IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 38, No. 3, June, 2021, pp. 911-917
Journal homepage: http://iieta.org/journals/ts

Recognition and Classification of Concrete Cracks under Strong Interference Based on Convolutional Neural Network

Ningyu Zhao[1,2], Yang Jiang[2*], Yi Song[2]

[1] State Key Laboratory of Mountain Bridge and Tunnel Engineering, Chongqing 400074, China
[2] School of Civil Engineering, Chongqing Jiaotong University, Chongqing 400074, China

Corresponding Author Email: jiangyang829@163.com

https://doi.org/10.18280/ts.380338

Received: 5 February 2021
Accepted: 30 April 2021

Keywords:
concrete cracks, image classification, convolutional neural network (CNN), block attention module

ABSTRACT

This paper proposes the UmNet model based on convolutional neutral network (CNN), aiming to improve the ability to recognize and classify concrete cracks in a background complicated by construction seams and seepage traces. The model was derived from the famous CNN AlexNet. Without changing the receptive field, large convolutional kernels were replaced with small ones to reduce the parameters, deepen the network, and increase nonlinear transforms. Next, convolutional block attention module (CBAM) was introduced to highlight the key information in images and focus on high-weight channels. Finally, Bayesian network (BN) layer and L2 regularization were added, and the number of nodes in fully connected layer were reduced. A series of comparative experiments were carried out on three datasets D, P, and W. The results show that the proposed UmNet surpassed AlexNet in the recognition accuracy on D, P, and W by 3.74%, 3.17%, and 5.74%, respectively, and reduced the number of parameters by 75.04%. Therefore, our model is an effective means to recognize and classify of concrete cracks under strong interference.

1. INTRODUCTION

Cracking is one of the most common diseases of concrete during long-term use. The manual detection of concrete cracks has several defects: long time consumption, lack of objectivity, and high risks. These defects can be overcome by applying image recognition techniques to identify the cracks in concrete images.

In recent years, many researchers adopted convolutional neural network (CNN) to recognize concrete crack images. For instance, Zhang et al. [1] built and trained the first CNN model, and proved that deep learning outperforms traditional image recognition algorithms in crack detection. Cha et al. [2] improved their CNNs with sliding windows, so that the models can detect any crack image surpassing the training resolution. Li et al. [3] enhanced datasets with deep convolutional generative adversarial network (GAN), and presented an algorithm that effectively detects complex road scenes. Referring to SegNet, Chen et al. [4] proposed an encoder-decoder structural model with a fully CNN, namely, PCSN, and verified its feasibility in crack detection. Lei et al. [5] developed an image recognition method for concrete crack detection, and demonstrated that the method can identify cracks in stained and moss-covered concretes. Deng et al. [6] employed region-based CNN (R-CNN) to detect bridge crack images containing handwritten traces, and observed that R-CNN can automatically recognize the cracks in original images. Li et al. [7] put forward a deep neural network based on attention mechanism and feature fusion, which effectively enhances the detection accuracy of narrow cracks on airfield pavement images with complex background and low contrast.

Laudable progress has been achieved on the image recognition of concrete cracks based on deep learning. However, most studies only deal with images with simple background and disturbances. Not many researchers have studied the recognition of concrete cracks with complex background involving pot holes, construction seams, and seepage traces. There is not yet a sufficiently precise model to recognize these cracks. To solve the problem, this paper acquires concrete crack image datasets with complex background, and derives an improved CNN model from AlexNet. In addition, the convolutional block attention module (CBAM) was improved from the attention mechanism, based on the existing channels and spaces. Then, CBAM was introduced to the improved CNN model, such that the latter focuses more on cracking and extract more details from images. Further, small convolutional kernels and nonlinear activation functions were adopted to reduce the number of parameters, while enhancing the learning of network features. In this way, the proposed model can accurately recognize and classify the surface cracks in concrete crack image datasets with complex background.

2. METHODOLOGY

2.1 CNN

With the development of computing technology, CNN has been extensively applied to image recognition [8-11]. This feedforward neural network [12] consists of input layer, convolutional layer, pooling layer, fully connected layer, and other supporting layers, as well as connection weights.

2.1.1 Convolution

Lying at the core of CNN, convolution layers aim to

perceive the local information of the input image through convolution. Each convolution layer contains multiple filters that learn the features from the input image. Convolution kernels of different sizes extract different types of features. Low-level kernels can extract simple features like edges and curves, while high-level kernels can obtain more abstract features. The convolution operation can be expressed as:

$$a_{i,j} = \sum_{m=0}^{2}\sum_{n=0}^{2} w_{m,n}x_{i+m,j+n} + w_b \quad (1)$$

where, $x_{i,j}$ is the element in the i-th row and j-th column of the image; $w_{m,n}$ is the weight of the filter in the m-th row and n-th column; w_b is the bias of filter; $a_{i,j}$ is the element in the i-th row and j-th column of the feature map.

The convolution operation is explained in Figure 1. Addition and multiplication are carried out in turn on the value of a kernel and the value at the corresponding value in the sample, producing a numerical value containing image information. Then, a top-down sliding calculation is performed from left to right with a fixed step length. The outputs constitute the image feature corresponding to the kernel.

To expand the receptive field and extract more global features from the input image, AlexNet adopts large kernels of the size 11×11, and 5×5. The large size brings a huge number of parameters and a huge computing load. For an input image of 28×28, the size of the feature map will be (28-5)/1+1=24, if ten large kernels (5×5) are arranged with the stride of 1 and padding of 0; the size of the feature maps will be (28-3)/1+1=26, and (26-3)/1+1=24, if three small kernels (3×3) are arranged with the same stride and padding. Since the feature maps are of the same size, our model replaces the large kernels in AlexNet with multiple 3×3 small kernels, without changing the receptive field. In the above example, the convolution layer has 10×5×5+10=260 parameters in the presence of large kernels, and (10×3×3+10)×2=200 parameters in the presence of two small kernels. After the kernel replacement, both the number of parameters and computing load are reduced, and the network is deepened. In addition, the multiple small kernels lead to more nonlinear activation layers, which contribute to the overall recognition ability for concrete cracks under disturbance.

2.1.2 Pooling

Pooling is a down-sampling process that reduces the dimensionality of features. The input feature map is down-sampled step by step to reduce the map dimensionality, the number of parameters, and computing load. This process enhances the fault tolerance of the model, and lowers the risk of overfitting. Pooling is typically arranged right after convolution. The common pooling strategies include max pooling and average pooling (Figure 1). Our model adopts max pooling, because it is the best strategy for image recognition [13, 14].

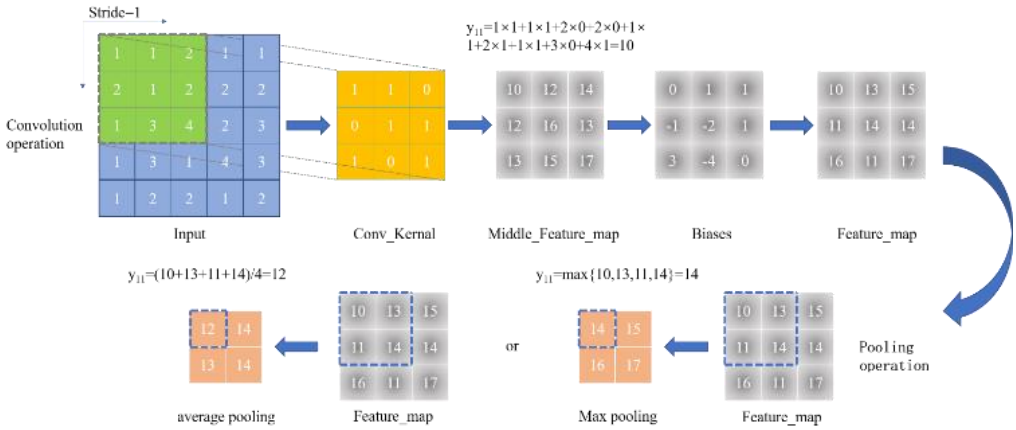


Figure 1. Illustration of convolution and pooling

2.1.3 Fully connected layers

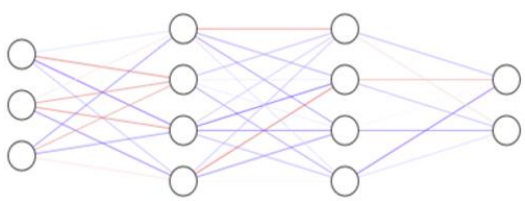


Figure 2. Illustration of fully connected layer

Fully connected layer is an ordinary layer of the neutral network. It is connected to all the nodes in the previous layer. As a classifier of the CNN, each fully connected layer encapsulates the previous local features into a weight matrix, and maps the learned distributed features to the sample labeling space. In this way, the local information of convolution and pooling layers can be integrated effectively. Coupled with the nonlinear mapping of activation functions, fully connected layer can theoretically simulate any nonlinear transform. This layer is illustrated in Figure 2.

During the design of our model, the number of nodes in the first and second fully connected layers of AlexNet was changed from 4,096 per layer to 1,024, and that in the last fully connected layer was set to 2 according to the classification task of the model. Due to parameter redundancy, fully connected layer contributes the most parameters to the network model. Therefore, the modified fully connected layers in our model greatly reduce the total number of parameters. To prevent the common problem of overfitting in neural network training, Dropout technology was adopted in the fully connected layers of our model. Thus, the nodes are discarded at a certain probability. For stochastic gradient descent (SGD), a different network is trained in each batch, because the nodes are discarded randomly. Therefore, the risk of overfitting could be minimized.

2.1.4 Activation functions

The essential task of our neural network model is function fitting. The activation functions are normally nonlinear. Adding these functions introduces nonlinearity to the model, such that it could theoretically approximate any function. Common activation functions include sigmoid, rectified linear unit (ReLU), tanh, and Leaky ReLU. The curves of these functions are displayed in Figure 3. ReLU is a nonlinear activation function introduced by Nair and Hinton [15]. The value of ReLU either equals 1 or 0. The function achieves a fast speed and converges much faster than sigmoid and tanh, because it only needs to judge whether the input is greater than zero. If the input to ReLU is negative, however, the nodes will be unable to update parameters. To solve the problem, a Leaky value is introduced to the negative half of the interval of ReLU, which to a certain extent improves the function performance. Hence, our model adopts both ReLU and Leaky ReLU as activation functions:

$$f(x) = \max(0, x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases} \tag{2}$$

$$f(x) = \begin{cases} x, x \geq 0 \\ ax, x < 0 \end{cases} \tag{3}$$

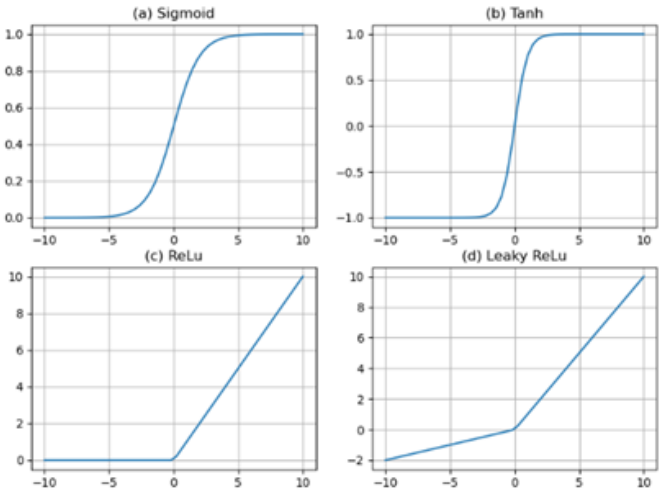


Figure 3. Nonlinear activation functions

2.1.5 Softmax layer

The feature map matrix generated through the above convolution, pooling, and fully connected layers can be applied to classify concrete cracks. The most common classification layer is softmax layer, whose principle can be explained by:

$$P\left(y^{(i)} = n \mid x^{(i)}; W\right) = \begin{bmatrix} P(y^{(i)} = 1 \mid x^{(i)}; W) \\ P(y^{(i)} = 2 \mid x^{(i)}; W) \\ \vdots \\ P(y^{(i)} = n \mid x^{(i)}; W) \end{bmatrix} = \frac{1}{\sum_{j=1}^{n} e^{w_j^T}} \begin{bmatrix} p^{w_1^T x^{(i)}} \\ p^{w_2^T x^{(i)}} \\ \vdots \\ p^{w_n^T x^{(i)}} \end{bmatrix} \tag{4}$$

where, i=1…m; $P(y^{(i)}=n|x^{(i)}; W)$ is the probability of the m-th training sample belonging to class n with weight W; $p^{w_n^T x^{(i)}}$ is the input of softmax layer, i.e., the feature map outputted from the previous pooling layer. For the i-th input, the sum on the right side always equals 1. Since the function values obey the normal distribution, the probability of each input belonging to each class will be returned.

2.2 AlexNet

The CNN AlexNet is the winner of the image classification competition ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2012. It is 10% more accurate than the winner of ILSVRC 2011. The network operates in the following manner: the first and second layers perform convolution, pooling, and normalization, in turn; the third and fourth layers only perform convolution; the fifth layer perform convolution and pooling; the eighth layer (fully connected layer) performs classification with softmax. Local response normalization (LRN), ReLU, and Dropout are adopted by the network.

2.3 CBAM

CBAM is an important mechanism to enhance CNN performance. Hu et al. [16] proposed the squeeze-and-excitation (SE) network (SENet), the first effective mechanism with channel attention. SENet establishes the spatial correlations between features, and effectively improves CNN performance. Later, CBAM was extended based on SENet [17]. The extended mechanism contains a channel attention module and a spatial attention module:

$$F' = M_c(F) \otimes F \tag{5}$$

$$F'' = M_s(F') \otimes F' \tag{6}$$

where, F is the feature map; F' is the feature map of the channel attention module; F'' is the feature map of the spatial attention module; $\otimes$ is element-wise multiplication; Mc and M_s are the weight coefficients of channel attention module and spatial attention module, respectively.

The channel attention module carries out global average pooling and max pooling of the feature map outputted from convolution operation. The pooled feature map is then imported to a two-layer neural network. The obtained features are processed by sigmoid activation function, producing a weight coefficient M_c. The product between M_c and the original feature F is a new feature F'. After that, the new feature is passed through the spatial attention module, and weighted into the final feature map. The spatial attention module carries out average pooling and max pooling of F'. The pooled results are stitched, convoluted, and processed by sigmoid to obtain a weight coefficient M_s. The product between M_s and F' is a new feature F''.

In this paper, CBAM is added between convolution layers. The addition of the module makes feature learning focus more on cracks, and suppresses the effect of disturbances. In this way, the recognition model can acquire more details about cracks, and identify concrete cracks more effectively.

2.4 UmNet structure

Figure 4 illustrates the structure of our improved model UmNet. In our model, batch normalization (BN) replaces the original LRN. Bayesian network (BN) and ReLU are added

behind each convolution layer. After the first convolution layer, ReLU is replaced with Leaky ReLU. The original large kernels (11×11; 5×5) are changed into small kernels (3×3). L2 regularization is adopted to regularize the network, and max pooling is selected for implementing pooling operations.

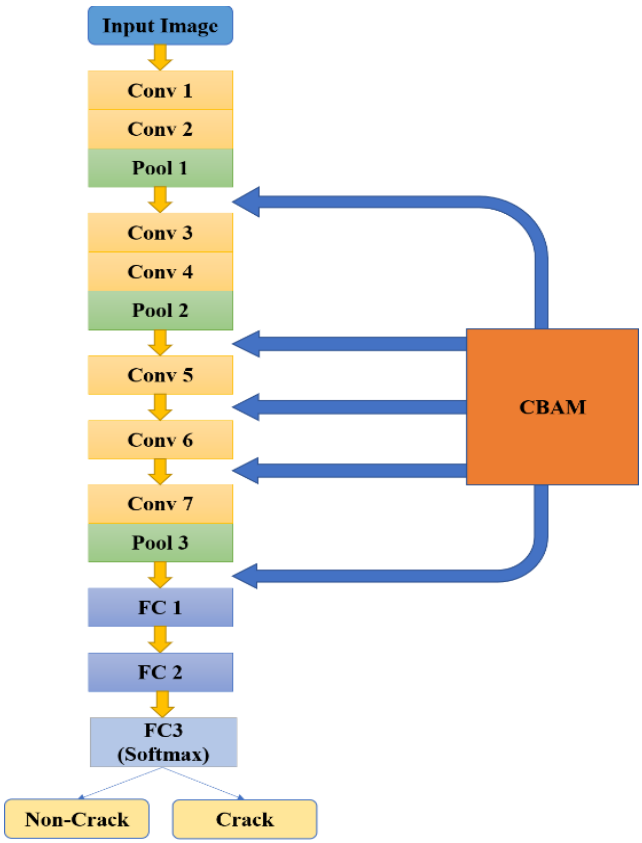


Figure 4. UmNet structure

Table 1. Sizes of model training data

Layer	Height	Width	Depth
Input	227	227	1
L1	27	27	96
L2	13	13	256
L3	13	13	384
L4	13	13	384
L5	6	6	256
L6	1	1	1,024
L7	1	1	1,024
L8	1	1	2

Table 2. Parameters of each convolution layer and pooling layer

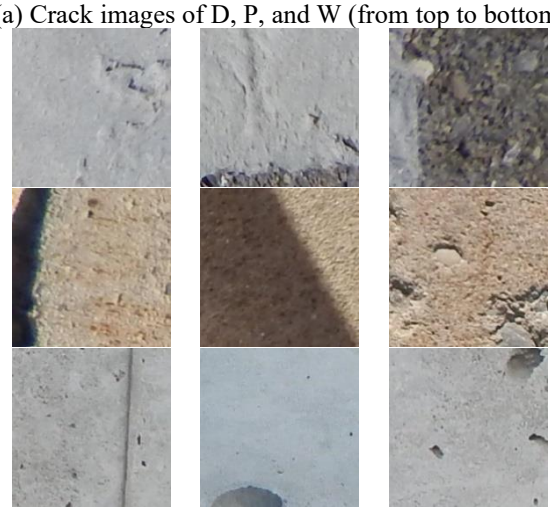
Operation	Height	Width	Depth	Step length	Padding	Output
C1	3	3	96	4	0	96
C2	3	3	96	1	0	96
P1	3	3	-	2	0	-
C3	3	3	256	1	1	256
C4	3	3	256	1	1	256
P2	3	3	-	2	0	-
C5	3	3	384	1	1	384
P3	3	3	-	2	0	-
C6	3	3	384	1	1	384
C7	3	3	256	1	1	256

In the improved model, the first and second convolution layers perform convolution, BN, and pooling, in turn. The third to fifth layers perform convolution and BN. The fifth layer performs max pooling. Channel attention module and spatial attention module are introduced between every pair of convolution layers. The six and seventh layers (fully connected layers) perform Dropout. The eighth layer (fully connected layer) outputs class labels with softmax function. Table 1 presents the sizes of model training data. Table 2 lists the parameters of each convolution layer and pooling layer in the model.

3. CONCRETE CRACK DATASETS WITH COMPLEX BACKGROUND

3.1 Dataset acquisition

(a) Crack images of D, P, and W (from top to bottom)



(b) Non-crack images of D, P, and W (from top to bottom)

Figure 5. Datasets D, P, and W

Our experiments adopt the concrete crack image dataset SDNET2018 published by Sattar Dorafshan et al. The dataset consists of three parts: Bridge deck (D), Wall (W), and Pavement (P). Dataset D contains 13,620 images, including 2,025 crack images and 11,595 non-crack images; Dataset W contains 18,138 images, including 3,851 crack images and 14,287 non-crack images; Dataset P contains 24,334 images, including 2,608 crack images, and 20,826 non-crack images. As shown in Figure 5, the selected database has complex background disturbances (pot holes, construction seams, seepage traces, and shadows). There are fewer crack images than non-crack images in the dataset. In the presence of

disturbances, the non-crack images should cover as many disturbed backgrounds as possible. Considering the performance of graphics processing unit (GPU), the authors did not adjust the situation that the dataset contains more non-crack images than crack images.

3.2 Image preprocessing and dataset construction

SDNET2018 provides color images of the resolution 256×256. According to the needs of our model, the original images should be cropped into color images of the resolution 227×227. Grayscale processing can accelerate image training, and save lots of memory. Therefore, the color images were subject to grayscale processing through the conversion from RGB to YUV:

$$g=0.299R+0.587G+0.114B \tag{7}$$

where, R, G and B are the brightness of red, yellow, and blue, respectively; g is the synthetic brightness. Figure 6 shows the grayscale image.

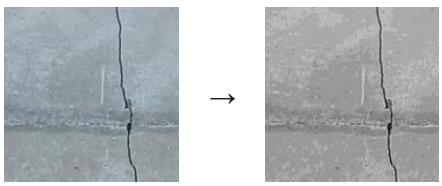


Figure 6. Grayscale processing of concrete crack images

Table 3. Training set, verification set, and test set of each dataset

	Dataset	Type	Crack images	Non-crack images	Total number
Grayscale	D	Training set	1619	9275	10894
		Verification set	203	1160	1363
		Test set	203	1160	1363
	W	Training set	3081	11431	14512
		Verification set	385	1428	1813
		Test set	385	1428	1813
	P	Training set	2086	10332	12418
		Verification set	261	1291	1552
		Test set	261	1291	1552

The color images in datasets D, W, and P were preprocessed through cropping and grayscale processing. After that, the preprocessed images of each dataset were divided into a training set, a verification set, and a test set by 8: 1: 1. To ensure the uniform distribution of crack and non-crack images across datasets, the ratio of crack images to non-crack images was kept the same as that in D, W, and P. The crack and non-crack images were randomly selected from each dataset by a Python program. Finally, one-hot coding was performed on the crack and non-crack images (10 for each crack image, and 01 for each non-crack image). Table 3 shows the training set, verification set, and test set.

4. CLASSIFICATION EXPERIMENTS

4.1 Experimental environment

In terms of software, the operating system is Windows 10 (64bit), the programming language is Python, the deep learning framework is TensorFlow 2.2. The dependencies mainly include CUDA 10.1 and CuDNN. In terms of hardware, our experiments adopt an Intel Core i7-10700 processor (eight cores), and a Nvidia GTX1660Ti (6G) GPU.

4.2 Hyperparameter setting

Our improved CNN model was trained through SGD. The loss function is categorical crossentropy. The learning rate was set to 0.01, the momentum to 0.9, the weight attenuation coefficient to 0.0001, the Dropout rate to 0.5, the number of iterations (epoch) to 50, the batch size to 32, and the size of input image to 227×227×1.

4.3 Experimental procedure and results

Our CNN model was constructed under the deep learning framework of TensorFlow 2.2, and applied to recognize the surface cracks of concrete members. The model was trained and tested by the above-mentioned datasets divided from D, P, and W. The following experiments were conducted to verify the effectiveness of each module of our model in the recognition and classification of concrete cracks.

4.3.1 Effectiveness of small kernels

The first experiment intends to compare the influence of large and small kernels on the classification of concrete cracks. The original AlexNet was defined as Model A. Then, the original large kernels (11×11; 5×5) in the first and second convolution layers were changed into small kernels (3×3), and the resulting model was defined as Model A1. After that, Model A and Model A1 (small kernels) were trained separately on D, P, and W. Table 4 records the test accuracies and losses.

Table 4. Test accuracies and losses of Models A and A1

Dataset	Model	Accuracy %	Loss
D	A	87.31	0.5047
	A1	89.35	0.3497
P	A	89.69	0.4049
	A1	91.07	0.2647
W	A	85.33	0.4328
	A1	88.29	0.3672

As shown in Table 4, the original AlexNet, after being trained by D, P, and W, achieved an accuracy of 87.31%, 89.69%, and 85.33% on the test sets, respectively. Model A1 improved the test accuracy to 89.35%, 91.07%, and 88.29%, respectively. Thus, small kernels can effectively enhance the recognition and classification performance of the model on datasets D, P, and W. The possible reasons for the enhancement are as follows: The replacement of large kernels with small kernels does not change the receptive field, but increases the depth of the network, which contributes to the network performance. Besides, small kernels bring more nonlinear transforms to convolution layers (nonlinear activation functions), further enhancing the generalization ability of the model.

4.3.2 Effectiveness of CBAM

The next experiment investigates the influence of CBAM on the recognition and classification of the network model. CBAM was added between convolution layers of AlexNet,

and the modified model was defined as Model A2. Then, Model A2 was trained by D, P, and W, respectively. Table 5 records the test accuracies and losses of Models A and A2.

Table 5. Test accuracies and losses of Models A and A2

Dataset	Model	Accuracy %	Loss
D	A	87.31	0.5047
	A2	88.41	0.3898
P	A	89.69	0.4049
	A2	91.17	0.3668
W	A	85.33	0.4328
	A2	86.93	0.3928

As shown in Table 5, the original AlexNet, after being trained by D, P, and W, achieved an accuracy of 87.31%, 89.69%, and 85.33% on the test sets, respectively. Model A2 improved the test accuracy to 88.41%, 91.17%, and 86.93%, respectively. Thus, CBAM can effectively enhance the recognition and classification performance of the model on datasets D, P, and W. The possible reasons for the enhancement are as follows: CBAM includes a channel attention module and a spatial attention module. The former learns the weight of each channel, and assigns a high weight to key channels. The latter identifies the regions of interest in images through training, enhancing the focusing ability of the model.

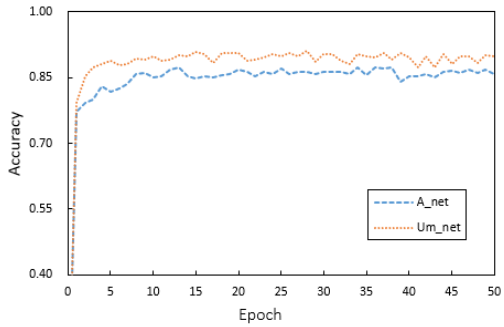
4.3.3 Effectiveness of UmNet

The third experiment tries to verify the effectiveness of the proposed network model. For this purpose, the UmNet model was constructed by introducing small kernels, CBAM, BN layer, and L2 regularization to the original AlexNet, and changing the number of nodes in the first and second fully connected layers to 1,024. Specifically, a BN layer was added behind each convolution layer, and supported with L2 regularization. Then, UmNet was trained on D, P, and W. The test accuracies, losses, and number of parameters are recorded in Table 6. The accuracy curves of Model A and UmNet are compared in Figure 7.

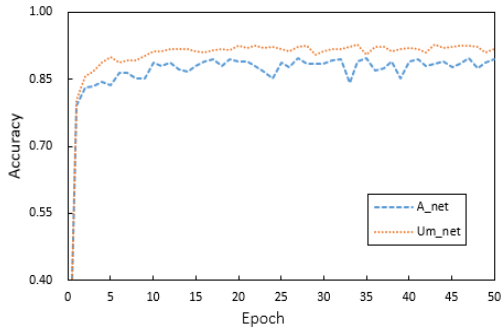
As shown in Table 6 and Figure 7, the recognition and classification accuracies of UmNet on D, P, and W were 3.74%, 3.17%, and 5.74% higher than those of original AlexNet. The most prominent improvement was realized on W. From the data volume of training sets, it can be learned that W provides more crack images than D and P, and has a more balanced ratio between crack images and non-crack images (background images). By reducing the number of nodes in fully connected layers, UmNet had 75.04% fewer training parameters than AlexNet. The experimental results fully demonstrate the effectiveness of the proposed UmNet model.

Table 6. Test accuracies, losses, and number of parameters of Model A and UmNet

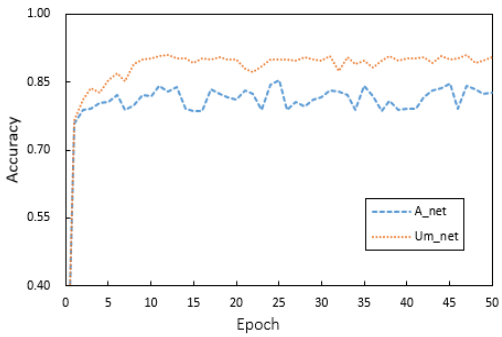
Dataset	Model	Accuracy %	Loss	Number of parameters
D	A	87.31	0.5047	58,267,714
	UmNet	91.05	0.4142	14,545,652
P	A	89.69	0.4049	58,267,714
	UmNet	92.85	0.3659	14,542,652
W	A	85.33	0.4328	58,267,714
	UmNet	91.07	0.3815	14,542,652



(a) Test accuracy on dataset D



(b) Test accuracy on dataset P



(c) Test accuracy on dataset W

Figure 7. Test accuracy curves

5. CONCLUSIONS

Based on AlexNet, this paper puts forward a CNN model called UmNet for recognizing and classifying concrete cracks in complex backgrounds (including pot holes, construction seams, and seepage traces).

(1) Small convolution kernels were adopted to deep the network, reduce parameters, and increase nonlinear transforms (nonlinear activation functions), without changing the receptive field. The effectiveness of small kernels was proved by an experiment on Model A1, which improved the test accuracies on D, P, and W by 2.04%, 1.38%, and 2.96%, respectively.

(2) CBAM was introduced between convolution layers. The channel attention module of CBAM assigns a high weight to key channels, while the spatial attention module identifies the regions of interest in images. The effectiveness of CBAM was proved by an experiment on Model A2, which improved the test accuracies on D, P, and W by 1.1%, 1.48%, and 1.6%, respectively.

(3) The proposed UmNet model combines multiple modules, including small kernels, CBAM, BN layer, and L2 regularization, and reduces the number of nodes in fully connected layers. Experimental results show that the UmNet

improved the test accuracies on D, P, and W by 3.74%, 3.17%, and 5.74%, respectively, and reduced 75.04% of parameters. Therefore, the proposed UmNet model outshines the original AlexNet in classification and parameter volume.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Grant No.: 51608081) and the Scientific and Technological Research Program of the Chongqing Municipal Education Commission (No. KJQN201800743) and Graduate Education Innovation Fund of Chongqing Jiaotong University (Grant No.: 2020S0020).

REFERENCES

[1] Zhang, L., Yang, F., Zhang, Y.D., Zhu, Y.J. (2016). Road crack detection using deep convolutional neural network. In 2016 IEEE International Conference on Image Processing (ICIP), pp. 3708-3712. https://doi.org/10.1109/ICIP.2016.7533052

[2] Cha, Y.J., Choi, W., Büyüköztürk, O. (2017). Deep learning-based crack damage detection using convolutional neural networks. Computer-Aided Civil and Infrastructure Engineering, 32(5): 361-378. https://doi.org/10.1111/mice.12263

[3] Li, L., Shao, R., Progress, O. (2019). Bridge crack detection algorithm based on image processing under complex background. Laser & Optoelectronics Progress, 56(6): 112-122.

[4] Chen, T., Cai, Z., Zhao, X., Chen, C., Liang, X., Zou, T., Wang, P. (2020). Pavement crack detection and recognition using the architecture of segNet. Journal of Industrial Information Integration, 18: 100144. https://doi.org/10.1016/j.jii.2020.100144

[5] Lei, S.D., Cao, H.Y., Kang, J.T. (2020). Concrete surface crack recognition in complex scenario based on deep learning. Journal of Highway and Transportation Research and Development (English Edition), 14(4): 48-58. https://doi.org/10.1061/JHTRCQ.0000754

[6] Deng, J., Lu, Y., Lee, V.C.S. (2020). Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network. Computer-Aided Civil and Infrastructure Engineering, 35(4): 373-388. https://doi.org/10.1111/mice.12497

[7] Li, H.F., Wu, Z.L., Nie, J. (2020). An automatic fine crack recognition algorithm for airport pavement under significant noises. Computer Engineering and Science, 42(11): 2020-2029.

[8] Barat, C., Ducottet, C. (2016). String representations and distances in deep convolutional neural networks for image classification. Pattern Recognition, 54: 104-115. https://doi.org/10.1016/j.patcog.2016.01.007

[9] Shi, B., Bai, X., Yao, C. (2016). Script identification in the wild via discriminative convolutional neural network. Pattern Recognition, 52: 448-458. https://doi.org/10.1016/j.patcog.2015.11.005

[10] Leng, B., Guo, S., Zhang, X., Xiong, Z. (2015). 3D object retrieval with stacked local convolutional autoencoder. Signal Processing, 112: 119-128. https://doi.org/10.1016/j.sigpro.2014.09.005

[11] Xu, J., Luo, X., Wang, G., Gilmore, H., Madabhushi, A. (2016). A deep convolutional neural network for segmenting and classifying epithelial and stromal regions in histopathological images. Neurocomputing, 191: 214-223. https://doi.org/10.1016/j.neucom.2016.01.034

[12] Simonyan, K., Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. arXiv preprint arXiv:1406.2199.

[13] Scherer, D., Müller, A., Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In International Conference on Artificial Neural Networks, pp. 92-101. https://doi.org/10.1007/978-3-642-15825-4_10

[14] Boureau, Y.L., Ponce, J., LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 111-118.

[15] Nair, V., Hinton, G.E. (2010). Rectified linear units improve restricted Boltzmann machines. In Icml.

[16] Hu, J., Shen, L., Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132-7141.

[17] Woo, S., Park, J., Lee, J.Y., Kweon, I.S. (2018). Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 3-19.