Hindawi
Advances in Materials Science and Engineering
Volume 2021, Article ID 5520515, 13 pages
https://doi.org/10.1155/2021/5520515

Research Article

Attention-Based Convolutional Neural Network for Pavement Crack Detection

Haifeng Wan,[1] **Lei Gao**,[2] **Manman Su**,[1] **Qirun Sun**,[1] **and Lei Huang**[1]

[1]*School of Civil Engineering, Yantai University, Yantai, Shandong 264005, China*
[2]*CSIRO, Waite Campus, Urrbrae, SA 5064, Australia*

Correspondence should be addressed to Haifeng Wan; wanhaifengytu2015@126.com

Received 3 February 2021; Revised 11 March 2021; Accepted 20 March 2021; Published 8 April 2021

Academic Editor: Fuat Kara

Achieving high detection accuracy of pavement cracks with complex textures under different lighting conditions is still challenging. In this context, an encoder-decoder network-based architecture named CrackResAttentionNet was proposed in this study, and the position attention module and channel attention module were connected after each encoder to summarize remote contextual information. The experiment results demonstrated that, compared with other popular models (ENet, ExFuse, FCN, LinkNet, SegNet, and UNet), for the public dataset, CrackResAttentionNet with BCE loss function and PRelu activation function achieved the best performance in terms of precision (89.40), mean IoU (71.51), recall (81.09), and F1 (85.04). Meanwhile, for a self-developed dataset (Yantai dataset), CrackResAttentionNet with BCE loss function and PRelu activation function also had better performance in terms of precision (96.17), mean IoU (83.69), recall (93.44), and F1 (94.79). In particular, for the public dataset, the precision of BCE loss and PRelu activation function was improved by 3.21. For the Yantai dataset, the results indicated that the precision was improved by 0.99, the mean IoU was increased by 0.74, the recall was increased by 1.1, and the F1 for BCE loss and PRelu activation function was increased by 1.24.

1. Introduction

Pavement cracks are a common symptom and an early sign of potential damages and degradation in the pavements [1–4]. However, due to heavy traffic and drastic environmental changes, cracks can significantly affect the performance and functionality of the asphalt pavement projects and eventually lead to structural deterioration and a shortened service time.

Early and frequent inspection can help collect asphalt pavement condition data for in-depth analysis and strategic decisions. Using this information, appropriate maintenance measures can be arranged to prevent pavement failure at an early stage. However, manual inspections of asphalt pavement require massive time involvement and labor costs, and the results are not accurate [5–9].

In this case, some automated vision-based techniques for road damage detection have been investigated. Meanwhile, the development of deep learning in computer vision [10, 11] has significantly improved the crack detection accuracy of camera-captured images [12–15]. Scholars have proposed pixel-level methods with deep convolutional neural networks (CNN) to detect cracks. For crack segmentation, UNet and DenseNet were performed. The results showed high performance and accuracy to a certain extent [16, 17].

In the past few decades, the crack detection technology based on captured images has been widely studied and applied [12, 18–20]. Compared with subjective and labor-intensive manual approaches [19], image-based methodologies were consistent and objective, thus leading to reduced labor costs and higher detection efficiency. In general, image-based methodologies can be divided into different categories, including rule-based methods, machine learning-based methods, and deep learning-based methods. Rule-based methods used a variety of filter algorithms and image processing techniques [12–15]. S. Chambon et al. [20] proposed a two-step multiscale method. The first step was to use adaptive filtering binarization, and the second step was

to refine the binarization by segmentation based on Markov model. James Tsai et al. [21] assessed the crack detection performance in different lighting conditions with poor contrast by emerging 3D laser technology. M. Salman et al. [22] developed a novel technology for the automatic detection and identification of cracks from digital pavement images. Merazi-Meksen et al. [23] used a mathematical method to extract the discontinuity-related pixels and used pattern recognition technology to characterize the discontinuity. Due to the complexity of the texture of pavement surfaces, and the irregularity of the crack morphologies, scholars tried to detect the cracks with machine learning-based algorithms. Qin Zou et al. [24] developed CrackTree, which was a fully automatic crack detection method using pavement images. Ghada Moussa et al. [17] presented a novel reliable method, which used flexible pavement images acquired in the asphalt pavement surveys to automatically detect, classify, and estimate the cracks. Miguel Gavilán et al. [25] proposed a seed-based crack detection method for asphalt pavement by combining Multiple Directional Non-Minimum Suppression (MDNMS) and a symmetry check.

However, machine learning-based approaches are highly dependent on input features and require the researchers to have a broad knowledge of the features. In recent years, as a branch of machine learning, deep learning, especially artificial neural network (ANN) [26, 27], has been rapidly applied due to its superior performance in various fields [28–33], including image classification and semantic segmentation. In the report by Gopalakrishnan et al. [31], a Deep Convolutional Neural Network (DCNN) was trained on the "big data" ImageNet database and applied in their study. Zhenqing Liu et al. [32] adopted UNet to detect the concrete cracks. The trained UNet can identify the location of cracks from the input original images under various conditions. Ankang Ji et al. [34] proposed an integrated crack detection method based on the convolutional neural network DeepLabv3+ and developed an algorithm to quantify the cracks at the pixel level. Cao Vu Dung et al. [35] proposed a crack detection method based on a deep full convolutional network (FCN) and applied this method for semantic segmentation of concrete crack images. Some other semantic segmentation methods were also proposed. However, none of the semantic segmentation methods were employed in crack detection involving engineering applications. Adam Paszke et al. [36] developed a new deep neural network architecture, i.e., efficient neural network (ENet) for tasks requiring low latency operation. Zhenli Zhang et al. [37] proposed a new framework called ExFuse to fill in the gap between low-level and high-level features. The developed framework leads to significant improvement of the segmentation quality (by 4.0%). Abhishek Chaurasia et al. [38] designed a novel neural network architecture from scratch, which can be specifically applied for semantic segmentation. Vijay Badrinarayanan et al. [39] designed an encoder network architecture with the same topology as the 13 convolutional layers in the VGG16 network [40, 41]. Abhishek Chaurasia et al. [38] designed a novel deep neural network architecture, namely, LinkNet, which can learn without significantly increasing the number of parameters. SegNet was designed by Vijay et al. [39]. It had a core trainable segmentation engine and contained an encoder network, a corresponding decoder network, and a pixel-wise classification layer. However, due to impact from shadows, uneven lighting conditions, or crack shape irregularity, it was still not accurate enough for pixel-level crack segmentation.

In this study, an encoder-decoder network-based architecture named CrackResAttentionNet was proposed to detect asphalt pavement cracks, as well as pixel-level image segmentation. The overall process of the CrackResAttentionNet training process is shown in Figure 1. In the following sections, the structures of CrackResAttentionNet, model evaluation method, and model optimization method are described. From the structure of CrackResAttentionNet itself, the encoder mainly used the convolutional layers of ResNet-34 to extract image features, and another encoder layer was added to better extract information. The decoder employed the deconvolutional layers to perform the semantic segmentation of pixels with and without crack. Additionally, the position attention module and channel attention module were connected after each encoder to summarize remote context information. The two attention modules were fused proportionally to emphasize more the position information. The output of each encoder layer was fused with attention output and linked with the corresponding decoder. In addition, the output of last decoder was fed into the decoder as input. In this way, the decoder and its upsampling operations can use spatial information to help improve the accuracy of prediction.

The organization of this paper is as follows. The "Introduction" section reviews some related studies on image-based crack detection and segmentation; the "Methodology" section describes the methodologies related to technical background, including Convolutional Neural Network (CNN) and a group of layers, deconvolution, unpooling, Position Attention, and Channel Attention, and encoder-decoder network; the "Network Architecture" section explains the CrackResAttentionNet architecture in detail; the "Experimental Program" section lists the details of data acquisition, data generation, and experimental program; the "Results and Discussion" section presents the experiment results and the analysis results; the "Summary and Conclusions" section summarizes the conclusions and our findings; and finally, the "Suggestions for Future Research" section looks at the future research prospects and suggestions.

2. Methodology

This section briefly describes the methodologies related to the technical background, including Convolutional Neural Network (CNN) and a group of layers, deconvolution, unpooling, Position Attention, and Channel Attention, and encoder-decoder network.

The convolution layer was a linear operation used to process the product of a set of weights and the input. The multiplication was performed between the input data array and a two-dimensional weight array, which was called filter

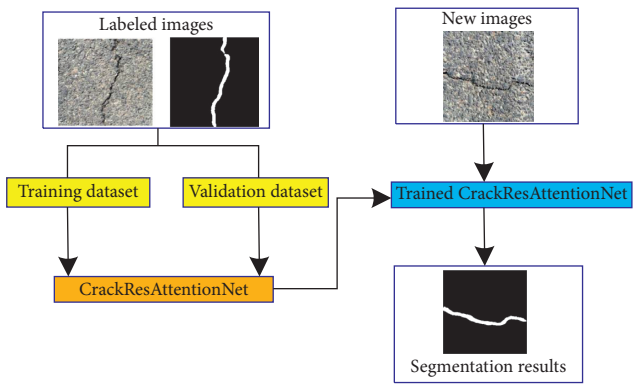

Figure 1: Process for detecting asphalt pavement crack segmentation by CrackResAttentionNet.

or kernel. The kernel had a smaller size than the input, and the multiplication calculation between the kernel and the input data was based on dot product. Dot product is an element-wise multiplication operation between two 2D arrays. Then, the products are added together, generating a single value as the output. As the calculation progressed, the kernel moved along the width and height of the image, which produced a 2D image representation of the receptive region. The generated image representation indicated the response of the kernel at each spatial position of the image. The kernel slid on the image with the step size of "stride" [8]. The pooling layer helped reduce the spatial dimension of the representation, which can lower the number of iterations, computation size, and weight [8]. Several pooling options are currently available, namely, the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and max pooling, which can obtain the maximum value from the subarrays of the input array.

A nonlinear function was employed as an activate function. Dropout was used to process the overfitting problem for neural networks. The connections between neurons were disconnected at a fixed dropout rate. During training, as the training progressed, the parameters of the preceding layers changed, which led to the change in the distribution of inputs. Thus, the current layer needed to be updated to reflect the actual distribution. The output from the previous activation layer was normalized by Batch normalization. In detail, the Batch mean was subtracted, and the resulting value was divided by the Batch standard deviation. Then, the most likely classification was identified by the softmax layer. In the most likely classification, the class had the highest probability [8]. The probability distribution with a sum of one was output. Pooling in a convolution network was used to abstract activations in a receptive field with a single representative value, thus filtering the noisy activations in the lower layer. Unpooling operation performed the reverse operation of pooling and reconstructed the original size of activations. The unpooling can be implemented using the approach proposed in References [9, 25]. Briefly, unpooling recorded the positions of the maximum activations obtained by the pooling operation, performed the reverse operation of pooling, and reconstructed the original size of the activations. As a result, each activation was placed back to the original position before pooling operation. The deconvolution layer used operations with multiple learned filters similar to convolution to make the sparse activations much denser. A convolutional layer connected multiple input activations within a filter window to a single activation; in contrast, deconvolutional layers associated a single input activation with multiple outputs. The deconvolutional layer then output an enlarged and dense activation map. Encoder-decoder network was a widely used form of semantic neural network for image segmentation. The encoder may consist of a few convolutional layers to obtain the feature maps, and the decoder was used to extend the features extracted by the encoder so that the output probability map and the input image had matching sizes.

3. Network Architecture

An encoder-decoder network-based architecture called "CrackResAttentionNet" was proposed to segment crack pixels. CrackResAttentionNet consisted of two parts: encoder and decoder. Between each encoder and decoder, an attention module was added behind the encoder and linked to the decoder. The encoder network consisted of pretrained ResNet-34 [41] as its main encoder layer, and the latter part was connected with non-ResNet-34 encoders. Each encoder included a corresponding decoder layer, and the final decoder output was fed into an output block containing a deconvolution function to perform pixel prediction.

Attention was obtained for each ResNet-34 encoder layer, and the output of the attention was added to the original encoder output. The outputs from the encoder layer and the previous decoder layer were fed into the next decoder layer as input.

3.1. Encoder. The architecture of CrackResAttentionNet is shown in Figure 2. From the figure, the encoder starts with the pretrained ResNet-34 in the block layer and performs convolution, Batch normalization, Relu activation, and max pooling. The next layers include layer 1, layer 2, layer 3, and layer 4 of ResNet-34.

The last part of the encoder is a diverse encoder block, as shown in Figure 3. The diverse encoder block performed convolution, in which the kernel size was 2×2, the stride was 2, and the padding was 0. The size of the output matrix was divided by 2. After the convolution, Dropout, Batch normalization, and PRelu activation were connected.

3.2. Decoder. The last encoder block was directly fed into decoder 5. As shown in Figure 4, decoder 5 contains convolution 1 module and convolution 2 module, deconvolution module, and convolution 3 modules. Each convolution module performed convolution. The kernel size was set to 1×1, the stride was 1, and the padding was 0. Then, the matrix with the same size can be obtained as the input. Dropout, Batch normalization, and PRelu activation were after the convolution. The deconvolution module first performed deconvolution by function ConvTranspose2d,

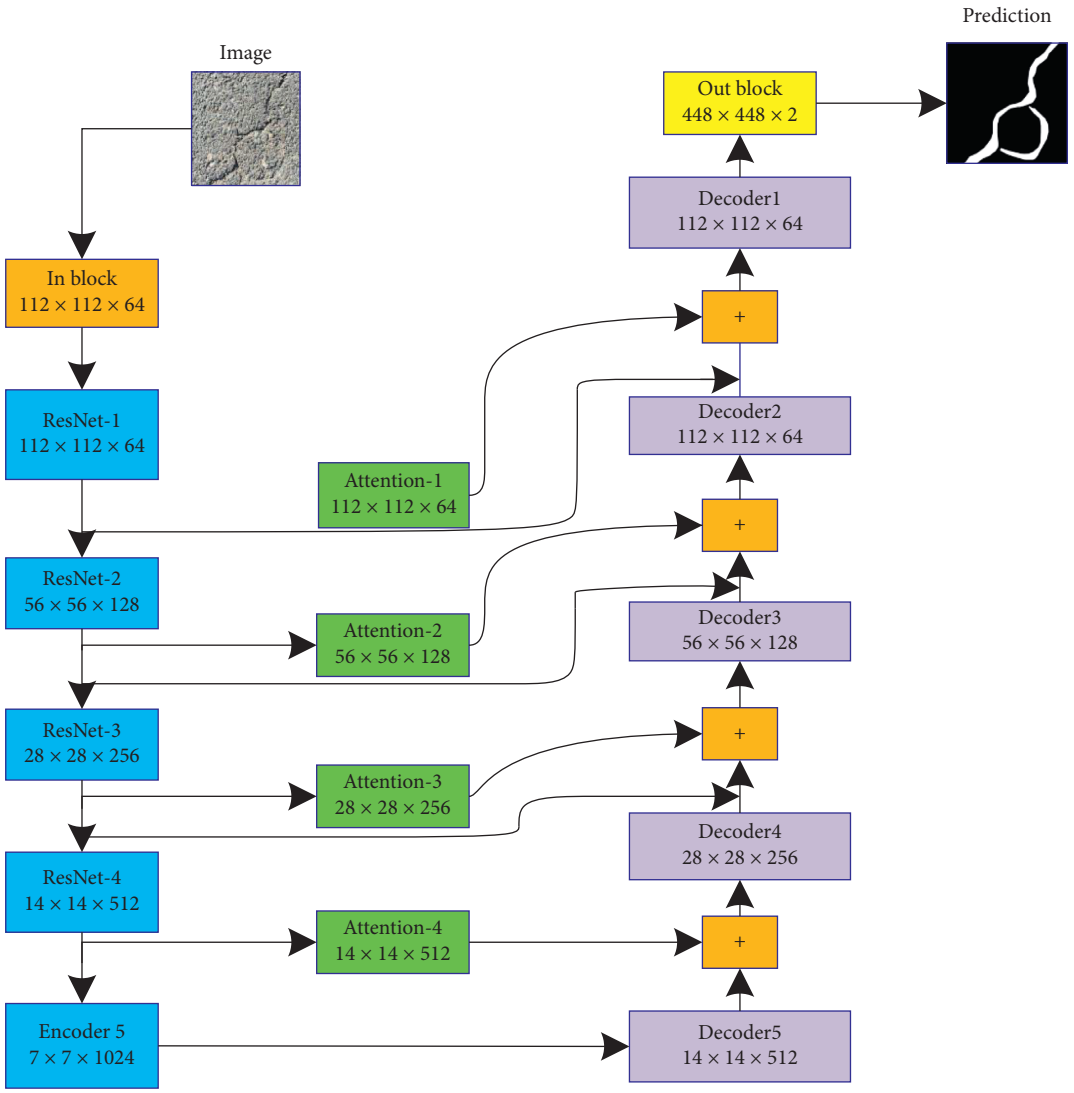

Figure 2: An illustration of the CrackResAttentionNet architecture.

with a kernel size of 2×2 and a stride of 2. As a result, the size of the input matrix was multiplied by 2. Then, the next step was the Batch normalization and PRelu.

As shown in Figure 5, the four decoders, i.e., decoder 4, decoder 3, decoder 2, and decoder 1, have the same structure. They contain convolution 1 module, deconvolution module, and convolution 2 module. Both convolution modules performed convolution with a kernel size of 1×1, a stride of 1, and a padding of 0. Then, the matrix with the same size was produced as the input. Dropout, Batch normalization, and PRelu activation were connected. The deconvolution module performed deconvolution by function ConvTranspose2d, in which the kernel size was 3×3 and the stride was 2. As a result, the input size was multiplied by 2. Then, the next step was the Batch normalization and PRelu.

As shown in Figure 6, the out block contains the deconvolution module 1, the convolution modules 1 and 2, and the deconvolution module 2. The first deconvolution module performed deconvolution through the function ConvTranspose2d, in which the kernel size was 3×3 and the stride was 2. As a result, the size of the input matrix was multiplied by 2. Then, the next step was the Batch normalization and PRelu. The two convolution modules had the same structure and performed convolution with a kernel size of 3×3, a stride of 1, and a padding of 0. Both the output matrix and the input matrix had the same size. Dropout, Batch normalization, and PRelu activation were connected. The last deconvolution module simply performed a deconvolution by function ConvTranspose2d, in which the kernel size was set to 2×2 and the stride was 2. This operation caused the size of the input matrix to be multiplied by 2. The output was the predicted image, which had the same size as the input image.

3.3. Attention. Cracks are different in scales, lighting, and views, which can affect the segmentation effect. Since the convolution operations would result in local reception, the pixels with the same label may have different features [42].

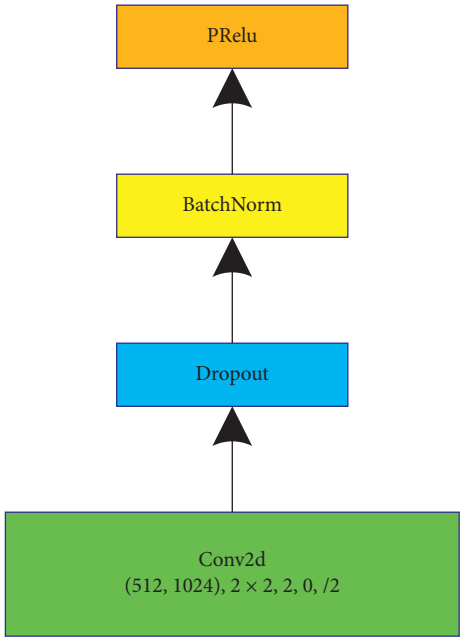

Figure 3: Encoder5 block.

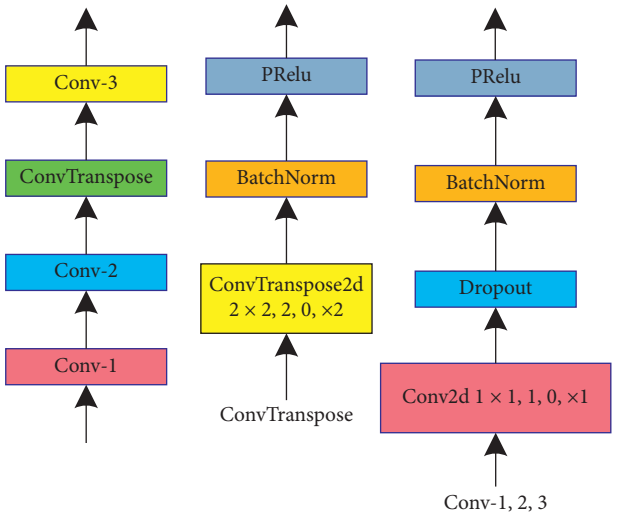

Figure 4: Decoder 5.

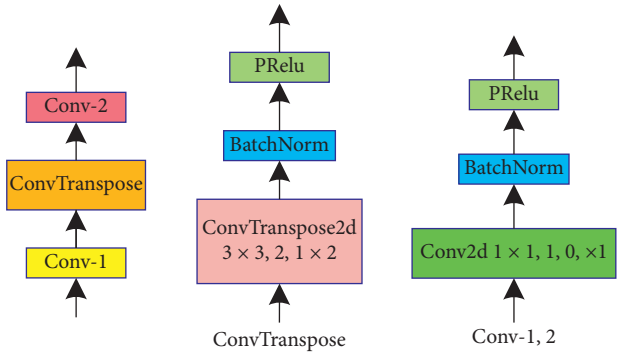

Figure 5: Decoders 4, 3, 2, and 1.

The differences can cause inconsistency across classes and influence the accuracy. Therefore, the attention mechanism was used to establish associations between features, thus extracting global context information. Paying more attention to crack segmentation can aggregate remote context information and enhance the feature representation capability.

As shown in Figure 7, in order to get the global context of local features from the network, two types of attention modules are added. Given the output from the ResNet-34 decoder layer, a convolution layer was firstly applied to obtain features through different layers, which would not change the shape of the input.

In the first attention module, i.e., position attention module [42], a wider range of context information was extracted into local features. The feature maps A, B, and C were generated from the convolution layer. The following condition was satisfied: $\{A, B, C, D\} \in R^{C\times H\times W}$. Then, A, B, and C were reshaped to $R^{C\times N}$, where N was the number of pixels and can be calculated by $N = H \times W$. B was transposed to $R^{N\times C}$, and a matrix multiplication was conducted between the transpose of C and B. The shape of the resulting matrix was $R^{N\times N}$. Then, the softmax layer was used to calculate the spatial attention map $S \in R^{N\times N}$ [42]:

$$s_{ji} = \frac{\exp\left(B_i \cdot C_j\right)}{\sum_{i=1}^{N} \exp\left(B_i \cdot C_j\right)}, \tag{1}$$

where s_{ji} is the influence of the ith position on jth position. When the feature representations of two locations were more similar to each other, the correlation between them was higher [42].

Then, S was transposed into the same shape $R^{N\times N}$, but the dimension was different. Matrix multiplication between A and the transposed S was performed, and the result was $R^{N\times C}$. This result was reshaped to be $R^{C\times H\times W}$. Finally, the result was multiplied by a scale parameter α, an element-wise sum operation was conducted with the original convolutional features D, and the final output $H \in R^{C\times H\times W}$ was obtained as follows [42]:

$$H_j = \alpha \sum_{i=1}^{N} \left(s_{ji} A_i\right) + D_j, \tag{2}$$

where α had the initial value of 0 and gradually learned to assign more weights [42].

Similarly, the channel attention module shown in Figure 7(b) can emphasize interdependent feature maps and improve the representation of specific semantic features.

A convolution firstly was performed to extract the feature maps E, F, G, H, and $\{E, F, G, H\} \in R^{C\times H\times W}$.

Then, the E, F, G were reshaped to $R^{C\times N}$, where N was the number of pixels and can be calculated by $N = H \times W$. F was transposed to $R^{N\times C}$, and a matrix multiplication of transposed F and E with the matrix was performed [42]. The resulting matrix with the shape of $R^{N\times N}$ was obtained, and a softmax layer was applied to calculate the spatial attention map $X \in R^{N\times N}$:

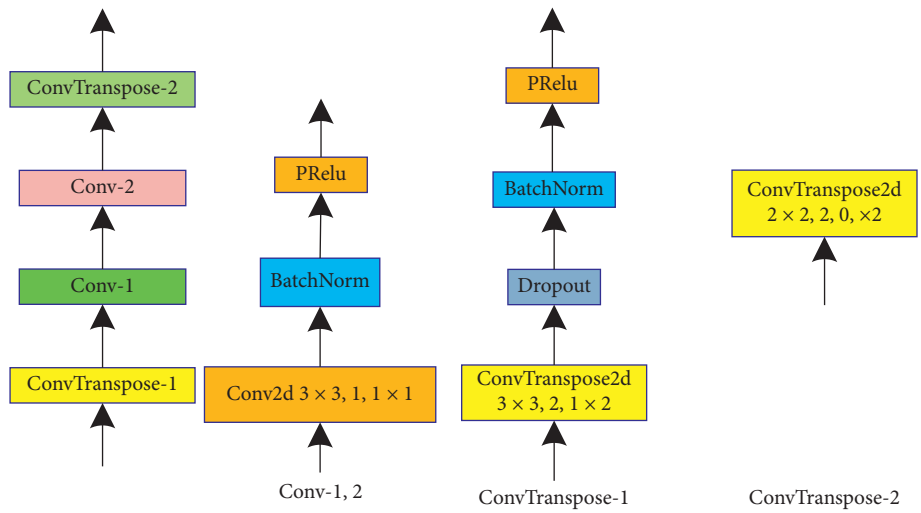

Figure 6: Out block.

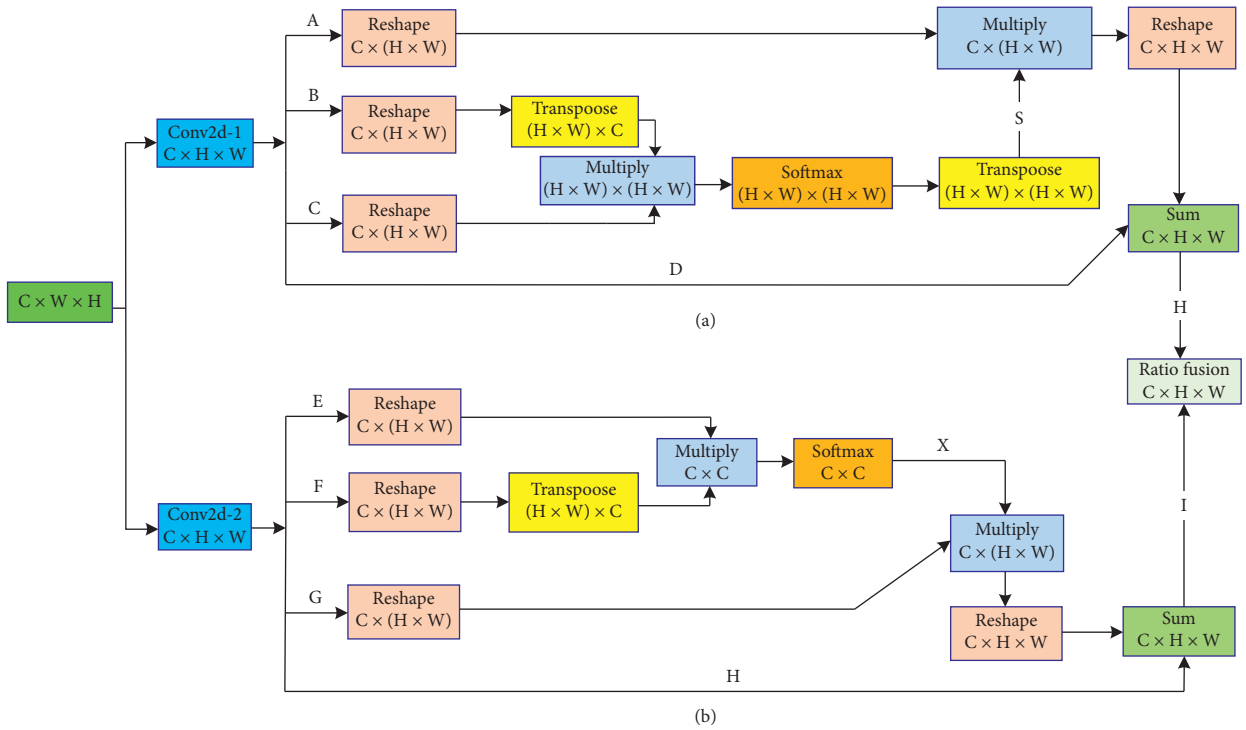

Figure 7: Attention modules. (a) Position attention module. (b) Channel attention module.

$$x_{ji} = \frac{\exp\left(E_i \cdot F_j\right)}{\sum_{i=1}^{C} \exp\left(E_i \cdot F_j\right)}, \tag{3}$$

where x_{ji} is the impact of the ith position on jth position.

Then, a matrix multiplication between the softmax result X and reshaped G was conducted, and the result of $R^{C\times N}$ was obtained. This result was reshaped to $R^{C\times H\times W}$. Finally, the result was multiplied by a scale parameter β, an element-wise sum operation was conducted with the original convolutional features H, and the final output $I \in R^{C\times H\times W}$ was obtained as follows [42]:

$$I_j = \beta \sum_{i=1}^{C} \left(X_{ji} G_i\right) + H_j, \tag{4}$$

where β had the initial value of 0 and gradually learned to assign more weights [42].

Therefore, the output feature of each channel was a weighted sum of the features in all channels and the original features. The output feature of each channel can model the remote semantic dependencies between feature maps.

After obtaining the results from both the position attention H and the channel attention with the shape

$C \times H \times W$, they can be fused. The position attention had a proportion of Ø, and correspondingly, the channel attention had a proportion of $(1 - Ø)$. The element-wise sum proportional operation can be calculated using equation (5):

$$J = Ø * H + (1 - Ø) * I, \tag{5}$$

where Ø is a hyperparameter, and 0.8 can be used to emphasize position attention for crack segmentation.

3.4. Encoder and Decoder Bridge. From Figure 2, each encoder layer is bridged with a decoder. This was done by fusing the output of each encoder layer, the results of attention are indicated in Section 4.3, and the output is from the previous decoder layer. By feeding this fusion output into the decoder, the information of the encoder layer and corresponding attention can be captured.

4. Experiment Setup

4.1. Data Preparation and Preprocessing. In this study, two datasets were used in the experiments. One was the public crack data set [43], which was an annotated dataset to train and validate machine learning-based crack detection and segmentation algorithms. It consisted of 2000 samples. The dimension of each image was 448×448 pixels. The dataset was divided into three datasets, i.e., training dataset, validation dataset, and test dataset. Among them, the images in the training dataset were used to fit the neural network model. In the training process, the fitted model repeatedly made predictions on the validation dataset, which evaluated the effect of the model and indicated whether overfitting occurred. After the training process was completed, the test dataset was used to assess the performance of different networks.

The second crack image dataset was denoted as "Yantai dataset," which was manually collected by a camera on Yantai highway. This dataset consisted of images with complex textures under different light conditions. The dataset consisted of 5000 samples. The images were firstly cropped to obtain the central part of the image, which was square. Then, the cropped images were resized to have 448×448 pixels. No other data preprocessing techniques were applied to the collected data, such as noise removal and crack enhancement. The dataset was also divided into three datasets, i.e., training dataset, validation dataset, and test dataset. The purpose of introducing this dataset was to further evaluate the performance of the proposed model in this work. The details are listed in Table 1.

4.2. Data Labeling. For training and validation purposes, the Yantai dataset was required to be annotated. The software of LabelMe was used to manually draw polygon for the crack. After that, the annotated images were generated through the python program, and the output consisted of a JSON file including the position information of the cracks. Therefore, the image background was black, while the cracks were white.

Table 1: Public dataset and Yantai dataset.

Dataset	Training	Validation	Test
Public dataset	1600	200	200
Yantai dataset	2800	500	500

4.3. Segmentation Metrics. The following evaluation indicators were applied for the crack segmentation task, i.e., precision (P), mean IoU, recall (R), and F1. In the image, the crack pixels (white) were defined as positive samples. Based on the combination of labelled cases and predicted cases, the segmentation of the pixels resulted in true positive (TP), false positive (FP), true negative (TN), and false negative (FN).

Precision was defined as the ratio of the number of correctly predicted crack pixels to the number of all the pixels predicted to be cracks. Recall was defined as the ratio of the number of correctly predicted crack pixels to the number of all the true crack pixels. F1 score was the harmonic mean of precision and recall [44].

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ F1 &= \frac{2.\text{precision} \cdot \text{Recall}}{\text{precision} + \text{Recall}}, \end{aligned} \tag{6}$$

Intersection over Union (IoU) reflected the overlapping degree between two objects. In this study, the IoU was evaluated on the "crack" class to measure the overlap between the truth crack objects and predicted crack objects.

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}. \tag{7}$$

4.4. Loss Functions. Three loss functions were applied and compared in this study. The pixel-wise cross-entropy loss was described as follows.

$$CE = -\sum_{i=0}^{n^*n} (p \log(\widehat{p}) + (1 - p)\log(1 - \widehat{p})), \tag{8}$$

where i is the index of the pixel, n*n indicates the size of the output image, p represents the true label of samples, 1 represents positive, 0 represents negative, and $\widehat{p}$ represents the possibility of samples being predicted as positive.

The balanced pixel-wise cross-entropy loss was similar to pixel-wise cross-entropy loss. The difference was that the balanced pixel-wise cross-entropy loss only assigned weights to positive and negative samples, and the sum of the weight was 1. The equation can be described as follows:

$$BCE = -\sum_{i=0}^{n^*n} (\beta p \log(\widehat{p}) + (1 - \beta)(1 - p)\log(1 - \widehat{p})), \tag{9}$$

where β represents the balanced factor, i is the index of the pixel, $n^* n$ indicates the size of the output image, p represents the true label of samples, 1 represents positive, 0 represents negative, and $\widehat{p}$ represents the possibility of samples being predicted as positive.

Dice loss was designed from the perspective of IoU and expressed in equation (10).

$$\text{Dice loss} = 1 - \frac{2 * \text{TP}}{2 * \text{TP} + \text{TP} + \text{FN}}, \tag{10}$$

where TP refers to true positive, and FN refers to false negative.

4.5. Computing Hardware and Software. All the experiments were performed on a computer with the following specifications: CPU was Intel (R) Xeon (R) CPU E5-2678 v3, and GPU was Nvidia GTX 1060 with 8G RAM. The software environment was based on Ubuntu 16.04, and python was used as the main programming language. Meanwhile, the experiments were conducted on Pytorch 1.5 deep learning framework.

4.6. Hyperparameter Setting. In this work, the training was optimized by the min-Batch stochastic gradient-descent algorithm with momentum. The hyperparameter settings were listed as follows: weight decay factor = 0.0002, momentum = 0.9, learning rate = 0.01, mini-Batch size = 4, and number of epochs = 60.

Two groups of experiments were designed to assess the performance of the proposed CrackResAttentionNet model. The first group of experiments was based on the public crack dataset, and the other group was based on the Yantai dataset. The detailed settings are listed in Table 2.

For each group of experiments, in addition to CrackResAttentionNet, the typical image segmentation models including ENet, ExFuse, FCN, LinkNet, SegNet, and UNet were also run. Three different loss functions, i.e., pixel-wise cross-entropy loss (CE), balanced pixel-wise cross-entropy loss (BCE), and dice loss (Dice), were used to train each model.

5. Results and Discussions

A series of experiments were performed, and the pavement images were processed with the models. Each experiment was evaluated from precision, mean IoU, recall, and F1. Precision indicated the ratio of the number of correctly classified crack pixels to the total number of all pixels, while recall indicated the ratio of the number of correctly classified crack pixels to the number of all true crack pixels. Mean IoU reflected the degree of overlap between predicted crack area and real crack area. F1 was the harmonic mean of precision and recall, reflecting the accuracy of an algorithm [45].

5.1. Experiment Results. The results are summarized in Tables 3–8.

Table 3 shows the performance of different models (ENet, ExFuse, FCN, LinkNet, SegNet, UNet, and CrackResAttentionNet) with CE loss for the public dataset [44]. Compared with other models, CrackResAttentionNet yielded higher precision (82.58), mean IoU (72.83), recall (85.13), and F1 (83.84). ExFuse had good precision (82.22), and ENet had good mean IoU (72.22), recall (83.94), and F1 (81.94).

Table 4 shows the performance of different models with BCE loss. Compared with other models, CrackResAttentionNet showed great precision (89.40). However, CrackResAttentionNet did not show advantages in the other three metrics (mean IoU, recall, and F1). Considering all four metrics together, CrackResAttentionNet still performed great.

Table 5 shows the performance of different models with Dice loss. CrackResAttentionNet showed great precision, mean IoU, and F1 gains, while its recall was not in the lead.

Table 6 shows the performance of different models (ENet, ExFuse, FCN, LinkNet, SegNet, UNet, and CrackResAttentionNet) with CE loss using the Yantai dataset. The maximum F1 score (93.33) was achieved by CrackResAttentionNet, which was 1.78 higher than that of the model in the 2nd place, ENet (91.55). Meanwhile, CrackResAttentionNet had much better performance in other metrics. For instance, the precision (94.64) and mean IoU (83.28) of CrackResAttentionNet were both much better than those of the other models.

Table 7 shows the performance of different models with BCE loss using the Yantai dataset. CrackResAttentionNet showed much better precision (96.17), mean IoU (83.69), recall (93.44), and F1 (94.79) than those of other models. LinkNet also obtained a good F1 score (93.55), but it was still lower than the F1 score of CrackResAttentionNet.

Table 8 shows the performance of different models with Dice loss using the Yantai dataset. CrackResAttentionNet exceeded other models in all the four metrics (precision 95.43, mean IoU 82.75, recall 94.2, and F1 94.81).

BCE loss can be selected as the best loss function for CrackResAttentionNet. It can be seen from these two datasets that BCE loss had higher precision (89.40 for the public dataset and 96.17 for the Yantai dataset).

Sample prediction segmentation output for BCE loss of each model is shown in Figures 8 and 9. From the public dataset sample prediction in Figure 8, SegNet misclassified some background as cracks, while FCN misclassified some cracks as background. CrackResAttentionNet made good segmentation, and the result was extremely close to the ground truth.

From the Yantai dataset sample prediction results in Figure 9, ENet and FCN misclassified some background as cracks, while SegNet and ExFuse misclassified some cracks as background. CrackResAttentionNet had perfect segmentation performance, and the result was extremely close to the ground truth.

5.2. Influence of Different Activation Functions. The effects of different activation functions on CrackResAttentionNet performance were further investigated. Firstly, we fixed the loss function as BCE loss and then used different activation

Table 2: Experimental settings.

Dataset	Epochs	Batch size	Optimizer	Learning rate	Weight decay factor	Momentum
Public dataset	60	4	SGD	0.01	0.0002	0.9
Yantai dataset	60	4	SGD	0.01	0.0002	0.9

Table 3: Public Crack Dataset- CE loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	80.03	72.22	83.94	81.94
ExFuse	82.22	71.77	81.17	81.69
FCN	81.87	71.02	77.72	79.74
LinkNet	81.15	70.97	82.62	81.88
SegNet	78.00	66.32	75.18	76.56
UNet	80.19	70.42	82.88	81.51
CrackResAttentionNet	82.58	72.83	85.13	83.84

Table 4: Public Crack Dataset- BCE loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	74.87	54.64	50.57	60.37
ExFuse	82.78	69.52	78.80	80.74
FCN	86.04	64.32	83.27	84.63
LinkNet	81.84	71.11	81.23	81.53
SegNet	73.53	53.51	61.39	66.91
UNet	86.19	70.40	83.32	84.73
CrackResAttentionNet	89.40	71.51	81.09	85.04

Table 5: Public Crack Dataset- Dice loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	76.18	55.45	56.68	65.00
ExFuse	48.92	48.88	50.00	49.45
FCN	80.17	67.83	79.11	79.64
LinkNet	89.90	68.18	79.66	84.47
SegNet	76.46	71.08	89.38	82.42
UNet	82.82	70.41	82.60	82.71
CrackResAttentionNet	90.72	71.69	81.93	86.10

Table 6: Yantai Dataset–CE loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	90.88	81.62	92.22	91.55
ExFuse	91.76	82.27	90.64	91.20
FCN	91.56	82.52	90.80	91.18
LinkNet	91.34	82.48	92.11	91.72
SegNet	91.73	81.98	89.73	90.72
UNet	93.54	78.63	89.24	91.34
CrackResAttentionNet	94.64	83.28	92.05	93.33

functions (Relu, PRelu, RRelu, and LeakyRelu). The same experimental setup was used as above.

As shown in Table 9, for the public crack dataset, if PRelu was used, mean IoU (71.51) was 0.28 higher than the second place, precision (89.4) was 2.37 higher than the second place, and F1 was also high (85.04). So PRelu has a better performance compared with the other three activation functions (LeakyRelu, RRelu, and Relu).

As shown in Table 10, for the Yantai dataset, PRelu had outstanding performance compared with the other three activation functions (LeakyRelu, RRelu, and Relu). Mean IoU (83.69) was 2.12 higher than the second place, and

Table 7: Yantai Dataset–BCE loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	94.67	81.82	92.34	93.49
ExFuse	95.18	82.03	91.85	93.48
FCN	93.05	82.95	90.64	91.83
LinkNet	95.07	82.53	92.08	93.55
SegNet	91.24	78.06	83.04	86.95
UNet	94.28	81.61	90.26	92.23
CrackResAttentionNet	96.17	83.69	93.44	94.79

Table 8: Yantai Dataset–Dice loss.

Model	Precision/(%)	Mean IoU/(%)	Recall/(%)	F1/(%)
ENet	94.80	82.17	92.10	93.43
ExFuse	92.10	74.12	87.66	89.87
FCN	90.23	77.65	89.16	89.69
LinkNet	94.45	80.76	91.62	93.01
SegNet	91.80	74.86	90.23	91.01
UNet	93.76	80.11	91.10	92.41
CrackResAttentionNet	95.43	82.75	94.2	94.81

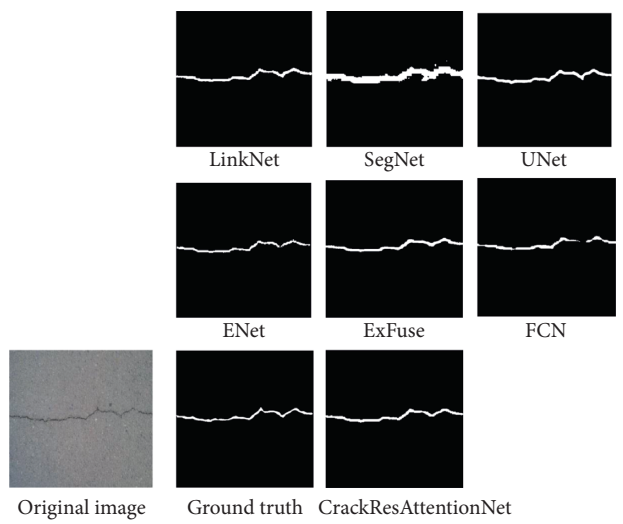

Figure 8: Public Crack Dataset Sample Prediction-BCE loss.

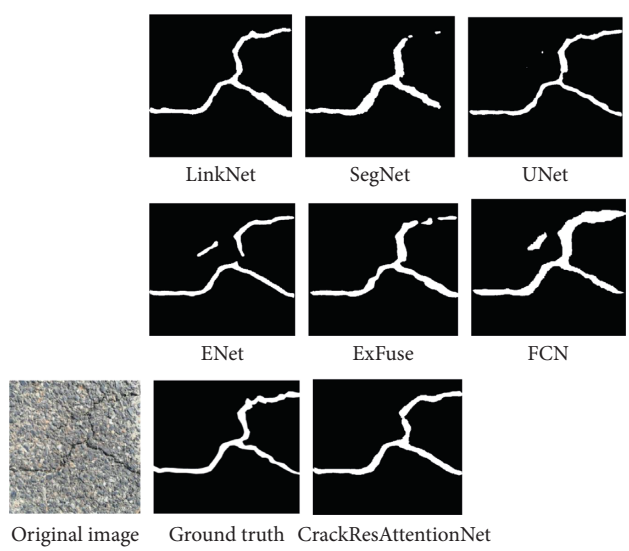

Figure 9: Yantai Dataset Sample Prediction-BCE loss.

Table 9: CrackResAttentionNet (BCE loss) on different activation functions (Public Dataset).

Model	Precision (%)	Mean IoU (%)	Recall (%)	F1 (%)
LeakyRelu	84.93	71.23	86.36	85.64
RRelu	87.03	71	83.77	85.37
Relu	82.09	69.5	86.42	84.2
PRelu	89.4	71.51	81.09	85.04

Table 10: CrackResAttentionNet (BCE loss) on different activation functions (Yantai Dataset).

Model	Precision (%)	Mean IoU (%)	Recall (%)	F1 (%)
LeakyRelu	94.7	78.89	92.52	93.6
RRelu	92	81.18	96.5	94.2
Relu	94.17	81.57	92.96	93.56
PRelu	96.17	83.69	93.44	94.79

precision (96.17) was 1.47 higher than the second place. Therefore, based on the performances of different functions for the public dataset and the Yantai dataset, PRelu was selected for CrackResAttentionNet.

6. Summary and Conclusions

In this study, an architecture based on encoder-decoder network was proposed to detect asphalt pavement cracks and perform pixel-level image segmentation. The results indicated that CrackResAttentionNet with BCE loss function achieved the best performance in precision (89.40), mean IoU (71.51), recall (81.09), and F1 (85.04) for the public dataset. Meanwhile, for the Yantai dataset, CrackResAttentionNet with BCE loss function had the precision of 96.17, mean IoU of 83.69, recall of 93.44, and F1 of 94.79. The performance of CrackResAttentionNet exceeded other popular models including ENet, ExFuse, FCN, LinkNet, SegNet, and UNet.

The summary and main findings are as follows:

(i) In the architecture of the developed algorithm, the encoder mainly used the convolutional layers of ResNet-34 to extract image features, and an additional encoder layer was added to better extract information. The decoder employed the deconvolutional layers to perform the semantic segmentation of crack and noncrack pixels.

(ii) After each encoder, position attention module and channel attention module were connected to summarize remote context information. The two attention modules were fused proportionally to emphasize more on the position information. The output of each encoder layer was fused with the output attention and linked with the corresponding decoder. In addition, the output of the previous decoder was fed into the decoder as an input. In this way, the decoder and its upsampling operations can use the spatial information to help improve the prediction accuracy.

(iii) The experiments were conducted for the public crack dataset and Yantai dataset using CrackResAttentionNet and other typical models including ENet, ExFuse, FCN, LinkNet, SegNet, and UNet. In addition, three different loss functions (CE, BCE, and Dice) were used in the experiments. The results demonstrated that CrackResAttentionNet with BCE loss function achieved the best performance.

(iv) For public dataset, CrackResAttentionNet performed well in precision (89.40), mean IoU (71.51), recall (81.09), and F1 (85.04). Especially, with BCE loss, the precision was increased by 3.21.

(v) For the Yantai dataset, the CrackResAttentionNet had a great performance in precision (96.17), mean IoU (83.69), recall (93.44), and F1 (94.79). With BCE loss, precision was increased by 0.99, mean IoU was increased by 0.74, recall was increased by 1.1, and F1 was increased by 1.24.

(vi) The effects of different activation functions were further analyzed. For the public dataset, if PRelu was used, mean IoU (71.51) was 0.28 higher than the second place, the precision (89.4) was 2.37 higher than the second place, and F1 was also high (85.04). For the Yantai dataset, if PRelu was used, mean IoU (83.69) was 2.12 higher than the second place, and the precision (96.17) was 1.47 higher than the second place. Therefore, PRelu was finally selected as the activation function for CrackResAttentionNet.

Overall, to the best of our knowledge, this research is the first attempt to use an encoder-decoder network with both encoder and attention information linked to the decoder layer. It also defined a few additional encoder and decoder modules. Another difference was that the position channel module and channel attention module were proportionally fused.

7. Suggestions for Future Research

Suggestions for future research on this topic include the following:

(i) More research should be conducted on the latest attention models (such as Transformer and SENet), which might be integrated into our network to improve the accuracy of the results.

(ii) More detailed experimentation is required to analyze the hyperparameters derived from the training and validation datasets. Meanwhile, it is necessary to study the impact of the hyperparameters on the model performance.

(iii) Multi-GPU can speed up the model training and inference. Real-time model training and prediction require running the training under a multi-GPU hardware environment.

(iv) A bigger public annotated public dataset is needed to cover all types of cracks with noises under different conditions. And a more accurate method related to pixel-level crack detection should be developed to characterize and quantify the spatial stereoscopic information of cracks in the pavement.

(v) In the future, the proposed detection method should be optimized, and a large-scale crack database for asphalt pavement crack detection should be built. The image recognition method of pavement cracks should be applied for pavement intelligent maintenance to help identify the microcracks on the pavement to the cloud platform for earlier repair.

Data Availability

The (https://github.com/wanhaifengytu/CrackSegmentation Project/tree/main/data) data used to support the findings of this study have been deposited in the Git Hub repository (https://github.com/wanhaifengytu/CrackSegmentationProject/).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Haifeng Wan conceptualized the study, developed the methodology, performed investigation, and wrote the original draft. Lei Gao conceptualized the study and performed investigation. Manman Su performed investigation and data curation. Qirun Sun performed investigation and reviewed and edited the manuscript. Lei Huang performed investigation and edited the manuscript.

Acknowledgments

The authors are grateful for the financial support from Natural Science Foundation of Shandong Province (Grant No. ZR2020ME238), the Shandong Provincial Key Laboratory of Highway Technology and Safety Assessment Research Fund Project (TM2018H46), and Yantai Science and Technology Innovation Development Plan Project (2020XDRH104).

References

[1] Q. Mei and G. Mustafa, "A cost effective solution for pavement crack inspection using cameras and deep neural networks," *Construction and Building Materials*, vol. 256, Article ID 119397, 2020.

[2] X. Feng, L. Xiao, W. Li et al., "Pavement crack detection and segmentation method based on improved deep learning fusion model," *Mathematical Problems in Engineering*, vol. 202022 pages, 2020.

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

[4] Y. Lu, A. J. Golrokh, and M. D. A. Islam, "Concrete pavement service condition assessment using infrared thermography," *Advances in Materials Science and Engineering*, vol. 20178 pages, 2017.

[5] A. Ayenu-Prah and N. Attoh-Okine, "Evaluating pavement cracks with bidimensional empirical mode decomposition," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, Article ID 861701, 1 page, 2008.

[6] Y. Zhou, F. Wang, N. Meghanathan, and Y. Huang, "Seed-based approach for automated crack detection from pavement images," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2589, no. 1, pp. 162–171, 2016.

[7] J. Zhou, P. Huang, and F. P. Chiang, "Wavelet-based pavement distress classification," *Transportation Research Record*, vol. 1940, no. 1, pp. 89–98, 2005.

[8] M. Flah, A. R. Suleiman, and M. L. Nehdi, "Classification and quantification of cracks in concrete structures using deep learning image-based techniques," *Cement and Concrete Composites*, vol. 114, 2020.

[9] A. Rezaie, R. Achanta, M. Godio, and K. Beyer, "Comparison of crack segmentation using digital image correlation measurements and deep learning," *Construction and Building Materials*, vol. 261, 2020.

[10] G. Li, Q. Liu, W. Ren, W. Qiao, B. Ma, and J. Wan, "Automatic recognition and analysis system of asphalt pavement cracks using interleaved low-rank group convolution hybrid deep network and SegNet fusing dense condition random field-Science Direct," *Measurement*, vol. 170, 2020.

[11] M. Alipour and D. K. Harris, "Increasing the robustness of material-specific deep learning models for crack detection across different materials," *Engineering Structures*, vol. 206, 2016.

[12] B. Wei, K. Hao, L. Gao, and X.-S. Tang, "Detecting textile micro-defects: a novel and efficient method based on visual gain mechanism," *Information Sciences*, vol. 541, pp. 60–74, 2020.

[13] B. Li, C. Kelvin, P. Wang, A. Zhang, F. Yue, and G. Sollazzo, "Automatic segmentation and enhancement of pavement cracks based on 3D pavement images," *Journal of Advanced Transportation*, vol. 20199 pages, 2019.

[14] S. E. Park, S.-H. Eem, and H. Jeon, "Concrete crack detection and quantification using deep learning and structured light," *Construction and Building Materials*, vol. 252, 2020.

[15] M. R. Jahanshahi and S. F. Masri, "Adaptive vision-based crack detection using 3D scene reconstruction for condition assessment of structures," *Automation in Construction*, vol. 22, pp. 567–576, 2012.

[16] D. Sattar, R. J. Thomas, and M. Marc, "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.

[17] G. Moussa and K. Hussain, "A new technique for automatic detection and parameters estimation of pavement crack," in

Proceedings of the 4th International Multi-Conference on Engineering and Technological Innovation (IMETI 2011), Kaohsiung, Taiwan, July 2011.

[18] D. Kang, S. S. Benipal, D. L. Gopal, and Y.-J. Cha, "Hybrid pixel-level concrete crack segmentation and quantification across complex backgrounds using deep learning," *Automation in Construction*, vol. 118, Article ID 103291, 2020.

[19] Y. Dong, C. Su, P. Qiao, and L. Sun, "Microstructural crack segmentation of three-dimensional concrete images based on deep convolutional neural networks," *Construction and Building Materials*, vol. 253, Article ID 119185, 2020.

[20] S. Chambon, C. Gourraud, J. M. Moliard, and P. Nicolle, "Road crack extraction with adapted filtering and Markov model-based segmentation: introduction and validation," in *Proceedings of the Visapp-Fifth International Conference on Computer Vision Theory & Applications*, DBLP, Angers, France, May 2010.

[21] Y.-C. J. Tsai and F. Li, "Critical assessment of detecting asphalt pavement cracks under different lighting and low intensity contrast conditions using emerging 3D laser technology," *Journal of Transportation Engineering*, vol. 138, no. 5, pp. 649–656, 2012.

[22] M. Salman, S. Mathavan, K. Kamal, and M. Rahman, "Pavement crack detection using the Gabor filter," in *Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013)*, IEEE, Hague, Netherlands, October 2014.

[23] T. Merazi-Meksen, M. Boudraa, and B. Boudraa, "Mathematical morphology for TOFD image analysis and automatic crack detection," *Ultrasonics*, vol. 54, no. 6, pp. 1642–1648, 2014.

[24] Z. Qin, Y. Cao, Q. M. Q. Li, and S. Wang, "CrackTree: automatic crack detection from pavement images," *Pattern Recognition Letters*, vol. 33, 2012.

[25] G. Miguel, D. Balcones, O. Marcos, D. Llorca, and A. Miguel, "Sotelo parra ignacio, ocaña manuel, aliseda pedro, pedro yarza, and alejandro amírola. Adaptive road crack detection system by pavement classification," *Sensors (14248220)*, vol. 11, no. 10, pp. 9628–9657, 2011.

[26] Y. Ö. Özgören, S. Çetinkaya, S. Sarıdemir, A. Çiçek, and F. Kara, "Artificial neural network based modelling of performance of a beta-type Stirling engine," *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, vol. 227, no. 3, pp. 166–177, 2013.

[27] A. Eser, E. A. Ayyıldız, M. Ayyıldız, and F. Kara, "Artificial intelligence-based surface roughness estimation modelling for milling of AA6061 alloy," *Advances in Materials Science and Engineering*, vol. 2021, 2021.

[28] M. Ayyıldız and K. Çetinkaya, "Predictive modeling of geometric shapes of different objects using image processing and an artificial neural network," *Journal of Process Mechanical Engineering*, vol. 231, 2016.

[29] Ö. Erkan, B. Işık, A. Çiçek, and F. Kara, "Prediction of damage factor in end milling of glass fibre reinforced plastic composites using artificial neural network," *Applied Composite Materials*, vol. 20, no. 4, pp. 517–536, 2013.

[30] F. Kara, M. Karabatak, M. Ayyıldız, and E. Nas, "Effect of machinability, microstructure and hardness of deep cryogenic treatment in hard turning of AISI D2 steel with ceramic cutting," *Journal of Materials Research and Technology*, vol. 9, no. 1, pp. 969–983, 2020.

[31] 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.

[32] 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.

[33] B. Wei, K. Hao, L. Gao, X.-S. Tang, and Y. Zhao, "A biologically inspired visual integrated model for image classification," *Neurocomputing*, vol. 405, pp. 103–113, 2020.

[34] A. Ji, X. Xue, Y. Wang, X. Luo, and W. Xue, "An integrated approach to automatic pixel-level crack detection and quantification of asphalt pavement," *Automation in Construction*, vol. 114, Article ID 103176, 2020.

[35] 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, 2018.

[36] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," *Computer Science*, 2016.

[37] Z. Zhang, X. Zhang, C. Peng, X. Xue, and S. Jian, *Exfuse: Enhancing Feature Fusion for Semantic Segmentation. European Conference on Computer Vision*, Springer, Berlin, Germany, 2018.

[38] A. Chaurasia and E. Culurciello, "LinkNet: exploiting encoder representations for efficient semantic segmentation," in *Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP)*, Petersburg, FL, USA, December 2017.

[39] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: a deep convolutional encoder-decoder architecture for image segmentation," *Institute of Electrical and Electronics Engineers Transactions on Pattern Analysis & Machine Intelligence*, vol. 20171 page, 2017.

[40] B. Wei, H. He, K. Hao, L. Gao, and X.-S. Tang, "Visual interaction networks: a novel bio-inspired computational model for image classification," *Neural Networks*, vol. 130, pp. 100–110, 2020.

[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *in Proceedings of the International Conference on Learning Representations,* San Diego, CA, USA, 2015.

[42] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, and Z. Fang, "Dual attention network for scene segmentation," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Long Beach, CA, USA, June 2020.

[43] R. Amhaz, S. Chambon, J. Idier, and V. Baltazart, "Automatic crack detection on two-dimensional pavement images: an algorithm based on minimal path selection," *Institute of Electrical and Electronics Engineers Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2718–2729, 2016.

[44] S. Zhou and W. Song, "Concrete roadway crack segmentation using encoder-decoder networks with range images," *Automation in Construction*, vol. 120, 2020.

[45] A. Zhang, F. Y. Wang, W. Liu et al., "Deep learning-based fully automated pavement crack detection on 3D asphalt surfaces with an improved CrackNet," *Journal of Computing in Civil Engineering*, vol. 32, 2018.