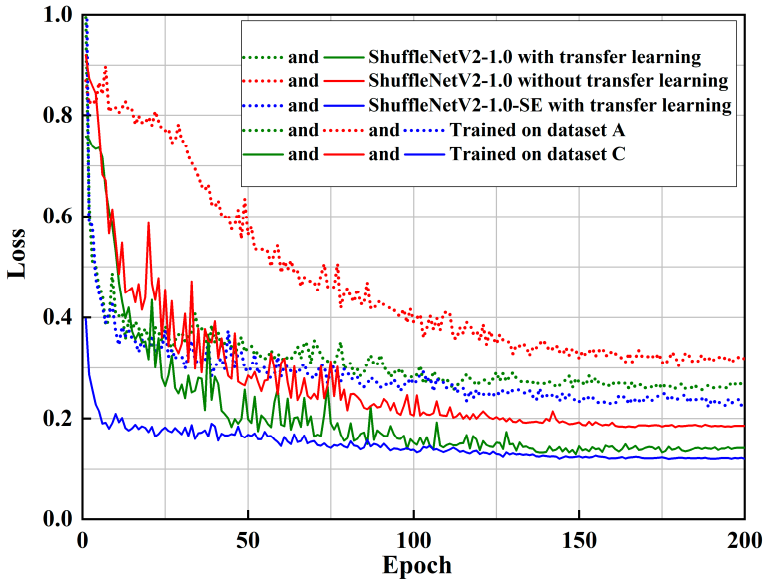


Figure 12. Training loss curve of ShuffleNetV2-1.0 under different strategies.

4.4.3. Analysis of the Impact of SE Modules and Transfer Learning on Network Performance

The effect of transfer learning and the SE module on the performance of ShufflenetV2-1.0 on Dataset C was analyzed, as shown in Figure 13. Without transfer learning, the network's crack image accuracy decreased, making it more prone to misclassifying background images as crack images. This shows that transfer learning was crucial for ShufflenetV2-1.0 to learn crack features. When the SE attention mechanism was introduced, compared to that of the background image, the accuracy for crack images improved significantly, demonstrating that the SE attention mechanism could effectively enhance the network's ability to extract crack features.

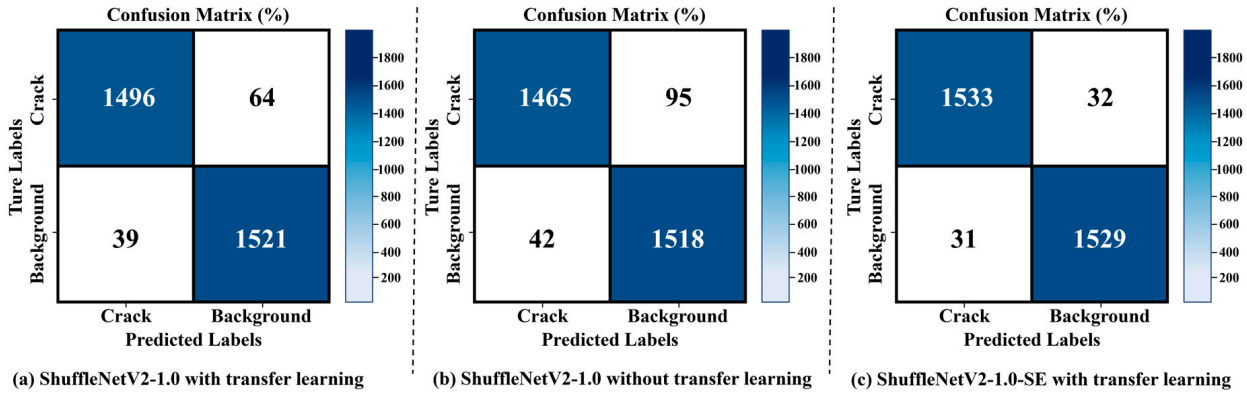


Figure 13. Confusion matrix for Shufflenetv2-1.0 validation results for Dataset C.

4.5. Visualization of Augmentation Using DCGAN with Transfer Learning

4.5.1. Effect of DCGAN Augmentation on Network Performance

Some of the test results achieved with ShuffleNetV2-1.0-SE trained on Datasets A and C are shown in Figure 14. The images included interference backgrounds (a)–(e) and various types of cracks (f)–(j). When using Dataset A, ShuffleNetV2-1.0-SE misclassified the background images with surface peeling (a) and black areas as cracks. It also misclassified the crack images with dark light (f), pits (g) and water seepage (i) backgrounds. When using Dataset C, ShufflenetV2-1.0-SE correctly classified all background images. The accuracy of most of the crack images also improved following training with Dataset C. The only exception was the crack images with a background of water seepage. This suggested that future dataset generation should include more images of this type to enhance the network's ability to handle water seepage interference cases.

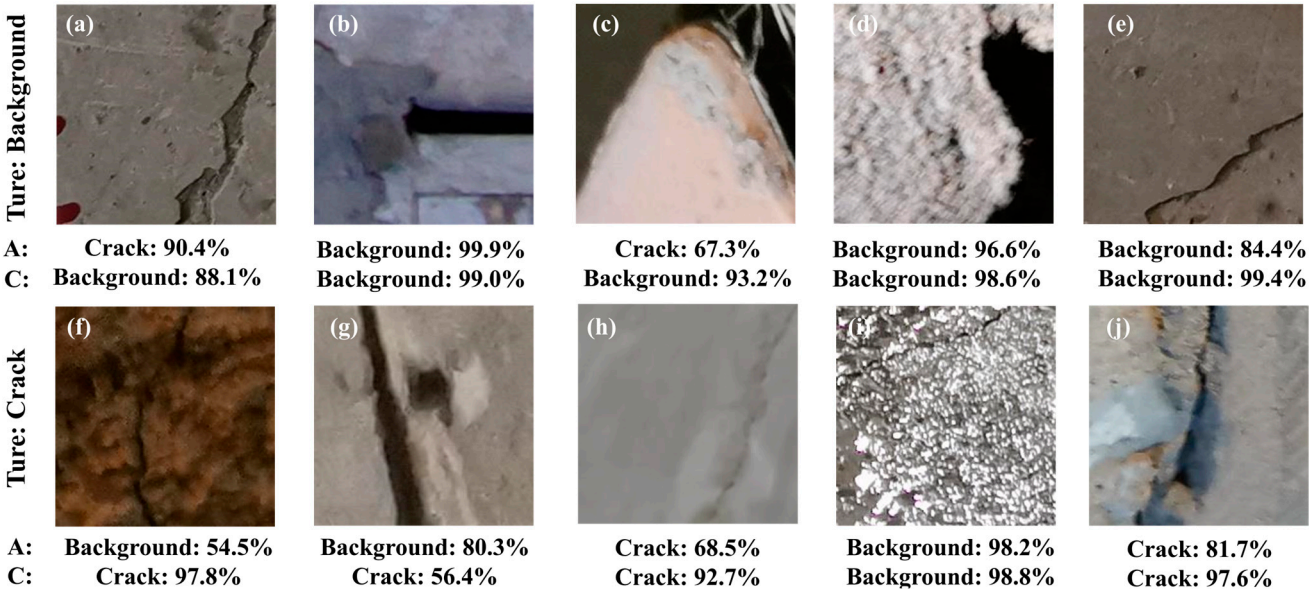


Figure 14. Comparison of the predictions of ShuffleNetV2-1.0-SE trained on different datasets: (**a**–**e**) represent interference background image; (**f**–**j**) represent crack image.

4.5.2. Generalization Performance Study

The generalization performance of ShuffleNetV2-1.0-SE trained with data augmentation using transfer learning and DCGAN was evaluated on a publicly available dataset containing tunnel lining cracks [3]. Figure 15 shows some test results for different types of cracks (disconnected, high luminance, interference background and edge cracks), where (a)–(d) show the test results achieved after training ShuffleNetV2-1.0-SE on Dataset C, while (e)–(h) show the results obtained after training the network on Dataset A. The results obtained after training on Dataset A, which contains DCGAN-augmented samples, are consistently correct, while those obtained after training on Dataset C are not. The results of ShuffleNetV2-1.0-SE trained on Dataset C, which correctly classifies crack images misclassified as backgrounds, demonstrate that DCGAN can effectively enhance the generalization ability of ShuffleNetV2-1.0-SE for crack detection.

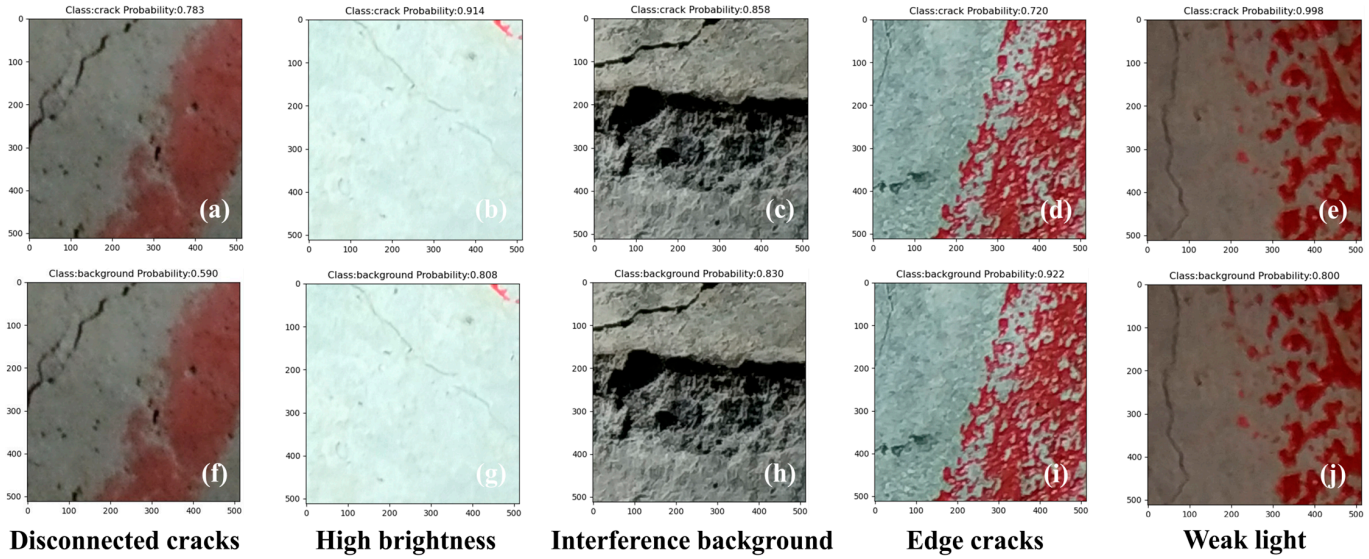


Figure 15. Comparison of the generalization performance of ShuffleNetV2-1.0-SE trained on different datasets: (**a**–**j**) represent crack image from [3].

5. Conclusions

In this paper, a method combining transfer learning and DCGAN to optimize ShuffleNetv2-1.0-SE for tunnel lining crack detection is presented. In addition, we apply transfer learning on a publicly available small-scale concrete dataset and compare data augmentation using DCGAN and conventional techniques. At last, an LCNN named ShuffleNetV2-1.0-SE is proposed, incorporating the SE attention mechanism into ShuffleNetV2-1.0. Our main findings are given below:

(1) Transfer learning on a small-scale dataset enhances the image generation quality of DCGAN, the classification accuracy of LCNN for crack images and the detection performance of LCNN for true positive crack images.
(2) The SE attention mechanism improves the crack feature extraction of ShuffleNetV2-1.0. An accuracy improvement from 94.23% to 98.14% was achieved with 1.45 million parameters and 017G floating-point computation using transfer learning and DCGAN data augmentation.
(3) ShuffleNetV2-1.0-SE shows better generalization on the public dataset after being trained with the DCGAN-augmented dataset, demonstrating the effectiveness of the proposed data augmentation method, which combines transfer learning and DCGAN.

Most LCNNs struggle to accurately classify crack images in a seepage background with or without data augmentation; therefore, we plan to further increase the number of images with a interference background in future work. In addition, quantization is required to detect cracks; therefore, the classification studied in this paper has limitations for practical applications. In the future, we will explore semantic segmentation models for crack images, which are simultaneously highly accurate and lightweight.

Author Contributions: Conceptualization, N.Z. and Y.S.; methodology, Y.S.; software, Y.S. and K.L.; validation, N.Z. and A.Y.; formal analysis, A.Y.; investigation, C.D.; resources, N.Z., K.L. and H.J.; data curation, C.D.; writing—original draft preparation, N.Z. and Y.S.; writing—review and editing, N.Z. and K.L.; visualization, A.Y. and H.J.; supervision, N.Z.; project administration, N.Z.; funding acquisition, N.Z. and H.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (Project no. 51608081) and the Natural Science Foundation of Chongqing City of China (No. cstc2021jcyj-msxmX1011).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Huang, H.W.; Li, Q.T.; Zhang, D.M. Deep learning based image recognition for crack and leakage defects of metro shield tunnel. *Tunn. Undergr. Space Technol.* **2018**, *77*, 166–176. [CrossRef]
2. Xue, Y.; Li, Y. A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects. *Comput.-Aided Civil Infrastruct. Eng.* **2018**, *33*, 638–654. [CrossRef]
3. Ren, Y.; Huang, J.; Hong, Z.; Lu, W.; Yin, J.; Zou, L.; Shen, X. Image-based concrete crack detection in tunnels using deep fully convolutional networks. *Construct. Build. Mater.* **2020**, *234*, 117367. [CrossRef]
4. Koch, C.; Georgieva, K.; Kasireddy, V.V.; Akinci, B.; Fieguth, P. A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. *Adv. Eng. Inform.* **2015**, *29*, 196–210. [CrossRef]
5. Sony, S.; Dunphy, K.; Sadhu, A.; Capretz, M.A.M. A systematic review of convolutional neural network-based structural condition assessment techniques. *Eng. Struct.* **2021**, *226*, 111347. [CrossRef]
6. He, Y.; Zhang, L.; Chen, Z.; Li, C.Y. A framework of structural damage detection for civil structures using a combined multi-scale convolutional neural network and echo state network. *Eng. Comput.* **2023**, *39*, 1771–1789. [CrossRef]
7. Tian, Y. Artificial intelligence image recognition method based on convolutional neural network algorithm. *IEEE Access* **2020**, *8*, 125731–125744. [CrossRef]
8. Guo, F.; Qian, Y.; Liu, J.; Yu, H. Pavement crack detection based on transformer network. *Autom. Constr.* **2023**, *145*, 104646. [CrossRef]
9. Liu, Y.; Cho, S.; Spencer, B.F., Jr.; Fan, J. Automated assessment of cracks on concrete surfaces using adaptive digital image processing. *Smart Struct. Syst.* **2014**, *14*, 719–741. [CrossRef]
10. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
11. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Adv. Neural Inform. Process. Syst.* **2012**, *25*, 1–9. [CrossRef]
12. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
13. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
14. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
15. Liu, Y.; Yao, J.; Lu, X.; Xie, R.; Li, L. DeepCrack: A deep hierarchical feature learning architecture for crack segmentation. *Neurocomputing* **2019**, *338*, 139–153. [CrossRef]
16. Song, W.; Jia, G.; Zhu, H.; Jia, D.; Gao, L. Automated Pavement Crack Damage Detection Using Deep Multiscale Convolutional Features. *J. Adv. Transp.* **2020**, *2020*, 6412562. [CrossRef]
17. Zhang, L.; Yang, F.; Zhang, Y.D.; Zhu, Y.J. Road Crack Detection Using Deep Convolutional Neural Network. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3708–3712.
18. Gao, Y.; Mosalam, K.M. Deep transfer learning for image-based structural damage recognition. *Comput.-Aided Civil Infrastruct. Eng.* **2018**, *33*, 748–768. [CrossRef]
19. Han, X.; Zhao, Z.; Chen, L.; Hu, X.; Tian, Y.; Zhai, C.; Wang, L.; Huang, X. Structural damage-causing concrete cracking detection based on a deep-learning method. *Constr. Build. Mater.* **2022**, *337*, 127562. [CrossRef]
20. Andrew, G.; Menglong, Z.J.M. Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
21. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
22. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 Ocober–2 November 2019; pp. 1314–1324.
23. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
24. Ma, N.; Zhang, X.; Zheng, H.-T.; Sun, J. Shufflenet v2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 116–131.
25. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. *Int. Conf. Mach. Learn.* **2019**, *97*, 6105–6114.
26. Li, G.; Liu, T.; Fang, Z.; Shen, Q.; Ali, J. Automatic bridge crack detection using boundary refinement based on real-time segmentation network. *Struct. Control Health Monit.* **2022**, *29*, e2991. [CrossRef]

27. Hou, Y.; Li, Q.; Han, Q.; Peng, B.; Wang, L.; Gu, X.; Wang, D. MobileCrack: Object Classification in Asphalt Pavements Using an Adaptive Lightweight Deep Learning. *J. Transp. Eng. Part B-Pavements* **2021**, *147*, 04020092. [CrossRef]
28. Chen, L.; Yao, H.; Fu, J.; Ng, C.T. The classification and localization of crack using lightweight convolutional neural network with CBAM. *Eng. Struct.* **2023**, *275*, 115291. [CrossRef]
29. Ni, F.; He, Z.; Jiang, S.; Wang, W.; Zhang, J. A Generative adversarial learning strategy for enhanced lightweight crack delineation networks. *Adv. Eng. Inform.* **2022**, *52*, 101575. [CrossRef]
30. Xu, B.Q.; Liu, C. Pavement crack detection algorithm based on generative adversarial network and convolutional neural network under small samples. *Measurement* **2022**, *196*, 111219. [CrossRef]
31. Zhong, J.; Huyan, J.; Zhang, W.; Cheng, H.; Zhang, J.; Tong, Z.; Jiang, X.; Huang, B. A deeper generative adversarial network for grooved cement concrete pavement crack detection. *Eng. Appl. Artif. Intell.* **2023**, *119*, 105808. [CrossRef]
32. Pei, L.; Sun, Z.; Xiao, L.; Li, W.; Sun, J.; Zhang, H. Virtual generation of pavement crack images based on improved deep convolutional generative adversarial network. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104376. [CrossRef]
33. Jin, T.; Ye, X.; Li, Z. Establishment and evaluation of conditional GAN-based image dataset for semantic segmentation of structural cracks. *Eng. Struct.* **2023**, *285*, 116058. [CrossRef]
34. Zhou, Z.; Zhang, J.; Gong, C.; Wu, W. Automatic tunnel lining crack detection via deep learning with generative adversarial network-based data augmentation. *Undergr. Space* **2023**, *9*, 140–154. [CrossRef]
35. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
36. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
37. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
38. Özgenel, Ç.F.; Sorguç, A.G. Performance Comparison of Pretrained Convolutional Neural Networks on Crack Detection in Buildings. In Proceedings of the International Symposium on Automation and Robotics in Construction, Berlin, Germany, 20–25 July 2018; pp. 1–8.
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
40. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1026–1034.