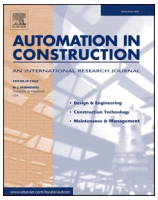
Automation in Construction 157 (2024) 105166

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon



Dual attention transformer network for pixel-level concrete crack segmentation considering camera placement

Yingjie Wu [a], Shaoqi Li [a,**], Jinge Zhang [a], Yancheng Li [a,b,*], Yang Li [c], Yingqiao Zhang [a]

[a] College of Civil Engineering, Nanjing Tech University, Nanjing 211816, China
[b] School of Civil and Environmental Engineering, University of Technology Sydney, Ultimo, NSW 2007, Australia
[c] Qilu University of Technology (Shandong Academy of Sciences), Institute of Automation, Jinan, Shandong 250014, China

ARTICLE INFO

Keywords:
Deep learning
Vision transformer
Crack detection
Semantic segmentation

ABSTRACT

Pixel-level crack segmentation remains a challenging task due to the trade-off between computational cost and accuracy, as well as the small size of real-world cracks, typically submillimeter in width, resulting in limited pixels for analysis. To address these challenges, this paper proposes a Pixel Crack Transformer Network (PCTNet) to investigate the impact of different camera placements on network performance. PCTNet adopts a hierarchical structure with Cross-Scale PatchEmbedding Layer and Dual Attention Transformer Block, enabling the generation of multi-scale feature maps and the fusion of global and local features. PCTNet achieves a reduction of up to 64% in computational cost compared to transformer networks while outperforming both convolutional and transformer networks, achieving 95.89% precision, 93.77% recall, 94.8% F1-score, and 90.53% mIoU. Furthermore, this work introduces Crack-R dataset, which encompasses crack images captured at varying distances, facilitating the evaluation of segmentation accuracy in real-world scenarios with different crack-to-pixel ratios.

1. Introduction

Crack detection holds immense significance in the field of structural health monitoring (SHM) and repairment, as cracks are the most common type of damage in concrete structures and can be a leading cause of structural collapse [1]. Timely and accurate crack detection is critical for assessing the safety and durability of concrete structures [2–4], as it allows for appropriate measures to be taken to ensure their integrity and longevity [5].

In recent years, deep learning-based methods have gained significant attention for detecting cracks in concrete structures. Several Convolutional Neural Networks (CNNs) architectures, such as VGG [6], ResNet [7], and R-CNN [8], have shown promising performance in object detection and classification tasks. For example, He et al. [9] proposed an UAV road crack objection detection algorithm, which has a satisfactory detection speed and accuracy; Liu et al. [10] utilized CNNs combined with infrared thermography to classify whether asphalt pavement has cracks and their severity. However, object detection algorithms can only provide the general location of cracks, and image classification algorithms can only categorize images based on the presence or absence of cracks. Semantic segmentation, a method that provides precise localization and generates high-resolution feature maps, has emerged as a viable solution for concrete crack detection [11]. This approach has been extensively investigated and gained popularity in SHM [12,13].

For instance, Dung et al. [2] proposed a VGG16-based Fully Convolutional Network (FCN) network to detect cracks. The network was trained with 500,227 × 227 pixel crack images and achieved an average accuracy of 90% on the test set. However, FCN-based approaches suffer from the loss of low-level semantic information during the up-sampling process, resulting in inaccurate segmentation of complex and fine cracks. To address this limitation, Liu et al. [14] were the first to employ a U-Net network for crack segmentation, in which U-Net utilizes skip connections to transfer low-level semantic information to the up-sampling path during up-sampling, allowing for the extraction of more feature information. Nevertheless, it can only accept 512 × 512 images as input, and for high-pixel images, the images need to be cropped or scaled, making it impractical for SHM. In addition, these CNNs are limited in their ability to capture long-range features due to the inherent limitations of convolution [15]. As a result, these networks may fail to extract adequate feature information to accurately distinguish cracks

* Corresponding author at: College of Civil Engineering, Nanjing Tech University, Nanjing 211816, China.
** Corresponding author.
E-mail addresses: shaoqi@njtech.edu.cn (S. Li), yancheng.li@uts.edu.au (Y. Li).

https://doi.org/10.1016/j.autcon.2023.105166
Received 17 May 2023; Received in revised form 28 October 2023; Accepted 31 October 2023
Available online 9 November 2023

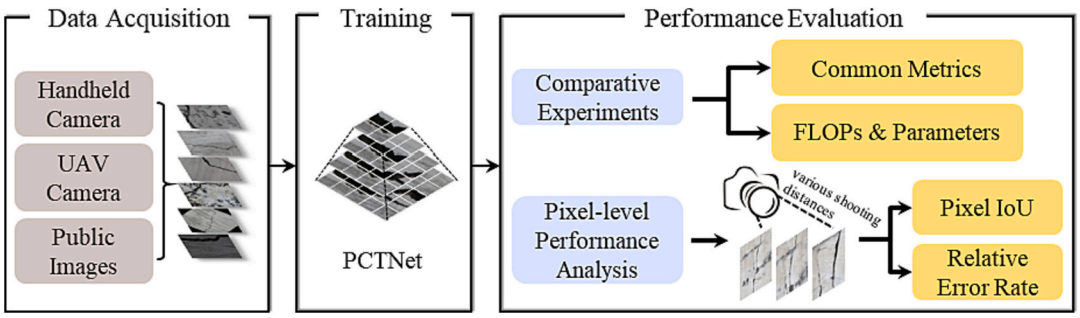


Fig. 1. Logic and arrangement of this work.

from the background under different shooting distances and camera placements, as the contrast between crack pixels and background pixels is undesirable in such cases.

Compared to traditional CNNs, transformer networks utilize self-attention mechanism to extract long-range feature information and have achieved outstanding achievements in the field of natural language processing (NLP) [16]. In 2020, Google proposed Vision Transformer (ViT) [17], which used transformer structures for image classification and became the state-of-the-art network at that time [18]. ViT employs self-attention mechanism to extract and integrate contextual information. However, the number and dimensions of the tokens in ViT are fixed, which makes it unable to train and predict at multiple scales. Moreover, using ViT algorithm to identify cracks requires high computational costs. To overcome these limitations, researchers have developed various variants [19,20]. Wang et al. [21] proposed Pyramid Vision Transformer (PVT) with a pyramid structure, which divides the feature map into patches and performs self-attention computation on each patch to reduce the computational cost and enhance the continuity of local information. Liu et al. [22] proposed Swin Transformer, using non-overlapping shift windows for self-attention computation, allowing different windows to interact with each other.

Following the success of transformer on various tasks, a series of transformer networks for crack detection have been proposed. Elyas et al. [23] proposed a hybrid structure of U-Net and ViT, applying ViT structure to the output feature maps of U-Net to compensate for the inherent bias of CNNs, i.e., the insufficient extraction ability of contextual features. However, it relies on a large number of convolutional layers for feature extraction, which is prone to information loss. Wang et al. [24] suggested a standard transformer-based network named SegCrack for concrete crack detection, and the online hard example mining strategy was utilized to improve the network performance. In order to reduce the computational complexity of SegCrack, the length of the sequences is reduced by a reduction factor R. However, this inevitably leads to the weakening or losing contextual relevance in the original sequences. Xu et al. [25] highlighted that while the attention mechanism can extract long-range feature information, it may lose local fine-grained feature information. To address this issue, LETNet transformer network was designed to detect pavement cracks with a local enhancement module composed of 1×1 convolution and 3×3 convolution to compensate for the transformer's lack of local characteristics. However, the compensation of local features in LETNet is limited due to small kernel size with the confined field of perception in the local enhancement module, resulting in feature information loss [26]. This disadvantage leads to blurring and discontinuity of the segmentation results for pixel-level width cracks, while in practical engineering applications, promising results for pixel-level width cracks are required.

In light of the past research presented, the advancement of CNNs and transformer networks has made it feasible to perform semantic segmentation for cracks on diverse civil structures, including buildings, bridges, and pavement. However, it should be noted that CNNs and transformer networks have inherent limitations that can compromise their performance in real-world engineering applications [27]. Therefore, to enable the practical application of crack segmentation algorithms, it is imperative to address the following two challenges:

1) Processing high-resolution images requires significant computation cost: According to the ACI Committee 224R-01 [28], cracks with a width greater than 0.3 mm are required to be monitored and treated. To meet such standards, high-resolution images of the structure's surface must be collected, trained, and tested. This necessitates substantial computation resources, with video memory costs reaching hundreds of gigabytes. In previous practices, training and testing images were often cropped and scaled to lower resolutions to enable feature extraction and achieve high accuracy. However, this approach results in the loss of detailed crack features, compromising the performance of segmentation algorithm in real-world engineering applications.
2) Considering the restriction on the camera placement: In general, capturing images of concrete structures requires the camera to be positioned at a minimum distance of 30 cm from the structure [29]. In more complex scenarios, such as bridge piers and tunnels, the distance can exceed 50 cm or even 100 cm, while a crack with a width of 1 mm appears as only one or two pixels at a shooting distance of 50 cm. Consequently, the contrast between crack and background is diminished, making it challenging to accurately detect cracks in real-world scenarios [30].

From these perspectives, this paper proposes a Dual Attention Transformer network named Pixel Crack Transformer Network (PCTNet) for pixel-level concrete crack segmentation. The logic and arrangement of this is shown in Fig. 1. A relatively comprehensive dataset named Crack 896 is collected for evaluating the overall performance of PCTNet and assessing it with commonly metrics and computational efficiency metrics. A new dataset named Crack-R is established, which contains high-resolution crack images with varying shooting distances, to explore the influence of different camera placement on network's crack detection performance. Two new metrics, Pixel IoU and relative error rate, are proposed to evaluate the cracks segmentation performance at the pixel-level. A series of experiments are conducted to prove the effectiveness of PCTNet for crack detection in high-resolution images and its robustness in complex environments.

The remainder of this paper keeps the following organization. Section 2 presents the specific structure of PCTNet and elaborates the details of each module; Section 3 describes the composition of the crack datasets for pixel-level performance analysis, initialization of the algorithm, and evaluation metrics; Section 4 analyzes and discusses the results of the comparative experiments; Section 5 summarizes this work and renders some recommendations for this algorithm.

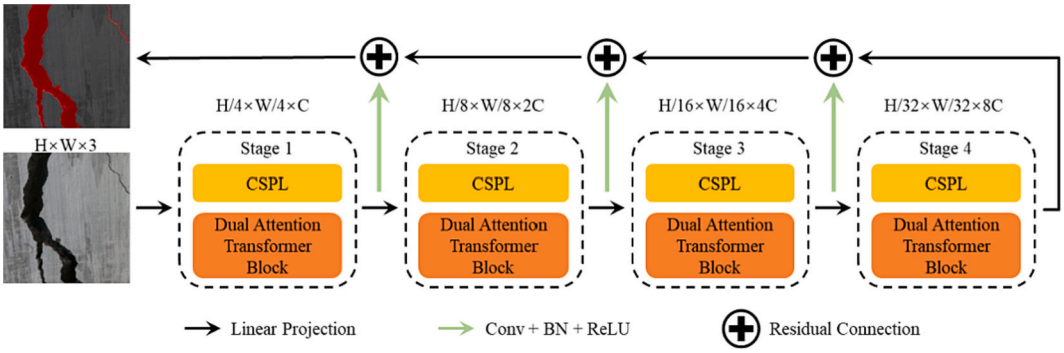


Fig. 2. Architecture of PCTNet network.

Table 1
Parameters of CSPL.

Type	Stride	Dilation	Padding	Dim
4 × 4 Conv	4 × 4	/	/	dim/2
8 × 8 Conv	4 × 4	/	2 × 2	dim/4
3 × 3 Dilated Conv	4 × 4	6 × 6	6 × 6	dim/4

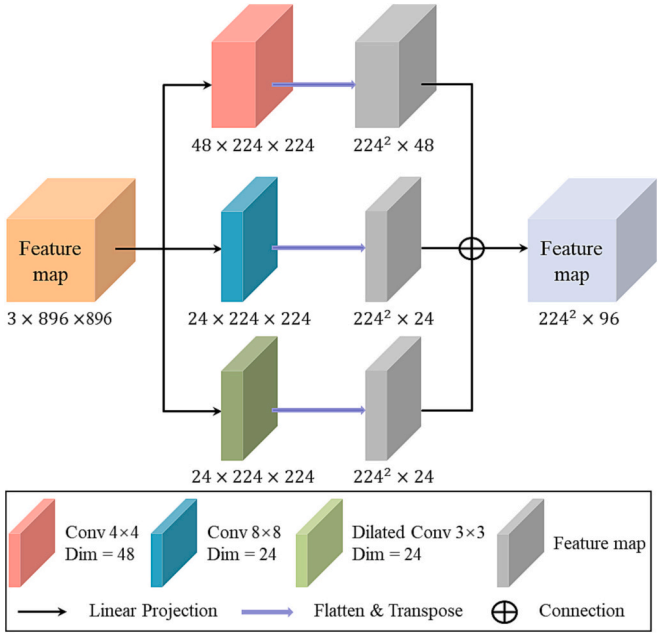


Fig. 3. Schematic of CSPL in Stage 1.

2. Proposed network

2.1. Overall network architecture

In this paper, given the consideration of the challenges mentioned in the introduction, PCTNet is inspired by transformer networks and introduces a hierarchical feature pyramid structure. The overall network structure of PCTNet, as shown in Fig. 2, comprises two main modules.

The lower module is a transformer encoder with a hierarchical feature pyramid structure that generates feature maps of different scale. In the encoder, lower-level feature maps have a large perceptual field to capture global contextual feature information about cracks, while higher-level feature maps have a limited perceptual field to capture the detail and texture information of cracks. Cracks usually contain subtle texture and morphological changes, and these details are important for accurate detection. The upper module is a feature pyramid decoder for fusing the multi-scale crack features extracted by the encoder. The lower layers in the decoder are responsible for recovering the local details and local contextual information such as crack edges, texture, etc. The higher decoder layers correct and refine based on global contextual feature information, such as crack shape.

The specific process of PCTNet is as follows: the Cross-Scale Patch-Embedding Layer (CSPL) divides the input image into small patches in the encoder module. For example, in the first stage, the input image of size H × W × 3 is split into $\frac{HW}{4^2}$ patches by CSPL. These patches are fed into the proposed Dual Attention Transformer Block, generating feature maps with a resolution of 1/4 of the original image. Notably, in stages 2, 3, and 4, the obtained feature maps are 1/8, 1/16, and 1/32 resolution of the original image, respectively. The extracted features are further extracted through Conv 1 × 1 → BatchNorm (BN) [31] → ReLU [32], and connected with low-level semantic features to generate the final segmentation map.

2.2. CSPL

Considering cracks can exist in different scales within an image, while previous Patch Embedding modules in transformer networks often employ convolution layers of the same size to generate embeddings [33]. This approach results in the generated embeddings containing limited single-scale feature information about cracks and restricts transformer networks performance on crack detection. Previous research has shown the advantages of establishing interactions between features of different scales in improving network performance [34,35], inspired by which CSPL module is proposed to integrate the multi-scale feature extraction into Patch Embedding module for a transformer network. CSPL module comprises three types of convolution layer: 4 × 4, 8 × 8, and 3 × 3 dilated convolution layers [36], to establish feature interactions at different scales [37]. The specific parameters of CSPL are listed in Table 1. The introduction of dilated convolution layer (3 × 3) expands the perceptual field while maintaining computational efficiency and capturing more spatial context information to better understand the contextual environment of cracks [38]. To manage computation complexity, the dimension of convolutional layers with larger kernel size (8 × 8) is set to dim/4 while those with smaller kernel size (4 × 4) are set to dim/2. The dimension of Dilated convolution layer is also set to dim/4. For stages 2, 3, and 4, the stride of CSPL is set to 2 × 2. As shown in Fig. 3, CSPL takes a crack feature map as input and generates patches using three types of convolutions. The stride of the three convolutions is set to 4 × 4 to ensure the same number of embeddings are generated, and resulting three embeddings are concatenated into one embedding on the dimension.

In stage 1 of PCTNet, the specific operations for CSPL are as follows: Given a crack RGB image with the dimension of 3 (Channel) × 896 (Height) × 896 (Width) as the input. CSPL, which contains three types of convolutions, is used to extract the multi-scale features of the crack image and generate three feature maps with different dimensions (48 ×

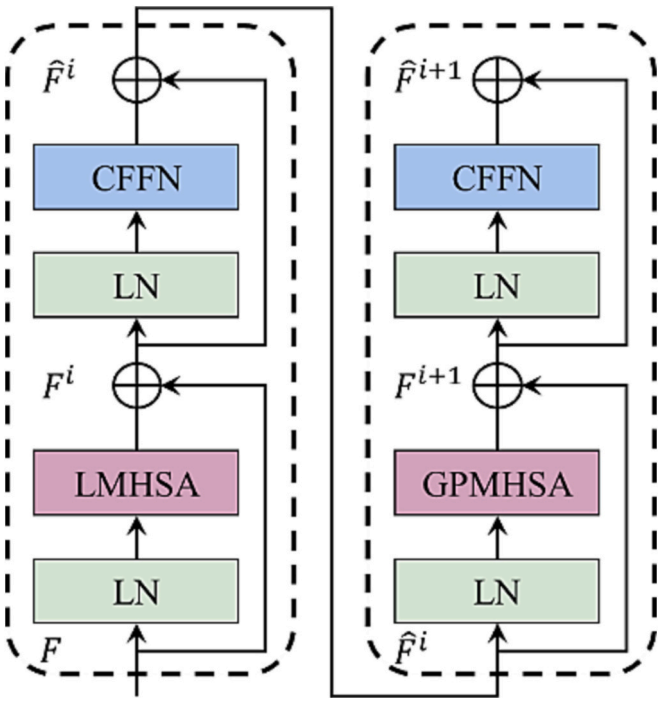


Fig. 4. Architecture of dual attention transformer block.

224 × 224, 24 × 224 × 224, and 24 × 224 × 224). Then, these feature maps are flattened, transposed, and connected to generate new feature maps (96 × 224 × 224) with smaller dimensions and retain the features of the original crack image. At last, the new feature maps are flattened into vector representations for subsequent processing and computation.

2.3. Dual attention transformer block

In PCTNet, Dual Attention Transformer Block shown in Fig. 4 comprises several key modules, including LayerNorm (LN) [39], Local-Multi-Head Self-Attention (LMHSA), Global Pooling-Multi-Head Self-Attention (GPMHSA), and a Convolution Feed-Forward Network (CFFN). These components work in synergy to enable effective feature interactions and contextual information capture within the transformer block, facilitating accurate feature representation of PCTNet. Unlike BN, LN calculates the normalized mean and variance for a specific dimension of a single training sample. LMHSA module further enhances local perception by dividing the feature map into sub-windows and performing self-attention calculations while reducing computational complexity and parameter count. On the other hand, GPMHSA module generates Query (Q) with global feature information through linear projection. In addition, a pooling and convolution residual block is introduced to generate Key (K) and Value (V), capturing rich contextual feature information. The preceding process can be expressed as follows:

$$\begin{aligned} F^i &= LMHSA(LN(F)) + F \\ \widehat{F}^i &= CFFN(LN(F^i)) + F^i \\ F^{i+1} &= GPMHSA(LN(\widehat{F}^i)) + \widehat{F}^i \\ \widehat{F}^{i+1} &= CFFN(LN(F^{i+1})) + F^{i+1} \end{aligned} \tag{1}$$

where F denotes the input features, i denotes the current block, $\widehat{F}^i$ denotes the output features of CFFN module, F^i and F^{i+1} denotes the output features of LMHSA and GPMHSA module, respectively.

2.3.1. LMHSA

LMHSA module is proposed to facilitate self-attention calculations within local sub-windows, allowing the network to capture fine-grained contextual information while reducing computational complexity. As depicted in Fig. 5, it divides the feature map into sub-windows and performs self-attention calculations. The specific calculation process is as follows: Initially, let $E \in \mathbb{R}^{HW\times C}$ denote the input embeddings, where H and W represent the height and width of the input image, respectively, and C denotes the channel of input embeddings. Subsequently, the input embeddings $E \in \mathbb{R}^{HW\times C}$ are reshaped into feature maps $F \in \mathbb{R}^{H\times W\times C}$. These features maps $F \in \mathbb{R}^{H\times W\times C}$ are then equally divided into m × n sub-windows ($W \in \mathbb{R}^{\frac{H}{m}\times\frac{W}{n}\times C}$), enabling self-attention operations to be conducted within each sub-window. The sub-windows generate $Q \in \mathbb{R}^{N\times h\times\frac{HW}{mn}\times d_h}$, $K \in \mathbb{R}^{N\times h\times\frac{HW}{mn}\times d_h}$, and $V \in \mathbb{R}^{N\times h\times\frac{HW}{mn}\times d_h}$ by linear projection operations. Here, N represents the number of sub-windows, h denotes the number of heads, $\frac{HW}{mn}$ denotes the size of each sub-window, and d_h denotes the dimension of each head. Thus, the calculation of attention is undertaken as follows:

$$Attention(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2}$$

where K^T denotes the transpose metric of K, and d_k denotes the dimension of K.

The computational complexity of LMHSA module is analyzed as follows: Matrix Q, K, and V are generated by matrix multiplication, where M_Q, M_K, and M_V denotes the respective transformation matrix, $\otimes$ is a matrix multiplication operation. The computational complexity of this process is $\Omega(3HWC^2)$. The calculation process can be expressed as:

$$\begin{aligned} W^{\frac{H}{m}\times\frac{W}{n}\times C} \otimes M_Q^{C\times C} &= Q^{\frac{H}{m}\times\frac{W}{n}\times C} \\ W^{\frac{H}{m}\times\frac{W}{n}\times C} \otimes M_K^{C\times C} &= K^{\frac{H}{m}\times\frac{W}{n}\times C} \\ W^{\frac{H}{m}\times\frac{W}{n}\times C} \otimes M_V^{C\times C} &= V^{\frac{H}{m}\times\frac{W}{n}\times C} \end{aligned} \tag{3}$$

Q, K, and V are utilized to perform self-attention operations. Initially, matrix Q and the transpose matric of K are multiplied to generate a weighting coefficients matrix X using matrix multiplication. Subsequently, matrix X and matrix V are multiplied to obtain the attention

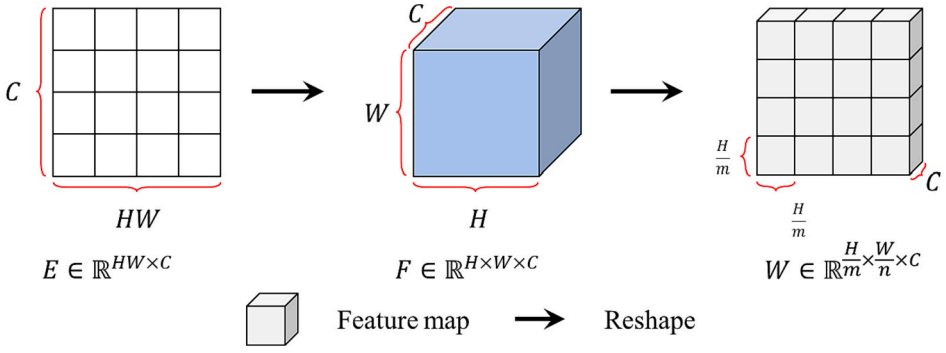


Fig. 5. Architecture of LMHSA module.

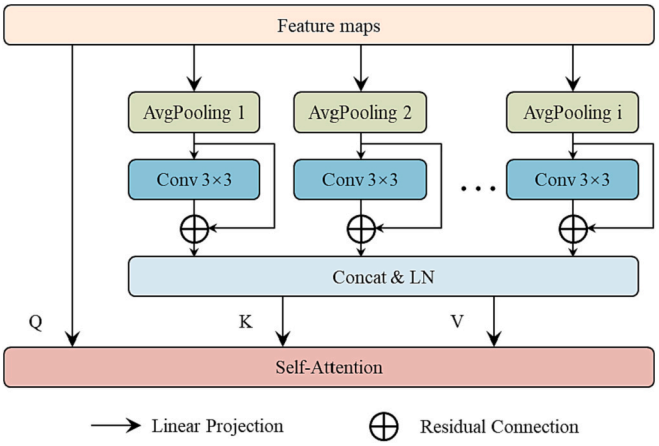


Fig. 6. Architecture of GPMHSA module.

value matrix A. Since LMHSA is a multi-headed self-attentive module, the attention value matrix A of each head is fused by the fusion matrix M. The computational complexity of this process is $\Omega\left(HWC^2 + 2\frac{H^2W^2}{mn}C\right)$, and the calculation process is expressed as:

$$\begin{aligned} &Q^{\frac{H}{m}\times\frac{W}{n}\times C} \otimes K^{T\left(C\times\frac{H}{m}\times\frac{W}{n}\right)} = X^{\frac{HW}{mn}\times\frac{HW}{mn}} \\ &X^{\frac{HW}{mn}\times\frac{HW}{mn}} \otimes V^{\frac{H}{m}\times\frac{W}{n}\times C} = A^{\frac{H}{m}\times\frac{W}{n}\times C} \\ &A^{\frac{H}{m}\times\frac{W}{n}\times C} \otimes M_O^{C\times C} = O^{\frac{H}{m}\times\frac{W}{n}\times C} \end{aligned} \tag{4}$$

Compared to the self-attention mechanism of ViT, the computational complexity of LMHSA in the proposed is reduced from $\Omega\left(4HWC^2 + 2H^2W^2C\right)$ to $\Omega\left(4HWC^2 + 2\frac{H^2W^2}{mn}C\right)$. In the specific case of stage 1, where the input feature maps are of size 224 × 224 with 96 channels and 7 × 7 sub-windows, the computational complexity is reduced by 4.735×10^{11} FLOPs when compared to the self-attention mechanism of ViT.

Since the sub-windows generate Q, K, and V matrices, the self-attention operation occurs within each sub-window, enabling LMHSA module to fully extract local information. However, to avoid information being limited to local processing and to prevent degradation of performance and shrinkage of the perceptual field, it is necessary to establish communication between different sub-windows. To address this issue, GPMHSA module is introduced to enable information interaction between windows. GPMHSA module will be described in detail in 2.3.2.

2.3.2. GPMHSA

GPMHSA module is utilized to establish global feature information interactions, as shown in Fig. 6. The pooling and convolution residual blocks are introduced to capture rich contextual information, which comprises adaptive averaging pooling layers, convolution layers, and a residual connection. As mentioned in 2.3.1, the input embedding $E \in \mathbb{R}^{HW\times C}$ is reshaped to feature maps $F \in \mathbb{R}^{H\times W\times C}$. The feature maps $F \in \mathbb{R}^{H\times W\times C}$ generate Q ($Q \in \mathbb{R}^{H\times W\times C}$) directly by linear projection, representing global feature information, and generates K and V by pooling and convolution residual block. The feature maps $F \in \mathbb{R}^{H\times W\times C}$ are fed into adaptive averaging pooling layers with different ratios, like:

$$\mathrm{F}_i \in \mathbb{R}^{N_i} = AdaptiveAvgPool_i(F), i = 1, 2, 3, 4 \tag{5}$$

where F_i denotes the generated feature maps with various sizes,N_i denotes the length of sequence of individual feature maps, and i is the number of pooling layers. The output size of four adaptive averaging pooling layers is 5 × 5, 4 × 4, 3 × 3, and 2 × 2, respectively. The generated feature maps are fed into 3 × 3 convolution for further feature

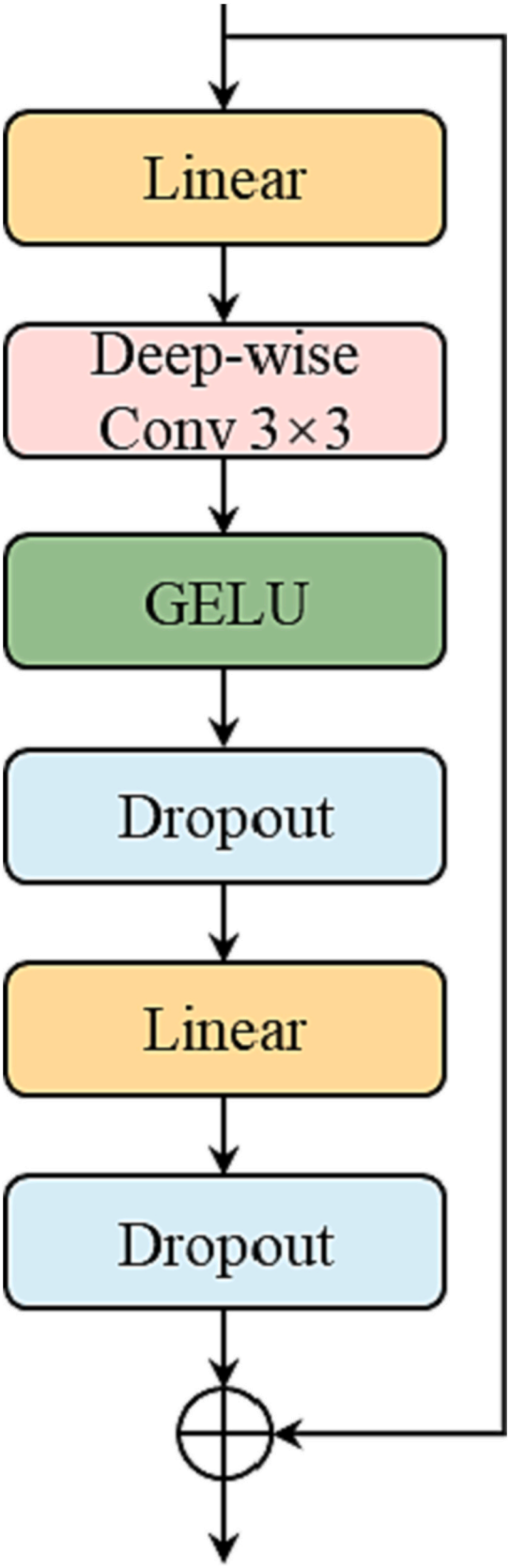


Fig. 7. Architecture of CFFN module.

Table 2
Details of different crack datasets.

Datasets	Image resolution	Training set	Test set
Crack 896	896 × 896	4788	1197
Crack-R	1512 × 2016	685	172

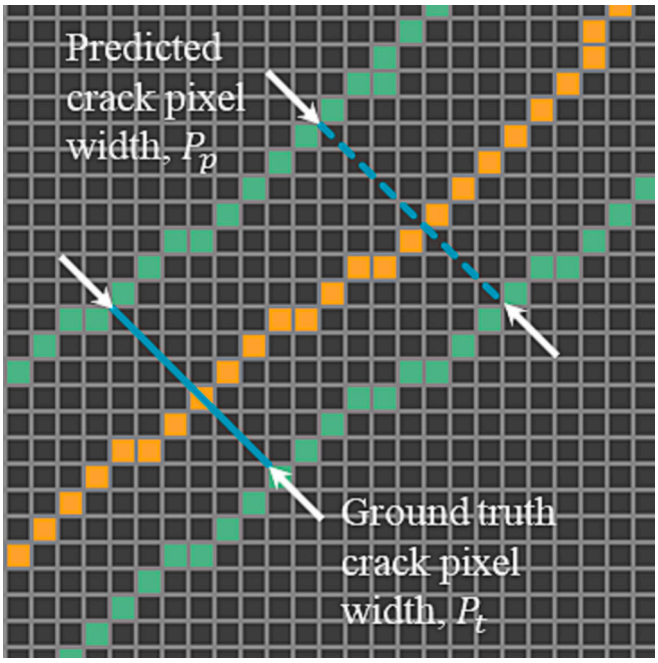


Fig. 8. Pixel IoU schematic representation.

Table 3
Comparison with other well-known networks (bold represents the best).

Network	Precision (%)	Recall (%)	F1-score (%)	mIoU (%)
DeepLabV3+	**96.61**	83.17	89.02	81.83
U-Net	95.32	87.67	91.12	84.79
SegNeXt	94.52	88.74	92.94	85.81
BiFormer	94.23	91.12	92.61	87.03
PVTv2-B4	89.51	89.34	91.87	85.9
PVTv2-B5	89.52	89.55	92.08	86.23
Swin-tiny	93.69	92.68	91.73	87.13
Swin-small	94.36	93.20	92.11	87.94
PCTNet	95.89	**93.77**	**94.8**	**90.53**

extraction, such that:

$$\widehat{F}_i \in \mathbb{R}^{C\times N_i} = Conv(F_i) + F, i = 1,2,3,4 \tag{6}$$

Subsequently, the output feature maps $\widehat{F}_i$ will be flattened, concatenated, and performed LN operation, as:

$$\mathrm{X} \in \mathbb{R}^{C\times N} = LayerNorm(Concat(\widehat{F}_1, \widehat{F}_2, \widehat{F}_3, \widehat{F}_4)) \tag{7}$$

where X denotes the output sequence, and N denotes the sequence length. The resulting X contains rich contextual information on feature maps F. Thus, it can replace F to generate K and V. Q, K, and V are fed into the attention module to compute the attention values.

Q contains the feature information from the entire feature maps, while K and V have richer multi-scale feature information. As a result, GPMHSA module is capable of extracting more contextual feature information and cross-utilizing it with LMHSA module mentioned above so that the local features can be well represented during the interaction with the global features.

The computational complexity of GPMHSA module is analyzed as follows: Matrix Q is generated by feature maps $F \in \mathbb{R}^{H\times W\times C}$, and K, V are generated by output sequence $X \in \mathbb{R}^{C\times N}$, M_Q, M_K and M_V denotes the respective transformation matrix, $\otimes$ is a matrix multiplication operation. The computational complexity of this process is $\Omega((HW+2N)C^2)$. The calculation process can be expressed as:

$$\begin{aligned} F^{H\times W\times C} \otimes M_Q^{C\times C} &= Q^{H\times W\times C} \\ X^{N\times C} \otimes M_K^{C\times C} &= K^{N\times C} \\ X^{N\times C} \otimes M_V^{C\times C} &= V^{N\times C} \end{aligned} \tag{8}$$

Q, K, and V are used to perform self-attention operations, and the process is similar to that of LMHSA. The computational complexity of this process is $\Omega(HWC^2 + 2HWCN)$, and the calculation process is expressed as:

$$\begin{aligned} Q^{H\times W\times C} \otimes K^{T(C\times N)} &= X^{H\times W\times N} \\ X^{H\times W\times N} \otimes V^{N\times C} &= A^{H\times W\times C} \\ A^{H\times W\times C} \otimes M_O^{C\times C} &= O^{H\times W\times C} \end{aligned} \tag{9}$$

The computational complexity of GPMHSA is reduced from $\Omega(4HWC^2 + 2H^2W^2C)$ to $\Omega(2[(HW+N)C^2 + HWCN])$, resulting in a reduction of 4.84×10^{11} FLOPs of computational cost at stage 1.

2.3.3. CFFN

Fig. 7 depicts the architecture of CFFN module. After the self-attention calculation, the matrix of self-attention values A is obtained. The first step is a linear transformation, which expands the input channels to four times of the original number. Notably, unlike other transformer networks, PCTNet employs a simple positional encoding method, i.e., zero paddings on the feature maps, followed by 3 × 3 Depthwise convolution [40] to extract the location information. Subsequently, the GELU activation [41] function is applied to enhance the expressiveness of the sequence. Specifically, the GELU activation function strengthens the parts with larger values relative to itself and suppresses the parts with small values, resulting in improved network generalization. The GELU function is defined as follows:

$$GELU(x) = x\phi(x) \tag{10}$$

where x denotes the input, and $\phi(x)$ is the probability function of normal distribution. The dropout layer [42] is incorporated to mitigate the risk of overfitting during neural networks training. Finally, a linear transformation is employed to reduce the output channels to the original channels, and the resulting feature maps are fed back into the dropout layer.

3. Methodology

In this section, two concrete crack datasets, including Crack 896 and Crack-Real (Crack-R), are first introduced, detailed hyperparameters for network training are also presented, and the end provides performance evaluation metrics for networks.

3.1. Dataset description

Previous researchers have focused on improving the overall performance of crack segmentation rather than the pixel-level segmentation capability of individual cracks [43,44]. Based on that, this paper establishes two datasets, Crack 896[1] and Crack-R[1], and the details are shown in Table 2. The former is used to assess PCTNet's overall performance for crack segmentation and as comparative benchmark with existing CNNs and transformer networks using common evaluation metrics. The latter is for evaluating PCTNet's effectiveness for crack

[1] https://drive.google.com/drive/folders/1vQXJ9VH6g-gz-W0F9IpU1HakKs7H2hNM?usp=drive_link

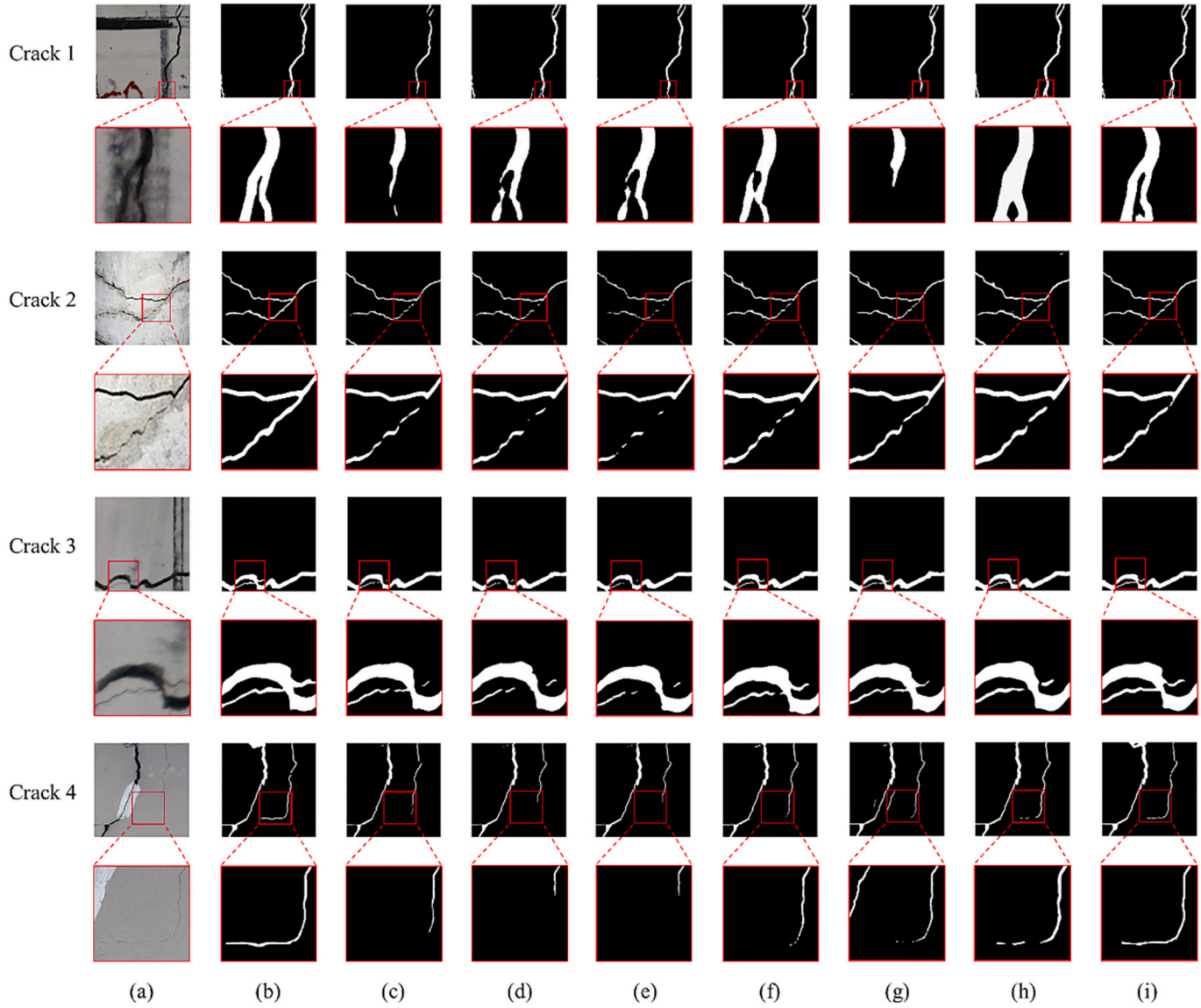


Fig. 9. Segmentation results by various networks. (a) Input image; (b) Ground truth; (c) DeepLabV3+; (d) U-Net; (e) SegNeXt; (f) BiFormer; (g) PVTv2-b5; (h) Swin-small; (i) PCTNet.

Table 4
Comparison of network computational efficiency. "GFLOPs" is calculated under the input scale of 896 × 896.

Network	Parameters (M)	FLOPs (G)	mIoU (%)	FPS
BiFormer	64.05	167.52	87.04	19.01
PVTv2-B4	62.04	157.12	85.90	12.66
PVTv2-B5	81.44	182.10	86.23	11.97
Swin-tiny	27.50	69.97	87.13	50.03
Swin-small	48.79	136.74	87.94	30.33
PCTNet	30.05	65.55	90.53	24.12

Table 5
Pixel-physical conversion coefficient k at different distances.

Distance (cm)	k
50	1 mm = 3.8 Pixels
80	1 mm = 1.9 Pixels
100	1 mm = 1.5 Pixels

segmentation at the pixel-level and demonstrating the applicability of PCTNet in practical engineering scenarios, and to compare it with DeepLabV3+ and Swin Transformer by the proposed two metrics (Pixel IoU and relative error rate). Efficient Interactive Segmentation (EISEG) [45] is used to mark concrete damage.

1) **Crack 896:** This dataset is relatively comprehensive, encompassing crack images with various shapes and backgrounds. The majority of Crack 896 consists of manually captured crack photos, as well as publicly available crack datasets. The DJI Mavic mini UAV was employed to capture the images of surface of structure, resulting in a total of 125 raw images with a resolution of 4000 × 3000 pixels. These images were then cropped to obtain 1500 crack images, each sized at 896 × 896 pixels. Data augmentation techniques, including brightness, gaussian, and others, were applied to augment the diversity of the dataset to enhance the generalization and robustness of PCTNet. Finally, 5985 crack images from the dataset were resized to 896 × 896, with 80% of images used for training and 20% for validation.

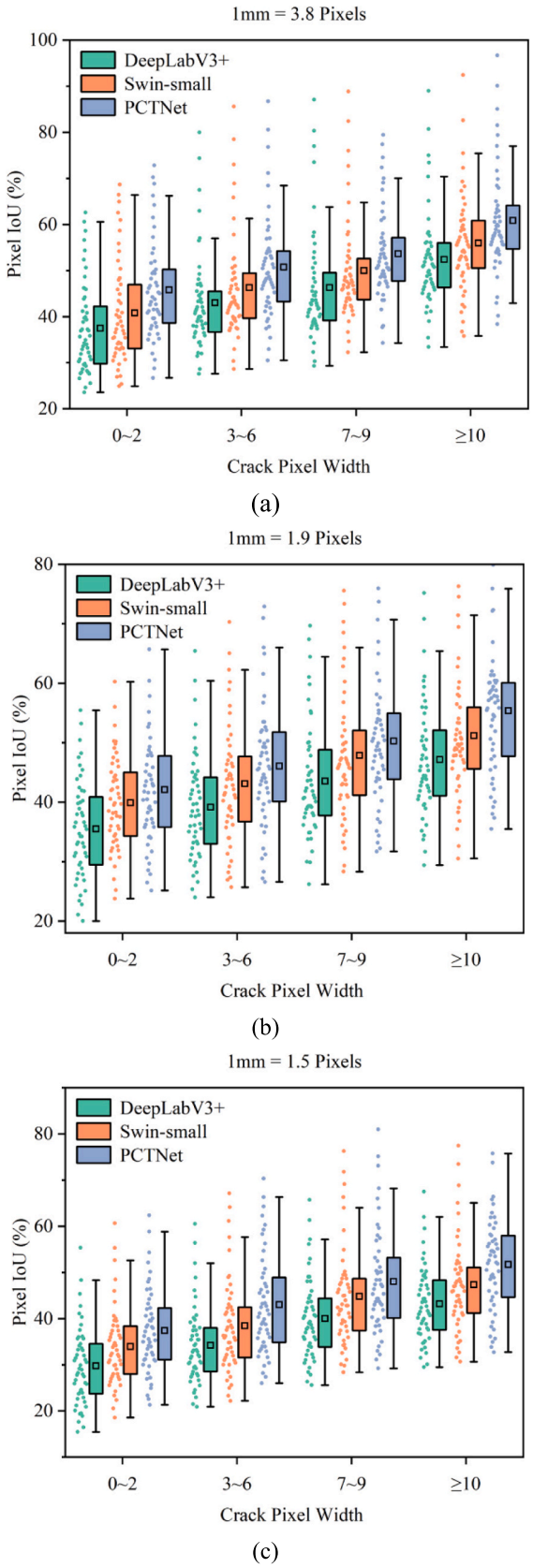


Fig. 10. Box and scatter plots for different crack width at the shooting distance of (a)50 cm; (b) 80 cm; and (c) 100 cm.

2) **Crack-R**: The distance between the handheld camera from the structure's surface was measured using a laser rangefinder at various distances (30, 40, 50, 60, 70, 80, 90, and 100 cm). The aperture on the iPhone camera was set to 1.6*f*, the focal length was 35 mm, and the raw resolution was 3024 × 4032 pixels. To better explore the effectiveness of PCTNet for pixel-level width cracks in practical engineering scenarios, and considering that current dominant GPU memory is limited to 24 GB, the resolution of 857 collected crack images was adjusted to 1512 × 2016 pixels to enable network training and validation.

3.2. Implementation details and network initialization

PCTNet network is built on the MMSegmentation [46] framework and trained on a Windows PC system with an Intel Xeon W-2150B CPU and a single NVIDIA RTX 3090 24 GB GPU. The weights of the network are initialized by Kaiming initialization strategy [47] and updated by an AdamW optimizer [48]. The initial learning rate is set to 4.5×10^{-5}, and weight decay is set to 0.0001. A total of 20,000 iterations are performed during each training process, with a mini-batch size of 2 used in each iteration. Regularization in the form of L2 norm [49] is employed to mitigate network overfitting during training. Additionally, transfer learning is applied to PCTNet network, utilizing pretraining on the ADE20K dataset [50] to achieve better performance and faster convergence of the network.

3.3. Loss function

Crack segmentation can be viewed as a binary classification task; however, in the crack images used for training, the proportion of background pixels is considerably higher than that of the crack pixels [51], potentially resulting in an imbalance in the categories. To address this issue, the Cross Entropy loss function [52] is employed to quantify the prediction errors. This loss function takes into account the proportion of each category in the dataset and sets the inverse of the proportion to the corresponding weight, as expressed by the following equation:

$$Loss = -[ylog(p) + (1-y)log(1-p)] \tag{11}$$

where y denotes the true label and p denotes the predicted probability.

3.4. Performance evaluation metrics

In this work, a comprehensive set of performance evaluation metrics, including precision, recall, F1-score, and mIoU are employed to assess the effectiveness of PCTNet and other networks for crack semantic segmentation. For the crack segmentation task, true positive (TP), false positive (FP), and false negative (FN) are as follows: TP represents a crack pixel that is correctly detected as a crack pixel, FP indicates a non-crack pixel that is incorrectly detected as a crack pixel, and FN represents a crack pixel that is incorrectly detected as a non-crack pixel.

Precision, which is defined in Eq. (12), is the ratio of the number of pixels predicted as cracks to the number of pixels that actually belong to cracks.

$$precision = \frac{TP}{TP + FP} \tag{12}$$

Recall is the ratio between the number of pixels that are correctly detected as cracks and the number of pixels that are actually cracks. It can be expressed as:

$$recall = \frac{TP}{TP + FN} \tag{13}$$

The F1-score is used to combine precision and recall as an evaluation metric, as shown in Eq. (14).

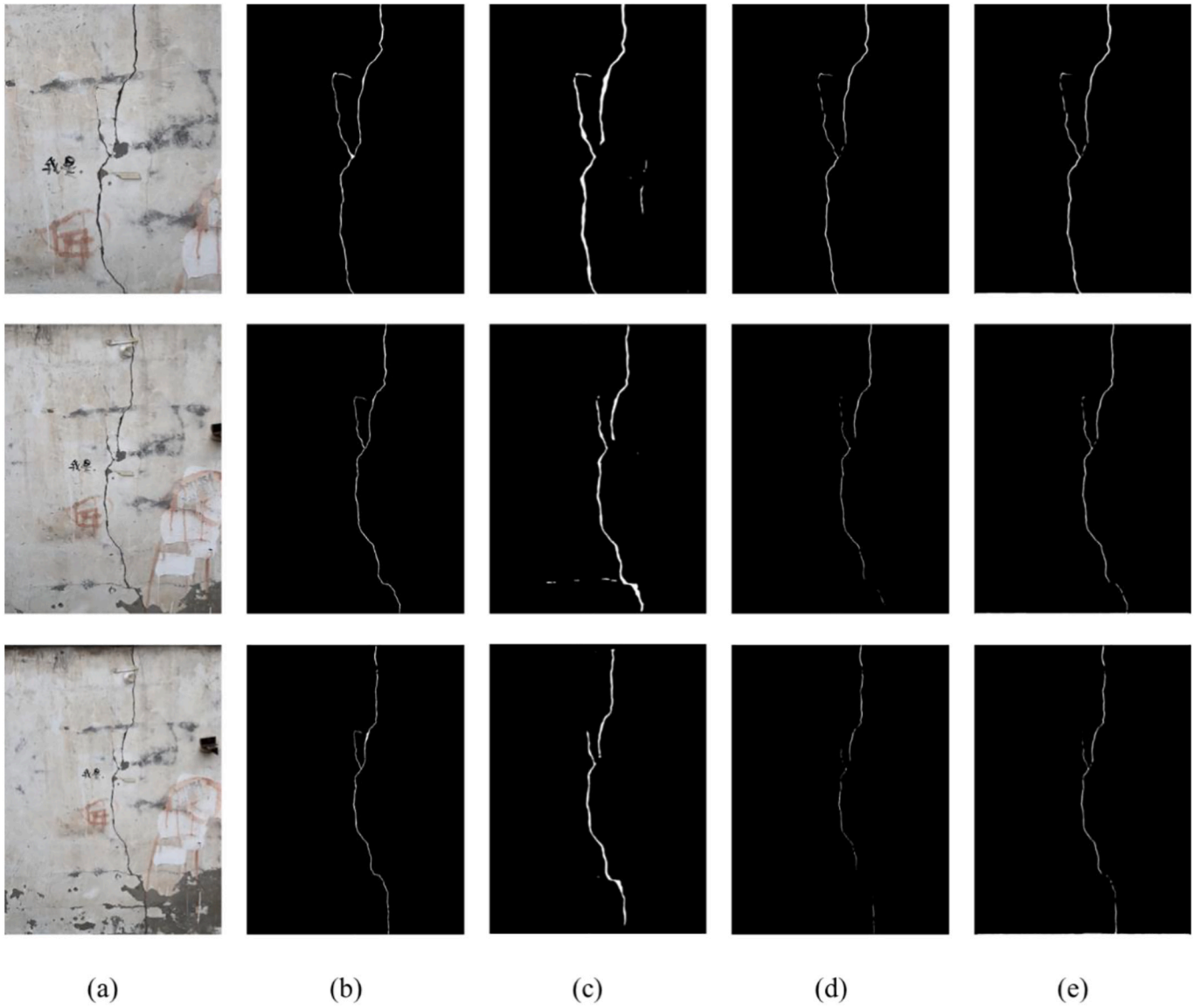


Fig. 11. Crack detection results compared with various networks at the distance of 50 cm, 80 cm, and 100 cm. (a) Input image; (b) Ground truth; (c) DeepLabV3+; (d) Swin-small; (e) PCTNet.

$$F1-score=2\times\frac{PR\times RE}{PR+RE} \tag{14}$$

The mean cross union (mIoU) is a metric to evaluate the overlap between ground truth crack pixels (T) and predicted crack pixels (P), where n represents the number of classes.

$$mIoU=\frac{1}{n}\times\frac{area(T\cap P)}{area(T\cup P)} \tag{15}$$

For pixel-wise accuracy evaluation, two new metrics, i.e., Pixel IoU and relative error rate, are proposed to evaluate the performance of PCTNet at pixel level. As shown in Fig. 8, the cyan line represents the crack's edge, and the yellow line represents the skeleton of the crack. The orthogonal skeleton method is used to measure the ground truth crack pixel width (solid line) and the predicted crack pixel width (dashed line). The Pixel IoU is defined as the intersection ratio of the ground truth crack pixel and the predicted crack pixel width, as shown in Eq. (16), with a higher ratio representing a more accurate detection of cracks with pixel-level widths and a stronger extraction capability of the network for boundary features.

$$Pixel\ IoU=\frac{P_t\cap P_p}{P_t\cup P_p} \tag{16}$$

Relative error rate, as defined in Eq. (17), is used to calculate the value of the relative difference between the number of pixels predicted to be cracks R_p and the number of pixels that belong to cracks R_t. The lower relative error rate, the more accurately the algorithm can differentiate between crack pixels and non-crack pixels.

$$Relative\ error\ rate=\left|\frac{R_p-R_t}{R_t}\times 100\%\right| \tag{17}$$

The proposed Pixel IoU and Relative error rate can provide a more accurate performance evaluation for the peculiarities of high-resolution image crack detection tasks, such as low contrast and fine crack boundaries. Most semantic segmentation networks [53,54] utilize mIoU as the major evaluation metric, but there are some limitations when applied in high-resolution crack segmentation tasks. As the crack details often take up a small part of the high-resolution images, two crack images can have the same mIoU score, whereas the detection accuracy at a specific location can be drastically different. By using the proposed

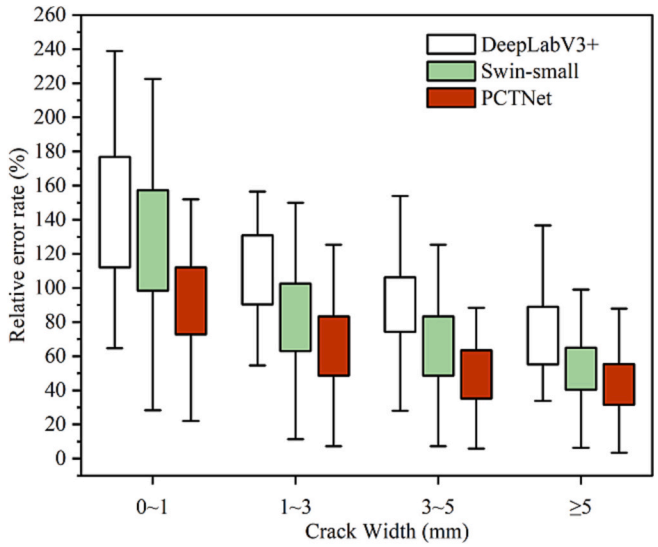


Fig. 12. Box-plot of relative error rates for different crack widths.

Table 6
Ablation study of PCTNet with different modules.

Network	Precision (%)	Recall (%)	F1-score (%)	mIoU (%)
PCTNet	95.89	93.77	94.80	90.53
PCTNet-PE	94.88	91.64	93.65	88.67
PCTNet-OPE	94.64	91.43	92.94	87.59
PCTNet- STA	94.32	90.86	92.64	86.78

Table 7
Detailed parameters for various levels of noises.

	Intensity				
Noise type	1	2	3	4	5
Contrast					
Parameter *alpha*	1.2	1.4	1.6	1.8	2
Gaussian Blur					
Parameter *ksize*	7	11	15	19	25
Parameter *sigma*	7	11	15	19	25
Gaussian Noise					
Parameter *sigma*	2	3	4	5	6
Motion Blur					
Parameter *kernel*	10	20	30	40	50
Salt & Pepper Noise					
Parameter *intensity*	0.4	0.5	0.6	0.7	0.8
Shot Noise					
Parameter *low*	1.2	1.3	1.4	1.5	1.6
Parameter *high*	1.2	1.3	1.4	1.5	1.6

metrics, it is feasible to assess the ability of the network in detecting fine cracks, the accuracy of boundary localization, and other critical performance.

4. Results and discussions

4.1. Results of crack 896 dataset

This section compares the performance of PCTNet with other CNNs and transformer networks on the Crack 896 dataset, including DeepLabV3+ [55], U-Net [56], SegNeXt [57], BiFormer [58], PVTv2, and Swin Transformer.

Table 3 lists the quantitative results of all compared networks on the dataset. It can be observed that PCTNet outperforms other networks in terms of precision, recall, F1-score, and mIoU at 95.89%, 93.77%, 94.8%, and 90.53%, respectively. Compared with CNNs, the precision of PCTNet is lower than that of DeepLabV3+ with 0.72%. However, the other metrics of PCTNet are significantly higher than those of CNNs, with at least 5.03% higher in recall, 1.86% higher in F1-score, and 4.72% higher in mIoU. This can be attributed to the inherent limitations of CNN convolution in extracting global feature information that is crucial for the semantic segmentation task, resulting in relatively poor crack detection performance compared to the transformer networks in this experiment. The self-attention mechanism employed by transformer networks allows for capturing more contextual information about the target and performs better than CNNs. Nevertheless, PCTNet outperforms the transformer networks in all metrics.

Fig. 9 displays the crack detection results of different compared networks on the Crack 896 dataset, and unideal segmentation areas of detection results are marked with a red box. It can be seen that all networks have satisfactory performances in terms of the overall segmentation result. However, other networks show weakness in detecting fine cracks. Consider Crack 4 as an example, CNNs fail to identify most of the cracks within the red box, and transformer networks do not detect cracks continuously. In contrast, PCTNet can precisely detect cracks by extracting local and global features.

To further evaluate the overall performance of compared networks, the following factors are employed to assess the computational efficiency: parameters, FLOPs, mIoU, and FPS. Table 4 shows the comparative results of transformer networks. Compared to Swin Transformer-tiny (Swin-tiny), which has the fewest network parameters, PCTNet has a higher number of parameters due to Dual Attention Transformer Block containing two attention mechanisms. In terms of running speed, PCTNet achieved 24.12 FPS, which is lower than Swin-tiny and Swin Transformer-small (Swin-small), as CSPL module in PCTNet uses multiple convolution kernels to extract feature information at different scales. However, PCTNet achieved the lowest computational complexity and the highest mIoU value of 65.55 GFLOPs and 90.53%, respectively.

4.2. Pixel-level performance analysis based on Crack-R dataset

In this section, Crack-R dataset is utilized for pixel-level performance analysis, considering the influence of different camera placements on the performance of networks. Crack-R dataset contains crack images with different shooting distances. Specifically, 50 crack images, each with a shooting distance of 50 cm, 80 cm, and 100 cm, are selected from Crack-R dataset for validation. Transfer learning (pre-trained on Crack 896 and Crack-R dataset) is used to improve the performance of networks, including PCTNet, DeepLabV3+, and Swin-small, on the Crack-R dataset.

To evaluate the pixel-wise accuracy performance of networks, 50 cracks were chosen from images with each of the three shooting distances, and the orthogonal skeleton line method is used to skeletonize the segmented cracks and then extract the pixel width of the cracks. The ground truth crack width is obtained according to the pixel-physical conversion coefficients at different shooting distances, as shown in Table 5. The crack width is then classified into four groups based on the label map of cracks, namely 0–2 pixels, 3–6 pixels, 7–9 pixels, and more than 10 pixels.

As shown in Fig. 10, the scatter represents the Pixel IoU values for different cracks, the box plots represent the distribution of these values, and the square box represents the mean Pixel IoU value. It can be seen that all networks exhibit better performance for wider cracks, while PCTNet performs best. Given a specific instance, for a crack image taken at a distance of 50 cm, the pixel-physical conversion factor k is 1 mm = 3.8 pixels. This means that the actual width of a crack with a pixel width of 0–2 is approximately 0.8 mm. In this case, the Pixel IoU values of PCTNet are concentrated from 38.62% to 50.25%, and its mean value of Pixel IoU is 45.82%, as illustrated in Fig. 10 (a). Comparatively, Swin-small and DeepLabV3+'s Pixel IoU values are concentrated from

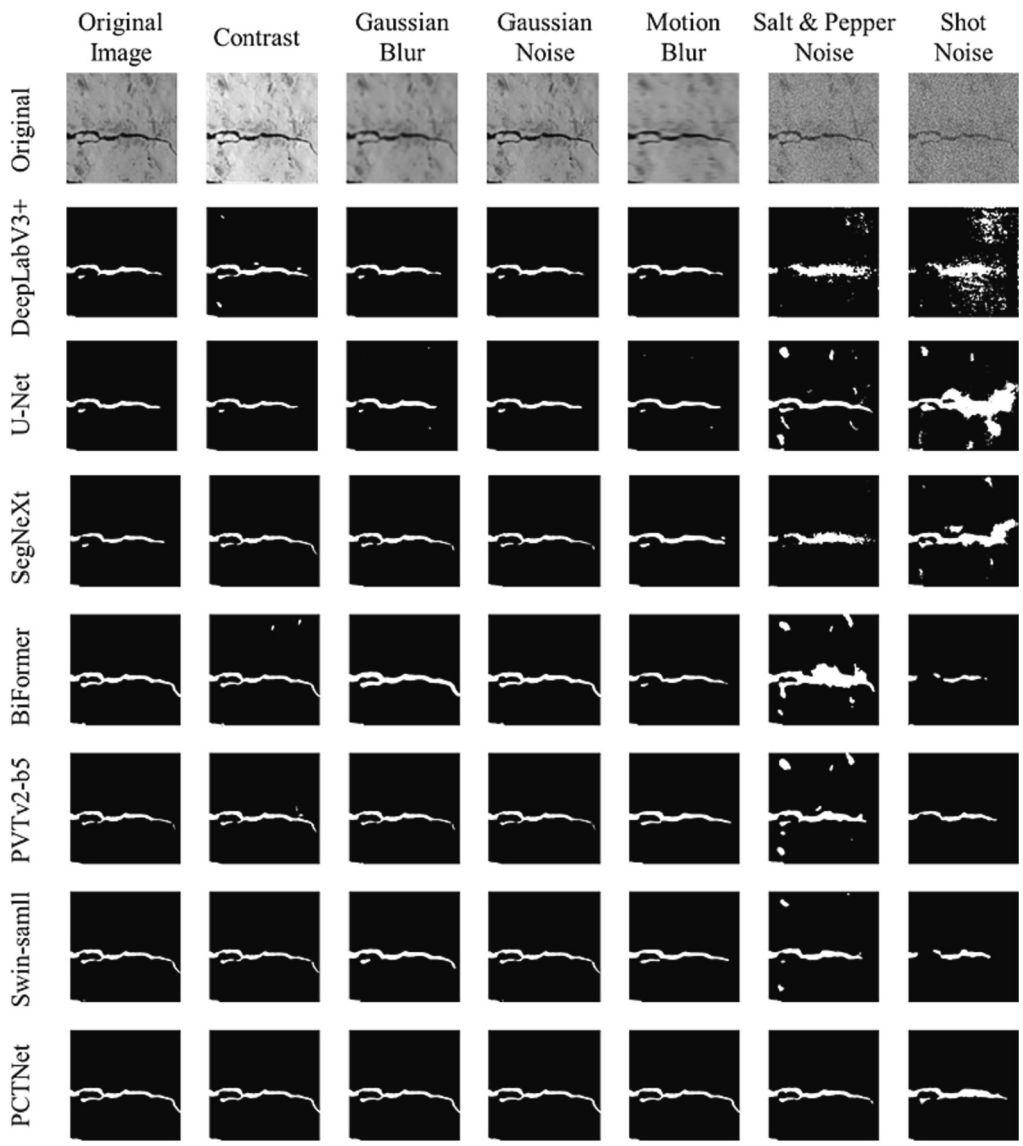


Fig. 13. Assessment of various noise effects.

33.12% to 46.95% and 29.78% to 42.24%, respectively, and their respective mean Pixel IoU value is 40.8% and 37.46%. This demonstrates that PCTNet outperforms current networks for the segmentation of millimeter-wide cracks. As the shooting distance increases, the detection performance of networks inevitably decrease due to the low contrast between the background pixels and the crack pixels, resulting in inaccurate segmentation results. Therefore, the distribution of Pixel IoU values is less concentrated, and the range of the box also expands. Although the box range of PCTNet is broader than the other two networks, its mean Pixel IoU is still much higher than both of them. At the shooting distance of 80 cm, the Pixel IoU of PCTNet for cracks wider than 10 mm is 55.38%, while the values of Swin-small and DeepLabV3+ are 51.19% and 47.16%, respectively. With the camera placement set to 100 cm from the structure surface, the Pixel IoU of the three is 51.7%, 47.36%, and 43.21%, accordingly. Despite the decreased performance, PCTNet performs better and more robustly for pixel-level crack segmentation to be suitable for practical SHM applications.

Fig. 11 displays the crack image with different shooting distances in Crack-R dataset and the detection results of various networks. DeepLabV3+ seems to have the best performance but is actually the least effective of three networks. The main reason for this phenomenon is that due to the limitation of convolution, it cannot accurately extract sufficient feature information to distinguish crack pixels from background pixels. Hence, the background tends to be incorrectly identified as cracks. In contrast, Swin-small is more precise in predicting the crack pixels, while the cracks it detects are discontinuous, which is mainly attributed to the self-attention mechanism of Swin-small that still restricts the perceptual field to the sub-window and consequently somewhat hampers its performance. On the contrary, PCTNet can accurately detect the defects, although the detected cracks are slightly discontinuous but very similar to the ground truth defects.

Furthermore, to investigate the overall performance of PCTNet in practical SHM, Crack-R dataset with 400 crack images of different shooting distances was classified into four categories (0–1 mm, 1–3 mm, 3–5 mm, and more than 5 mm) based on the ground truth crack width. Fig. 12 shows the relative error rate plots for different crack widths. It can be seen that the relative error rate of PCTNet for different crack widths is lower than others, which means that it can accurately detect crack pixels despite the different camera placements and the low contrast between crack pixels and background pixels. In addition, the relative error rate of PCTNet is more concentrated, which indicates that it has better generalization and robustness to meet the challenges of complex environments.

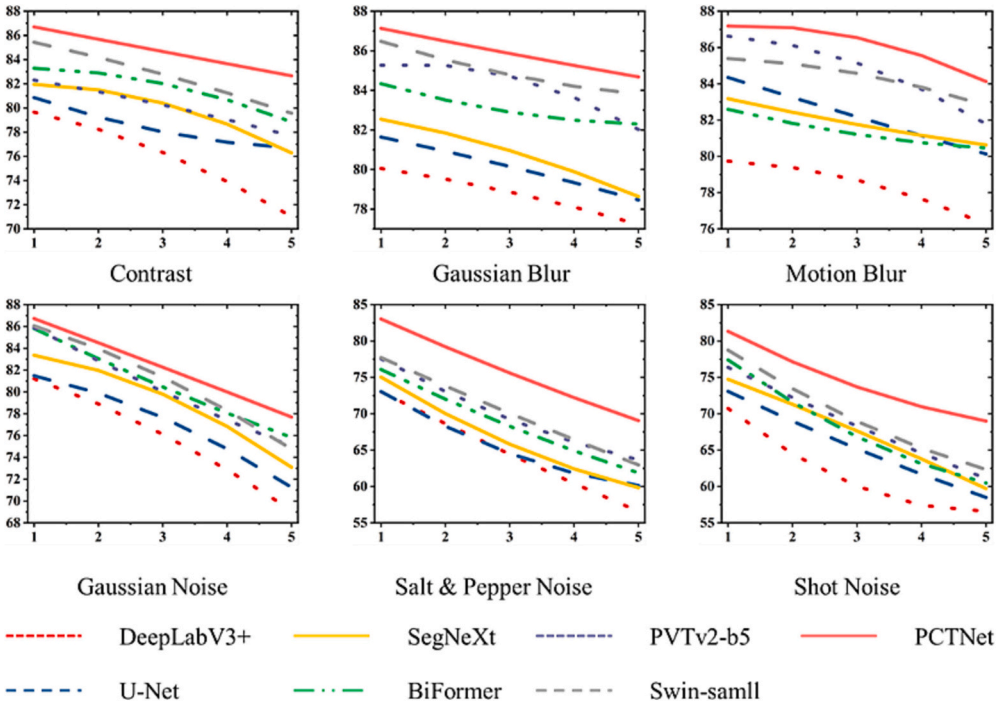


Fig. 14. Quantitative assessment of various noise effects; x-axis: intensity y-axis: mIoU (unit: %).

Table 8
Robustness test results of comparison networks under various noise intensities (unit: %).

	Contrast	Gaussian Blur	Motion Blur	Gaussian Noise	Salt & Pepper Noise	Shot Noise
DeepLaBV3+	75.83	78.75	78.35	75.65	64.63	61.85
U-Net	78.39	80.10	82.21	77.02	65.57	65.49
SegNeXt	79.76	80.77	81.83	79.01	66.63	67.44
BiFormer	81.54	83.11	81.36	80.64	68.63	67.89
PVTv2-b5	80.13	84.18	84.67	80.24	69.92	68.52
Swin-small	82.64	84.96	84.34	80.89	70.21	69.75
PCTNet	84.67	85.88	86.10	82.24	75.82	74.43

Table 9
Performance on public datasets (bold represents the best).

Network	Crack 458				Crack level				SDNET 2018			
	Precision (%)	Recall (%)	F1-score (%)	mIoU (%)	Precision (%)	Recall (%)	F1-score (%)	mIoU (%)	Precision (%)	Recall (%)	F1-score (%)	mIoU (%)
DeepLabV3+	97.79	92.57	95.42	90.49	97.65	97.01	97.33	93.88	**95.74**	90.51	91.32	83.72
U-Net	97.63	96.18	96.89	93.07	97.27	97.01	97.14	93.54	95.17	91.06	93.12	87.54
SegNeXt	96.77	**97.52**	97.14	93.53	97.2	96.88	**97.66**	93.26	94.35	92.92	94.09	88.92
BiFormer	97.11	97.06	97.09	94.44	97.36	97.24	97.22	94.66	93.34	92.73	93.52	88.36
PVTv2-B5	97.33	97.12	96.99	93.69	97.47	96.9	97.18	93.62	93.1	93.77	93.8	88.77
Swin-tiny	97.72	96.97	97.34	93.9	97.76	97.17	97.46	94.13	93.17	93.93	93.54	88.4
Swin-small	98.06	97.14	97.59	94.37	98.04	97.13	97.58	94.34	93.54	93.51	93.63	89.17
PCTNet	**98.32**	97.38	**97.65**	**96.13**	**98.56**	**97.45**	97.47	**96.29**	94.54	**95.68**	**95.04**	**92.26**

4.3. Ablation study

To illustrate the effectiveness of different modules of PCTNet, ablation experiments are carried out. Specifically, CSPL, Dual Attention Transformer Block modules, and CFFN will be replaced with other modules to validate their effectiveness.

4.3.1. Effect of CSPL module

The original PatchEmbeding module of ViT network and the Overlapping PatchEmbeding module of PVTv2 network is used to replace CSPL, and named PCTNet-PE and PCTNet-OPE, respectively. As shown in Table 6, compared to PCTNet, the PCTNet-PE, and PCTNet-OPE decrease at least 1.01%, 2.13%, 1.15%, and 1.86% in precision, recall, F1-score, and mIoU, respectively. This reveals that CSPL module used in PCTNet is more effective than other PatchEmbedding modules at extracting multi-scale feature information from feature maps.

4.3.2. Effect of dual attention transformer block module

To demonstrate the effectiveness of Dual Attention Transformer Block in allowing local feature information to interact with global feature information, Dual Attention Transformer Block is replaced with a standard transformer block, resulting in a new network named PCTNet-STA. As seen in Table 6, PCTNet outperformed PCTNet-STA by 1.57%, 2.91%, 2.16%, and 3.75% in precision, recall, F1-score, and

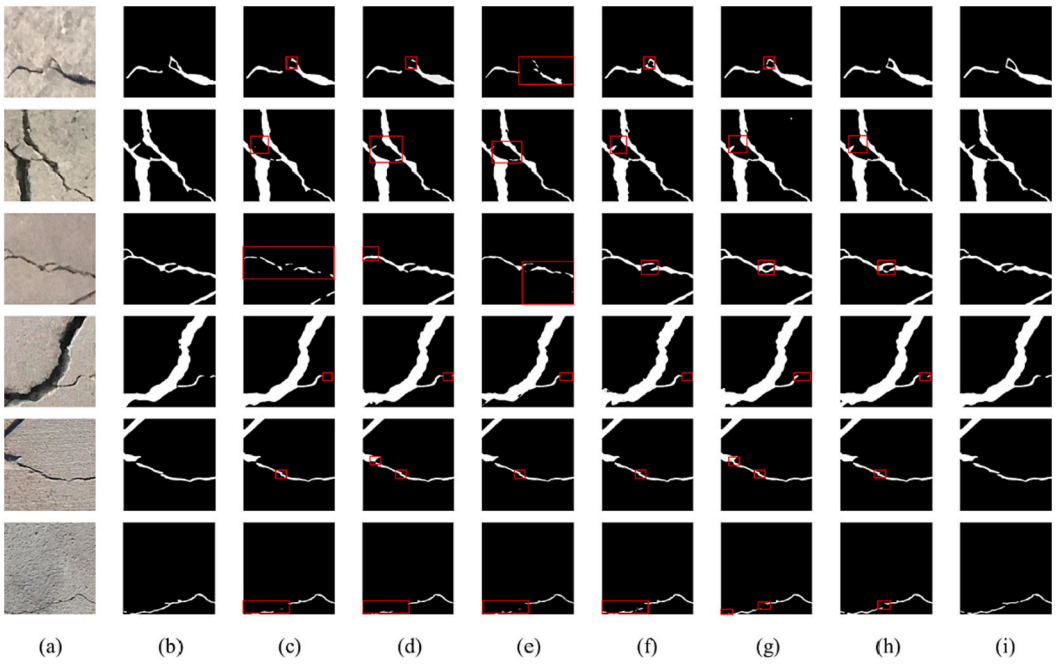


Fig. 15. Crack detection results compared with various networks on public datasets. First two rows contain samples taken from Crack 458, next two rows from Crack level, and final two rows contain samples taken from SDNET 2018. (a) Input image; (b) Ground truth; (c) DeepLabV3+; (d) U-Net; (e) SegNeXt; (f) BiFormer; (g) PVTv2-b5; (h) Swin-small; (i) PCTNet.

mIoU values, respectively, showing that Dual Transformer Block can extract rich contextual information using LMHSA and GPMHSA.

4.4. Robustness test

In this paper, the robustness of PCTNet was studied on different types of noises with varying intensity applied to the original crack images of Crack 896 dataset. Six different noises, including contrast, gaussian blur, motion blur, gaussian noise, salt & pepper noise, and shot noise, were considered. The intensity level of noises is increased from 1 to 5. The detailed parameters for various levels of noises are shown in Table 7. Fig. 13 shows crack images with applied various noises and detection results of networks. It can be seen that compared with other networks, PCTNet shows better robustness and generalization, especially for crack images under the perturbation of Salt & Pepper Noise and Shot Noise, which are the most destructive to the image.

Fig. 14 shows the changes in mIoU experienced by various studied networks against the intensity of noise, which indicates that as the intensity of noise increases, PCTNet exhibits promising stability and robustness and can maintain good performance in the face of noise interference. In contrast, the other networks suffered a more pronounced performance degradation as the noise intensity increased.

Table 8 exhibits the mean mIoU values for different networks on the dataset with various intensities of noise. The applied noise types with the most detrimental negative effects on the network's detection performance are Salt & Pepper Noise and Shot Noise. Salt & Pepper Noise randomly changes the pixel values in the image, which may lead to blurring or disappearance of crack edges and mislead the crack detection algorithm to recognize the noisy points as cracks. Furthermore, Shot Noise induces brightness changes in the pixel points around the cracks, which makes the boundary-blurring challenging to recognize. However, PCTNet shows significant advantages over other networks in terms of mean mIoU values which are at least 5.61% and 4.68% higher than others under the interference of these two types of noise, respectively.

4.5. Evaluation of PCTNet on public datasets

To further demonstrate generalization of PCTNet, comparison experiments are conducted on three public datasets: the Crack 458 [59], Crack level [60], and SDNET 2018 [61] datasets.

The Crack 458 dataset was collected from various METU Campus Buildings and contains 458 high-resolution images (4032 × 3024 pixels) taken in illuminated conditions, from which 400 cracked images of 896 × 896 pixels are chosen for training and 100 for validation.

Table 9 demonstrates that PCTNet outperforms other SOTA networks, with 98.32% on precision, 97.38% on recall, 97.65% on F1-score, and 96.13% on mIoU, respectively. Compared with Swin-small, PCTNet obtains a gain of 1.76% on mIoU. The first two rows of Fig. 15 show samples of the Crack 458 dataset and the results of the comparison of the different networks, which demonstrates that PCTNet can detect fine cracks with high precision.

The Crack level dataset consists of 2000 images of concrete with varying crack levels, from which 400 are used for training and 100 for validation, and all images are resized to 896 × 896 pixels. As shown in Table 9, PCTNet achieved the best performance. Compared to DeepLabV3+ and Swin-small, the gains on mIoU are 2.41% and 1.95%, respectively. In addition, PCTNet detection results are more continuous, as seen in the middle two rows of Fig. 15.

The SDNET 2018 dataset was collected from USU Campus Buildings and captured using a 16MP Nikon digital camera, and it includes photos of pavement, bridge decks, and walls, from which 400,896 × 896 bridge deck crack images are utilized for training and 100 for validation. It can be seen from Table 9 that PCTNet outperformed other networks on recall, F1-score, and mIoU at least 1.75%, 0.95%, and 3.09%, respectively. The last two rows of Fig. 15 indicate that PCTNet has superior robustness and generalization and performs well in detecting pavement cracks.

5. Conclusions

In this paper, a Dual Attention Transformer network for pixel-level concrete crack segmentation named PCTNet is proposed to address the

challenges faced in practical SHM. PCTNet integrates the advantages of CNNs and transformer networks to achieve accurate segmentation of cracks by performing the self-attention calculation of local and global features. Considering the limitations of complex environments for structural surface image acquisition in piratical SHM, Crack-R dataset with crack images of different shooting distances is established to handle this challenge. Compared to common performance evaluation methods, the proposed method utilizes the orthogonal skeleton method to extract the pixel width of cracks and proposes two new metrics, Pixel IoU and relative error rate, to evaluate the performance of PCTNet for pixel-level width cracks in Crack-R dataset, which can precisely represent the detection performance of network for cracks of different widths.

A series of experiments have demonstrated PCTNet exhibits promising performance in detecting cracks in high-resolution images and cracks at millimeter-level resolution. Robustness test experiments and the results on public datasets confirm the robustness and generalization of PCTNet, making it suitable for on-site SHM tasks. However, there are limitations that need to be addressed. Although PCTNet performs well in terms of crack detection accuracy, its computational efficiency is lower than other transformer networks, which poses challenges for real-time inspection on mobile devices. Additionally, this work does not consider the assessment of crack damage severity. Future research will focus on designing lightweight and efficient crack detection networks and combining multiple methods, such as ultrasonic measurement of crack depth, to evaluating the crack damage severity.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the Education Integration Innovation Pilot Program from Qilu University of Technology (Shandong Academy of Sciences) – International Collaboration Project (2022GH006).

References

[1] R. Ali, J.H. Chuah, M.S.A. Talip, N. Mokhtar, M.A. Shoaib, Structural crack detection using deep convolutional neural networks, Autom. Constr. 133 (2022), 103989, https://doi.org/10.1016/j.autcon.2021.103989.
[2] C.V. Dung, Autonomous concrete crack detection using deep fully convolutional neural network, Autom. Constr. 99 (2019) 52–58, https://doi.org/10.1016/j.autcon.2018.11.028.
[3] N. Sholevar, A. Golroo, S.R. Esfahani, Machine learning techniques for pavement condition evaluation, Autom. Constr. 136 (2022), 104190, https://doi.org/10.1016/j.autcon.2022.104190.
[4] Q. Yang, W. Shi, J. Chen, W. Lin, Deep convolution neural network-based transfer learning method for civil infrastructure crack detection, Autom. Constr. 116 (2020), 103199, https://doi.org/10.1016/j.autcon.2020.103199.
[5] A. Rezaie, R. Achanta, M. Godio, K. Beyer, Comparison of crack segmentation using digital image correlation measurements and deep learning, Constr. Build. Mater. 261 (2020), 120474, https://doi.org/10.1016/j.conbuildmat.2020.120474.
[6] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv (2014), https://doi.org/10.48550/arXiv.1409.1556 preprint arXiv:1409.1556.
[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2016) 770–778, https://doi.org/10.1109/CVPR.2016.90.
[8] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2014) 580–587, https://doi.org/10.1109/CVPR.2014.81.
[9] X. He, Z. Tang, Y. Deng, G. Zhou, Y. Wang, L.J.A. Li, UAV-based road crack object-detection algorithm 154 (2023) 105014, https://doi.org/10.1016/j.autcon.2023.105014.
[10] F. Liu, J. Liu, L.J. Wang, Asphalt pavement fatigue crack severity classification by infrared thermography and deep learning, Autom. Constr. 143 (2022), 104575, https://doi.org/10.1016/j.autcon.2022.104575.
[11] F. Panella, A. Lipani, J. Boehm, Semantic segmentation of cracks: data challenges and architecture, Autom. Constr. 135 (2022), 104110, https://doi.org/10.1016/j.autcon.2021.104110.
[12] J. Hang, Y. Wu, Y. Li, T. Lai, J. Zhang, Y. Li, A deep learning semantic segmentation network with attention mechanism for concrete crack detection, Struct. Health Monit. (2023), https://doi.org/10.1177/14759217221126170, 14759217221126170.
[13] H. Zhang, G. Yang, H. Li, W. Du, J.J.A. Wang, Pixel-wise detection algorithm for crack structural reconstruction based on rock CT images, Autom. Constr. 152 (2023), 104895, https://doi.org/10.1016/j.autcon.2023.104895.
[14] Z. Liu, Y. Cao, Y. Wang, W. Wang, Computer vision-based concrete crack detection using U-net fully convolutional networks, Autom. Constr. 104 (2019) 129–139, https://doi.org/10.1016/j.autcon.2019.04.005.
[15] R. Yamashita, M. Nishio, R.K.G. Do, K. Togashi, Convolutional neural networks: an overview and application in radiology, Insights Imag. 9 (2018) 611–629, https://doi.org/10.1007/s13244-018-0639-9.
[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Proces. Syst. 30 (2017), https://doi.org/10.48550/arXiv.1706.03762.
[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, An image is worth 16x16 words: Transformers for image recognition at scale, arXiv (2020), https://doi.org/10.48550/arXiv.2010.11929 preprint arXiv:2010.11929.
[18] S. Paul, P.-Y. Chen, Vision transformers are robust learners, Proc. AAAI Conf. Artific. Intellig. 36 (2022) 2071–2081, https://doi.org/10.1609/aaai.v36i2.20103.
[19] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, A survey on vision transformer, IEEE Trans. Pattern Anal. Mach. Intell. 45 (1) (2022) 87–110, https://doi.org/10.1109/TPAMI.2022.3152247.
[20] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan, M. Shah, Transformers in vision: a survey, ACM Comp. Surv. (CSUR) 54 (10s) (2022) 1–41, https://doi.org/10.1145/3505244.
[21] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pyramid vision transformer: a versatile backbone for dense prediction without convolutions, Proc. IEEE/CVF Int. Conf. Comp. Vision (2021) 568–578, https://doi.org/10.48550/arXiv.2102.12122.
[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: hierarchical vision transformer using shifted windows, Proc. IEEE/CVF Int. Conf. Comp. Vision (2021) 10012–10022, https://doi.org/10.48550/arXiv.2103.14030.
[23] E.A. Shamsabadi, C. Xu, A.S. Rao, T. Nguyen, T. Ngo, D.J.A. Dias-da-Costa, Vision transformer-based autonomous crack detection on asphalt and concrete surfaces, Autom. Constr. 140 (2022), 104316, https://doi.org/10.1016/j.autcon.2022.104316.
[24] W. Wang, C.J. Su, Automatic concrete crack segmentation model based on transformer 139 (2022) 104275, https://doi.org/10.1016/j.autcon.2022.104275.
[25] Z. Xu, H. Guan, J. Kang, X. Lei, L. Ma, Y. Yu, Y. Chen, J. Li, Pavement crack detection from CCD images with a locally enhanced transformer network, Int. J. Appl. Earth Obs. Geoinf. 110 (2022), 102825, https://doi.org/10.1016/j.jag.2022.102825.
[26] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, W. Wu, Incorporating convolution designs into visual transformers, Proc. IEEE/CVF Int. Conf. Comp. Vision (2021) 579–588, https://doi.org/10.1109/ICCV48922.2021.00062.
[27] X. Ye, T. Jin, C. Yun, A review on deep learning-based structural health monitoring of civil infrastructures, Smart Struct. Syst. 24 (5) (2019) 567–585, https://doi.org/10.12989/sss.2019.24.5.567.
[28] M.N. Abou-Zeid, D.W. Fowler, E.G. Nawy, J.H. Allen, G.T. Halvorsen, R. Poston, J. P. Barlow, W. Hansen, R.J. Rhoads, M.E. Brander, N. Hassoun, A. Scanlon, K. L. Carlson, H.H. Haynes, E.K. Schrader, P. Hedli, W. Suaris, T.C.Y. Liu, Z. A. Zielinski, F.G. Barth, Chairman R.J. Frosch, 224R-01 Control of Cracking in Concrete Structures, Rep. ACI Comm 224, 2001, pp. 12–16, https://doi.org/10.14359/10632.
[29] A. Mohan, S. Poobal, Crack detection using image processing: a critical review and analysis, Alex. Eng. J. 57 (2) (2018) 787–798, https://doi.org/10.1016/j.aej.2017.01.020.
[30] H. Liu, X. Miao, C. Mertz, C. Xu, H. Kong, Crackformer: transformer network for fine-grained crack detection, Proc. IEEE/CVF Int. Conf. Comp. Vision (2021) 3783–3792, https://doi.org/10.1109/ICCV48922.2021.00376.
[31] N. Bjorck, C.P. Gomes, B. Selman, K.Q. Weinberger, Understanding batch normalization, Adv. Neural Inf. Proces. Syst. 31 (2018), https://doi.org/10.48550/arXiv.1806.02375.
[32] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814, https://doi.org/10.5555/3104322.3104425.
[33] H. Wan, L. Gao, Z. Yuan, H. Qu, Q. Sun, H. Cheng, R. Wang, A novel transformer model for surface damage detection and cognition of concrete bridges, Expert Syst. Appl. (2022), 119019, https://doi.org/10.1016/j.eswa.2022.119019.
[34] J. Deng, A. Singh, Y. Zhou, Y. Lu, V.C.-S.J.C. Lee, B. Materials, Review on computer vision-based crack detection and quantification methodologies for civil structures, Constr. Build. Mater. 356 (2022), 129238, https://doi.org/10.1016/j.conbuildmat.2022.129238.
[35] Z. Liu, X. Gu, J. Chen, D. Wang, Y. Chen, L.J. Wang, Automatic recognition of pavement cracks from combined GPR B-scan and C-scan images using multiscale feature fusion deep neural networks, Autom. Constr. 146 (2023), 104698, https://doi.org/10.1016/j.autcon.2022.104698.

[36] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv (2015) preprint arXiv:1511.07122. doi:arxiv.org/abs/1511.07122.
[37] W. Wang, W. Chen, Q. Qiu, L. Chen, B. Wu, B. Lin, X. He, W. Liu, CrossFormer++: a versatile vision transformer hinging on cross-scale attention, arXiv (2023), https://doi.org/10.48550/arXiv.2303.06908 preprint arXiv:2303.06908.
[38] W. Ye, S. Deng, J. Ren, X. Xu, K. Zhang, W. Du, Deep learning-based fast detection of apparent concrete crack in slab tracks with dilated convolution, Constr. Build. Mater. 329 (2022), 127157, https://doi.org/10.1016/j.conbuildmat.2022.127157.
[39] J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, arXiv (2016), https://doi.org/10.48550/arXiv.1607.06450 preprint arXiv:1607.06450.
[40] F. Chollet, Xception: deep learning with depthwise separable convolutions, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2017) 1251–1258, https://doi.org/10.48550/arXiv.1610.02357.
[41] D. Hendrycks, K. Gimpel, Gaussian error linear units (GELUs), arXiv (2016), https://doi.org/10.48550/arXiv.1606.08415 preprint arXiv:1606.08415.
[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958, doi:abs/10.5555/2627435.2670313.
[43] F. Guo, Y. Qian, J. Liu, H. Yu, Pavement crack detection based on transformer network, Autom. Constr. 145 (2023), 104646, https://doi.org/10.1016/j.autcon.2022.104646.
[44] L. Yang, S. Bai, Y. Liu, H. Yu, Multi-scale triple-attention network for pixelwise crack segmentation, Autom. Constr. 150 (2023), 104853, https://doi.org/10.1016/j.autcon.2023.104853.
[45] D. Acuna, H. Ling, A. Kar, S. Fidler, Efficient interactive annotation of segmentation datasets with polygon-rnn++, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2018) 859–868, https://doi.org/10.48550/arXiv.1803.09693.
[46] M. Contributors, OpenMMLab Semantic Segmentation Toolbox and Benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.
[47] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on imagenet classification, Proc. IEEE Int. Conf. Comp. Vis. (2015) 1026–1034, https://doi.org/10.48550/arXiv.1502.01852.
[48] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv (2014), https://doi.org/10.48550/arXiv.1412.6980 preprint arXiv:1412.6980.
[49] C. Cortes, M. Mohri, A. Rostamizadeh, L2 regularization for learning kernels, arXiv (2012), https://doi.org/10.48550/arXiv.1205.2653 preprint arXiv:1205.2653.
[50] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba, Scene parsing through ade20k dataset, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2017) 633–641, https://doi.org/10.1109/CVPR.2017.544.
[51] T. Chen, Z. Cai, X. Zhao, C. Chen, X. Liang, T. Zou, P. Wang, Pavement crack detection and recognition using the architecture of segNet, J. Ind. Inf. Integr. 18 (2020), 100144, https://doi.org/10.1016/j.jii.2020.100144.
[52] Z. Zhang, M. Sabuncu, Generalized cross entropy loss for training deep neural networks with noisy labels, Adv. Neural Inf. Proces. Syst. 31 (2018), https://doi.org/10.48550/arXiv.1805.07836.
[53] B. Chen, H. Zhang, G. Wang, J. Huo, Y. Li, L.J. Li, Automatic concrete infrastructure crack semantic segmentation using deep learning, Autom. Constr. 152 (2023), 104950, https://doi.org/10.1016/j.autcon.2023.104950.
[54] T.S. Tran, S.D. Nguyen, H.J. Lee, V.P.J.C. Tran, B. Materials, Advanced crack detection and segmentation on bridge decks using deep learning, Constr. Build. Mater. 400 (2023), 132839, https://doi.org/10.1016/j.conbuildmat.2023.132839.
[55] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 801–818, https://doi.org/10.48550/arXiv.1802.02611.
[56] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, medical image computing and computer-assisted intervention–MICCAI 2015, in: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, Springer, 2015, pp. 234–241, https://doi.org/10.48550/arXiv.1505.04597.
[57] M.-H. Guo, C.-Z. Lu, Q. Hou, Z. Liu, M.-M. Cheng, S.-M.J. Hu, Segnext: rethinking convolutional attention design for semantic segmentation, Adv. Neural Inf. Proces. Syst. 35 (2022) 1140–1156, https://doi.org/10.48550/arXiv.2209.08575.
[58] L. Zhu, X. Wang, Z. Ke, W. Zhang, R.W. Lau, BiFormer: vision transformer with bi-level routing attention, Proc. IEEE/CVF Conf. Comp. Vision Patt. Recognit. (2023) 10323–10333, https://doi.org/10.48550/arXiv.2303.08810.
[59] Ç.F. Özgenel, Concrete crack images for classification, Mendeley Data V2 (2019), https://doi.org/10.17632/5y9wdsg2zt.2.
[60] Z. Qi, Concrete cracking level, Mendeley Data V1 (2020), https://doi.org/10.17632/bs7rjwywfm.1.
[61] T.R.J. Dorafshan, SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks, Data Brief (2018), https://doi.org/10.1016/j.dib.2018.11.015.