CrackCLF: Automatic Pavement Crack Detection Based on Closed-Loop Feedback

Chong Li, Zhun Fan, *Senior Member, IEEE*, Ying Chen, Huibiao Lin, Laura Moretti, Giuseppe Loprencipe, Weihua Sheng, *Senior Member, IEEE*, and Kelvin C. P. Wang

***Abstract*— Automatic pavement crack detection is an important task to ensure the functional performances of pavements during their service life. Inspired by deep learning (DL), the encoder-decoder framework is a powerful tool for crack detection. However, these models are usually open-loop (OL) systems that tend to treat thin cracks as the background. Meanwhile, these models can not automatically correct errors in the prediction, nor can it adapt to the changes of the environment to automatically extract and detect thin cracks. To tackle this problem, we embed closed-loop feedback (CLF) into the neural network so that the model could learn to correct errors on its own, based on generative adversarial networks (GAN). The resulting model is called CrackCLF and includes the front and back ends, i.e. segmentation and adversarial network. The front end with U-shape framework is employed to generate crack maps, and the back end with a multi-scale loss function is used to correct higher-order inconsistencies between labels and crack maps (generated by the front end) to address open-loop system issues. Empirical results show that the proposed CrackCLF outperforms others methods on three public datasets. Moreover, the proposed CLF can be defined as a plug and play module, which can be embedded into different neural network models to improve their performances.**

***Index Terms*— Automatic pavement crack detection, generative adversarial network, encoder-decoder, deep learning, closed-loop feedback.**

Manuscript received 8 May 2023; revised 26 September 2023; accepted 1 November 2023. This work was supported in part by the Science and Technology Planning Project of Guangdong Province of China under Grant 180917144960530, in part by the Project of Educational Commission of Guangdong Province of China under Grant 2017KZDXM032, in part by the Project of Robot Automatic Design Platform Combining Multi-Objective Evolutionary Computation and Deep Neural Network under Grant 2019A050519008, and in part by the State Key Laboratory of Digital Manufacturing Equipment and Technology under Grant DMETKF2019020. The Associate Editor for this article was D. F. Wolf. *(Corresponding authors: Zhun Fan; Giuseppe Loprencipe.)*

Chong Li, Zhun Fan, Ying Chen, and Huibiao Lin are with the Key Laboratory of Digital Signal and Image Processing of Guangdong Province, College of Engineering, Shantou University, Shantou 515063, China (e-mail: 15cli@stu.edu.cn; zfan@stu.edu.cn; 1316957264@qq.com; 13hblin@stu.edu.cn).

Laura Moretti and Giuseppe Loprencipe are with the Department of Civil, Construction and Environmental Engineering, Sapienza University of Rome, 00184 Rome, Italy (e-mail: laura.moretti@uniroma1.it; giuseppe.loprencipe@uniroma1.it).

Weihua Sheng is with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: weihua.sheng@okstate.edu).

Kelvin C. P. Wang is with the School of Civil and Environmental Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: kcpwang@gmail.com).

Digital Object Identifier 10.1109/TITS.2023.3332995

I. Introduction

PAVEMENT crack is a common road distress, mainly caused by temperature, materials aging, fatigue, and traffic loads [1]. They often start from the bottom of the upper layers and propagate to surfaces [2]. These discontinuities affect the structures and functional performances of the pavement and shorten its service life, causing discomfort for road users and costs for road managers [3]. Moreover, there are potential threats to road safety with social costs [4]: cracks may be penetrated by rain, forming potholes, which can cause accidents [5]. Crack detection is a necessary step for road management to repair them, prevent their diffusion, and maintain structural and functional conditions of the pavement [6].

In the past few decades, many structural health monitoring research were reported [7], [8]. The development and application of intelligent robotic systems for defect detection in civil infrastructure is advancing rapidly. Oh et al. proposed a bridge detection system, including a specially designed car, a robot mechanism, and a machine vision system [9]. Lim et al. [10] proposed a crack detection system that uses a camera to collect images. Laplacian of Gaussian was employed to inspect crack information, while camera calibration and robot localization were applied to obtain a global crack map. Zalama et al. [11] proposed a Gabor filter to detect crack types. Prasanna et al. [12] presented spatially tuned robust multi-feature (STRUM) classifier for crack detection.

In early years, crack detection methods are mainly based on traditional image processing algorithms, which are divided into three steps: preprocessing, preliminary detection, and refinement of cracks. Kaul et al. [13] applied Minimal Path Selection algorithm without any prior knowledge to detect cracks within gray pavement images, which can extract the continuous cracks. Lim et al. [10] used edge detection method for crack detection based on robotic crack inspection system. However, traditional methods for crack detection are time-consuming and expensive.

In recent years, automatic methods were applied to accelerate crack detection and reduce costs for road managers and users. With the development of computer vision, deep learning has entered the field of vision of researchers. Convolutional neural networks (CNNs) have achieved state-of-the-art performance in various computer vision applications.

Encoder-decoder framework with an U-shape is a typical backbone for segmentation tasks [4]. Fan et al. [14] proposed a U-HDN method based on encoder-decode architecture,

including the proposed Multi-Dilation Module and Hierarchical Feature Module. These modules can obtain rich global context information and integrate multi-scale feature information, which can improve crack detection accuracy. Liu et al. [15] proposed the DeepCrack for crack detection, which adopted the VGG16 neural network and hierarchical feature learning to improve neural network performance. However, it may fail to extract very thin cracks with complex topology.

Although many methods adopt the encoder-decoder framework to detect cracks and obtain satisfactory results [4,24,27,35,36], these models are usually open-loop (OL) systems that fail to detect the thin cracks, and tend to treat them as the background with a good loss.

In 2014, Goodfellow et al. [16] proposed the generative adversarial networks (GAN), which is widely used in machine learning and image generation tasks. GAN consists of two parts: a generative and an adversarial network. The generative one is used to generate fake images, and the latter distinguishes between real and generated results. After an alternative training, the generative network produces an image that resembles a real image, which can hardly be distinguished by the adversarial network. Inspired by this method, Zhang et al. [17] proposed CrackGAN model to synthesise truth crack image, which uses crack patch supervised GAN and U-shape architecture to detect cracks. However, this method ignores the global feature of crack image, and could fail to detect complex road textures and obtain a relatively low accuracy for CFD dataset. Zhou et al. proposed Unet++ [18] to address the issue of the global feature extraction of crack image using multiple skip connection operation based on U-net architecture. Although Unet++ indirectly integrates the features of different receptive fields, it only integrates the information of the next layer, and the information of the upper layer is not integrated. As a result, the granularity of its decoder part is still not fine enough, resulting in the loss of edge and position information in the segmentation results. Gao et al. [19] modified the U-net with a different way of cross-layer concatenate ways and combined the discriminative network to perform crack segmentation. However, the output of the discriminative network adopts the two-class networks, which may simply divide the output patch image into a fake or real image, ignore the relationship between pixels, and cannot extract thin cracks with complex topology.

To tackle above problems, we embed closed-loop feedback (CLF) into the neural network so that the model could learn to correct errors on its own, which is called CrackCLF, including the front and back ends (i.e., segmentation and adversarial network, respectively). The proposed CrackCLF can perform end-to-end training based on generative adversarial networks (GAN) (Figure 1).

The proposed segmentation network (i.e., the front end) adopts an encoder-decoder architecture to perform crack segmentation (the dashed box on the left in Figure 1). In addition, we propose an upsampling-convolutional block attention module (UCBAM) to replace the convolution operation based on the decoder part, which can fuse features from the encoder and decoder parts. An innovative convolutional block attention module+ (CBAM+) being a lightweight model is embedded into UCBAM, which can reduce the number of parameters and calculations, and accelerate neural network training. Then, we designed the global attention pooling (GAP) based on CBAM+, which can assign different weights for the feature channel to focus on the crack information. We developed a hierarchical feature learning method to perform deep supervision of the crack output for accelerating the convergence of neural network.

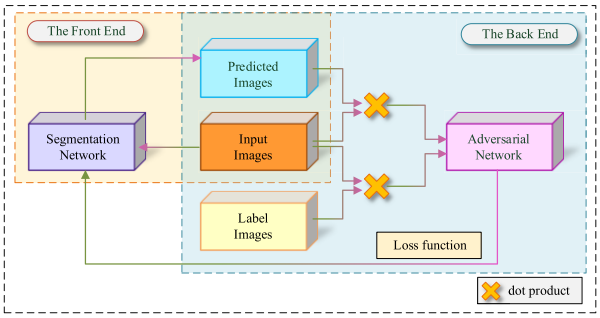

Fig. 1. The proposed CrackCLF framework. The CrackCLF architecture contains two parts: Segmentation (the front end) and Adversarial network (the back end).

The adversarial network (the back end) is employed to enforce the segmentation and adversarial network to learn global, local and thin crack information, which can capture a range of spatial relationships between pixels (the dashed box on the right in Figure 1). Meanwhile, it can correct higher-order inconsistencies between labels and crack maps (generated by the front end) to address open-loop system issues. In the training process of CrackCLF, the segmentation and adversarial networks are trained in an alternating manner. The contributions of CrackCLF are:

1) A new crack detection framework, called CrackCLF, was proposed with a closed loop feedback, in which the front end is employed to segment crack, and the back end is used to distinguish images between the generated ones by the front end and the real images. To the best of our knowledge, this is the first time that a closed-loop system is integrated with a mixed domain attention to perform crack detection task.
2) An upsampling-convolutional block attention module (UCBAM) is proposed to construct the decoder part in the proposed segmentation network (the front end), which can reduce the number of parameters and calculations costs, accelerating neural network training speed. In particular, a global attention pooling (GAP) is designed and embedded into the proposed CBAM+, which is part of UCBAM and can assign different weights for the feature channel to focus on the crack information.
3) A hierarchical feature learning module is carefully designed and embedded into the segmentation network to perform deep supervision of the crack output for accelerating convergence of the neural network.
4) An adversarial network (the back end) is proposed to perform closed-loop feedback to capture the difference between the predicted cracks and the ground truth, which

can better learn features of global and local cracks to improve the performance of segmentation network.

This study is organized as follows: In section II, we review the related works on crack detection. We describe the details of CrackCLF, including the segmentation and adversarial networks in Section III. Then, we perform comprehensive experiments to demonstrate the performance of CrackCLF in Section IV. Finally, Sections V is the conclusion of our study.

II. Related Works

A. Traditional Methods

In 2006, Subirats et al. [20] adopted a wavelet transform to detect cracks, based on different frequency sub-bands and amplitudes, which cannot be adapted to detect various types of cracks. Other researchers employed the threshold method to detect pavement cracks from images [21], [22], followed by morphological operations to refine the cracks. Oliveira and Correia [23] proposed CrackIT toolbox with threshold and edge detection methods et al., which is convenient for workers to detect crack images [24]. However, edge detection, morphology and thresholding methods are sensitive to the background noises, such as oil stains, shadows, and leaves, which reduce the accuracy of crack detection. Minimal path-based methods consider brightness and connectivity to reduce noises and improve the accuracy of the continuous crack. Kaul et al. [13] adopted the minimal path selection (MPS) method without any prior knowledge, which can extract the continuous cracks. However, it fails to detect discontinuous cracks. In summary, the traditional methods normally need to adopt several steps (such as, eliminating noises and adjusting hyperparameters et al.) to detect cracks. Therefore, these methods are too computationally intensive for practical applications.

B. Artificial Intelligence Methods

To address the above issues, some researchers have proposed the use of artificial intelligence [25], [26] to analyze cracks in road pavements. In particular, machine learning (ML) and deep learning (DL) have been proposed to perform semantic segmentation tasks [27]. Shi et al. [28] proposed a random structured forests method with the designed feature descriptors and public pavement dataset (CFD) to help road managers to analyze and evaluate the pavement surfaces. However, this method heavily relied on the selection of feature descriptors cannot be generalized for different pavement types. In general, the ML method cannot deal with and represent the features of the different pavement types for crack detection tasks. In recent years, DL is widely used to perform crack detection tasks.

Zhang et al. [29] used crack patch images to train a convolutional neural network (CNN) to detect cracks on road pavements. However, the patch-based methods are usually very time-consuming and ignore the global crack information, resulting in relatively low accuracy. In CrackNet, Zhang et al. [30], [31] adopted a CNN without pooling layers to avoid crack information loss, improve the pixel-perfect accuracy, and achieve a higher speed in crack detection. Fan et al. developed a structured prediction and an ensemble network method with the patch-based to extract cracks [32] and make measurements [33]. However, these methods may ignore the global crack feature and can fail to extract complex and thin cracks. Meanwhile when these methods are used to train large-scale datasets for crack detection, they obtain a lower accuracy with smaller convolution layers.

Encoder-decoder framework with an U-shape is a typical backbone for segmentation tasks [4]. Fan et al. proposed a U-HDN [14] method based on encoder-decode architecture, including the proposed Multi-Dilation Module and Hierarchical Feature Module. These modules can obtain rich global context information and integrate multi-scale feature information, which can improve crack detection accuracy. Qu et al. [34] proposed a new method based on deeply supervised convolutional neural network for pavement crack detection with Multi-Scale Features Fusion, which adopted U framework and hierarchical feature learning to improve neural network performance. Guo et al. [35] proposed BARNet method, which includes three modules: Edge Adaptation Module, Base Predictor Module and Refinement Module. Knig et al. [36] proposed using the Weakly-supervised method to segmentation crack image, which can obtain a better accuracy than others. Sun et al. [37] proposed a DMA (DeepLab With Multi-Scale Attention) method based on DeepLab, which integrates the ASPP(Atrous Spatial Pyramid Pooling) and attention mechanism. Zhou et al. [38] proposed an Enhanced Convolution and Dynamic Feature Fusion (ECDFFNet) method, which can improve its overall performance by focusing on the local details.

III. Methods

According to the literature in this study, the segmentation network is employed to detect cracks, and the adversarial one is used to perform feedback operation and enforce the segmentation to learn global, local and thin crack information.

A. Segmentation Network (the Front End)

The encoder-decoder architecture is used in the segmentation network, which consists of three parts: an encoder part, a decoder part, and a hierarchical feature learning module.

(1) Encoder Part:

In the encoder part, each encoder structure consists of two 3×3 convolutional layers and activation function (ReLU) [39]. Meanwhile, the pad operation is embedded into the 3×3 convolution operation to maintain the original resolution. Then, a pooling layer is adopted to downsampling operation to reduce the image size. After these operations, the image size is halved, and the number of channels is doubled, such as 64, 128, 256, 512, and 1024 (Figure 2).

(2) Decoder Parts:

The decoder part is composed of 4 successive UCBAM modules to restore the image size. Finally, the number of channels is gradually reduced to 1 (Figure 2).

1) The Proposed UCBAM: The resolution of the feature maps was reduced after passing through many downsampling operations in the final part of the encoder. The feature maps

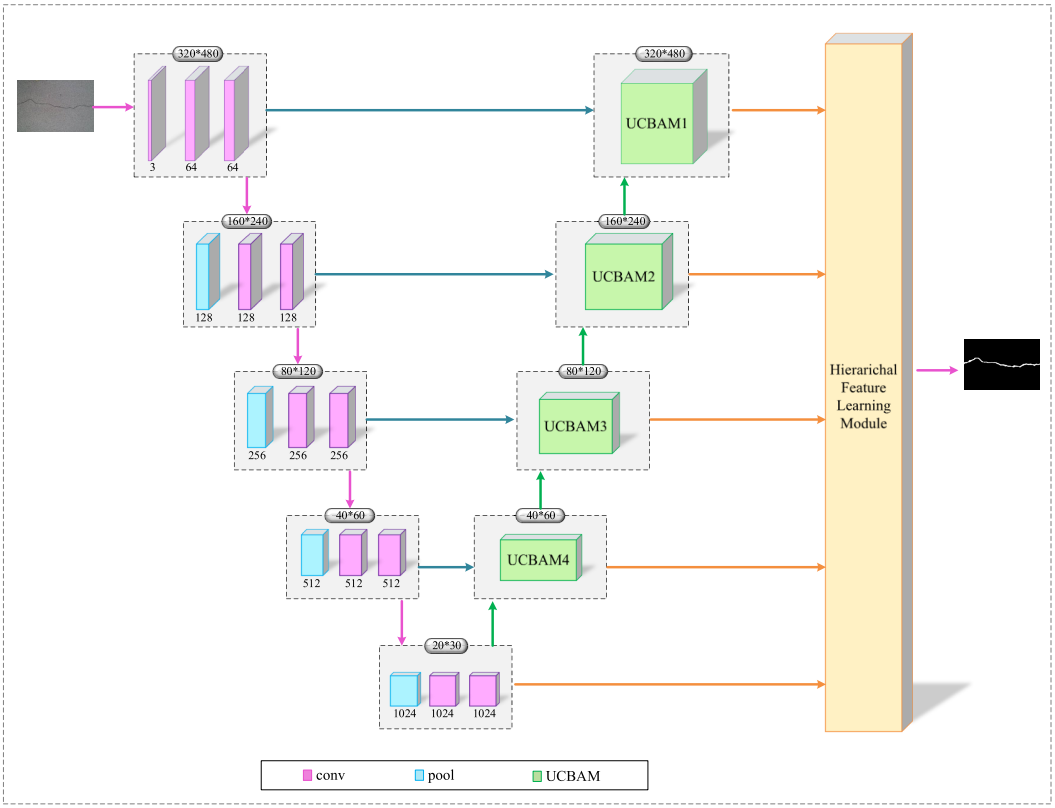

Fig. 2. The proposed Segmentation Network framework (the front end). The Segmentation Network contains three parts: encoder part, decoder part and hierarchical feature learning module.

in the encoder part have more information and are blurred and discarded after the downsampling operation. If we extract the crack image with a simple upsampling operation from the encoder part, the extracted crack results can be rough. To perform crack segmentation, the size of the final features may be restored to the original size. The function of the decoder is to restore the size of the features using transposed convolution operation. To fuse and incorporate more information, the skip connection from the encoder features is used to connect the decoder part. However, if these two different feature maps are simply added, the different contributions of edge and texture cannot be presented in the segmentation process. Hence, to overcome this issue, restore the image size, and highlight the important semantic information, UCBAM is proposed to perform channel and spatial attention mapping (Figure 3).

The inputs are the features from the encoder part with a skip connection along with the previous decoder part, and the output features are optimized after weight fusion. UCBAM first restores the size of the images from the previous decoder part using transposed convolution (light blue cuboid in Figure 3). Then, it adds the features (yellow cuboid in Figure 3) from the encoder part to generate the fused feature maps (pink cuboid in Figure 3), which can obtain and highlight the important semantic information, such as edges and textures.

In Figure 3 the yellow cuboid represents the feature maps from the encoder part by skip connection. The light blue cuboid is used to restore the image size that is from the previous decoder part after transpose convolution. Then, these two types of features perform an addition operation to fuse feature maps. Finally, the fused feature maps (pink cuboid) are input into the proposed CBAM+ module to obtain weighted features. Subsequently, the proposed CBAM+ based on CBAM [40] is employed to use the fused features to learn the weights of different features and suppress unnecessary features. UCBAM can adaptively assign different weights to the channels which can learn and highlight the different crack semantic information.

Specifically, the feature maps-based decoder part performs transposed convolution operations to restore the resolution and reduce the number of channels. The modified features are then concatenate to features from encoder to generate fused features. Finally, the fused features are given as input to the proposed CBAM+ to generate weight feature maps. The proposed UCBAM is able to reduce parameters number and calculations costs and accelerate the neural network training speed.

2) The Proposed CBAM+: The attention mechanism is widely embedded into the neural network to enhance the performance of models, reduce training time, and make the neural network lightweight. In CBAM [40], channel and spatial attention are employed to focus on significant features and ignore redundant features. This combination is superior to channel attention. Meanwhile, the CBAM is only able to obtain local features in the convolution process, which cannot capture and extract long-range dependency.

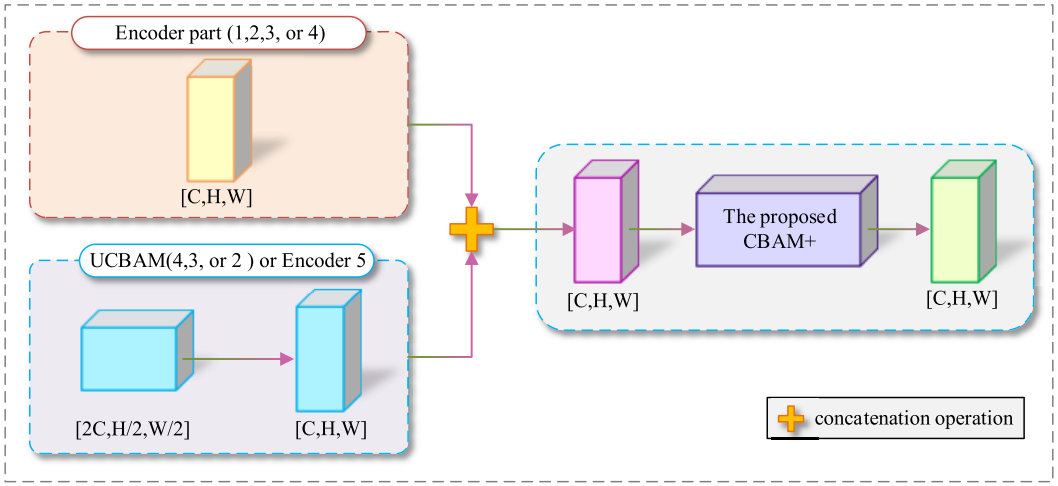

Fig. 3. The proposed UCBAM module framework. The number of the feature maps channel, height, and width are [C, H, W], respectively.

To overcome this issue, the CBAM+ module is proposed to capture the global context and make the neural network lightweight in this study. It contains channel attention and spatial attention modules (Figure 4). Specifically, the channel attention (CA) module extracts long-range dependency, capture the global edges, patterns, textures and enhance the performance of the model, and the spatial attention (SA) module was employed based on CBAM [40]. CBAM+ embedded into the UCBAM can improve calculation speed for the decoder part.

a) The proposed channel attention module: We propose a Channel Attention (CA) module to replace the original one based on CBAM [40]. CA can exploit the inter-channel relationship of features and focus on the important information, given an input image. The designed global attention pooling (GAP) performs pooling operations (Figure 4(a)), which can generate the attention weights to be assigned to global context features. Equation 1 gives the output of the GAP (F_{gap}):

$$F_{gap} = \sum_{j=1}^{N_p} \frac{e^{W_k \cdot x_j}}{\sum_{m=1}^{N_p} e^{W_k \cdot x_m}} x_j \tag{1}$$

where x is defined as an input feature, W_k is the linear transform matrix (1×1 convolution operation), N_p is the number of pixels (positions), and x_j enumerates all possible pixels (positions). F_{gap} is the output of the context modeling. Meanwhile, $\alpha_j = \frac{e^{W_k \cdot x_j}}{\sum_{m=1}^{N_p} e^{W_k \cdot x_m}}$ is defined as the weight of the global attention pooling. Specifically, the global attention pooling is passed through a 1×1 convolution, followed by a softmax function to generate attention weights, which can assign different weights to the features to obtain the context feature maps and capture long-range dependency.

On the other hand, the max pooling (F_{max}) gives significant information about desired features to infer finer channel attention. F_{gap} and F_{max} are employed to perform the channel attention feature extraction. Subsequently, the two pooling results are passed through a shared network to produce the proposed CA feature maps (Equation 2).

$$\begin{aligned} F_1 &= \sigma(Conv(GAPooling(F)) + Conv(MaxPooling(F))) \\ &= \sigma(W_1(W_0(F_{gap})) + W_1(W_0(F_{max}))) \end{aligned} \tag{2}$$

where F_1 and σ denote the outputs of the channel attention maps and sigmoid function, respectively. $W_0 \in \mathbb{R}^{C/r \times C}$, $W_1 \in \mathbb{R}^{C \times C/r}$, and $Conv$ is a 1×1 convolution operation.

b) Spatial attention module: In this study, we used the SA module based on CBAM [40] to produce a spatial attention map that captures the inter-spatial relationship of features. Compared with CA, this module focuses on the position of the region containing important information (Equation 3). The average and max pooling (S_{avg} and S_{max}, respectively) are employed to perform spatial operations to produce the SA maps with a convolution. The spatial attention module is represented by equation (3):

$$\begin{aligned} S_1 &= \sigma(Conv([AvgPooling(S); MaxPooling(S)])) \\ &= \sigma(Conv([S_{avg}; S_{max}])) \end{aligned} \tag{3}$$

where S is the input feature and AvgPooling and Maxpooling are defined as the average pooling and max pooling, respectively. Average pooling features: $S_{avg} \in \mathbb{R}^{1 \times H \times W}$, max pooling features: $S_{max} \in \mathbb{R}^{1 \times H \times W}$. $Conv$ is a 3×3 convolution operation.

c) Hierarchical feature-learning module: The different-level features include the different complex context, patterns, and textures information. Therefore, we adopted the hierarchical feature learning module to extract crack for obtaining the different crack semantic information individually (Figure 5). The designed hierarchical feature learning module can accelerate the convergence of neural network with deeply supervised nets (DSN) [41].

Each side output and fused output are supervised by DSN [41] according to the holistically-nested edge detection network (HED) [42]. A training dataset is defined as $S = (X_n, Y_n), n = 1, \ldots, N$, where X_n and Y_n are the raw original image and ground truth crack map, respectively. Each side network is followed by a classifier, and the weights for each side network are denoted as $w = (w^{(1)}, \ldots, w^{(M)})$. W and

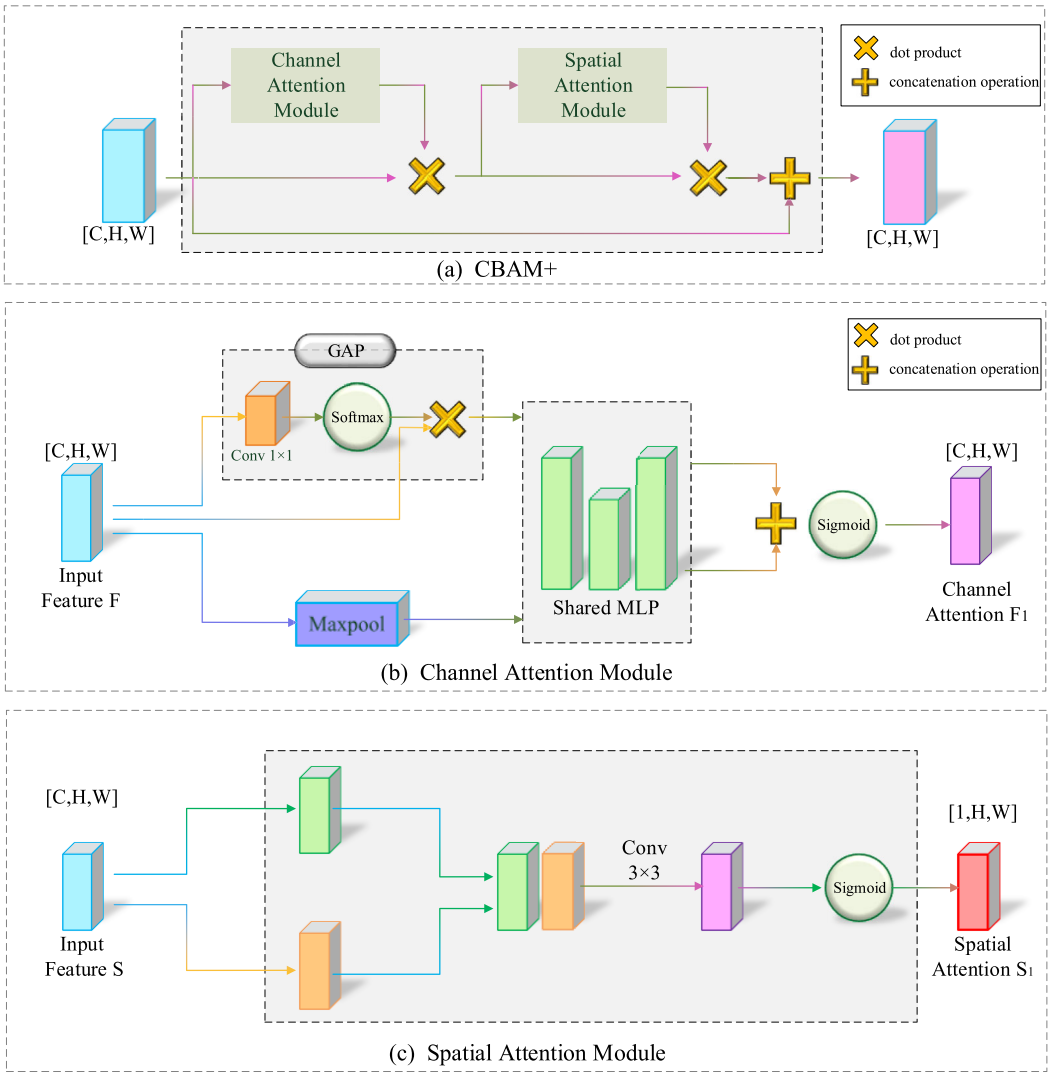

Fig. 4. Diagram of the proposed CBAM+ module.

M are the number of network parameters and side networks, respectively. Equation (4) represents the loss function for the side networks.

$$L_{side}(W, w) = \sum_{m=1}^{M} \alpha_m l_{side}^m(W, w^m) \tag{4}$$

where l_{side} and α_m are the loss function and a weight hyper-parameter at each side. The l_{side} loss function was applied to distinguish the non-crack and crack pixels with equation (5):

$$l_{side} = \frac{1}{N} \sum_{i=1}^{N} \{\beta y_i \log \hat{y}_i + \gamma(1 - y_i) \log(1 - \hat{y}_i)\} \tag{5}$$

where β and γ are hyperparameters, and N is defined as the number of pixels in one image. y_i and $\hat{y}_i$ are the label and predicted result, respectively.

The entitle outputs of the side networks are concatenated to produce fused feature maps (5 channels), and the fused loss function L_{fuse} is equation (6):

$$L_{fuse} = \frac{1}{N} \sum_{i=1}^{N} \{\beta y_i \log \hat{y}_i + \gamma(1 - y_i) \log(1 - \hat{y}_i)\} \tag{6}$$

where β and γ have the same meanings as in Equation (5). Finally, the total loss function is defined according to equation (7):

$$L_{total} = L_{side} + L_{fuse} \tag{7}$$

B. Adversarial Network (the Back End)

Unlike traditional GAN, the adversarial network developed in this study is based on the multi-scale L_1 loss to deal with mapping relationship between input and generated segmentation images (Figure 6) to perform closed-loop feedback for correcting high-order inconsistencies between ground truth and the predicted result and improve the detection accuracy of unrecognized small cracks in the segmentation network.

The input follows two paths (Figure 6): the former is the result of the predicted images being multiplied by the input images and the latter is the result of multiplying the labelled images by the input images. The multi-scale object loss function is employed to calculate the mean absolute error operation for features between two paths (Equation 6) and transfer to segmentation network with backpropagation

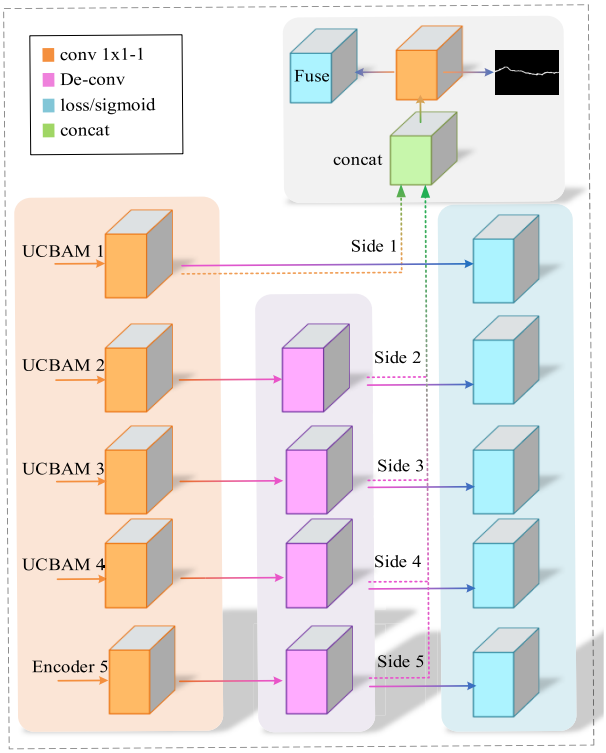

Fig. 5. The hierarchical feature learning module framework.

operation (Figure 1), which can improve the detection accuracy of unrecognized small cracks, and correct high-order inconsistencies between ground truth and the predicted result in the training process.

An adversarial network adopts the multi-scale L_1 loss function based on pixels level to solve the mapping relationship between input images and generated segmentation images. The multi-scale L_1 loss function is demonstrated with equation (8):

$$\min_{\theta_S}\max_{\theta_C} L_1(\theta_S,\theta_C) = \frac{1}{N}\sum_{n=1}^{N}\ell_{mae}(f_C(x_n\circ S(x_n)), f_C(x_n\circ y_n)) \quad (8)$$

where N is the number of images, and x_n and y_n are the input image and corresponding label image, respectively. ℓ_{mae} denotes the mean absolute error. $x_n \circ S(x_n)$ is defined as the result of multiplying the predicted image by input image. $x_n \circ y_n$ denotes the results obtained by multiplying the label image by the input image. $f_C(x)$ is a hierarchical feature extracted from image x by the adversarial network. ℓ_{mae} can be represented by equation (9):

$$\ell_{mae}(f_C(x), f_C(x')) = \frac{1}{N}\sum_{i=1}^{N}\|f_C^i(x) - f_C^i(x')\|_1 \quad (9)$$

where N is the number of layers in the adversarial network, and $f_C^i(x)$ is the extracted feature at the ith layer based on the adversarial network.

The multi-scale L_1 loss can capture long- and short- spatial relations between pixels by using the different level features, i.e., low-, middle- and high-level features, which can perform feedback operation (closed-loop feedback) to enforce segmentation and capture the difference between the predicted cracks and the ground truth and improve the accuracy of the segmentation network.

IV. Experiments and Results

Firstly, we mainly introduce experimental details of the CrackCLF in this section. Next, we present the evaluation metrics and compare them. Finally, we analyze and demonstrate experimental results.

A. Implementation Details

The proposed CrackCLF method was programmed using the Pytorch library [43] as the deep learning framework for training and testing based on the GPU server with four NVIDIA TITAN Xp GPU, each having 12GB of memory.

1) Crack Dataset: The public databases CFD [28] were used to evaluate the CrackCLF. The CFD database contains 118 images (resolution 320 × 480). 72 images and 46 images were used to train and test the proposed CrackCLF, respectively.

The Crack500 dataset [44] contains 500 images of size about 2000×1500, which were collected by phones in the main campus of the Temple University. The dataset is cropped into 16 non-overlapping image regions. Finally, training, validation, and test data contained 1896, 346, and 1124 images. All images share the same size of 256 × 256 in the training, validation, and test phases.

The Crack700 dataset was collected from Harbin, China [45], which contains 776 different types of images, such as concrete walls and bridge surfaces. The images were taken at different distances, which led to different resolutions. All images share the same size of 256 × 256 in the training, validation, and test processes.

2) Evaluate Metrics: The precision (Pr), recall (Re), and F1 score (F1) were used to evaluate CrackCLF using the following equations:

$$Pr = \frac{TP}{TP+FP} \quad (10)$$

$$Re = \frac{TP}{TP+FN} \quad (11)$$

$$F1 = \frac{2\times Pr\times Re}{Pr+Re} \quad (12)$$

where TP, FP, and FN are the number of true positives, false positives, and false negatives, respectively. $F1$ was employed to evaluate the overall performance of crack detection. Specifically, two different metrics based on $F1$ were adopted: the best $F1$ on the public database for a fixed threshold (ODS), and the aggregate $F1$ on the public database for the best threshold in each image (OIS).

ODS and OIS are defined as the max value according to Equations 13 and 14, respectively.

$$ODS = \frac{2\times Pr_t\times Re_t}{Pr_t+Re_t} : t = 0.001,\ldots 0.999 \quad (13)$$

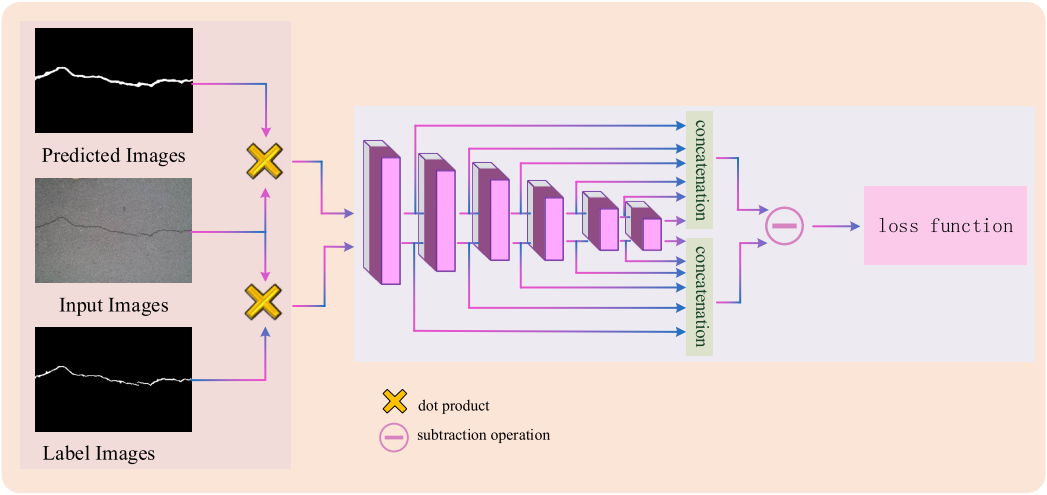

Fig. 6. An adversarial network framework (the back end). An adversarial network adopts the multi-scale L_1 loss to solve the mapping relationship between input images and generated segmentation images.

$$OIS = \frac{1}{N_{img}} \sum_{i}^{N_{img}} max \frac{2 \times Pr_t^i \times Re_t^i}{Pr_t^i + Re_t^i} : t = 0.001, \ldots 0.999 \tag{14}$$

where t, i, and N_{img} are the threshold, index, and number of images, respectively. Pr_t, Re_t, Pr_t^i, and Re_t^i are the precision and recall based on the threshold t and image i, respectively.

We consider the transitional areas between the non-crack and crack pixels before computing TP, FP, and FN. 2 pixels distances are also accepted in this study [1], [32], [33], [46], [47]. The decision threshold is defined as 0.5. The hyperparameter contains: batch size (4 images for CFD, 16 images for Crack 500 and Crack700), selecting adam optimzer, epochs 500, learning rate (0.001).

B. Experimental Results and Analysis

In this section, we mainly compare different algorithms based on different datasets to demonstrate the performance of the proposed CrackCLF method.

1) CFD Dataset: Table I lists the results in terms of Pr, Re, and F1 for different methods on CFD. Bold characters refer to the highest values of each column. The local thresholding method [48] is sensitive to noise and unable to inspect cracks (Pr equals to 0.7727 and F1 equals to 0.7418). The CrackForest method [28] was able to extract the wider cracks than the ground truth, which leads to low Pr (0.74466) and high Re (0.9514) values: it can extract many non-crack areas. As is shown in Figure 7, it was observed that U-net [49] and SegNet [50] can extract the crack skeleton, but there are some misidentifications in the images (Pr equals to 0.9119, 0.9325, 0.9256, 0.8876, respectively). U-HDN [14], DeepCrackLiu [15], and DeepCrackZou [4] adopt the U-shape framework method to perform crack detection based on hierarchical feature learning, which can extract the crack skeleton. Although CrackGAN [17] has a larger recall, it has a lower precision and F1, which can overestimate the crack regions with lower accuracy (Figure 7).

TABLE I
CRACK DETECTION EXPERIMENTAL RESULTS ON CFD DATASET

Method	Pr	Re	F1
Canny [51]	0.4377	0.7307	0.457
Local thresholding [48]	0.7727	0.8274	0.7418
CrackForest [28]	0.7466	0.9514	0.8318
CrackSeg [19]	0.7618	0.5959	0.6413
MFCD [52]	0.899	0.8947	0.8804
Method [46]	0.907	0.846	0.87
Structured prediction [32]	0.9119	0.9481	0.9244
Ensemble network [33]	0.9256	**0.9611**	0.934
U-net [49]	0.9325	0.932	0.928
U-HDN [14]	0.945	0.936	0.938
U-net ++ [18]	0.9412	0.935	0.938
DSMS [38]	0.9422	0.9388	0.941
DMA [37]	0.9435	0.9331	0.9382
BARNET [35]	0.9333	0.941	0.9356
Segnet [50]	0.8876	0.9073	0.8937
DeepCrackZou [4]	0.9447	0.9276	0.9364
DeepCrackLiu [15]	0.912	0.936	0.92
CrackGAN [17]	0.8803	**0.9611**	0.9189
CrackCLF	**0.9451**	0.9344	**0.9406**

However, traditional methods consider only pixel-to-pixel comparisons in the prediction phase, while CrackCLF transforms the feedback from the adversarial to the segmentation network improving the detection accuracy (Pr equals to 0.9451) and ensuring satisfactory performance. The F1 values demonstrate that CrackCLF gave the best overall performance of the crack detection (i.e., 0.9406) and the highest Pr value (i.e., 0.9451).

Table II lists the ODS and OIS values of competing methods based on CFD. Bold characters refer to the highest values of each column. Compared with other methods, CrackCLF gave the best performances in terms of ODS (0.9374) and OIS (0.9455). So, the CrackCLF can extract cracks information based on global and local databases (Figure 7).

2) Crack500 Dataset: Figure 8, Table III and Table IV present some examples of detection on Crack500. Bold characters refer to the highest values of each column. Segnet [50] and DeepCrackLiu [15] were not able to recognize and detect

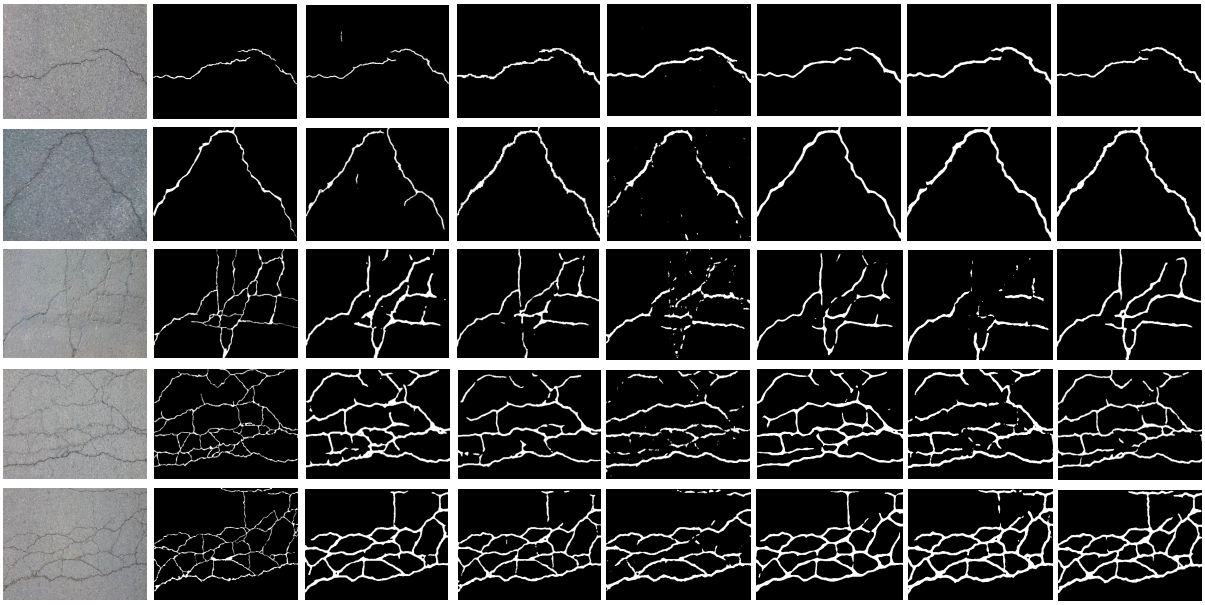

Fig. 7. Experimental results of comparison of the proposed CrackCLF with other methods based on CFD (from left to right: 1) original image, 2) groundTruth, 3) U-net, 4) U-HDN algorithm, 5) Segnet, 6) DeepCrackZou, 7) DeepCrackLiu, 8) CrackCLF).

TABLE II
THE ODS AND OIS OF COMPETING METHODS ON CFD DATASET

Methods	ODS	OIS
HED [42]	0.593	0.626
RCF [53]	0.542	0.607
FCN [54]	0.585	0.609
CrackForest [28]	0.104	0.104
FPHBN [44]	0.683	0.705
U-net [49]	0.923	0.931
U-net ++ [18]	0.933	0.943
DSMS [38]	0.936	0.942
DMA [37]	0.932	0.9422
BARNET [35]	0.9362	0.9435
U-HDN [14]	0.935	0.928
Segnet [50]	0.8938	0.9091
DeepCrack(Zou) [4]	0.9305	0.94
DeepCrack(Liu) [15]	0.9242	0.937
CrackCLF	**0.9374**	**0.9455**

TABLE III
CRACK DETECTION EXPERIMENTAL RESULTS ON CRACK500 DATASET

Method	Pr	Re	F1
Structured prediction [32]	0.3867	0.7984	0.4751
Ensemble network [33]	0.4147	0.7763	0.4953
SegNet [50]	0.7591	0.8504	0.7855
U-net [49]	0.7526	0.8632	0.7864
U-net ++ [18]	0.7625	0.853	0.7872
DSMS [38]	0.7616	0.8322	0.7885
DMA [37]	0.765	0.813	0.784
BARNET [35]	0.7733	0.831	0.7862
DeepCrackZou [4]	0.6655	**0.9064**	0.7499
DeepCrackLiu [15]	0.7661	0.8503	0.7885
U-HDN [14]	0.7744	0.8234	0.7788
CrackCLF	**0.7793**	0.8404	**0.7913**

thinner cracks (Pr equal to 0.7591 and 0.7661, respectively), and thereby unable to extract consecutive cracks (F1 equal to 0.7855 and 0.7885, respectively). Based on large Re (0.8632) and low Pr (0.7526), U-net [49] can not obtain the satisfactory results because it can overestimate and extract non-crack areas: it can inspect the thinner cracks, but it overestimates the crack width, which can lead to lower accuracy (Figure 8).

U-HDN [14] can inspect thinner cracks, but discontinuous cracks also occur in the image as is showing in Figure 8. Furthermore, it was observed that some isolated pixels were recognized as cracks. DeepCrackZou [4] cannot detect thin cracks and overestimates the crack width, which can cause high Re (0.9064) and low Pr (0.6655) values. CrackCLF can recognize and inspect the thinner crack and extract the continuous crack pixels obtaining satisfactory performance: compared with other methods, CrackCLF gave the best performances in terms of Pr (0.7793) and OIS (0.8225).

In summary, it is clear that CrackCLF can obtain two best performance results (Pr:0.7793, F1:0.7913) than others, and CrackCLF can obtain satisfactory accuracy. Pr metrics is higher than second method U-HDN (Pr:0.7744). In term of ODS and OIS, the proposed method obtain the best OIS (0.8225) and the fourth ODS (0.8096). It is obvious that the CrackCLF can get best accuracy in each image for different threshold value t with the OIS equation. Although the ODS is not the best result, the CrackCLF can obtain three best results (Pr, F1, and OIS) in five metrics. Therefore, CrackCLF ensures satisfactory performances.

3) Crack700 Dataset: Figure 9, Table V and Table VI list some examples of detection on Crack700. Bold characters refer to the highest values of each column. Under such conditions, Segnet [50] was able to extract the skeleton architect of the crack, but there were some isolated holes pixels of non-crack and crack. U-net [49], DeepCrackLiu [15], DeepCrackZou [4], and U-HDN [14] algorithms were able to extract the crack skeleton, but they tend to overestimate the crack width, which can lead to high Re (0.9760, 0.9677,

TABLE IV
THE ODS AND OIS OF COMPETING METHODS ON CRACK500 DATASET

Method	ODS	OIS
HED [42]	0.575	0.625
RCF [53]	0.49	0.586
FCN [54]	0.513	0.577
CrackForest [28]	0.199	0.199
FPHBN [44]	0.604	0.635
SegNet [50]	0.8116	0.8117
U-net [49]	0.8128	0.8131
U-net ++ [18]	0.8148	0.8188
DSMS [38]	0.813	0.8191
DMA [37]	0.818	0.821
BARNET [35]	0.8193	0.8211
DeepCrackZou [4]	0.8023	0.8155
DeepCrackLiu [15]	**0.8197**	0.819
U-HDN [14]	0.807	0.8091
CrackCLF	0.8096	**0.8225**

TABLE V
CRACK DETECTION EXPERIMENTAL RESULTS BASED ON CRACK700 DATASET

Method	Pr	Re	F1
Structuted prediction [32]	0.8217	0.9317	0.8584
Ensemble network [33]	0.827	0.9385	0.8631
U-HDN [14]	0.904	0.955	0.922
U-net [49]	0.867	**0.976**	0.9101
U-net ++ [18]	0.8825	0.953	0.9172
DSMS [38]	0.9027	0.9327	0.9185
DMA [37]	0.9078	0.9344	0.9211
BARNET [35]	0.9033	0.9331	0.9206
Segnet [50]	0.8511	0.9635	0.8934
DeepCrackLiu [15]	0.8756	0.9677	0.9105
DeepCrackZou [4]	0.9008	0.9533	0.9187
CrackCLF	**0.9151**	0.9457	**0.9237**

TABLE VI
THE ODS AND OIS OF COMPETING METHODS ON CRACK700 DATASET

Method	ODS	OIS
U-HDN [14]	0.916	0.934
U-net [49]	0.911	0.933
U-net ++ [18]	0.916	0.935
DSMS [38]	0.913	0.931
DMA [37]	0.912	0.935
BARNET [35]	0.917	0.933
Segnet [50]	0.892	0.918
DeepCrackLiu [15]	0.9151	**0.9368**
DeepCrackZou [4]	0.909	0.929
CrackCLF	**0.9182**	0.9364

0.9533, and 0.9550, respectively) and low Pr (0.8670, 0.8756, 0.9008, and 0.9040, respectively) values. Compared with other algorithms, CrackCLF gave low Re (0.9457), the highest Pr (0.9151) and the highest F1 (0.9237) because it can extract thin and consecutive crack pixels.

From Table VI, it is clear that the proposed CrackCLF can obtain the best ODS (0.9181) and the second OIS (0.9364, only small 0.0004). It presents that CrackCLF can extract more crack information for different thresholds based on the global database with equation 13. In short, the CrackCLF get three best results (Pr:0.9151, F1:0.9237, and ODS:0.9182) than others based on five metrics.

C. Ablation Study

To demonstrate the superiority of the CrackCLF framework, ablation study is adopted to verify the performance of the CrackCLF and the advantages of the different modules from different crack datasets (Table VII) with the encoder U-net. CBAM, CBAM+, the adversarial network, and the hierarchal feature learning modules.

CBAM+ could perform better accuracy (larger Pr, F1, ODS and OIS for CFD) than CBAM, and CBAM gives a higher Re than CBAM+. When the adversarial network was embedded into CrackCLF, Pr and F1 were enhanced (Pr equal to 0.9459 for CFD with adversarial network, 0.7793 for Crack500 with adversarial and hierarchical network, and 0.9191 for Crack700 with adversarial network, showing the adversarial network can improve the detection accuracy; F1 equal to 0.9604 for CFD with adversarial and hierarchical network, 0.7913 for Crack500 with adversarial and hierarchical network, and 0.9237 for Crack700 with adversarial network), and Re was reduced (Re equal to 0.9337 for CFD, 0.8346 for Crack500, and 0.9289 for Crack700 with adversarial network).

The proposed CBAM+ embedded into the UCBAM can not only exploit the inter-channel relationship of features and focus on the important information and enhance the accuracy for different databases from Table VII, but also reduce the parameters' number and calculations costs, accelerate the neural network training speed.

Meanwhile, some metrics values are further to enhancing and the neural network can extract thinner cracks by the embedded adversarial network with closed-loop feedback for correcting high-order inconsistencies between the label and the predicted image with segmentation network. Finally, the hierarchical feature module with a deep-supervised network and fusing feature maps accelerates the convergence speed and improves model performance.

Figure 10 shows the comparison of the proposed method with existing methods in terms of detecting thinner cracks. U-net method can overestimate the crack features, as is shown in Figure 10, which result in a low accuracy. In this example, DeepCrack(Liu) cannot extract the continuous shallow cracks very well, which results in a low precision. U-HDN method was not able to very well detect continuous shallow cracks either and fail to extract some thin cracks. DeepCrack(Zou) method tends to extract the wider cracks and ignore the continuous thin ones, which can cause the higher recall and lower accuracy. Figure 10 also shows that the CrackCLF without the adversarial network cannot extract the thin cracks well, which leads to lower precision. Meanwhile, it can be seen that the CrackCLF without adversarial network overestimates the crack regions, causing larger recall and lower precision, as showing in Table VII. The proposed CrackCLF with closed-loop feedback can better extract continuous thin crack features, which effectively improves the accuracy of crack detection. Meanwhile, the width for crack feature is better identified by our method in comparison with other methods. In summary, the proposed CrackCLF with closed-loop feedback performs better in extracting shallow cracks.

D. Model Complexity

We utilize the number of the parameters (Params), floating-point operations per second (FLOPs) and frames per second (FPS) to evaluate the processing efficiency of CrackCLF

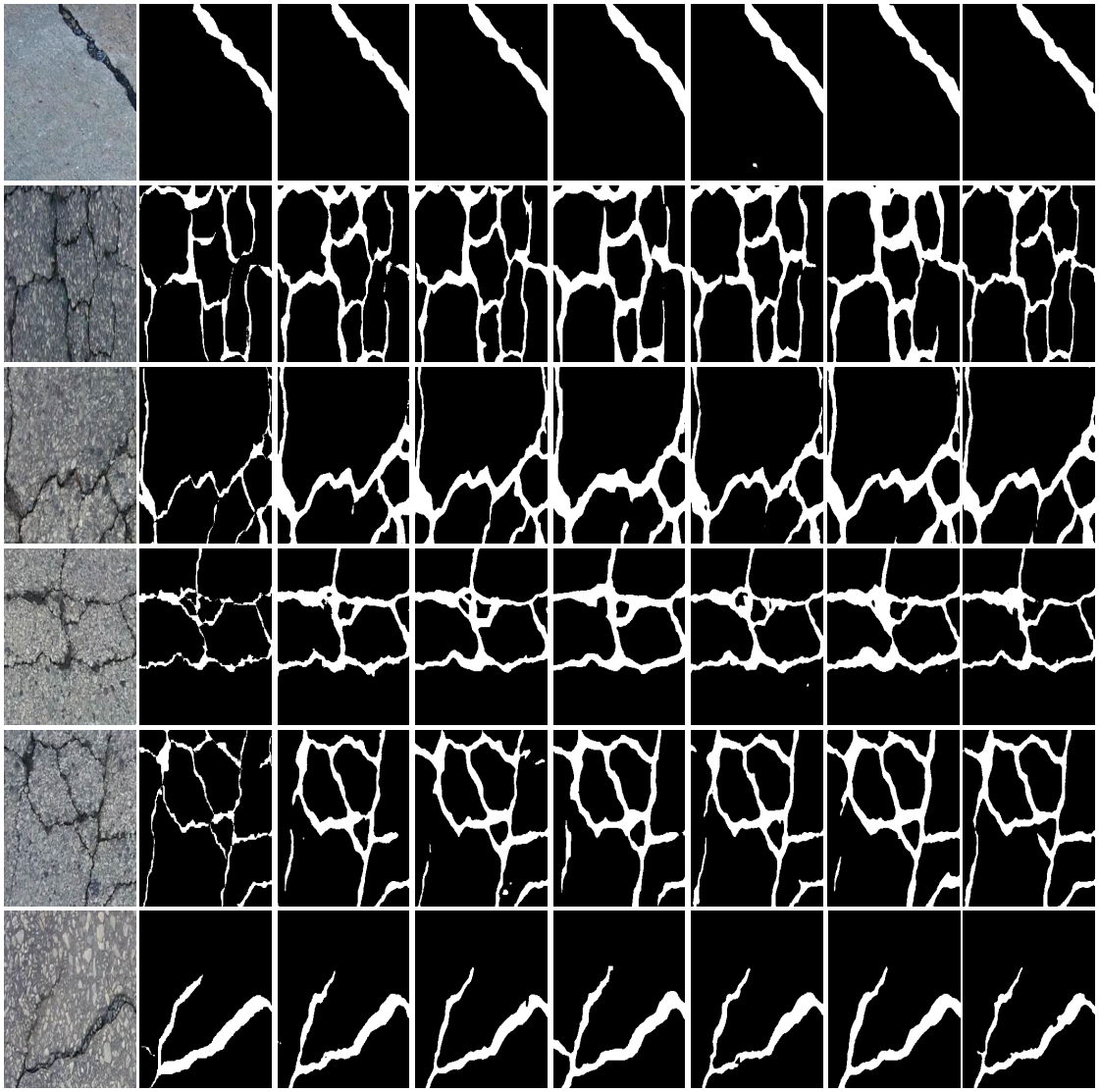

Fig. 8. Experimental results of comparison of the proposed CrackCLF with other methods based on Crack500 (from left to right: 1) original image, 2) groundTruth, 3) Segnet, 4) U-net, 5) DeepCrackLiu, 6) U-HDN algorithm, 7) DeepCrackZou, 8) CrackCLF).

TABLE VII
THE COMPARISON OF ABLATION STUDY BASED ON DIFFERENT MODULES

Dataset	Encoder	CBAM	CBAM+	Adversarial Network	Hierarichal Feature	Pr	Re	F1	ODS	OIS
CFD	**U-net Encoder**	✓				0.9178	**0.9509**	0.9321	0.9343	0.9411
			✓			0.9335	0.9441	0.937	0.9367	0.9421
			✓	✓		**0.9459**	0.9337	0.9379	0.9351	0.9432
			✓	✓	✓	0.9451	0.9344	**0.9406**	**0.9374**	**0.9455**
Crack500		✓				0.7222	**0.8755**	0.7729	0.7997	0.8064
			✓			0.7299	0.874	0.7776	0.8016	0.8092
			✓	✓		0.7756	0.8346	0.7861	0.8055	0.8163
			✓	✓	✓	**0.7793**	0.8404	**0.7913**	**0.8096**	**0.8225**
Crack700		✓				0.9048	0.9471	0.9174	0.9125	0.9292
			✓			0.9061	**0.9479**	0.9198	0.9111	0.9314
			✓	✓		**0.9191**	0.9289	0.9159	0.9124	0.9338
			✓	✓	✓	0.9151	0.9457	**0.9237**	**0.9182**	**0.9364**

and other models (Table VIII). It can be observed from the results that compared with other methods, the proposed CrackCLF achieves satisfactory efficiency based on the experimental results although FLOPs and Params are not the smallest (Table VIII). However, the costs of calculation for the proposed CrackCLF only with the segmentation network

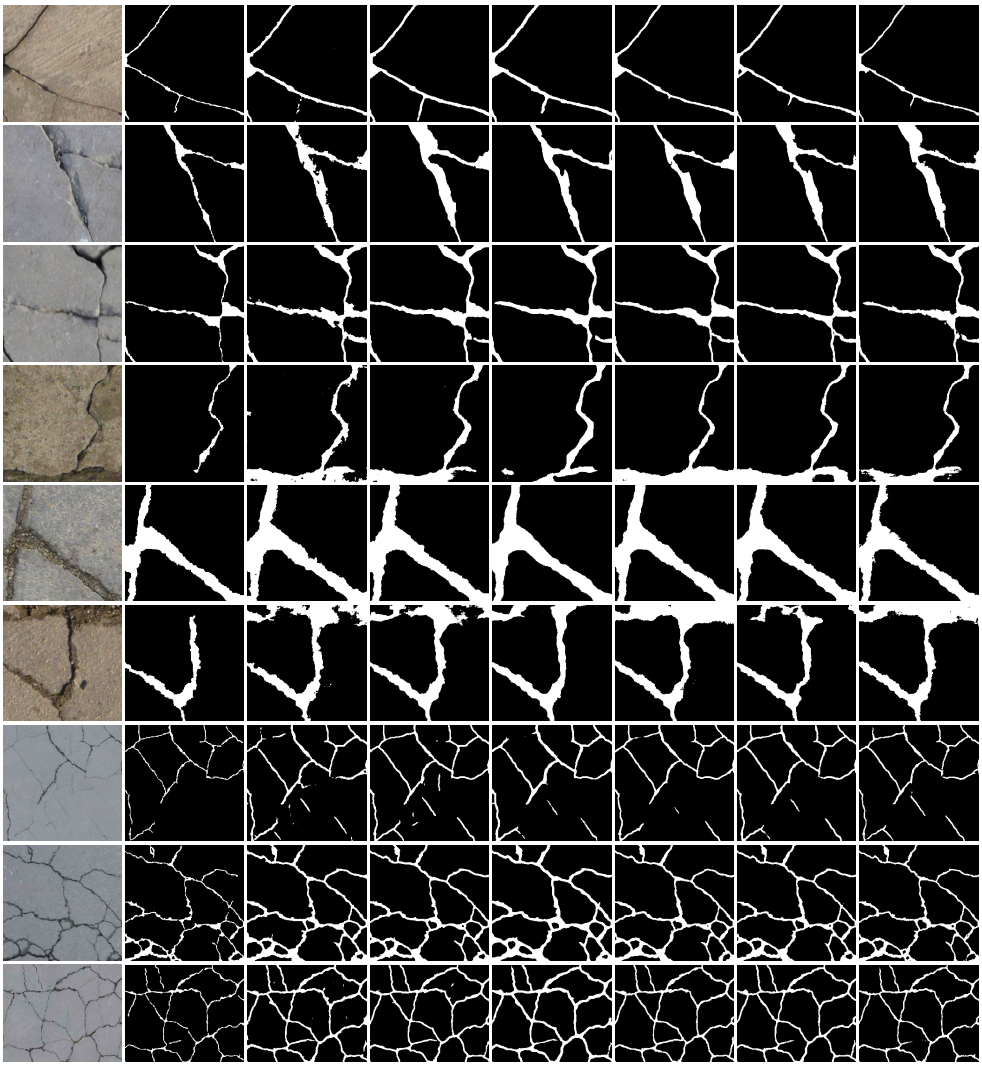

Fig. 9. Experimental results of comparison of the proposed CrackCLF with other methods based on Crack700 (from left to right: 1) original image, 2) groundTruth, 3) Segnet, 4) U-net, 5) DeepCrackLiu, 6) DeepCrackZou, 7) U-HDN, 8) CrackCLF).

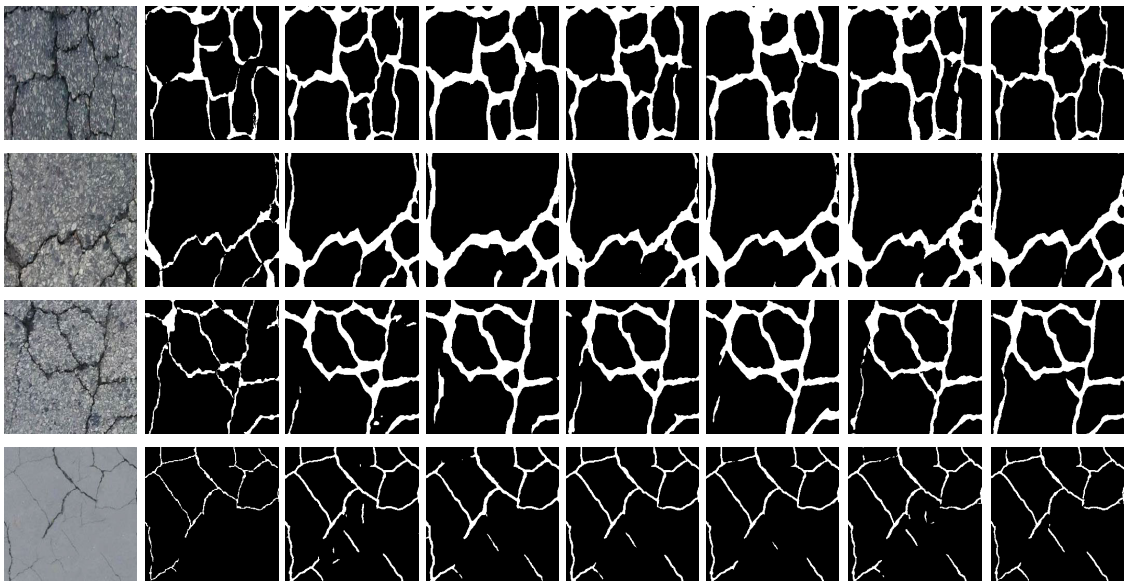

Fig. 10. Comparison of the CrackCLF with or without adversarial network (from left to right: 1) original image, 2) Groundtruth, 3) U-net, 4) DeepCrackLiu, 5) U-HDN, 6) DeepCrackZou, 7) CrackCLF without adversarial network, 8) CrackCLF).

(FLOP:17.02G) are smallest among the compared ones, which can run pretty fast (30FPS) to detect crack images during the inference stage with relatively small Params (18.84M). Thanks to the fewest FLOPs and least parameter number,

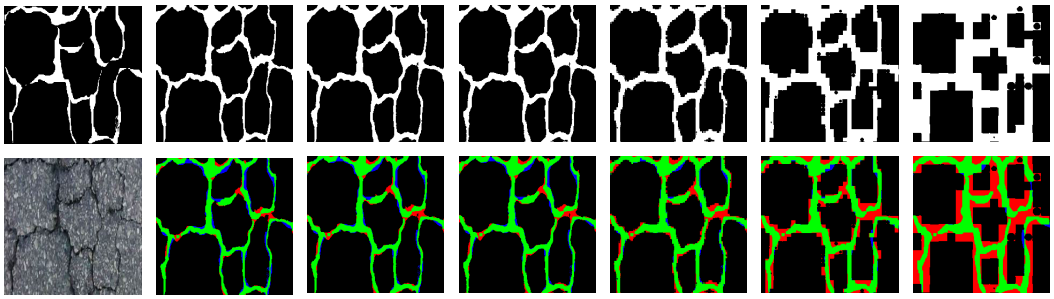

Fig. 11. Visualization of feature maps for each UCBAM module. The green, red, and blue pixels represent true positives (TP), false positives (FP), and false negatives (FN), respectively. (from left to right: 1) Original Image or Groundtruth, 2) Output, 3) UCBAM 1, 4) UCBAM 2, 5) UCBAM 3, 6) UCBAM 4, 7) Encoder 5).

TABLE VIII
THE COMPARISON OF MODEL COMPLEXITY

Method	FLOPs	Params	FPS(F/s)
HED [42]	20.07G	**14.72M**	**33**
RCF [53]	20.44G	14.80M	33
U-net [49]	65.47G	34.53M	13
Segnet [50]	40.16G	29.44M	16
DeepCrackZou [4]	42.12G	31.78M	16
DeepCrackLiu [15]	20.11G	**14.72M**	**33**
U-HDN [14]	38.66G	29.12M	16
CrackGAN [17]	51.54G	34.77M	13
Unet++ [18]	107.78G	43.23M	6
CrackCLF(only Segmentation)	**17.02G**	18.84M	30
CrackCLF	26.04G	26.39M	18

TABLE IX
CRACK DETECTION EXPERIMENTAL RESULTS BASED ON THREE DATASETS. W/ AND W/O MEAN WITH AND WITHOUT, RESPECTIVELY

Datasets	Methods	Pr	Re	F1
CFD	U-net w/ CLF [49]	**0.9345**	0.9315	**0.9326**
	U-net w/o CLF [49]	0.9325	**0.932**	0.928
	U-net ++ w/ CLF [18]	**0.9457**	0.9287	**0.9391**
	U-net ++ w/o CLF [18]	0.9412	**0.935**	0.938
	DeepCrackZou w/ CLF [4]	**0.9452**	0.9256	**0.9377**
	DeepCrackZou w/o CLF [4]	0.9447	**0.9276**	0.9364
	DeepCrackLiu w/ CLF [15]	**0.9266**	0.9355	**0.9299**
	DeepCrackLiu w/o CLF [15]	0.912	**0.936**	0.92
	U-HDN w/ CLF [14]	**0.9458**	**0.9378**	**0.9392**
	U-HDN w/o CLF [14]	0.945	0.936	0.938
	CrackCLF w/ CLF	**0.9451**	0.9344	**0.9406**
	CrackCLF w/o CLF	0.9335	**0.9441**	0.937
Crack500	U-net w/ CLF [49]	**0.7645**	0.8533	**0.7886**
	U-net w/o CLF [49]	0.7526	**0.8632**	0.7872
	U-net ++ w/ CLF [18]	**0.7689**	0.8466	**0.7911**
	U-net ++ w/o CLF [18]	0.7625	**0.853**	0.7872
	DeepCrackZou w/ CLF [4]	**0.6956**	0.8823	**0.7612**
	DeepCrackZou w/o CLF [4]	0.6655	**0.9064**	0.7499
	DeepCrackLiu w/ CLF [15]	**0.7713**	0.8423	**0.7894**
	DeepCrackLiu w/o CLF [15]	0.7661	**0.8503**	0.7885
	U-HDN w/ CLF [14]	**0.7812**	0.8123	**0.7823**
	U-HDN w/o CLF [14]	0.7741	**0.8234**	0.7788
	CrackCLF w/ CLF	**0.7793**	0.8404	**0.7913**
	CrackCLF w/o CLF	0.7299	**0.874**	0.7761
Crack700	U-net w/ CLF [49]	**0.8811**	0.9522	**0.9156**
	U-net w/o CLF [49]	0.867	**0.976**	0.9101
	U-net ++ w/ CLF [18]	**0.8956**	0.9423	**0.9201**
	U-net ++ w/o CLF [18]	0.8825	**0.953**	0.9172
	DeepCrackZou w/ CLF [4]	**0.9128**	0.9425	**0.9219**
	DeepCrackZou w/o CLF [4]	0.9008	**0.9533**	0.9187
	DeepCrackLiu w/ CLF [15]	**0.8934**	0.9422	**0.9188**
	DeepCrackLiu w/o CLF [15]	0.8756	**0.9677**	0.9105
	U-HDN w/ CLF [14]	**0.9149**	0.9423	**0.9256**
	U-HDN w/o CLF [14]	0.904	**0.955**	0.922
	CrackCLF w/ CLF	**0.9151**	0.9457	**0.9237**
	CrackCLF w/o CLF	0.9061	**0.9479**	0.9198

DeepCrack(Liu) and HED achieve the fastest inference speed of 33 FPS (0.03 second per image), and can detect cracks in real time (normally, 24 FPS for human eye). Our method obtains a running speed of 30 FPS, exactly 0.033 second per image, also achieving a real-time detection. This is because in practice, only the segmentation network (the front end) is employed at the crack inference stage after training is finished, and the adversarial network (the back end) is not needed in real world deployment. In short, the proposed CrackCLF is able to obtain superior detection accuracy with a satisfactory detection speed, which can meet the requirements of most real-world application scenarios.

E. Visualization for UCBAM Modules

The proposed CBAM+ module can exploit the inter-channel relationship of features and focus on the important information, given an input image. It can improve the network ability to extract high- and low- level semantic information, which is crucial for accurate crack segmentation. In Figure 11, we visualize the feature maps UCBAM modules and encoder 5, to intuitively observe the effect of these modules in the proposed network. After being processed by these modules, the feature maps are clearly enhanced on the crack areas. Based on the enhanced processed features, the classifier can more readily recognize the crack areas and perform an even more accurate crack segmentation.

Meanwhile, this design enables the module to consider both high-level and low-level characteristics in determining the weights of channels and screening channels that better characterize cracks. By visualizing the output of the UCBAM module, we can clearly observe that the discrimination between crack and non-crack pixels is greatly facilitated by focusing on the significantly reduced regions highlighted by the proposed attention mechanism.

F. Generalization Capability of CLF Module

It is also worthwhile to point out that one of the most important contributions of this work is that we propose a

CLF mechanism that can be integrated into almost any current models as a plug-in module, and further improve their performances by introducing a close loop feedback to address the open-loop issue of existing models.

In order to validate this, we conducted a comprehensive ablation study, in which we compared many existing models with and without the CLF mechanism, as shown in Table IX. It can be observed that the state-of-the-art methods without CLF tend to have a higher Re, but lower Pr and F1 score than their counterparts with CLF mechanism, which means that integrating CLF can effectively improve their performances. In particular, the back end (CLF) is used to correct higher-order inconsistencies between labels and crack maps (generated by the front end) to address the issue of open-loop systems, which can help improve the performance of the networks. In a word, the proposed CLF can be defined as a plug and play module, which can be embedded into different neural network models to improve their performances.

V. Conclusion

Automatic pavement crack detection is an imperative task to ensure functional and structural performances of road pavements. To tackle an open-loop (OL) system with encoder-decoder framework, we introduce closed-loop feedback (CLF) in the neural network so that the model could learn to correct errors on its own, based on generative adversarial networks (GAN), which is called CrackCLF and includes the front and back end (segmentation and adversarial network). The segmentation network contains two parts: encoder and decoder. In the encoder part, we employ the U-net encoder pat and we propose the UCBAM module to replace the convolution operation in the decoder part. Meanwhile, the proposed CBAM+ module is embedded into the UCBAM module, and the designed hierarchical feature module is employed to obtain the multiscale feature maps of different decoder parts. An adversarial network is used to enforce the segmentation and adversarial networks to learn global and local crack information to overcome open-loop system issues. The proposed framework has been compared with other methods (i.e., Canny [51], Local thresholding [48], CrackForest [28], MFCD [52], Structured prediction [32], Ensemble network [33], U-net [49], U-HDN [14], Segnet [50], DeepCrackZou [4], DeepCrackLiu [15] using three public datasets (i.e., CFD, Crack500, and Crack700). CrackCLF can give satisfactory output in terms of Re, Pr, and F1 (not less than 0.7793, 0.8404, and 0.7913, respectively). Finally, the ablation study verifies the performance of the CrackCLF with different modules based on different crack datasets. In summary, the proposed CrackCLF can give satisfactory results on three public datasets. Moreover, the proposed CLF can be defined as a plug and play module, which can be embedded into different neural network models to improve their performances.

In this study, CrackCLF gave promising results. However, there are still some limitations to be addressed in future work.

- Because the automatic crack detection system is only used to detect individual images so far. Video streaming will be tested in future studies.
- Moreover, the artificially designed neural network may contain redundant feature maps. Designing a neural network that can automatically optimize and prune its structure and parameters will be further investigated in our future work.
- The size of CrackCLF can be further reduced to become a more lightweight model, so that it can be more easily deployed in devices with limited computing resources, or adversary environments where computing capability of the devices will be largely restrained.
- The mechanisms of CrackCLF may be integrated with those of SAM to further improve its performance. This can be a new direction that is worthwhile to be investigated in the future.

References

[1] R. Amhaz, S. Chambon, J. Idier, and V. Baltazart, "Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2718–2729, Oct. 2016.

[2] D. Jones, "Pavement cracking: Mechanisms, modeling, detection, testing and case histories," *Acta Oto-Laryngol.*, vol. 135, no. 10, pp. 1–5, 2008.

[3] J. Zaniewski and M. Mamlouk, "Pavement preventive maintenance: Key to quality highways," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1680, no. 1, pp. 26–29, Jan. 1999.

[4] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "DeepCrack: Learning hierarchical convolutional features for crack detection," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1498–1512, Mar. 2019.

[5] C. Y. Chan, B. Huang, X. Yan, and S. Richards, "Investigating effects of asphalt pavement conditions on traffic accidents in Tennessee based on the pavement management system (PMS)," *J. Adv. Transp.*, vol. 44, no. 3, pp. 150–161, Jul. 2010.

[6] A. Cubero-Fernandez, F. J. Rodriguez-Lozano, R. Villatoro, J. Olivares, and J. M. Palomares, "Efficient pavement crack detection and classification," *EURASIP J. Image Video Process.*, vol. 2017, no. 1, pp. 1–11, Dec. 2017.

[7] H. S. Park, H. M. Lee, H. Adeli, and I. Lee, "A new approach for health monitoring of structures: Terrestrial laser scanning," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 22, no. 1, pp. 19–30, Jan. 2007.

[8] M. O'Byrne, B. Ghosh, F. Schoefs, and V. Pakrashi, "Regionally enhanced multiphase segmentation technique for damaged surfaces," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 29, no. 9, pp. 644–658, Oct. 2014.

[9] J.-K. Oh et al., "Bridge inspection robot system with machine vision," *Autom. Construct.*, vol. 18, no. 7, pp. 929–941, Nov. 2009.

[10] R. S. Lim, H. M. La, and W. Sheng, "A robotic crack inspection and mapping system for bridge deck maintenance," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 367–378, Apr. 2014.

[11] E. Zalama, J. Gómez-García-Bermejo, R. Medina, and J. Llamas, "Road crack detection using visual features extracted by Gabor filters," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 29, no. 5, pp. 342–358, May 2014.

[12] P. Prasanna et al., "Automated crack detection on concrete bridges," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 591–599, Apr. 2016.

[13] V. Kaul, A. Yezzi, and Y. Tsai, "Detecting curves with unknown endpoints and arbitrary topology using minimal paths," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 10, pp. 1952–1965, Oct. 2012.

[14] Z. Fan et al., "Automatic crack detection on road pavements using encoder–decoder architecture," *Materials*, vol. 13, no. 13, p. 2960, Jul. 2020.

[15] Y. Liu, J. Yao, X. Lu, R. Xie, and L. Li, "DeepCrack: A deep hierarchical feature learning architecture for crack segmentation," *Neurocomputing*, vol. 338, pp. 139–153, Apr. 2019.

[16] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[17] K. Zhang, Y. Zhang, and H.-D. Cheng, "CrackGAN: Pavement crack detection using partially accurate ground truths based on generative adversarial learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 1306–1319, Feb. 2021.

[18] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: Redesigning skip connections to exploit multiscale features in image segmentation," *IEEE Trans. Med. Imag.*, vol. 39, no. 6, pp. 1856–1867, Jun. 2020.

[19] Z. Gao, B. Peng, T. Li, and C. Gou, "Generative adversarial networks for road crack image segmentation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[20] P. Subirats, J. Dumoulin, V. Legeay, and D. Barba, "Automation of pavement surface crack detection using the continuous wavelet transform," in *Proc. Int. Conf. Image Process.*, Oct. 2006, pp. 3037–3040.

[21] J. Tang and Y. Gu, "Automatic crack detection and segmentation using a hybrid algorithm for road distress analysis," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 3026–3030.

[22] Q. Li and X. Liu, "Novel approach to pavement image segmentation based on neighboring difference histogram method," in *Proc. Congr. Image Signal Process.*, vol. 2, 2008, pp. 792–796.

[23] H. Oliveira and P. L. Correia, "Automatic road crack detection and characterization," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 155–168, Mar. 2013.

[24] H. Oliveira and P. L. Correia, "CrackIT—An image processing toolbox for crack detection and characterization," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 798–802.

[25] H. Adeli, "Neural networks in civil engineering: 1989–2000," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 16, no. 2, pp. 126–142, Mar. 2001.

[26] H. Adeli and S. L. Hung, "Machine learning—Neural networks, genetic algorithms and fuzzy systems," *Kybernetes*, vol. 28, no. 3, pp. 317–318, 1999.

[27] A. Tedeschi and F. Benedetto, "A real-time automatic pavement crack and pothole recognition system for mobile Android-based devices," *Adv. Eng. Informat.*, vol. 32, pp. 11–25, Apr. 2017.

[28] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3434–3445, Dec. 2016.

[29] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3708–3712.

[30] A. Zhang et al., "Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 10, pp. 805–819, Oct. 2017.

[31] A. Zhang et al., "Automated pixel-level pavement crack detection on 3D asphalt surfaces with a recurrent neural network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 34, no. 3, pp. 213–229, Mar. 2019.

[32] Z. Fan, Y. Wu, J. Lu, and W. Li, "Automatic pavement crack detection based on structured prediction with the convolutional neural network," 2018, *arXiv:1802.02208*.

[33] Z. Fan et al., "Ensemble of deep convolutional neural networks for automatic pavement crack detection and measurement," *Coatings*, vol. 10, no. 2, p. 152, Feb. 2020.

[34] Z. Qu, C. Cao, L. Liu, and D.-Y. Zhou, "A deeply supervised convolutional neural network for pavement crack detection with multiscale feature fusion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 4890–4899, Sep. 2022.

[35] J.-M. Guo, H. Markoni, and J.-D. Lee, "BARNet: Boundary aware refinement network for crack detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7343–7358, Jul. 2022.

[36] J. König, M. D. Jenkins, M. Mannion, P. Barrie, and G. Morison, "Weakly-supervised surface crack segmentation by generating pseudo-labels using localization with a classifier and thresholding," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24083–24094, Dec. 2022.

[37] X. Sun, Y. Xie, L. Jiang, Y. Cao, and B. Liu, "DMA-Net: DeepLab with multi-scale attention for pavement crack segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18392–18403, Oct. 2022.

[38] Q. Zhou, Z. Qu, S.-Y. Wang, and K.-H. Bao, "A method of potentially promising network for crack detection with enhanced convolution and dynamic feature fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18736–18745, Oct. 2022, doi: 10.1109/TITS.2022.3154746.

[39] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[40] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.

[41] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. Artif. Intell. Statist.*, 2015, pp. 562–570.

[42] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1395–1403.

[43] A. Paszke et al., "Automatic differentiation in PyTorch," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2017, pp. 1–4.

[44] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, "Feature pyramid and hierarchical boosting network for pavement crack detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 4, pp. 1525–1535, Apr. 2020.

[45] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, and X. Yang, "Automatic pixel-level crack detection and measurement using fully convolutional network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 33, no. 12, pp. 1090–1109, Dec. 2018.

[46] D. Ai, G. Jiang, L. S. Kei, and C. Li, "Automatic pixel-level pavement crack detection using information of multi-scale neighborhoods," *IEEE Access*, vol. 6, pp. 24452–24463, 2018.

[47] J. König, M. D. Jenkins, P. Barrie, M. Mannion, and G. Morison, "A convolutional neural network for pavement surface crack segmentation using residual connections and attention gating," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 1460–1464.

[48] H. Oliveira and P. L. Correia, "Automatic road crack segmentation using entropy and image dynamic thresholding," in *Proc. 17th Eur. Signal Process. Conf.*, Aug. 2009, pp. 622–626.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Germany: Springer, 2015, pp. 234–241.

[50] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[51] H. Zhao, G. Qin, and X. Wang, "Improvement of Canny algorithm based on pavement edge detection," in *Proc. 3rd Int. Congr. Image Signal Process.*, vol. 2, Oct. 2010, pp. 964–967.

[52] H. Li, D. Song, Y. Liu, and B. Li, "Automatic pavement crack detection by multi-scale image fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 2025–2036, Jun. 2019.

[53] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3000–3009.

[54] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

Chong Li received the M.S. degree in electronic engineering from Shantou University, Shantou, China, in 2018, where he is currently pursuing the Ph.D. degree in structural engineering. His current research interests include image processing and machine learning.

Zhun Fan (Senior Member, IEEE) received the B.S. and M.S. degrees in control engineering from the Huazhong University of Science and Technology, Wuhan, in 1995 and 2000, respectively, and the Ph.D. degree in electrical engineering from Michigan State University, East Lansing, MI, USA, in 2004. He is currently a Full Professor and the Head of the Department of Electronic and Information Engineering, Shantou University. His research interests include artificial intelligence and computer vision.

Ying Chen received the M.S. degree from Shantou University, Shantou, in 2022. Her current research interests include medical image analysis, image processing, and deep learning.

Huibiao Lin received the M.S. degree in artillery automatic weapons and ammunition engineering from Shenyang Ligong University, Shenyang, in 2008. He is currently pursuing the Ph.D. degree in structural engineering from the College of Engineering, Shantou University, Shantou, China. His research interests include computational intelligence, data mining, computer vision, image processing, and robotics.

Laura Moretti received the Ph.D. degree in infrastructure and transportation. She is an Associate Professor at the Department of Civil, Building and Environmental Engineering, Sapienza University of Rome. Her research interests are risk analysis, life cycle assessment, management, and construction of transport infrastructures. Her teaching activities are about the safety of road works, the design of transportation during emergencies, road infrastructures, and concrete pavements for bachelor's, master's, and second-level master's degrees. She is involved in several research on life cycle assessment, green public procurement, airport safety, environmental impact, design, construction, and maintenance of roads and airports. She is the author of more than 70 articles published in conference proceedings and international journals and she is a member of scientific committees at several conferences.

Giuseppe Loprencipe received the M.S. degree in civil engineering and the Ph.D. degree in transportation infrastructures from the Sapienza University of Rome, Italy, in 1994 and 2000, respectively. He is currently a Full Professor at the Sapienza University of Rome. His research interests include pavement management, road, railway, and airport engineering. He is a member of the Italian Society of Road Infrastructures (SIIV) Governing Council and a Member of the Commission of the Italian Ministry of Infrastructure and Transportation on Climate Change.

Weihua Sheng (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from Michigan State University, East Lansing, MI, USA, in 2002. He is currently an Associate Professor with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, USA. His current research interests include wearable computing and intelligent transportation systems.

Kelvin C. P. Wang received the B.S. degree from Southwest Jiaotong University, China, the M.S. degree from Beijing Jiaotong University, China, and the Ph.D. degree from Arizona State University. His professional career started at Arizona DOT in 1989, where he has been a University Faculty since 1993. He is currently a Full Professor of civil engineering with Oklahoma State University, Stillwater, OK, USA, where he holds the Dawson Chair. He received the ASCE 2011 Frank M. Masters Transportation Engineering Award and the ASCE 2018 Turner Award. He was the President of the Transportation and Development Institute of the American Society of Civil Engineers for FY 2017.