Research Article

An Improved Nondestructive Semantic Segmentation Method for Concrete Dam Surface Crack Images with High Resolution

Jun Zhang¹ and Jia Zhang²

¹State Grid Hunan Electric Power Company Limited Research Institute, Changsha, Hunan, China

²Changsha Jinfengtan High-Tech Company Limited, Changsha, Hunan, China

Correspondence should be addressed to Jun Zhang; sghnzj1988@163.com

Received 8 June 2020; Revised 16 September 2020; Accepted 28 October 2020; Published 17 November 2020

Academic Editor: Yu-Sheng Shen

Copyright © 2020 Jun Zhang and Jia Zhang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To nondestructive semantic segment the crack pixels in the image with high resolution, previous methods often use sliding window and the crack patches to train the FCNs, and then use the trained FCNs for crack recognition. However, the FCNs will produce a higher proportion of false crack predictions with messy distributions in the high-resolution image. A CNN-to-FCN method is proposed to solve this problem. The CNN is trained by all the patches for large-scale crack and background recognition, and the screened crack predictions are then segmented by the FCN. A real-world concrete dam surface crack image database is firstly established to verify the improved method. The results indicated that (1) the improved method can extremely avoid the higher proportion of false crack predictions and their messy distributions in the high-resolution image through the full utilization of background patches and large-scale background recognition; (2) the ResNetv2 backbone and DeepLabv3 architecture recommended by the improved method can be further modified by reducing the bottleneck channels and adding a DUC module to achieve better performance; (3) the improved method can also reduce the prediction time when the image has low proportion of crack patches, which becomes more practicable for the engineering applications.

1. Introduction

Dam safety monitoring is one of the most important tasks in reservoir project management. The “monitoring” includes not only the instrument observation and analysis of the fixed measuring points [1, 2] but also the visual inspection and instrument exploration on the dam. Through monitoring, the abnormal state of the dam can be discovered and dealt with in time to avoid serious consequences.

Crack detection plays a vital role in dam safety monitoring. The crack occurs on the concrete dam since the construction stage. Some cracks are beneficial, such as the artificial crack which is used to prevent the hydration heat temperature crack. Some cracks are harmful, which can decline the dam concrete strength or format leakage passage. Manual inspection is a traditional way to detect cracks. However, the vast space of the dam will limit the inspection scope and lead to time-consuming and inefficient.

Moreover, the crack detail and developing process are difficult to track and evaluate.

Automated crack detection with computer vision method is an effective way to replace manual inspection. In the past, researchers often adopted image processing, edge detection, and morphological operations to detect crack [3–9]. For dam concrete, Fan Xinnan et al. [10] uses local-global clustering analysis and the image processing method to identify the visual detection of underwater dam surface cracks. These methods have been proved to be useful in some situations, such as bright characteristics, high contrast, less noise, and strong continuity of crack. However, they may not be applicable to the dam concrete surface with diverse background texture, various types of noise, and irregular crack distribution (Figure 1(b)).

In recent years, a more powerful image recognition and detection technology based on the convolutional neural network (CNN) has been proposed by Lecun et al. [11], and

(a)

FIGURE 1: Continued.

FIGURE 1: The crack patches from high-resolution images of dam surface: (a) simple cracks; (b) intricate cracks.

then shows excellent potential in dealing with these complex scenarios above. To further classify the image in pixel level, Long et al. [12] proposed a specific CNN named the fully convolutional network (FCN). The basic FCN model is then developed by some optimized backbones and algorithms, including decoder variants and integrating context knowledge, such as the SG-net, U-net, and DeepLab [13]. Meanwhile, the crack detection techniques based on CNN or FCN have already applied in various civil structures such as

the bridges, roads, buildings, highways, and tunnel, and proved to be a more efficient way than the previous methods.

For classification tasks, Dorafshan et al. [14] compare the performance of common edge detectors and deep convolutional neural networks (AlexNet DCNN) for image-based crack detection in concrete panels. Hongyan Xu et al. [15] use the techniques of atrous convolution, atrous spatial pyramid pooling (ASPP) module, and depthwise separable convolution to improve the traditional CNN, and the bridge

crack dataset is established to verify the proposed model. Umme Hafsa Billah et al. [16] propose the ResNet to detect cracks of various roads, highways, and bridge decks at different times of the day and for different light orientation conditions. With the ResNet-based classifier, Chen Feng et al. [17] use AL to reduce the number of civil infrastructure surface images required for annotation and thus reduce the effort and cost of annotation by domain experts. Meanwhile, the backbones of VGG and GoogLeNet are also used to detect concrete crack [18–20].

For pixel-level classification tasks, Jianming Zhang et al. [21] use FCN and dilated convolution to detect cracks from various campus buildings. Zhenqing Liu et al. [22] propose U-Net to detect the concrete cracks of campus buildings. Allen Zhang et al. [23] developed a CNN architecture without pooling layers named CrackNet for automated pavement crack detection on 3D asphalt surfaces. Weidong Song et al. [24] used multiscale dilated convolutional module for automated pavement crack detection. Yupeng Ren et al. [25] used techniques of dilated convolution, spatial pyramid pooling, skip connections, and an optimized loss function for concrete crack detection in tunnels. For backbones, the VGG, DenseNet, and ResNet are frequently used to detect pixel-wise concrete crack [26–28].

In this study, we establish a crack image database of dam concrete surface for the first time. A high-resolution camera acquires these images. However, the high-resolution image is inappropriate as the FCN input directly due to the limited calculating conditions such as GPU memory. To pursue the speed, a destructive way is to shrink the image size, but it will reduce the accuracy, especially for the small pixel width of cracks, as shown in Figure 1(b). A nondestructive solution is to apply the sliding window to divide the high-resolution image into small patches to fit the input size of FCN. However, previous FCNs only adopt the crack patches to alleviate the category imbalance and extensive computation, which has a small percentage of all the patches. Frustratingly, it will produce a higher proportion of false crack predictions with messy distributions in the high-resolution image prediction.

Therefore, we provide a CNN-to-FCN nondestructive semantic segmentation method to solve the problem above. The method firstly adopts sliding window and CNN to locate the crack approximately and then uses FCN to segment the crack predictions. CNN and FCN are trained separately by different datasets. The CNN trains the crack and background patches to ensure the background information is fully utilized, while the FCN trains only the crack patches.

The contributions of this paper mainly include five aspects, as follows:

- (1) Propose a CNN-to-FCN nondestructive semantic segmentation method for dam crack detection
- (2) Apply the ResNetv2 backbone and the DeepLabv3 architecture for dam crack semantic segmentation
- (3) Crop the ResNetv2 backbone bottleneck channels to accelerate networks and improve their performance

- (4) Propose the dense upsampling convolution (DUC) in the upsampling stage of DeepLabv3
- (5) Establish a concrete dam crack dataset (Dam-CrackDataset) for the first time

The content of this paper is described as follows: Section 2 demonstrates the basic CNN and FCN, the mainstream backbone ResNet and the DeepLab architecture, and the DUC module. Section 3 introduces our improved nondestructive semantic segmentation method and improved model for concrete dam crack detection. The Dam-CrackDataset is established to verify the performance of different methods and models in Section 4. Finally, Section 5 concludes this article.

2. CNN and FCN

2.1. Basic CNN and FCN. The basic CNN consists of three parts: the input layer, convolution layers, and pooling layers, and a fully connected multilayer perceptron classifier. Convolution and pooling operations can greatly simplify model complexity and reduce model parameters. The basic CNN is a feature extractor with the advantage that human engineers do not need to design multiple layers of features. Compared to the standard feedforward neural networks, CNN has better learning and adaptive ability due to its unique designs.

The main function of the convolution layer is to convolve the convolution kernel with the input data of the upper layer, in which the convolution kernel in the convolutional layer is applied to feature extraction. It can also reduce the connection between different layers to prevent overfitting and too many parameters. The next layer output can be expressed as follows:

$$y(i, j) = \sum_{m=1}^M \sum_{n=1}^N x(i+m-1, j+n-1)k(m, n), \quad (1)$$

$$i = 1, 2, \dots, p, j = 1, 2, \dots, q,$$

where $y(i, j)$ is the element of the output matrix $\mathbf{y}$ with order $p \times q$; $x(i+m-1, j+n-1)$ is the element of the input matrix $\mathbf{x}$ with order $p \times q$; and $k(m, n)$ is the element of the convolution kernel $\mathbf{k}$ with order $M \times N$.

The pooling layer divides the input data into multiple nonoverlapping regions, and take the maximum value (maximum pooling) or average value (average pooling) of each region. It can eliminate noncritical feature samples, thus improving the training efficiency and estimation accuracy, and the pooling formula is

$$y' = \max_R(y) \text{ or } \text{mean}_R(y), \quad (2)$$

where R is the pooling region.

The fully connected layer expands the two-dimensional data passing through the convolution layer and pooling layer into one-dimensional data. It can be expressed as follows:

$$o = f(\mathbf{w}^T \mathbf{c} + b), \quad (3)$$

where $\mathbf{w}$ is the connection weight vector; $\mathbf{c}$ is the expanded one-dimensional data; b is the bias; o is the output; and f is an activation function which can enhance the network's nonlinear characteristics. For example, rectified linear unit (ReLU) is a popular activation function which can activate the neurons of the neural network sparsely and can be expressed as follows:

$$\text{ReLU}(y(i, j)) = \begin{cases} y(i, j), & y(i, j) > 0, \\ 0, & y(i, j) \leq 0. \end{cases} \quad (4)$$

The difference between FCN and CNN is that FCN converts the fully connected layer of CNN to the convolution layer. The converted convolution layer adopts upsampling to make the output to be the same size as the input image, which can achieve pixel-level prediction and retaining the original input space information of images.

2.2. ResNet Backbone. The ResNet backbone [29, 30] is put forward to solve the problem that deeper networks are more challenging to train. This kind of network is equivalent to add a new channel of the input so that the input can reach the output directly. Then, the optimization objective changes from the original output $H(x)$ to the residual between the $H(x)$ and input x . The ResNet backbone shows excellent characteristics in precision and convergence by using an extremely deep network. The ResNet series have two versions named ResNetv1 and ResNetv2 (Figure 2). ResNetv2 adopts the identity after-addition activation to make information propagation smoother. The asymmetric after-addition activation is equivalent to constructing a preactivation residual unit. In this study, the ResNetv2 model is adopted.

2.3. DeepLab Architecture. DeepLab series [31–34] are semantic segmentation deep learning models developed from FCN model. There are two technical hurdles in the application of basic FCN model: downsampling and spatial invariance. The downsampling method will reduce the resolution, especially at the high-level layers. The spatial invariance means that obtaining object-centric decisions from a classifier requires invariance to spatial transformations, inherently limiting the spatial accuracy. DeepLabv1 employs the atrous convolution algorithm and conditional random field (CRF) to address the downsampling and the spatial invariance, respectively. DeepLabv2 uses atrous spatial pyramid pooling (ASPP) to robustly segment objects at multiple scales. To encode multiscale information, DeepLabv3 proposes a cascaded module and an improved ASPP module. The cascaded module gradually doubles the atrous rates, and the improved ASPP module augmented with image-level features probes the features with filters at multiple sampling rates and effective field-of-views. DeepLabv3+ extends DeepLabv3 by adding a simple yet effective decoder module to refine the segmentation results, especially along object boundaries. For backbone, DeepLabv1 is constructed by VGG-16. DeepLabv2 and DeepLabv3 use the ResNet. DeepLabv3+ adopts the ResNet and Xception.

3. Improved Method and Model

3.1. CNN-to-FCN Method. The flowchart of our CNN-to-FCN nondestructive semantic segmentation method is shown in Figure 3 and demonstrated as follows:

- (i) Step 1: get high-resolution images of dam concrete surface. The images are then manual labelled by pixel level.
- (ii) Step 2: use the sliding window to get the image patches and their annotations. Meanwhile, the patches can be augmented, but the annotations should be nondestructive for FCN. For CNN, annotation of each patch should be further used to generate a new global crack dichotomy annotation. For FCN, the pixel-level annotations are used directly.
- (iii) Step 3: split the patches into training, validation, and test patches. The training patches are applied to train the model, while the validation patches are to choose the best parameters. The test patches are used to estimate the performance of the model.
- (iv) Step 4: train and optimize CNN using the training and validation patches, including crack and background patches.
- (v) Step 5: train and optimize FCN using the training and validation patches, including only the crack patches.
- (vi) Step 6: predict the test patches and judge whether each patch is crack by using the trained CNN.
- (vii) Step 7: classify the crack predictions according to Step 6 on pixel level by using the trained FCN.

Significantly, previous methods only train and test the crack patches, while the improved method trains the crack and background patches to fully utilize the background information. The large-scale background recognition by the improved method can make us focus on the fewer crack patches of the high-resolution image, thus avoiding the messy distributions of false crack predictions.

3.2. Improved Model. In this study, we use the ResNetv2-50 backbone and integrate a dense upsampling convolution (DUC Figure 4) module proposed by Wang et al. [35] into DeepLabv3 architecture (Figure 5) because the DUC operation can be naturally integrated into the DeepLabv3, makes the whole encoding and decoding process end-to-end trainable, and only increases the FLOPs by a small amount.

At the stage of upsampling, the DeepLab series models often used the bilinear interpolation method. As most of the width pixels of cracks are fewer than 4 pixels in our DamCrackDatabase, the bilinear interpolation will more likely miss the fine-detailed information of crack when the downsampling rate is less than 1/4. DUC is more effective than the bilinear interpolation method, and the deconvolution method used by the basic FCN model. The key idea of DUC is to transform the whole label map into a smaller label map with multiple channels. At the upsampling stage, DUC

FIGURE 2: ResNetv1 and ResNetv2 units. The units consist of the same components, while the orders are different. (a) ResNetv1. (b) ResNetv2.

FIGURE 3: The CNN-to-FCN method for dam crack detection.

only needs to reshape the output feature map into the whole label map.

ResNet is invented to deal with the large datasets, such as the ImageNet [36]. However, our DamCrackDatabase is a relatively smaller dataset which may not need many

parameters to learn. Therefore, we cut the bottleneck channels of ResNetv2-50 in half (named ResNetv2-50s) to reduce the learning parameters and accelerate the computing speed. As shown in Table 1, the FLOPs of ResNetv2-50 and DeepLabv3-DUC are shortened about three times as

FIGURE 4: Dense upsampling convolution (DUC) module.

FIGURE 5: Illustration of the DeepLabv3 with the DUC layer. DUC is applied on top of the network for decoding purpose.

TABLE 1: Different backbones and architectures and their FLOPs.

Layer name	Output size	ResNet v2-50	ResNet v2-50 s	DeepLabv3-DUC ResNetv2-50	DeepLabv3-DUC+ ResNetv2-50 s
conv1	112 × 112	7 × 7, 64, stride 2 3 × 3 max pool, stride 2			
conv2_x	56 × 56	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 128 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 128 \end{bmatrix} \times 3$
conv3_x	28 × 28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 4$
conv4_x	14 × 14	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 6$
conv5_x	7 × 7 1 × 1	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$ Average pool, 2-d fc, softmax	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 3$		
conv5_d	14 × 14			$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 3$
conv6_d	14 × 14 1 × 1			$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 256 \end{bmatrix} \times 3$

TABLE 1: Continued.

Layer name	Output size	ResNet v2-50	ResNet v2-50 s	DeepLabv3-DUC ResNetv2-50	DeepLabv3-DUC+ ResNetv2-50 s
conv7_d	14 × 14			[1 × 1, 256]	[1 × 1, 256]
	14 × 14			[1 × 1, 512]	[1 × 1, 512]
DUC	224 × 224			Reshape, 14 × 14 × 512 to 224 × 224 × 2, softmax	
FLOPs		3.87 × 10 ⁹ [33]	1.04 × 10 ⁹	8.72 × 10 ⁹	3.08 × 10 ⁹

TABLE 2: mIoU index of FCNs on validation crack set.

Model name	FCN architecture	Backbone	Bath size	Max mIoU	Total epoch	Relative/absolute training time
A	DeepLabv3	ResNetv2-50s	30	0.27	400	1.00x/59 h
B	DeepLabv3-DUC	ResNetv2-50s	30	0.57 + 111.1%	400	1.04x/61 h
C	DeepLabv3+	ResNetv2-50s	30	0.56 + 107.4%	400	1.73x/102 h

before when using the ResNetv2-50 s backbone. Furthermore, our experiments will demonstrate that the improved backbone and architecture can achieve better performance.

We have also considered the U-Net and DeepLabv3+ architectures. The U-Net is proved to be an effective way to detect concrete crack [22]. However, its training set is less than 1×10^2 , and the VGG-16 backbone has the FLOPs of 1.54×10^{10} , which is about 15 times more than our proposed model's FLOPs (1.04×10^9) by adopting the same input size. That is time-consuming and unacceptable to train our data over 5×10^4 . The DeepLabv3+ with ResNetv2-50 s backbone is used for comparison in Table 2.

4. Experiments and Results

We collected 344 images of cracks from a concrete arch dam surface at different elevations. These images with the resolution of 3456×4608 are labelled on pixel level by domain experts and then cut into 224×224 resolution patches using the sliding window with a stride of 112. We assign each patch a crack label if its centring 200×200 region contains at least one pixel. Otherwise, patches are labelled as background. Meanwhile, the crack patches are augmented by rotating the angle to 90, 180, and 270 degrees. Finally, we get 484092 patches including 402796 background patches and 81296 crack patches with the ratio about 5:1. For CNN, all the patches are divided into training, validation, and test data with the ratio of 4:1:1, respectively. For FCN, we only use the crack patches in the training, validation, and test data, and these data are also set as the ratio of 4:1:1, respectively.

For the two networks, the initial learning rates are both set as 0.001 and the Adam updater which designs an independent adaptive learning rate for different parameters by calculating the first-order moment estimation and the second-order moment estimation of the gradient is used. It is very robust and usually converges quickly and gives pretty good performance and the hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10e - 8$ [37]. The weight decay to use for regularizing the model is set $10e - 5$ to prevent overfitting.

The batch normalization (BN) parameter is set $\epsilon = 10e - 6$ [38], which is a small constant to prevent division by zero when normalizing activations by their variance in BN.

All these algorithms are implemented using TensorFlow (v1.14.1), and used in their original version and performed via a laptop (CPU: Intel i9 9900K @ 3.6 GHz, RAM: 32 GB, GPU: Nvidia GeForce RTX 2080). The class-balanced cross-entropy loss function is used in equation (5) to alleviate the category imbalance:

$$\text{Loss} = -\alpha y \log y' - (1 - \alpha)(1 - y) \log(1 - y'), \quad (5)$$

where $\alpha = (Y_+/Y)$ and $1 - \alpha = (Y_-/Y)$; Y_+ and Y_- denote the crack and background label sets, respectively; y denotes the positive and negative label; and y' denotes the output probability.

The performance indexes are as follows:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ F &= \frac{(1 + \beta^2) * \text{Recall} * \text{Precision}}{\text{Recall} + \beta^2 * \text{Precision}}, \end{aligned} \quad (6)$$

where TP denotes the true positive predictions; FP means the false positive predictions; FN denotes the false negative predictions; F score measures the weighted harmonic mean of Precision and Recall; and β is a weighted factor.

When focusing on the Recall, the β is set larger than 1, while focusing on the Precision, the β should be smaller than 1. We increase the Recall to obtain the TP predictions from the true positive labels as many as possible at the CNN stage. So, the β is set as 2. For CNN, we use the indexes to value all the patches. For FCN, we use the above indexes to estimate each patch, and their mean indexes to value all the patches. If each patch's Precision or Recall is

FIGURE 6: CNN loss and F during training and validation: (a) loss; (b) F .TABLE 3: F index of different CNN backbones on validation set.

Model name	CNN backbone	Bath size	Max F	Total epoch	Relative/absolute training time
I	ResNetv2-50	50	0.87	60	1.00x/49 h
II	ResNetv2-50s	50	$0.88 \pm 1.1\%$	60	0.43x/21 h
III	ResNetv2-50s	50	$0.86 \pm 1.1\%$	60	0.29x/14 h

FIGURE 7: FCN loss and mIoU during training and validation: (a) loss; (b) mIoU.

FIGURE 8: Different models' prediction results on test crack set: (a) original patch, (b) labelled patch, (c) model A, (d) model B, and (e) model C.

TABLE 4: Indexes of ONLY-FCN and CNN-FCN by different thresholds on test set.

Method name	Model II threshold	Model B threshold	TP	FP	TN	FN	mR	mP	mIoU	Relative/absolute prediction time per patch
ONLY-FCN	—	0.999	—	29227	—	—	0.188	0.865	0.182	1.00x/7.9 ms
	—	0.99	—	40719	—	—	-76.4%	+33.5%	-66.7%	
	—	0.9	—	54502	—	—	0.408	0.806	0.370	
	—	0.9	—	54502	—	—	-48.7%	+24.4%	-32.4%	
	—	0.5	—	69316	—	—	0.645	0.729	0.514	
CNN-FCN	0.001	0.5	13496	56779	10364	43	-18.9%	+12.5%	-6.0%	0.73x/5.8 ms
	0.01	0.5	13410	41299	25844	129	0.795	0.648	0.547	
	0.1	0.5	12885	10988	56155	654	+0.0%	+0.0%	+0.0%	
	0.5	0.5	12151	2851	64292	1388	0.795	0.648	0.548	
	0.5	0.5	12151	2851	64292	1388	+0.0%	+0.0%	+0.2%	
CNN	—	—	—	—	—	—	0.798	0.649	0.550	0.84x/6.6 ms
	—	—	—	—	—	—	0.801	0.650	0.552	0.73x/5.8 ms
	—	—	—	—	—	—	+0.8%	+0.3%	+0.9%	0.53x/4.2 ms

TN denotes the true background predictions.

FIGURE 9: R_{time} and D_{FP} of all the image: (a) R_{time} ; (b) D_{FP} .

FIGURE 10: Continued.

FIGURE 10: Different methods for high-resolution image nondestructive semantic segmentation: (a) original image, (b) labelled image, (c) ONLY-FCN, and (d) CNN-FCN.

incomputable, then the patch is not taken into account, respectively.

For FCN, we also add the common index as follows:

$$mIoU = \frac{1}{M} \sum_{m=1}^M \frac{TP}{TP + FP + FN}, \quad (7)$$

where M denotes the number of patches which are computable.

The CNNs are trained for 60 epochs. Observing the curves in Figure 6(a), the loss decreases gradually on the training set, while the validation set converges after about the 20th epoch. As shown in Table 3 and Figure 6(b), model II changes the backbone of model I to ResNetv2-50 s, and obtain the best F and saves more than half of the training time. Model III cut the bottleneck channels of ResNetv2-50 to a quarter (named ResNetv2-50 s) and obtain the worst F , which shows that the bottleneck channels cannot be cut too much. Therefore, we finally choose the ResNetv2-50 s as the backbone for the next FCNs.

The FCNs are trained for 400 epochs because we found that model B and model C need more time to obtain a better mIoU (Table 2 and Figure 7(b)). As can be seen from Figure 7, the loss and mIoU of model A converge the most quickly than model B and model C on both the training and validation sets. Model B substantial increases the mIoU more than 110% by adding the DUC module, and shows the best prediction performance on the test crack set (Table 2

and Figure 8). Model C is the most time-consuming over 100 hours, and its mIoU is worse than model B.

Significantly, the loss in model B has the poorest convergence performance than model A and model C after about the 250th epoch, which shows that the loss function may not be the best training way to achieve a better mIoU.

Thresholds are vital parameters to tune the models' indexes. To compare the distributions of false crack (FP) predictions by using the methods ONLY-FCN and CNN-FCN (our improved method), we assign each patch predicted by model B and model II FP if its region contains at least one FP pixel. As shown in Table 4, the threshold of model B can be applied to reduce the FP and increase the mean Precision (mP) for method ONLY-FCN. However, it will reduce the mean Recall (mR) and mIoU at the same time. The threshold of model II can be applied to reduce the FP more efficiently than the threshold of model B. As the FN growth rate is much less than the rate of FP decline, and the mR and mIoU rise by the increase of the model II threshold without loss of mean Precision (mP). Therefore, we can coordinate the FP and FN by tuning the threshold of model II to adapt to different requirement on recognition. Meanwhile, we use the index as follows to evaluate model II FP distribution:

$$DFP = \frac{FP}{TP + FP + TN + FN}. \quad (8)$$

For prediction time-consuming, we can conclude the function as follows:

$$\begin{aligned}
T_{\text{FCN}} &= t_{\text{FCN}}N, \\
T_{\text{CNN-FCN}} &= t_{\text{CNN}}N + t_{\text{FCN}}N_{\text{cp}}, \\
R_{\text{time}} &= \frac{T_{\text{CNN-FCN}}}{T_{\text{FCN}}} = \frac{t_{\text{CNN}}}{t_{\text{FCN}}} + \frac{N_{\text{cp}}}{N},
\end{aligned} \tag{9}$$

where $T_{\text{CNN-FCN}}$ and T_{FCN} denote the time-consuming on all the patches using the methods CNN-FCN and ONLY-FCN, respectively; t_{CNN} and t_{FCN} denote the time-consuming on each patch using CNN and FCN, respectively; N represents the number of all the patches; N_{cp} represents the number of crack predictions from model II; and R_{time} denotes the ratio of $T_{\text{CNN-FCN}}$ to T_{FCN} .

For our proposed model, the $(t_{\text{CNN}}/t_{\text{FCN}})$ is 0.53 (Table 4), so if the (N_{cp}/N) is smaller than 0.47, our CNN-FCN method can reduce the prediction time-consuming. Note that the test set has a low proportion of crack patches (about 15%) and the crack predictions (TP + FP) can be controlled in a close range. As we have augmented the crack patches, for the real-world dam surface, the crack patches will have a lower proportion. Therefore, our improved method will reduce the prediction time, which becomes more practicable for the engineering applications.

The R_{time} and D_{FP} are used to estimate all the 344 images. The thresholds of model II and model B are set 0.1 and 0.5, respectively. As shown in Figures 9 and 10, our proposed method CNN-FCN can reduce the prediction time to 76% as before and extremely eliminate the messy distribution of FP from 84% to 15%.

5. Conclusions

Image with high resolution can acquire more information and ensure clarity when the camera frame is enlarged. It is beneficial for the dam with wide spaces and small width of cracks which are hard to discover.

To nondestructive segment the dam surface crack pixels of the image with high resolution, this study proposed a CNN-to-FCN method. Comparing with the previous FCNs, our method can extremely avoid the higher proportion of false crack predictions and their messy distributions in the high-resolution image through the large-scale background information recognition by CNN. The improved method also makes the computer easy to get the primary information of dam surface crack and achieve better mIoU than previous FCNs. A few true crack patches are missed at the CNN prediction stage, which should be controlled in a reasonable range by tuning the threshold. Meanwhile, the improved method can also reduce the prediction time when the image has a low proportion of crack patches, which becomes more practicable for the engineering applications.

To obtain better results, we also modify the ResNetv2 backbone and DeepLabv3 architecture by reducing the bottleneck channels and adding a DUC module. The results indicated that the modified model is efficient and dramatically increases the mIoU on dam crack recognition.

Note that our proposed method is universal. However, for a deep learning model, both data and model selection affect the final performance. We have augmented the data from the 344 images to improve the deep learning model's generalization ability. Meanwhile, we proposed an improved model for nondestructive semantic segmentation. However, more data from different dam scenarios will be more representative. We need to collect more representative data to verify and further improve our proposed model in the future.

Finally, the DamCrackDataset is established for the first time and can be found in https://figshare.com/articles/DamCrackDataset_crack_encrypted_zip/12362159 for further study on concrete dam surface crack detection.

Data Availability

The DamCrackDataset used to support the findings of this study is currently under embargo while the research findings are commercialized. Requests for data, 6 months after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the State Grid Hunan Electric Power Company Limited Science Project (no. 5216A518000N). The authors would like to thank Yan Zhaohui, Chen Shiqiao, and Zhang Zhichao of the State Grid Hunan Hydropower Company for providing the verification data.

References

- [1] F. Kang, J. Li, and J. Dai, "Prediction of long-term temperature effect in structural health monitoring of concrete dams using support vector machines with jaya optimizer and salp swarm algorithms," *Advances in Engineering Software*, vol. 131, no. 5, pp. 60–76, 2019.
- [2] F. Kang, J. Li, S. Zhao, and Y. Wang, "Structural health monitoring of concrete dams using long-term air temperature for thermal effect simulation," *Engineering Structures*, vol. 180, no. 2, pp. 642–653, 2019.
- [3] A. Mohan and S. Poobal, "Crack detection using image processing: a critical review and analysis," *Alexandria Engineering Journal*, vol. 57, no. 2, pp. 787–798, 2017.
- [4] H.-B. Yun, S. Mokhtari, and L. Wu, "Crack recognition and segmentation using morphological image-processing techniques for flexible pavements," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2523, no. 1, pp. 115–124, 2015.
- [5] Y. Fujita, Y. Mitani, and Y. Hamamoto, "A method for crack detection on a concrete structure," in *Proceedings of the 18th International Conference on Pattern Recognition*, Hong Kong, China, August 2006.
- [6] I. Abdel-Qader, O. Abudayyeh, and M. E. Kelly, "Analysis of edge-detection techniques for crack identification in bridges,"

- Journal of Computing in Civil Engineering*, vol. 17, no. 4, pp. 255–263, 2003.
- [7] P. Prasanna, K. J. Dana, N. Gucunski et al., “Automated crack detection on concrete bridges,” *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 591–599, 2016.
 - [8] T. Nishikawa, J. Yoshida, T. Sugiyama, and Y. Fujino, “Concrete crack detection by multiple sequential image filtering,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 27, no. 1, pp. 29–47, 2012.
 - [9] H. Kim, E. Ahn, M. Shin, and S.-H. Sim, “Crack and noncrack classification from concrete surface images using machine learning,” *Structural Health Monitoring*, vol. 18, no. 3, pp. 725–738, 2019.
 - [10] X. Fan, J. Wu, P. Shi, X. Zhang, and Y. Xie, “A novel automatic dam crack detection algorithm based on local-global clustering,” *Multimedia Tools and Applications*, vol. 77, no. 20, pp. 26581–26599, 2018.
 - [11] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
 - [12] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 3431–3440, Boston, MA, USA, June 2015.
 - [13] A. Garciagarcia, S. Ortsescolano, S. Oprea, V. Villenamartinez, and J. Garciarodriguez, “A review on deep learning techniques applied to semantic segmentation,” 2017, <https://arxiv.org/abs/1704.06857>.
 - [14] S. Dorafshan, R. J. Thomas, and M. Maguire, “Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete,” *Construction and Building Materials*, vol. 186, pp. 1031–1045, 2018.
 - [15] 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.
 - [16] U. H. Billah, H. La, N. Gucunski, and A. Tavakkoli, “Classification of concrete crack using deep residual network,” in *Proceedings of the 9th International Conference on Structural Health Monitoring of Intelligent Infrastructure*, St Louis, MO, USA, August 2019.
 - [17] C. Feng, M.-Y. Liu, C.-C. Kao, and T.-Y. Lee, “Deep active learning for civil infrastructure defect detection and classification,” in *Proceedings of the ASCE International Workshop on Computing in Civil Engineering*, June 2017.
 - [18] W. Ricardo Leal Da Silva and D. S. D. Lucena, “Concrete cracks detection based on deep learning image classification,” in *Proceedings of the Eighteenth International Conference of Experimental Mechanics*, Brussels, Belgium, July 2018.
 - [19] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, “Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection,” *Construction and Building Materials*, vol. 157, pp. 322–330, 2017.
 - [20] K. Jang, N. Kim, and Y.-K. An, “Deep learning-based autonomous concrete crack evaluation through hybrid image scanning,” *Structural Health Monitoring*, vol. 18, no. 5–6, pp. 1722–1737, 2019.
 - [21] J. Zhang, C. Lu, J. Wang, L. Wang, and X.-G. Yue, “Concrete cracks detection based on FCN with dilated convolution,” *Applied Sciences*, vol. 9, no. 13, p. 2686, 2019.
 - [22] Z. Liu, Y. Cao, Y. Wang, and W. Wang, “Computer vision-based concrete crack detection using U-net fully convolutional networks,” *Automation in Construction*, vol. 104, pp. 129–139, 2019.
 - [23] A. Zhang, K. C. P. Wang, B. Li et al., “Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network,” *Computer-aided Civil and Infrastructure Engineering*, vol. 32, no. 10, pp. 805–819, 2017.
 - [24] W. Song, G. Jia, H. Zhu, D. Jia, and L. Gao, “Automated pavement crack damage detection using deep multiscale convolutional features,” *Journal of Advanced Transportation*, vol. 2020, Article ID 6412562, 11 pages, 2020.
 - [25] Y. Ren, J. Huang, Z. Hong et al., “Image-based concrete crack detection in tunnels using deep fully convolutional networks,” *Construction and Building Materials*, vol. 234, Article ID 117367, 2020.
 - [26] S. Li, X. Zhao, and G. Zhou, “Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network,” *Computer-aided Civil and Infrastructure Engineering*, vol. 34, no. 7, pp. 616–634, 2019.
 - [27] C. V. Dung and L. D. Anh, “Autonomous concrete crack detection using deep fully convolutional neural network,” *Automation in Construction*, vol. 99, pp. 52–58, 2019.
 - [28] M. M. Islam and J. Kim, “Vision-based autonomous crack detection of concrete structures using a fully convolutional encoder-decoder network,” *Sensors*, vol. 19, no. 19, p. 4251, 2019.
 - [29] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.
 - [30] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” 2016, <https://arxiv.org/abs/1603.05027>.
 - [31] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Semantic image segmentation with deep convolutional nets and fully connected CRFs,” 2014, <https://arxiv.org/abs/1412.7062>.
 - [32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.
 - [33] L. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” 2017, <https://arxiv.org/abs/1706.05587>.
 - [34] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Proceedings of the European Conference on Computer Vision*, Munich, Germany, September 2018.
 - [35] P. Wang, P. Chen, Y. Yuan et al., “Understanding convolution for semantic segmentation,” in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1451–1460, Lake Tahoe, NV, USA, March 2018.
 - [36] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Feifei, “ImageNet: a large-scale hierarchical image database,” in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 248–255, Miami, FL, USA, June 2009.
 - [37] D. P. Kingma and J. B. Adam, “A method for stochastic optimization,” in *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA, May 2015.
 - [38] S. Ioffe and C. Szegedy, “Batch normalization: accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the International Conference on Machine Learning*, Lille, France, July 2015.