Automation in Construction 139 (2022) 104275

Contents lists available at ScienceDirect

Automation in Construction

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

Automatic concrete crack segmentation model based on transformer

Wenjun Wang *, Chao Su

College of Water Conservancy and Hydropower Engineering, Hohai University, Nanjing 210098, China

ARTICLE INFO

Keywords:
Concrete crack
Pixel-wise segmentation
Visual transformer
Self-attention
Encoder-decoder

ABSTRACT

Routine visual inspection of concrete structures is essential to maintain safe conditions. Therefore, studies of concrete crack segmentation using deep learning methods have been extensively conducted in recent years. However, insufficient performance remains a major challenge in diverse field-inspection scenarios. In this study, a novel SegCrack model for pixel-level crack segmentation is therefore proposed using a hierarchically structured Transformer encoder to output multiscale features and a top-down pathway with lateral connections to progressively up-sample and fuse features from the deepest layer of the encoder. Furthermore, an online hard example mining strategy was adopted to strengthen the detection of hard samples and improve the model performance. The effect of dataset size on the segmentation performance was then investigated. The results indicated that SegCrack achieved a precision, recall, F1 score, and mean intersection over union of 96.66%, 95.46%, 96.05%, and 92.63%, respectively, using the test set.

1. Introduction

Cracking is a type of concrete damage that can cause catastrophic structural failure resulting in considerable economic losses if not inspected and repaired in a timely manner. Thus, it is important to conduct regular inspections to obtain accurate surface damage information and thereby assess the health of concrete structures. In the past few decades, expert-dependent visual inspection has been widely employed to do so, but this approach is typically labor-intensive, subjective, and even dangerous [1–3].

To address these issues, researchers have proposed several computer vision-based algorithms to develop defect detection systems that mimic on-site human inspections. Many machine-learning crack detection algorithms [4–7] that rely on feature engineering have been proposed accordingly. However, these algorithms were unable to operate effectively under real-world conditions owing to shadows, the inhomogeneity of distress, and other factors. Convolutional neural networks (CNNs) have therefore attracted the interest of computer vision researchers since AlexNet [8] won the 2012 ImageNet championship, as they do not require complex feature engineering during training, and rely on end-to-end learning methods to extract data features and achieve state-of-the-art performance in computer vision tasks [9].

Recently, researchers have begun to apply CNNs to concrete damage-detection tasks, and many promising results have been achieved. Notably, sufficient training of a CNN model requires a large quantity of data. To address the issue of insufficient training data, the Gaussian kernel and Brownian motion processes have been used to generate crack images to expand the training dataset [10]. A feature pyramid network was introduced in [11,12] to fuse the features extracted by an encoder and make pixel-level predictions of the crack path using consecutive convolutional layers. Critically, the pyramid module achieved this objective by reserving information from low-level features. A fully CNN was developed using a bridge database from Niigata Prefecture to detect delamination and rebar exposure that provided promising results for an automated concrete damage-detection system [13]. To address the problem of insufficient accuracy and generalization associated with traditional detection methods, a context-aware deep neural network [14] was proposed that fused the predicted image patches using cross-state and cross-space potential functions. Researchers [15,16] have also designed lightweight segmentation networks for real-time crack detection, such as SDDNet—which consists of standard convolution, densely connected separable convolution, and modified atrous spatial pyramid pooling—and exhibited good performance while significantly reducing parameters [15]. Furthermore, an efficient crack segmentation model was proposed in [16] that retains sufficient spatial information along the spatial path and obtains a sufficient receptive field via the context path. The impact of dataset size and model depth on detection performance was studied in [17], which found that model performance improved with a larger dataset, but did not increase linearly with

* Corresponding author.
E-mail address: wenjunwang@hhu.edu.cn (W. Wang).

https://doi.org/10.1016/j.autcon.2022.104275
Received 20 December 2021; Received in revised form 12 March 2022; Accepted 16 April 2022
Available online 23 April 2022

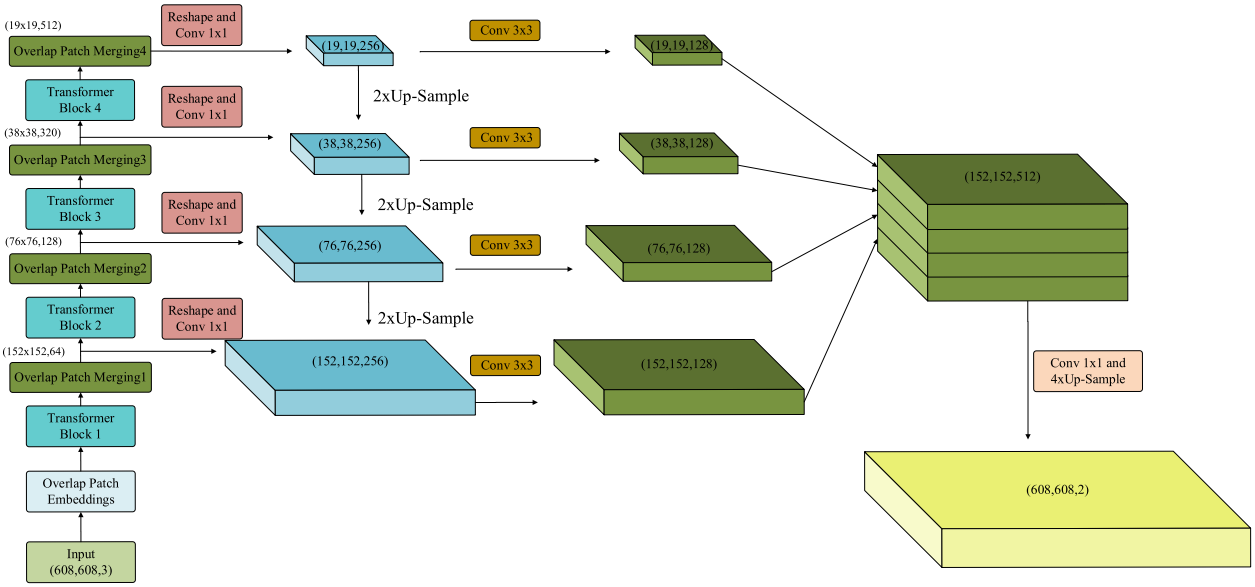

Fig. 1. Overall structure of SegCrack.

increasing model depth. In addition, Mei et al. [18] proposed a loss function that considered pixel connectivity and smoothed the prediction boundary in the crack detection model by reusing the multilevel features in the densely connected convolutional layers and transposing convolutional layers to fuse features. Other studies introduced attention mechanisms to improve prediction model performance. Attention-guided technology [19] was applied in SegNet to improve its performance in detecting and segmenting infrastructure damage. The attention gate module [20,21] was introduced into UNet to effectively extract crack features by focusing on critical areas and reconstructing semantic information, thereby improving its crack-segmentation performance. U-hierarchical dilated network [22] was developed using a multi-dilation and hierarchical feature module to extract different context sizes and multi-scale feature maps, achieving satisfactory segmentation results. A concrete crack segmentation network based on semi-supervised learning was proposed in [23] to utilize unlabeled data and thereby improve model performance. The detection and measurement of pavement crack was implemented in [24] through an ensemble network based on probabilistic fusion. The prediction accuracy of the ensemble model was improved by removing the pooling layers. A local weighting factor with a sensitivity map was proposed in [25] to remove the network bias and address the imbalance between crack and non-crack pixels in the training and test data.

Based on this literature review, it was determined that existing research has two major limitations: insufficient segmentation performance and a small number of training and test datasets. Previous studies used CNNs to extract information from crack images, as CNNs can expand the receptive field by stacking convolutional layers. In recent years, the Transformer architecture has achieved excellent results in computer vision applications compared to state-of-the-art CNNs. The Vision Transformer (ViT) [26] was the first to use a Transformer for computer vision tasks, but the number and dimensions of the tokens in ViT are fixed, preventing the model from capturing fine spatial details. The introduction of a pyramid structure into the Pyramid Vision Transformer (PVT) [27] enhanced the local continuity of features. Swin Transformer [28] restricts self-attention computation to non-overlapping local windows while allowing cross-window connections to enhance interactions.

In this study, a Transformer-based crack segmentation model named SegCrack was proposed that employs a hierarchical Transformer architecture for the encoder that outputs multi-scale feature information. This decoder uses a top-down pathway with lateral connections to gradually upsample from the deepest layer of the network. Furthermore, SegCrack does not require positional encoding as it introduces positional information into the feature maps using zero padding during convolution. Our contributions include the following:

- The proposal of a segmentation model (SegCrack) that adopts a hierarchical Transformer as the encoder and employs a top-down pathway with lateral connections as the decoder.
- The use of the Online Hard Example Mining (OHEM) strategy to improve model performance.
- A study of the influence of the training dataset size on the model performance.
- Comparison of the proposed method with four other CNNs, showing superior performance.

The remainder of this paper is organized as follows. Section 2 provides a detailed description of the SegCrack model developed in the study. Section 3 describes the concrete crack datasets used to train and demonstrate SegCrack. Details of the implementation of SegCrack are presented in Section 4, the experimental research results are analyzed and compared in Section 5, and finally the conclusions are presented in Section 6.

2. Methodology

A simple and effective concrete crack semantic segmentation model called SegCrack is proposed in this paper. As shown in Fig. 1, SegCrack takes 608 × 608 RGB images as input and consists of two main modules: a hierarchical Transformer encoder for generating high-resolution coarse features and low-resolution fine features, and a feature pyramid decoder for multi-scale feature fusion that extracts features on different scales from different layers of the network for prediction. Under this approach, a given input image is first divided into small patches using the overlap patch embedding module. These patches are then input to the hierarchical transformer block to obtain feature maps with resolutions equivalent to 1/4, 1/8, 1/16, and 1/32 that of the original image. The high-level features are then upsampled to ensure a top-down connection with the low-level features to predict the segmentation mask. This remainder of this section explains the composition of SegCrack in further detail.

Table 1
Detailed settings of overlap patch-merging modules.

Module	Kernel size	Stride	Padding
Overlap Patch-Merging 1	7	4	3
Overlap Patch-Merging 2	3	2	1
Overlap Patch-Merging 3	3	2	1
Overlap Patch-Merging 4	3	2	1

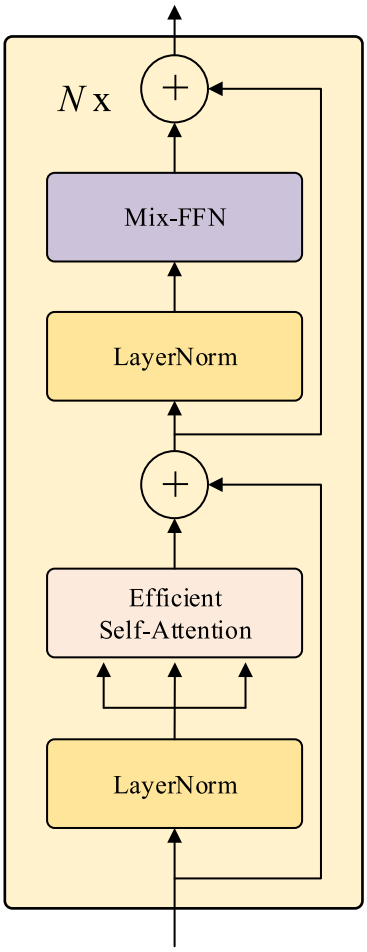

Fig. 2. Transformer block.

2.1. Overlap patch merging

Overlap patch merging is utilized to generate multilevel feature maps realized through 2D convolution. Feature maps with different resolutions are generated by changing the kernel size and stride. To ensure local continuity around each patch, they are designed to overlap. The segmentation model includes four overlapping patch-merging modules, the specific parameters of which are listed in Table 1. The feature map resolutions of the input image after the four overlapping patch-merging modules are (152,152), (76,76), (38,38), and (19,19).

2.2. Transformer block

As shown in Fig. 2, the transformer block is composed of a layer norm [29], efficient self-attention, and the Mix Feed-Forward Network (Mix-FFN). Batch normalization [30] is used to standardize the data distribution of the same batch to obtain the mean and variance, and is sensitive to the batch size. The layer normalization operation is similar to the transposing batch normalization: layer normalization is realized by calculating the normalized mean and variance of all weighted inputs of the neurons in a layer in a single training example. For example, when training a neural network with a batch size of 16, 16 means and variances will be obtained. Each mean and variance is standardized by all the channels of a single image, thereby preventing layer normalization from being affected by the batch size. After the input features pass through layer normalization, the self-attention of each feature is calculated and added to the input features using residual connections to prevent gradient vanishing and aid in training.

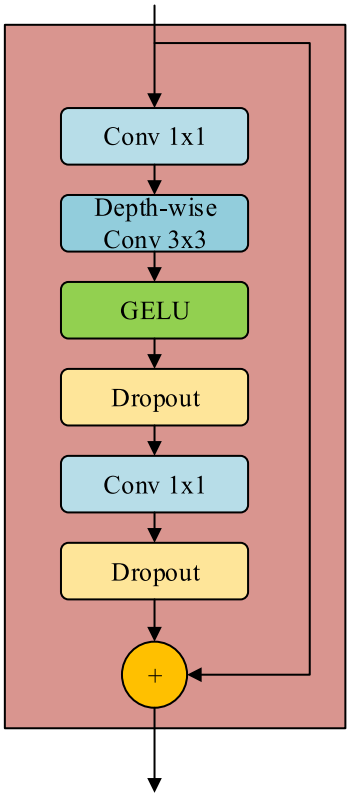

Fig. 3. Mix-FFN module.

2.2.1. Efficient self-attention

The attention mechanism calculates the correlation between the query and each key to obtain their weight coefficients. The attention value is obtained by calculating the sum of the products of each value and the corresponding weight coefficient. Self-attention refers to queries, keys, and values from the same location. The primary role of self-attention is to model the rich interactions between pixels. This approach can capture interactions between any two positions on the feature map regardless of the distance separating them, making the Transformer model highly flexible in modeling long-range dependencies in vision tasks.

Assuming an input query Q, the context information is stored in the form of key–value pairs (K, V). The calculation of attention is thus undertaken as follows [31]:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where Q, K, and V are matrices describing the queries, keys, and values, respectively, T is the transpose of a matrix, d_k is the dimension of K.

To reduce the number of calculations in the self-attention layer, a reduction ratio R is introduced to control the sequence length.

First, converting the shape of K from $(N \times C)$ to $\frac{N}{R} \times (C \bullet R)$:

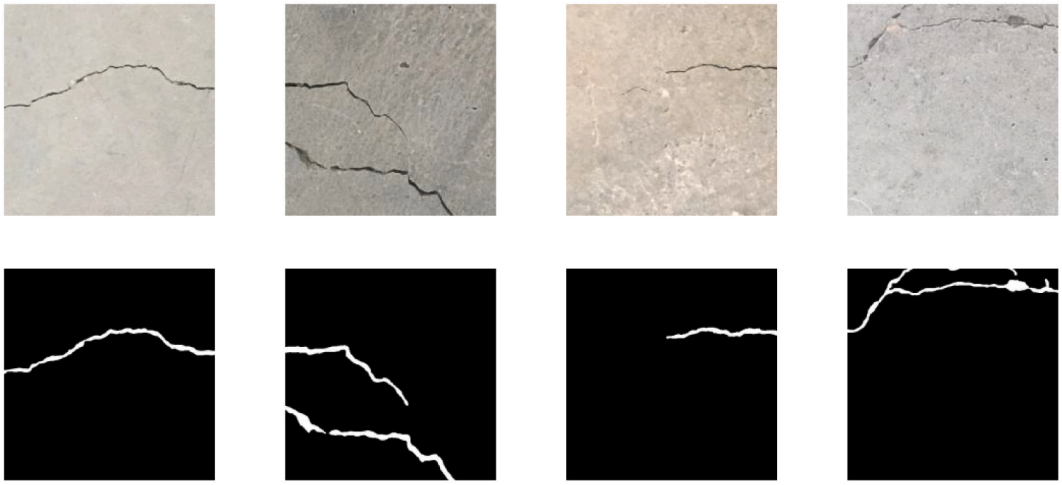

Fig. 4. Sample images with mask labeling.

$$\widehat{K} = Reshape\left(\frac{N}{R}, C, R\right)(K) \tag{2}$$

Then, converting the shape of $\widehat{K}$ from $\frac{N}{R} \times (C \bullet \mathrm{R})$ to $\frac{N}{R} \times C$:

$$K = Linear(C \bullet R, C)(\widehat{K}) \tag{3}$$

where N is the length of the sequence, R is a reduction ratio, C is the channel dimension, $(\bullet)$ is a linear layer.

The computational complexity is thus reduced from $O(N^2)$ to $O\left(\frac{N^2}{R}\right)$.

For Transformer Blocks 1, 2, 3, and 4, the value of R was set to 64, 16, 4, and 1, respectively.

Compared with conventional CNNs, the distance between elements when applying self-attention is shortened from the logarithmic path length to a constant path length, and the receptive field is changed from fixed-size perceptive to variable-sized perceptive.

Unlike single-attention pooling, efficient self-attention adopts a multi-head attention design. The queries, keys, and values are transformed using h independently learned linear projections. Then, the h projected queries, keys, and values are fed in parallel into the attention pooling. Finally, the h attention pooling outputs are concatenated and transformed by another linear projection to produce the final output.

2.2.2. Mix-FFN

Fig. 3 illustrates the Mix-FFN module, in which it can be seen that the number of input channels is expanded to four times the original number using 1×1 convolution in order to extract richer semantic information. Unlike other Transformer networks, positional embedding is not used in the proposed model; instead, a 3×3 convolution is used to transmit the position information. During the convolution operation, zero padding unintentionally introduces position information into the feature map, which ensures that when the test resolution is different from the training resolution, the accuracy will not be reduced owing to position encoding interpolation.

Subsequently, the nonlinear Gaussian Error Linear Unit (GELU) activation function [32] is employed. Nonlinearity is an important property of any neural network model. Random regularization, such as dropouts, must therefore be added to realize the generalization ability of the model. To do so, GELU introduces the idea of random regularity in activation, which represents a probabilistic description of the neuron input. The expression for the GELU [32] is as follows:

$$GELU(x) = x\Phi(x) \tag{4}$$

where x is the input and $\Phi(x)$ is the probability function of the normal distribution.

Finally, it is restored to the original channel number through 1×1 convolution and added to the input features.

2.3. Feature pyramid decoder

The feature pyramid decoder utilizes four different feature map output sizes during the forward propagation of the Transformer encoder to predict the mask. It employs a top-down structure with lateral connections to achieve multi-scale feature fusion. The top-down path starts from the deepest layer of the network, and the low-resolution fine features are bilinearly upsampled to fuse with the feature map of the same size. This process iterates until the final resolution map is generated; this map enables the full use of the position information in the low-level features to locate detailed information. The feature dimensions of all the feature maps are adjusted to a fixed value of $d = 256$ through 1×1 convolution. After fusion, the number of channels of the feature maps is further adjusted to 128 and then upsampled to 1/4 resolution of the original image for concatenation. Finally, the number of channels in the concatenated feature maps is adjusted, and the resolution of the original image is restored through bilinear interpolation.

3. Dataset

A publicly available dataset was applied to train the proposed SegCrack model for concrete crack segmentation [33]. This dataset contained 458 high-resolution images captured from various buildings located at Middle East Technical University with manual crack annotations. For each high-resolution RGB image, a binary mask indicating the position of the crack was hand-drawn using labeling software to serve as the ground truth. The large images were then cropped into smaller tiles of 608×608 pixels to establish a dataset for neural network training. Images that did not contain any cracks were removed to reduce the already large imbalance in cracked/uncracked pixel ratio. These images were further divided into a training set (1971 images), validation set (216 images), and test set (548 images), respectively used to fit the model parameters, adjust the hyperparameters of the model and make a preliminary assessment of the model's crack identification ability, and evaluate the generalization ability of the best model. Fig. 4 shows samples of the annotated images.

Essentially, a neural network model uses a series of linear and nonlinear functions to fit the output of the target data; the more samples employed to do so, the more accurate the results. Semantic segmentation training data require pixel-level manual labeling, which is a labor-intensive and time-consuming task, yet label quality is crucial for

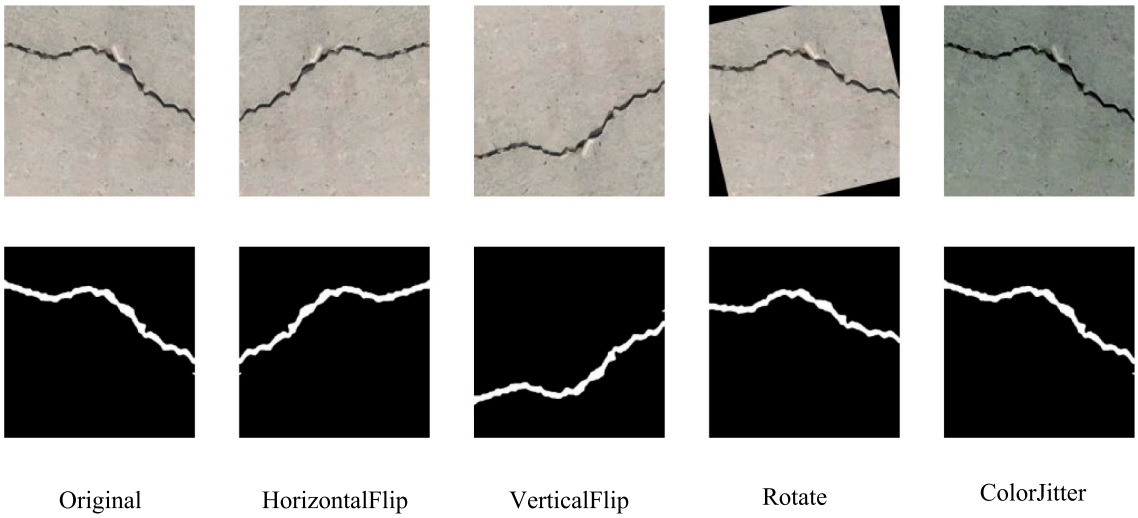

Fig. 5. Sample images with mask labeling after data augmentation.

Table 2
Performance comparison with and without OHEM strategy.

	Pr (%)	*Re* (%)	*F1_score* (%)	*mIoU* (%)
With OHEM	96.66	95.46	96.05	92.63
Without OHEM	96.58	94.10	95.30	91.34

Table 3
Test results when training with different numbers of crack images.

Number of crack images	*Pr* (%)	*Re* (%)	*F1_score* (%)	*mIoU* (%)
394	95.64	91.23	91.23	88.06
788	95.16	92.58	93.83	88.90
1183	95.92	93.49	94.66	90.27
1577	96.15	94.43	95.28	91.29
1971	96.66	95.46	96.05	92.63

ensuring model performance. In addition, successful training of deep learning models must ensure sufficient data diversity to prevent overfitting. These issues make it challenging to achieve accurate concrete damage detection. To overcome these challenges, online data augmentation was used to expand the dataset during the training process. The goal of data augmentation [34] is not simply to stack data, but to cover to the extent possible situations not addressed by the original data, but still possible in the real world. Therefore, after obtaining batch images, the RandomFlip, RandomRotate, and ColorJitter operations were performed. RandomFlip applies either a horizontal or vertical flip, RandomRotate rotates the image by a randomly selected angle, and ColorJitter randomly transforms the brightness, contrast, saturation, and hue of an image within a certain range to simulate real-world changes in conditions such as different lighting environments. Fig. 5 shows several sample images of cracks after data augmentation.

4. Implementation and loss function

4.1. Implementation

SegCrack was coded using the MMSegmentation [35] and trained on

Table 4
Test results of compared methods.

	Pr (%)	*Re* (%)	*F1_score* (%)	*mIoU* (%)
FCN	96.09	89.85	92.72	87.14
UPerNet	97.53	87.35	91.76	85.67
EMANet	96.93	90.00	93.16	87.83
BiSeNetv2	95.39	89.33	92.11	86.20
SegCrack	96.66	95.46	96.05	92.63

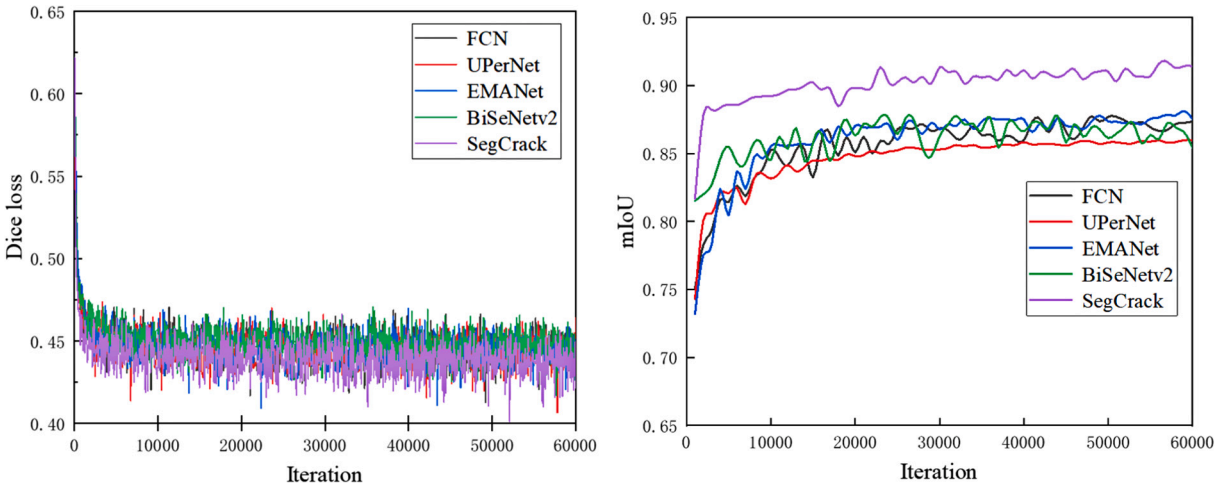

Fig. 6. Loss curves during training and *mIoU* curves during validation.

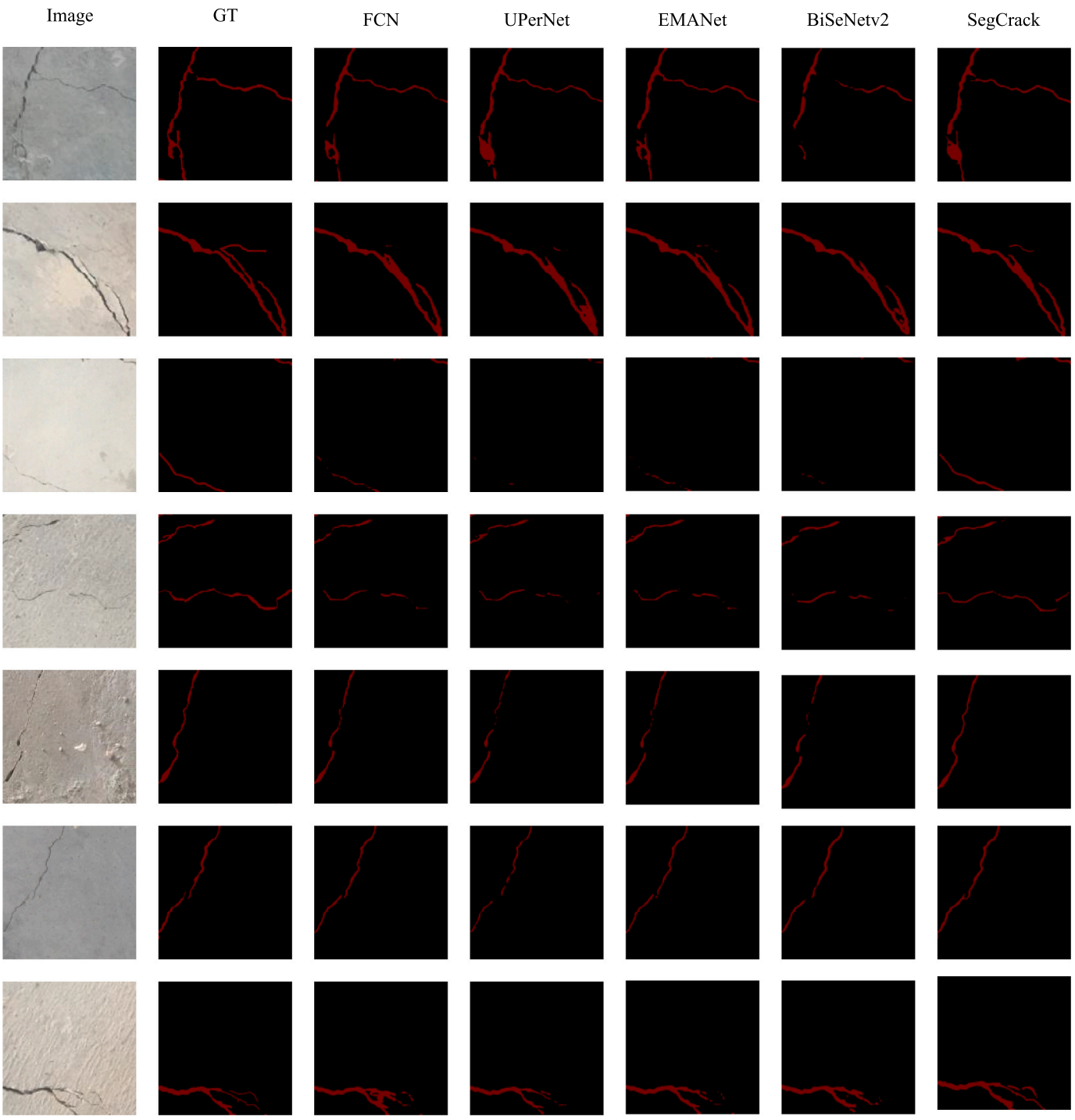

Fig. 7. Example results of different segmentation models.

Table 5
The comparison results of model efficiency.

	Parameters (M)	FLOPs (G)	Inference time (FPS)	*mIoU* (%)
FCN	49.48	279.17	13.90	87.14
UPerNet	85.39	361.50	11.52	85.67
EMANet	42.08	237.91	15.50	87.83
BiSeNetv2	14.76	17.35	105.54	86.20
SegCrack	28.44	48.30	21.81	92.63

a PC running Ubuntu with a GeForce GTX 1080Ti 11 GB GPU. The optimization method has always been a very important part of machine learning and represents the core algorithm of the learning process. Though the Adam [36] optimizer has been widely used since it was proposed, its convergence is not guaranteed. However, AdamW [37] uses the gradient calculated from the overall loss function to adjust the updating strategy and has achieved good performance in computer vision tasks. The AdamW optimizer with an initial learning rate of 2e-5 was therefore utilized in this study to train the model for 60,000 iterations on the concrete crack segmentation dataset. A "poly" learning rate schedule with default factor of 1.0 was used to dynamically adjust the

Table 6
Robustness test results of SegCrack.

		Severity					
		1	2	3	4	5	Mean
Noise	Gaussian	89.22	88.47	87.85	86.61	83.39	87.11
	Shot	88.90	88.34	87.69	85.42	82.71	86.61
	Speckle	89.02	88.57	87.62	86.60	84.79	87.32
Blur	Defocus	90.43	88.84	87.21	86.17	84.93	87.52
	Glass	90.42	89.47	84.02	83.92	83.05	86.18
	Zoom	76.09	72.52	71.41	69.35	68.04	71.48
Digital	Brightness	92.56	92.08	90.84	86.93	82.79	89.04
	Contrast	88.95	87.03	83.70	76.91	64.54	80.23
	Saturate	92.54	92.51	92.42	91.20	87.38	91.21

Table 7
Robustness test results of EMANet.

		Severity					
		1	2	3	4	5	Mean
Noise	Gaussian	75.82	67.52	54.27	48.15	47.47	58.65
	Shot	73.98	62.65	52.40	47.83	47.50	56.87
	Speckle	76.44	70.70	55.59	50.83	48.28	60.37
Blur	Defocus	87.20	86.49	84.88	83.03	81.01	84.52
	Glass	86.15	86.09	82.92	82.86	80.99	83.80
	Zoom	74.74	70.34	69.11	66.70	65.94	69.37
Digital	Brightness	87.08	85.08	84.02	80.22	72.61	81.80
	Contrast	76.09	66.24	52.30	47.49	47.48	57.92
	Saturate	87.70	87.53	86.37	79.51	51.82	78.59

learning rate during training, L2 regularization [38] was used to reduce model overfitting, and the weight decay was set to 0.01. The model used an image with a resolution of 608 × 608 pixels and a batch size of two as the input. The data augmentation procedure was as outlined in Section 3. The selection of the initial weights of the neural network parameters is critical and is related to the optimization efficiency and generalization ability of the network. Owing to the limited dataset size in this study, it was impossible to train the entire crack segmentation network from scratch. Therefore, transfer learning was applied to fine-tune SegCrack, where the encoder was initialized with pre-trained weights on the ImageNet dataset, and the decoder parameters were initialized with kaiming_uniform initialization [39].

4.2. Loss function

The Dice coefficient is a measure of overlap that has been widely used to assess segmentation performance. It has also been adapted to serve as a loss function, known as Dice Loss [40]. This study employed Dice Loss to compute the distance between the current and expected output of the algorithm, and is defined as [40]:

$$Loss_{dice}(y,\hat{y}) = 1 - \frac{2y\hat{y}+\epsilon}{y+\hat{y}+\epsilon} \tag{5}$$

where y is the label value, $\hat{y}$ is the predicted value, and ϵ is added in the numerator and denominator to ensure loss function stability by avoiding a "divide by zero" when $y = \hat{y} = 0$.

5. Results and discussion

5.1. Evaluation criteria

To measure the performance of the proposed segmentation model, it was necessary to provide a test set, use the model to identify each sample, and calculate the evaluation score based on the resulting identification. In these experiments, four commonly used metrics were selected to evaluate and compare the performances of the different models: precision (*Pr*), recall (*Re*), *F1_score*, and mean intersection over union (*mIoU*)—a standard measure of semantic segmentation tasks, respectively defined as:

$$Pr = \frac{TP}{TP+FP} \tag{6}$$

$$Re = \frac{TP}{TP+FN} \tag{7}$$

$$F1\_score = 2\bullet\frac{Pr\bullet Re}{Pr+Re} \tag{8}$$

$$mIoU = mean\left(\frac{TP}{TP+FP+FN}\right) \tag{9}$$

where true positive (*TP*) indicates that the real category of a pixel is a crack, and that it was correctly identified as a crack; a false positive (*FP*) means that the real category of a pixel is the background, and it was incorrectly identified as a crack; a false negative (*FN*) means that the real category of a pixel is a crack, and it was incorrectly identified as the background.

5.2. Experimental testing of SegCrack

This section describes the evaluation of the proposed SegCrack model, including the impact of the OHEM strategy [41] on model performance, the impact of training dataset size on model performance, a comparison of the performance of the proposed model with those of other models, and a robustness test and demonstration of model application. These results were obtained using the test set, which never participated in the training process.

5.2.1. Online hard example mining (OHEM)

The background and crack pixels in the training set images exhibited a sample imbalance phenomenon. Therefore, to conduct targeted training on hard examples, the OHEM [41] was adopted in this study as the concrete damage dataset contained a large number of easy examples and a small number of hard examples. Automatic selection of these hard examples can realize more efficient training. Images in each mini-batch contain hundreds of thousands of candidate pixels, and the OHEM was used to subsample the candidate pixels according to a distribution conducive to high loss. The OHEM uses an online selection method without setting the ratio of positive and negative samples. In this study, only pixels with a confidence score of less than 0.7 were be used for training. Simultaneously, a dataset of at least 100,000 pixels was maintained during training. Table 2 shows the impact of the OHEM strategy on model performance, in which it can be observed that the OHEM strategy increased the *F1_score* and *mIoU* by 0.75% and 1.29%, respectively. Although the performance improvement owing to the use of OHEM was not significant, this approach was introduced as a training trick in the crack segmentation task, and the test results indeed demonstrate that OHEM has a positive effect on improving model performance.

5.2.2. Influence of dataset size on model performance

Since AlexNet [8] won the 2012 ImageNet competition, CNNs have been made increasingly accurate by increasing their size. However, an increase in CNN size requires additional training data. To study the influence of dataset size on model performance, the training set was redivided to contain 20, 40, 60, 80, and 100% of the maximum number of training data points when training the proposed SegCrack model, while the validation and test sets were left unchanged. The results are presented in Table 3, in which it can be observed that the performance of the proposed SegCrack model gradually improved as the quantity of training set data increased. Compared with its worst performance, the best performance of SegCrack improved the *F1_score* and *mIoU* by 4.82%

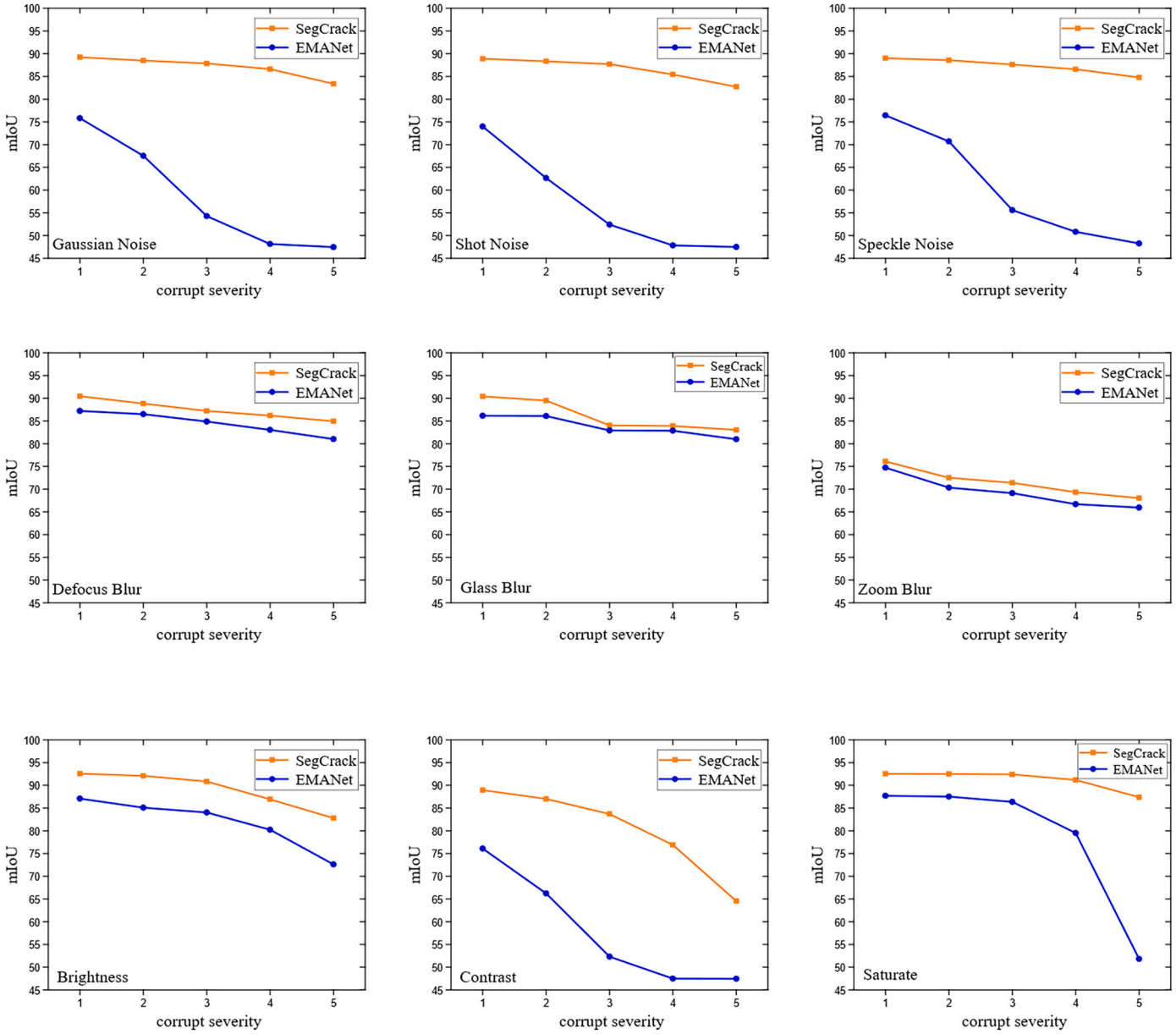

Fig. 8. Comparison of SegCrack and EMANet robustness.

and 4.57%, respectively, yet still achieved satisfactory results using only 20% of the training set, with an *F1_score* and *mIoU* of 91.23% and 88.06%, respectively. These results indicate that SegCrack can effectively generalize the learned features to the images in the test set, even under small-sample conditions. This ability to learn under small-sample conditions is particularly beneficial to scalability and practicality owing to the difficulty of acquiring large-scale datasets in civil engineering otherwise.

5.2.3. Comparative study

The superior crack segmentation performance of the proposed SegCrack model was demonstrated by comparison with recently developed networks that have shown state-of-the-art performance in the field of semantic segmentation: FCN [42], UPerNet [43], EMANet [44], and BiSeNetv2 [45]. To ensure a valid comparison, these networks were trained and tested on the same training, validation, and test data sets using the same data augmentation method. During training, Dice Loss was applied to monitor the parameter learning state, and during validation, the *mIoU* was utilized to quantify the model performance. Fig. 6 illustrates the loss curve during training and the *mIoU* curve during validation; Table 4 presents the quantitative results of the five models for the test set. It can be observed that the proposed SegCrack demonstrated superior performance in terms of *Re*, *F1_score*, and *mIoU*. Although SegCrack exhibited a 0.87% lower *Pr* than UPerNet, it exhibited a 8.11% higher *Re*, 4.29% higher *F1_score*, and 6.96% higher *mIoU*. Indeed, as UPerNet sacrifices recall for higher precision, it cannot achieve a good balance between them, which is disadvantageous for segmentation tasks. The proposed SegCrack model considered the importance of both precision and recall, and realized higher *F1_score* and *mIoU* values of 96.05% and 92.63%, respectively, surpassing the second best performance by a large margin. This demonstrates that SegCrack provides better robustness and generalization than the other state-of-the-art methods. This superior performance can be attributed to the high-efficiency hierarchically structured Transformer encoder, which outputs multi-scale features, and the multi-scale feature fusion decoder, which adds a light top-down pathway with lateral connections. The FCN, UPerNet, EMANet, and BiSeNetv2 all exhibited unbalanced precision and recall, indicating that these models experienced false-positive

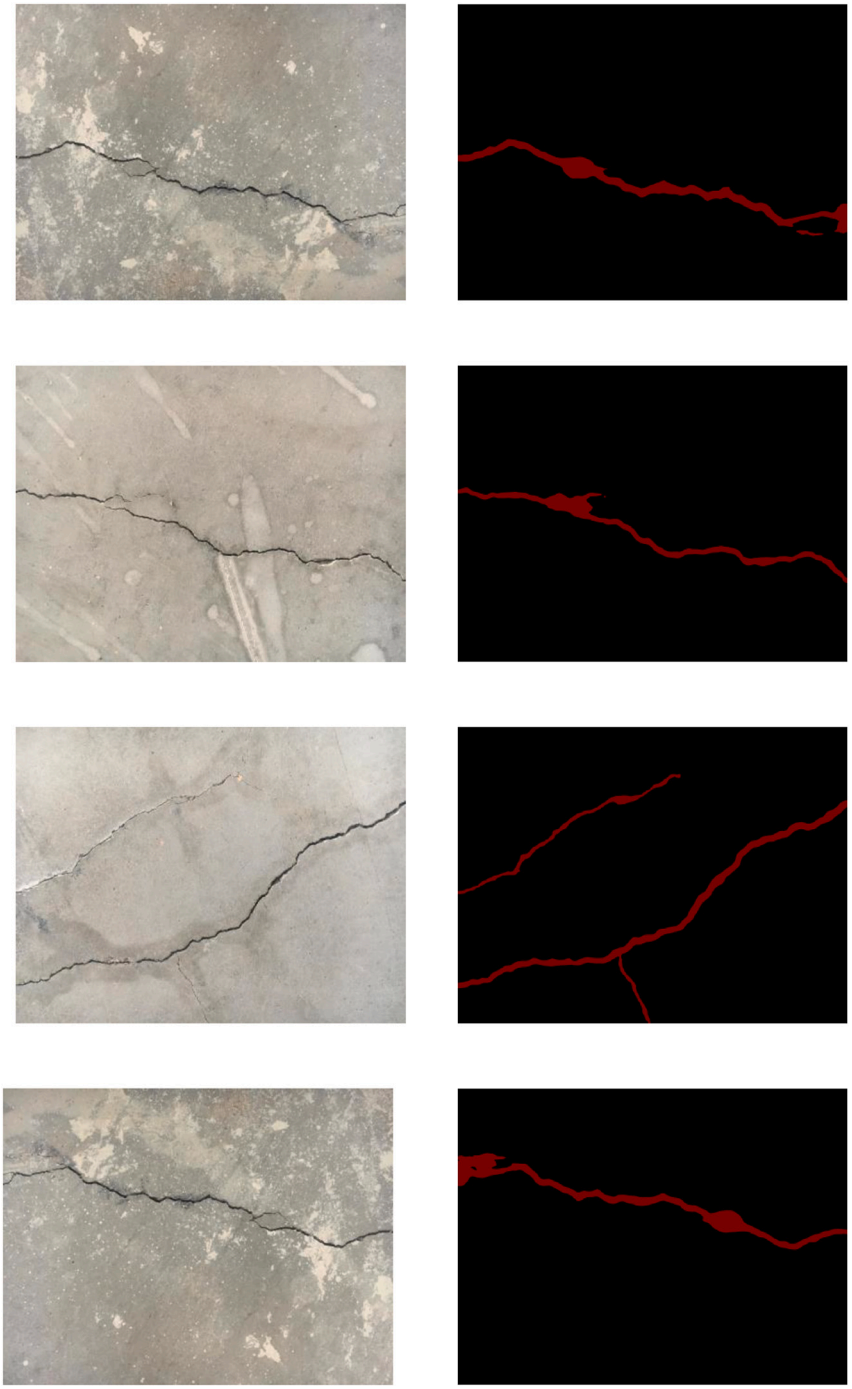

Fig. 9. Segmentation results for high-resolution images.

or false-negative detection issues. Fig. 7 displays sample images of the test results obtained using the different evaluated methods.

Table 5 presents a comparison of the efficiency of the five evaluated methods in terms of parameters, FLOPs, inference time, and *mIoU*. As shown in Table 5, SegCrack exhibited a 92.63% *mIoU* using only 28.44 M parameters and 48.30 G FLOPS, outperforming all other models except BiSeNetv2 in terms of parameters, flops, and inference time. For example, compared with EMANet, SegCrack exhibited a faster inference time (21.81 FPS) while maintaining a 4.8% better *mIoU* using only 68% of the parameters and 20% of the FLOPs. BiSeNetv2 was designed as a real-time semantic segmentation network, so it sacrifices accuracy to realize faster inferences, whereas SegCrack was designed to achieve accurate crack segmentation. Although SegCrack exhibited no advantage in efficiency compared with BiSeNetv2, its *mIoU* was 6.43% higher.

5.2.4. Robustness test and model application

Robustness is an important indicator of deep learning model performance and is typically applied to indicate the ability of a model to maintain identification accuracy when the input data changes. In this experiment, the robustness of SegCrack was extensively evaluated using a broad range of real-world image corruptions. Nine different image corruptions and perturbations in terms of blur, noise, and digital artifacts were evaluated in different severities increasing from 1 to 5. The robustness test results for SegCrack and EMANet are summarized in Tables 6 and 7, and shown in Fig. 8. As can be seen in Tables 6 and 7,

SegCrack exhibited significant advantages for most perturbation types compared to EMANet. The perturbation that had the greatest impact on EMANet was Gaussian noise with a severity of 5. Compared with clean data, the performances of SegCrack and EMANet as indicated by *mIoU* decreased by 9.24% and 40.36%, respectively, when Gaussian noise was introduced. As shown in Fig. 8, as the severity of corruption increased, EMANet exhibited considerable performance degradation, whereas the performance of SegCrack remained relatively stable and exhibited excellent robustness.

Note that owing to computer hardware limitations, high-resolution images must be cropped into many sub-images to train deep learning models. During the application phase, high-resolution images should be directly processed. Fig. 9 shows the segmentation results of the proposed model for the original high-resolution images. It can be seen that SegCrack was not disturbed by background noise and achieved satisfactory crack segmentation results.

6. Conclusion

A novel vision-based semantic concrete crack segmentation method, SegCrack, was developed in this study that adopts a hierarchical Transformer architecture to output multiscale features and uses a top-down pathway with lateral connections to output pixel-level identification of cracks. An OHEM strategy was used to alleviate the class imbalance problem. The following conclusions were drawn from this study:

1. The OHEM strategy improved model performance by subsampling the candidate pixels conducive to high loss.
2. SegCrack achieved satisfactory results (88.06% *mIoU*) when using only 20% of the training data.
3. The *F1_score* and *mIoU* achieved by SegCrack were 96.05% and 92.63%, respectively, outperforming previously proposed CNN-based methods.
4. The robustness test results show that the performance of SegCrack was stable against a broad range of real-world image corruptions.

Although SegCrack exhibited suitable performance in concrete crack segmentation, some research limitations remain to be addressed. First, the assessment of structural damage severity using the segmentation results was not investigated in this study. In addition, the design of SegCrack focused on the improvement of segmentation performance, resulting in relatively low efficiency; the utilization of model compression methods such as pruning and knowledge distillation to improve model efficiency while maintaining accuracy has not been investigated. In future work, we plan to use the segmentation results to provide long-term monitoring of structural conditions and performance evaluation. Advanced model compression methods will be utilized to improve the efficiency of SegCrack. In addition, new data sources such as satellite, ultrasonic, and impact-echo images will be applied to realize structural health monitoring tasks.

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.

Acknowledgments

We gratefully acknowledge the financial support of this research from the National Natural Science Foundation of China (Grant No. 51579089).

References

[1] K. Jang, N. Kim, Y.-K. An, Deep learning–based autonomous concrete crack evaluation through hybrid image scanning, Struct. Health Monit. 18 (5–6) (2019) 1722–1737, https://doi.org/10.1177/1475921718821719.
[2] J.K. Chow, Z. Su, J. Wu, Z. Li, P.S. Tan, K.-F. Liu, X. Mao, Y.-H. Wang, Artificial intelligence-empowered pipeline for image-based inspection of concrete structures, Autom. Constr. 120 (2020), 103372, https://doi.org/10.1016/j.autcon.2020.103372.
[3] Y. Jiang, D. Pang, C. Li, A deep learning approach for fast detection and classification of concrete damage, Autom. Constr. 128 (2021), 103785, https://doi.org/10.1016/j.autcon.2021.103785.
[4] L. Liu, G. Meng, Crack detection in supported beams based on neural network and support vector machine, International Symposium on Neural Networks, Springer (2005) 597–602, https://doi.org/10.1007/11427469_95.
[5] M. O'Byrne, F. Schoefs, B. Ghosh, V. Pakrashi, Texture analysis based damage detection of ageing infrastructural elements, Comput.-Aided Civil Infrastruct. Eng. 28 (3) (2013) 162–177, https://doi.org/10.1111/j.1467-8667.2012.00790.x.
[6] D. Lattanzi, G.R. Miller, Robust automated concrete damage detection algorithms for field applications, J. Comput. Civ. Eng. 28 (2) (2014) 253–262, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000257.
[7] P. Prasanna, K.J. Dana, N. Gucunski, B.B. Basily, H.M. La, R.S. Lim, H. Parvardeh, Automated crack detection on concrete bridges, IEEE Trans. Autom. Sci. Eng. 13 (2) (2014) 591–599, https://doi.org/10.1109/TASE.2014.2354314.
[8] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Proces. Syst. 25 (2012) 1097–1105, https://doi.org/10.1145/3065386.
[9] S. Albawi, T.A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, in: 2017 International Conference on Engineering and Technology (ICET), Ieee, 2017, pp. 1–6, https://doi.org/10.1109/ICEngTechnol.2017.8308186.
[10] D. Lee, J. Kim, D. Lee, Robust concrete crack detection using deep learning-based semantic segmentation, Int. J. Aeronaut. Space Sci. 20 (1) (2019) 287–299, https://doi.org/10.1007/s42405-018-0120-5.
[11] F. Ni, J. Zhang, Z. Chen, Pixel-level crack delineation in images with convolutional feature fusion, Struct. Control. Health Monit. 26 (1) (2019), e2286, https://doi.org/10.1002/stc.2286.
[12] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, H. Ling, Feature pyramid and hierarchical boosting network for pavement crack detection, IEEE Trans. Intell. Transp. Syst. 21 (4) (2019) 1525–1535, https://doi.org/10.1109/TITS.2019.2910595.
[13] J.J. Rubio, T. Kashiwa, T. Laiteerapong, W. Deng, K. Nagai, S. Escalera, K. Nakayama, Y. Matsuo, H. Prendinger, Multi-class structural damage segmentation using fully convolutional networks, Comput. Ind. 112 (2019), 103121, https://doi.org/10.1016/j.compind.2019.08.002.
[14] X. Zhang, D. Rajan, B. Story, Concrete crack detection using context-aware deep semantic segmentation network, Comput.-Aided Civil Infrastruct. Eng. 34 (11) (2019) 951–971, https://doi.org/10.1111/mice.12477.
[15] W. Choi, Y.-J. Cha, SDDNet: real-time crack segmentation, IEEE Trans. Ind. Electron. 67 (9) (2019) 8016–8025, https://doi.org/10.1109/TIE.2019.2945265.
[16] W. Wang, C. Su, Deep learning-based real-time crack segmentation for pavement images, KSCE J. Civ. Eng. 25 (12) (2021) 4495–4506, https://doi.org/10.1007/s12205-021-0474-2.
[17] L. Zhang, J. Shen, B. Zhu, A research on an improved Unet-based concrete crack detection algorithm, Struct. Health Monit. 20 (4) (2021) 1864–1879, https://doi.org/10.1177/1475921720940068.
[18] Q. Mei, M. Gül, M.R. Azim, Densely connected deep neural network considering connectivity of pixels for automatic crack detection, Autom. Constr. 110 (2020), 103018, https://doi.org/10.1016/j.autcon.2019.103018.
[19] E. Karaaslan, U. Bagci, F.N. Catbas, Attention-guided analysis of infrastructure damage with semi-supervised deep learning, Autom. Constr. 125 (2021), 103634, https://doi.org/10.1016/j.autcon.2021.103634.
[20] J. König, M.D. Jenkins, P. Barrie, M. Mannion, G. Morison, A convolutional neural network for pavement surface crack segmentation using residual connections and attention gating, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pp. 1460–1464, https://doi.org/10.1109/ICIP.2019.8803060.
[21] X. Cui, Q. Wang, J. Dai, Y. Xue, Y. Duan, Intelligent crack detection based on attention mechanism in convolution neural network, Adv. Struct. Eng. (2021) 1859–1868, https://doi.org/10.1177/1369433220986638.
[22] Z. Fan, C. Li, Y. Chen, J. Wei, G. Loprencipe, X. Chen, P. Di Mascio, Automatic crack detection on road pavements using encoder-decoder architecture, Materials 13 (13) (2020) 2960, https://doi.org/10.3390/ma13132960.
[23] W. Wang, C. Su, Semi-supervised semantic segmentation network for surface crack detection, Autom. Constr. 128 (2021), 103786, https://doi.org/10.1016/j.autcon.2021.103786.
[24] Z. Fan, C. Li, Y. Chen, P.D. Mascio, X. Chen, G. Zhu, G. Loprencipe, Ensemble of deep convolutional neural networks for automatic pavement crack detection and measurement, Coatings 10 (2) (2020) 152, https://doi.org/10.3390/coatings10020152.
[25] R. Ali, J.H. Chuah, M.S.A. Talip, N. Mokhtar, M.A. Shoaib, Automatic pixel-level crack segmentation in images using fully convolutional neural network based on residual blocks and pixel local weights, Eng. Appl. Artif. Intell. 104 (2021), 104391, https://doi.org/10.1016/j.engappai.2021.104391.
[26] 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 preprint, arXiv:2010.11929, https://arxiv.org/abs/2010.11929, 2020.
[27] 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, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 568–578. https://arxiv.org/abs/2102.12122.
[28] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, arXiv preprint, arXiv:2103.14030, https://arxiv.org/abs/2103.14030, 2021.
[29] J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, arXiv preprint, arXiv:1607.06450, https://arxiv.org/abs/1607.06450, 2016.
[30] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: Int. Conf. Machine Learn. PMLR, 2015, pp. 448–456, https://arxiv.org/abs/1502.03167.
[31] 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://arxiv.org/abs/1706.03762.
[32] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), arXiv preprint, arXiv:1606.08415, https://arxiv.org/abs/1606.08415, 2016.
[33] Çağlar Fırat Özgenel, Concrete crack segmentation dataset, Mendeley Data 1 (2019), https://doi.org/10.17632/jwsn7tfbrp.1.
[34] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 1–48, https://doi.org/10.1186/s40537-019-0197-0.
[35] M. Contributors, MMSegmentation: Openmmlab Semantic Segmentation Toolbox and Benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.
[36] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint, arXiv:1412.6980, https://arxiv.org/abs/1412.6980, 2014.
[37] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv preprint, arXiv:1711.05101, https://arxiv.org/abs/1711.05101, 2017.
[38] C. Cortes, M. Mohri, A. Rostamizadeh, L2 regularization for learning kernels, arXiv preprint, arXiv:1205.2653, https://arxiv.org/abs/1205.2653, 2012.
[39] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034, https://doi.org/10.1109/ICCV.2015.123.
[40] C.H. Sudre, W. Li, T. Vercauteren, S. Ourselin, M. Jorge Cardoso, Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations, Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Springer, 2017, pp. 240–248, https://doi.org/10.1007/978-3-319-67558-9_28.
[41] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 761–769. https://arxiv.org/abs/1604.03540.
[42] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440. https://arxiv.org/abs/1411.4038.
[43] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, J. Sun, Unified perceptual parsing for scene understanding, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 418–434, https://doi.org/10.1007/978-3-030-01228-1_26.
[44] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, H. Liu, Expectation-maximization attention networks for semantic segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9167–9176. https://arxiv.org/abs/1907.13426.
[45] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, N. Sang, Bisenet v2: bilateral network with guided aggregation for real-time semantic segmentation, Int. J. Comput. Vis. (2021) 1–18, https://doi.org/10.1007/s11263-021-01515-2.