CrackFormer Network for Pavement Crack Segmentation

Huajun Liu, *Member, IEEE*, Jing Yang, Xiangyu Miao, Christoph Mertz, and Hui Kong, *Member, IEEE*

***Abstract*— In this paper, we rethink our earlier work on self-attention based crack segmentation, and propose an upgraded CrackFormer network (CrackFormer-II) for pavement crack segmentation, instead of only for fine-grained crack-detection tasks. This work embeds novel Transformer encoder modules into a SegNet-like encoder-decoder structure, where the basic module is composed of novel Transformer encoder blocks with effective relative positional embedding and long range interactions to extract efficient contextual information from feature-channels. Further, fusion modules of scaling-attention are proposed to integrate the results of each respective encoder and decoder block to highlight semantic features and suppress non-semantic ones. Moreover, we update the Transformer encoder blocks enhanced by the local feed-forward layer and skip-connections, and optimize the channel configurations to compress the model parameters. Compared with the original CrackFormer, the CrackFormer-II is trained and evaluated on more general crack datasets. It achieves higher accuracy than the original CrackFormer, and the state-of-the-art (SOTA) method with 6.7× fewer FLOPs and 6.2× fewer parameters, and its practical inference speed is comparable to most classical CNN models. The experimental results show that it achieves the F-measures on Optimal Dataset Scale (ODS) of 0.912, 0.908, 0.914 and 0.869, respectively, on the four benchmarks. Codes are available at https://github.com/LouisNUST/CrackFormer-II.**

***Index Terms*— Automatic crack segmentation, SegNet, ConvNet, transformer, CrackFormer.**

I. Introduction

PAVEMENT crack is the most common type of surface distress [1]. Manual crack detection is considerably tedious and requires rich experienced domain experts. To facilitate the progress of pavement survey, it is necessary to achieve automatic crack segmentation. Because of the in-homogeneous intensity, non-consistent contrast, and very cluttered background [2], crack segmentation is a very tough problem. In addition, road of different types exhibits different types of textures, which share similar appearance and extreme weak contrast with cracks, and makes crack detection more difficult. Moreover, exploring a general model to detection both fine-grained cracks and coarse ones is more challenging.

Up to now, numerous efforts have been devoted to applying computer vision technologies to perform automatic crack detection. Previously, traditional edge detection operators are popular approaches to detect cracks on pavement that has stronger edges and lower-level noisy background [1]. For more complicated scenes, such as asphalt surfaces, advanced image analysis techniques [3], [4], [5], [6], [7] have been proposed. However these methods perform crack detection in a style-specific and parameter manually-tuned pattern, their performance is not superior when dealing with cracks with intensity in-homogeneity or complex topology. Owing to lacking robust feature representation and ignoring inter-dependency among cracks, these methods were not general enough. To overcome these shortcomings, CrackForest [8] fuses complementary features of multiple levels to characterize cracks and takes advantage of the contextual information in crack patches. It is shown to achieve SOTA performance, and outperforms CrackTree [9], CrackIT [10], Free-Form Anisotropy (FFA) [11], and Minimal Path Selection (MPS) [12]. However, CrackForest [8] still do crack detection based on manual features, which is somehow not discriminative enough to detect the cracks from complex background with low level cues.

Recently, more and more detection and segmentation works are proposed to find road damages based on deep convolution neural networks (CNN) [13], [14], [15], [16], [17], [18], [19]. Generally, classification of small image patches could not achieve pixel-level accuracy in finding cracks. In medical image segmentation, fully convolutional network and extended fully convolutional encoder-decoder network have been developed for pixel-level segmentation. In essence, crack segmentation is similar to retinal vessel segmentation. For example, UNet [20], [21] and FusionNet [22] usually for medical images segmentation can be used to pixel-level crack segmentation. We find that SegNet [23], UNet [24] and their variants [25] have been introduced to crack segmentation [17]. Both SegNets and UNets adopt an encoder-decoder structure, where the encoder module can obtain high-level semantic features based on a set of convolution and pooling layers, and the decoder part exploits memorized pooling indices or skip connections to re-use high-resolution features of the encoder to recover lost spatial information. Generally, these methods are based on stacked 3×3 convolution and pooling layers,

Manuscript received 15 December 2021; revised 17 August 2022, 14 November 2022, and 23 February 2023; accepted 4 April 2023. Date of publication 26 April 2023; date of current version 30 August 2023. The Associate Editor for this article was S. S. Nedevschi. *(Corresponding authors: Huajun Liu; Hui Kong.)*

Huajun Liu, Jing Yang, and Xiangyu Miao are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: liuhj@njust.edu.cn).

Christoph Mertz is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: cmertz@andrew.cmu.edu).

Hui Kong is with the State Key Laboratory of Internet of Things for Smart City (SKL-IOTSC), the Department of Electromechanical Engineering (EME), and the Department of Computer and Information Science (CIS), University of Macau (UM), Taipa, Macau, China (e-mail: huikong@um.edu.mo).

Digital Object Identifier 10.1109/TITS.2023.3266776

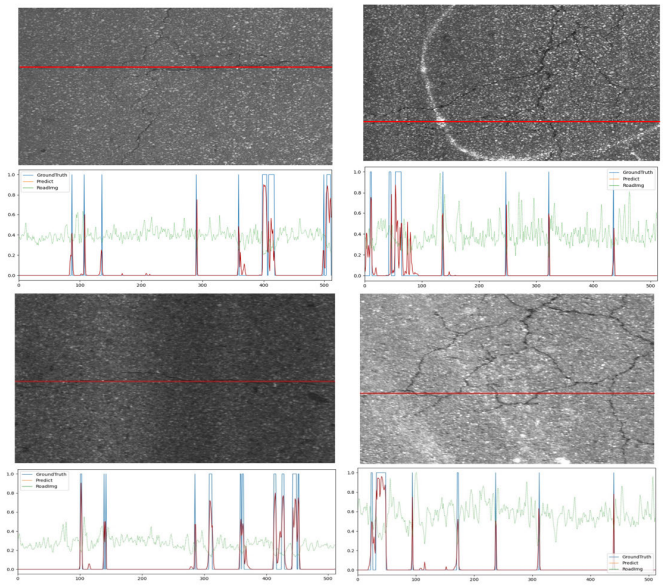

Fig. 1. Visual results of the CrackFormer-II method. The upper part in each example is a classical crack image with a selected profile reference line. The bottom part demonstrates a profile in grey level, the ground truth and the prediction probabilities of crack.

and could not achieve pixel-level precision, resulting in coarse segmentation. Due to the limited receptive field with the small convolutional kernels, they usually fail in segmenting long cracks with discontinuous results.

We notice that the CNNs with pure self-attention modules in Transformers [26], [27], i.e., without skip connections and feed-forward layers after replacing 3×3 convs in Vanilla CNNs models have stronger expressive power on most vision tasks. In particular, by leveraging the cascaded encoder and decoder blocks of specifically stacking self-attention modules, better convergence can be shown than normal CNN models [28] for crack segmentation. CrackFormer-II also adopts a SegNet-like network structure, but introduces attention mechanisms in two different aspects. Fig. 1 shows some results from it. The main contributions of this work can be concluded as follows:

1) We upgrade our earlier work to a new CrackFormer-II by combining novel Transformer encoder block, i.e., local self-attention layers, local feed-forward layers and skip connections, for crack segmentation;
2) We further optimize the channel settings to configure a more efficient CrackFormer-II model which performs better on ODS, OIS, and mIoU but need fewer parameters;
3) CrackFormer-II is a more general pipeline for both fine and coarse crack segmentation and achieves the state-of-the-art (SOTA) performance on several public datasets.

II. Related Work

A. ConvNet Models

Since the asphalt crack detection [16] based on CNN models, some more accurate methods analyze pavement damage using deep neural networks [14], [17], [18], [19], [29]. However, most these CNNs were to classify small image patches and could not achieve pixel-perfect accuracies. In recent years, CNN-based segmentation models provide pixel-level crack segmentation.

For example, Liu et al. [25] proposed a pyramid features aggregation network and a Condition Random Fields (CRFs) post-processing scheme for crack segmentation. Zou et al. [17] provided a multi-stage fusion on the SegNet encoder-decoder architecture for crack segmentation. Yang et al. [18] proposed a feature pyramid and hierarchical boosting network, which integrates context information to low-level features in a feature-pyramid way. Fei et al. [19] proposed the CrackNet-V model, which stacks several 3×3 convolutional layers and a 15×15 convolution kernel for deep abstraction to achieve a high performance for crack segmentation.

Moreover, multi-scale aggregation has been widely applied for image contour analysis. For example, some scale-associated side output methods have been developed for image edge detection or skeleton detection, such as HED [30], DeepSkeleton [31], RCF [32], DeepCrack [17] etc, which achieves better performance on edge detection accuracy and sensitivity. Although these segmentation-based crack detection methods have obtained promising results, they cannot afford to achieve satisfying performance at the pixel-level segmentation precision and result in coarse and discontinuous segmentation.

B. Visual Transformer Models

Vision Transformer (ViT) [33] is the first work to showcase how transformers can 'altogether' replace standard convolutions in deep neural networks on large-scale computer vision datasets. Originally, researchers applied the original transformer blocks on a sequence of image 'patches', and now they are exploring the flexible combination of transformer blocks and convolutions for complex visual tasks. Recent related work focused on designing Transformer pyramid models [34], [35], [36] or Transformer encoder-decoder architectures [37], [38], [39], [40], [41], which has sparked great interest in the computer vision community to dig these models on vision tasks.

Transformer pyramid models are usually for sparse prediction tasks, such as image classification. For instance, the Feature Pyramid Transformer [34] model transforms any feature pyramid into another one using specially designed transformers in self-level, top-down and bottom-up interaction fashion to enrich contexts. The PVT [35] adopts spatial-reduction attention and progressive shrinking pyramid to construct pyramid features with Transformer, which makes PVT flexible to learn multi-scale and high-resolution feature maps. The PVT-v2 [42] further improves the performance by overlapping patch embedding, convolutional feed-forward networks, and linear complexity attention layers. Furthermore, the CMT [36] proposes a new transformer based hybrid network by taking advantage of transformers to capture long-range dependencies, and of CNNs to model local features, and scales it to obtain a family of models, called CMTs, obtaining much better accuracy and efficiency than previous convolution and transformer based models.

Transformer encoder-decoder architectures are popular for dense prediction tasks, such as image segmentation. UNet

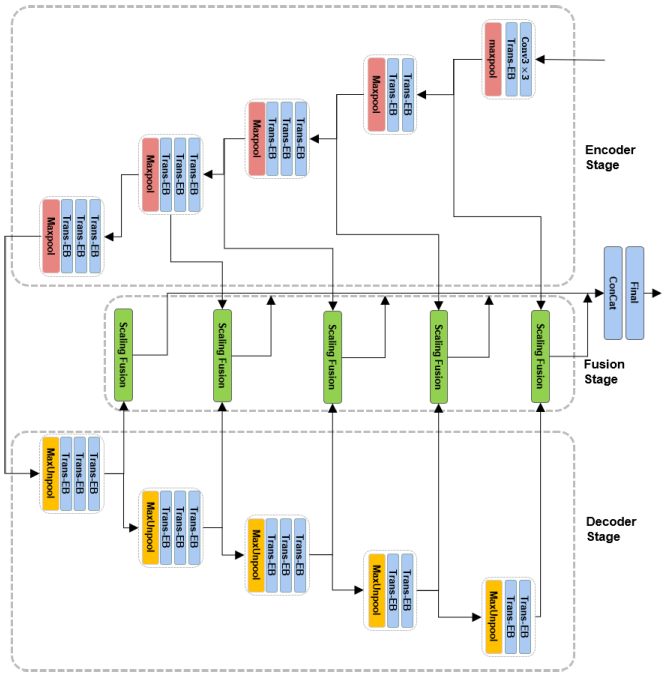

Fig. 2. The structure of CrackFormer-II.

Transformer [37] adopts a U-shaped structure for image segmentation with self- and cross-attention blocks, and it overcomes the problem of UNets in dealing with long-range interactions and spatial dependencies. In TransUNet [38], a transformer encoder block encodes image patches as the input sequence for extracting global contexts, and the decoder upsamples the encoded features, which are combined with the high-resolution CNN feature maps to enable precise localization. Transformer-UNet [39] uses transformer blocks in raw images in UNet, and achieves better segmentation accuracy than many previous UNet variants. DS-TransUNet [40] adopts Swin Transformer to extract coarse- and fine-grained feature representations of different semantic scales. DPT [41], [43] assembles tokens from various stages of the vision transformer into image-like representations at various resolutions and progressively combine them into full-resolution predictions using a convolutional decoder, and uses transformer backbone to process representations at a constant and relatively high resolution and has a global receptive field at every stage.

III. Our Work

A. CrackFormer-II

The CrackFormer-II adopts the basic structure of the SegNet [23] and is an extended version of our earlier work [28] on fine-grained crack segmentation. The flowchart of CrackFormer-II is shown in Fig.2. To establish simultaneous long-range interaction and local-context perception between the low-level feature maps, we propose a novel Transformer encoder block as the basic module. To enhance crack crispness, we propose a scaling fusion block between encoder and the corresponding decoder features to generate attention masks. Additionally, a multi-scale side fusion is exploited to refine multi-scale crack fusion result.

The CrackFormer-II contains three main stages: encoder stage, decoder stage and fusion stage. Firstly, at the start of the model, a stem block consists of a 3×3 conv enriching the image RGB channels to 64 dimensional feature channels. Then at the end of each encoder block, a Max-Pooling is used to downsample the feature maps by one half. At the symmetrical position of decoder, a Max-Unpooling is used to upsample the feature maps by two (Sec. III-B).

The CrackFormer-II's encoder stage consists of 13 feature encoding blocks, and they are deployed in 5 scales according to the layout of $\{2, 2, 3, 3, 3\}$. Meanwhile, the corresponding decoder stage has a symmetrical layout of $\{3, 3, 3, 2, 2\}$. This part in SegNet [23] exploits 3×3 convolution modules at each layer. In contrast, the convolution modules are replaced by Transformer encoder blocks in this work. It is noted that each of these blocks was a pure self-attention module used in our previous work [28], which will be upgraded to a novel bottleneck encoder block consisting of a local self-attention layer and a local feed-forward layer with skip connection in this work (Sec. III-C).

For the corresponding features of each scale of encoder and decoder, a scaling fusion stage is introduced to nonlinearize the salient semantic features and generate attention masks to refine the crack (Sec. III-D).

Finally, the predicted results in all scales are then fused to generate the final result. The predicted results in each scale and fused features are resized to the original dimension of the input image, and the model is supervised by a multi-loss function in training phase.

B. Spatial Domain Downsampling and Upsampling

Similar with the SegNet [23], we utilize the Max-Pooling to downsample and Max-Unpooling to upsample input tensors in the spatial domain at each scale of the encoder part and decoder part, respectively, as shown in Fig. 2. We use a 2D maximum pooling with a 2×2 window and a stride of 2 for tensor down-sampling. For instance, given an input tensor of size $h \times w \times c$, we feed it to the max-pooling layer and return the indices for upsampling. The output size will be $\frac{h}{2} \times \frac{w}{2} \times c$.

Correspondingly, at each scale of the decoder part, a 2D Max-Unpooling with indices is used for tensor upsampling. For instance, given an input tensor of size $h \times w \times c$, we feed it to a max-unpooling layer with a stride of 2 and a kernel size of 2×2, and use the indices in upsampling for tensor alignment. The output size will be $2h \times 2w \times c$.

C. Transformer Block for Feature Encoding and Decoding

The absolute position encoding used in previous transformers, initially designed to leverage the order of tokens, damages translational invariance because it adds unique positional encoding to each patch. Besides, our earlier CrackFormer work [28] ignores the local context and structure information inside the patch. To alleviate these limitations, we propose local self-attention layers and local feed-forward layers. In this work, the Transformer Encoder Block (Trans-EB in short, Fig. 3(b)) in encoder and decoder is upgraded from a stand-alone self-attention layer (Fig. 3(a))) as used in

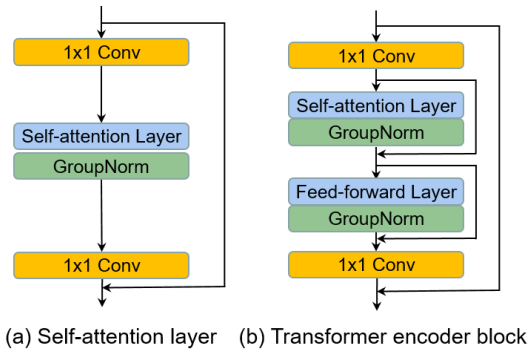

Fig. 3. The Transformer encoder block (Trans-EB).

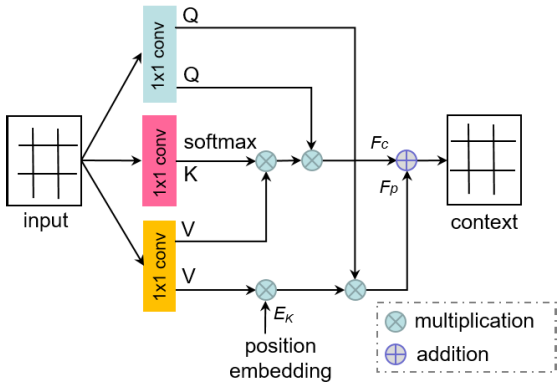

Fig. 4. The local self-attention layer.

the CrackFormer. The upgraded encoder block is a bottleneck module with two 1×1 convolutional blocks, and a local self-attention (LSA) layer (Fig. 4) and a local feed-forward (LFF) layer (Fig. 5) between them. A GroupNorm operation is used after the kernel LSA and LFF block to normalize the feature channels, respectively.

For a segmentation task, given a tensor of size $C \times H \times W$ as input, our transformer encoder block operates on the tensor instead of flattening the spatial 2D dimension to a 1D vector. In stead of using the same channel width in feature encoding as in the CrackFormer, our upgraded Transformer encoder block uses a 1×1 convolution operation to reduce the channels by a ratio of r. Thus, we get a tensor of size $\frac{C}{r} \times H \times W$, and this tensor is fed to our self-defined LSA block and a GroupNorm layer, outputting a new tensor of size $\frac{C}{r} \times H \times W$. The input to LSA and the output from LSA are connected with a skip connection. Subsequently, the output tensor from LSA module is further fed to the LFF block and a GroupNorm layer, and we can get another tensor of size $\frac{C}{r} \times H \times W$. Likewise, the input to LFF and the output from LFF are connected with another skip connection. Finally, the tensor is further expanded by another 1×1 convolution to the same size $C \times H \times W$ of the input tensor, and the third skip connection is also used for these two tensors for fast divergence.

1) Local Self-Attention Layer: The local self-attention layer is a relative position embedded self-attention block, which is shown as Fig. 4. Let $X \in \mathbb{R}^{d_{in} \times WH}$ be an input tensor, where W and H represent the width and height of image and d_{in} denotes the channel of input tensor. This layer applies 1×1 convolutions to generate the keys, queries, and values using Eq. 1, and then to generate global content features $\boldsymbol{F}^{\mathrm{c}} \in \mathbb{R}^{d_{out} \times WH}$ (d_{out} denotes the channel of output features) with long range dependency based on Eq. 2.

$$\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V} = \boldsymbol{\otimes_{1\times1}}(X), \tag{1}$$

$$\boldsymbol{F}^{\mathrm{c}}(X) = \boldsymbol{Q} \otimes \left(\sigma\left(\boldsymbol{K}^{\top}\right) \otimes \boldsymbol{V}\right), \tag{2}$$

where $\boldsymbol{\otimes_{1\times1}}(\cdot)$ is the 1×1 convolution operation and $\otimes$ is a matrix dot-product operation. Let h be the number of head, d_u be the intra-depth dimension, $d_k \times d_u$ and $d_v \times d_u$ be the channel's dimension of tensor K and V, respectively. Then we have $\boldsymbol{Q} \in \mathbb{R}^{d_k \times h \times WH}$, $\boldsymbol{K} \in \mathbb{R}^{d_k \times d_u \times WH}$, and $\boldsymbol{V} \in \mathbb{R}^{d_v \times d_u \times WH}$. We can get the content branch $\boldsymbol{F}^{c} \in \mathbb{R}^{h \times d_v \times HW}$ after self-attention computation in Eq. 2. Actually, we know $d_{out} = h \times d_v$ beforehand. Let σ denote the operation of applying softmax normalization on the tensor. This attention operation can be interpreted as first aggregating the pixel features in $\boldsymbol{V}$ into global context vectors using the weights in $\sigma\left(\boldsymbol{K}^{\top}\right)$, and then redistributing the global context vectors back to individual pixels using the weights in $\boldsymbol{Q}$. We notice its similarity to the one used in Bello [44], but it uses group normalization on queries and values. Softmax normalizing on the keys constrains the output features to be convex combinations of the global context vectors.

The relative position embedding can make the global context vector obtain an effective receptive field in a neighbour region. A relative positional embedded kernel $\boldsymbol{E}_K \in \mathbb{R}^{d_k \times d_u \times 1 \times K \times K}$ is defined as learnable weight parameters, where K indicates the possible relative position indexes within $K \times K$ spatial neighbors. For each pixel, our relative position embedding component attends to operate on its spatial neighbor with sliding $K \times K$ windows. The contextual weight parameter $\boldsymbol{E}_K$ as a learnable convolutional kernel will be embedded to the context vectors through a 3D convolution operation on the tensor $\boldsymbol{V}$.

$$\boldsymbol{F}^{\mathrm{p}}(X) = \boldsymbol{Q} \otimes \left(\widehat{\otimes}_{\boldsymbol{E}_K}(\boldsymbol{V})\right), \tag{3}$$

where $\widehat{\otimes}_{\boldsymbol{E}_K}(\cdot)$ is a 3D convolution operation with $\boldsymbol{E}_K$ as its convolutional kernel. This convolution operation on $\boldsymbol{V}$ $\widehat{\otimes}_{\boldsymbol{E}_K}(\boldsymbol{V})$ will get a tensor of size $d_k \times d_v \times HW$ after tensor reshape. After multiplying with tensor $\boldsymbol{Q}$ in Eq. 3, we can get the relative positional features $\boldsymbol{F}^{\mathrm{p}} \in \mathbb{R}^{h \times d_v \times HW}$.

Finally, the output context vector of the self-attention layer is the element-wise addition of the global content features and relative positional features as Eq. 4.

$$\boldsymbol{LSA}(X) = \boldsymbol{F}^{\mathrm{c}}(X) \oplus \boldsymbol{F}^{\mathrm{p}}(X), \tag{4}$$

where $\oplus$ is the matrix element-wise addition operator. Note that the computational and memory complexities of this layer are $O(N)$ in the number of pixels.

2) Local Feed-Forward Layer: Research indicates that feed-forward networks (FFN) in vision Transformer can slow down the convergence by increasing their Lipschitz constant [45]. But the original FFN is lack of enough information interaction across the patches. To handle this issue, we add a 3×3 depth-wise convolution in between the two point-wise MLPs that form the local FFN in Vision transformer in a

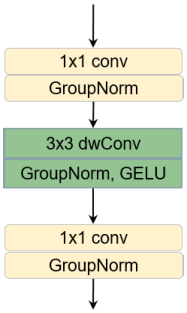

Fig. 5. The local feed-forward layer.

new pipeline as MLP(DWConv(MLP(·))). The Fig. 5 shows a specific structure of how FFN with 3×3 depth-wise convolution updates the 2D input features.

The local feed-forward layer (LFF) is used to extract local information, which is defined as:

$$\boldsymbol{LFF}(X) = \otimes_{\mathbf{1\times1}}(\bar{\otimes}_{\mathbf{3\times3}}(\otimes_{\mathbf{1\times1}}(X))), \tag{5}$$

where $\otimes_{\mathbf{1\times1}}(\cdot)$ is the 1×1 convolution operation and $\bar{\otimes}_{\mathbf{3\times3}}(\cdot)$ is the 3×3 depth-wise convolution operation.

For instance, we take a tensor with the dimension of $C \times H \times W$ as input, after the first 1×1 conv expanding the channels to a ratio of r, we can get a tensor $Cr \times H \times W$, and when passing through the 3×3 depth-wise convolution we can get a tensor with $Cr \times H \times W$ after enough local information interacting. Subsequently, the tensor is reduced to the same dimension as $C \times H \times W$ by another 1×1 conv. There is a GroupNorm after each convolution to normalize the features on channels, and a GELU activation function follows the dwConv subsequently.

D. Scaling Fusion for Feature Refining

The scaling-fusion block (Fig. 6) is inherited from our first CrackFormer version [28]. It is helpful to generate the salient and crisp crack boundary map. Actually, the attention-gate mechanism in Attention UNet [46] inspires that the feature vectors in a specific decoder block can boost segmentation performance by combining the feature vectors in the corresponding encoder block. In essence, the attention-gate mechanism generates an attentive mask, which is normalized to $\alpha_i \in [0, 1]$ by a Sigmoid activation function, and multiplies element-wise those features to be refined. In this way, it acts as a filter to activate some features within a region of interest and simultaneously suppress other irrelevant features. Specifically, at each stage of the CrackFormer, we use features in encoder to generate an attentive mask as attention coefficients and multiply them element-wise with the corresponding features in decoder to active crack features and suppress the non-crack ones.

Let's take the k^{th}-stage fusion as an example, features from the encoder and decoder are $\{X_1^k, X_2^k, X_3^k\}$ and $\{Y_1^k, Y_2^k, Y_3^k\}$, respectively. Based on Fig. 6, the mask is generated according to

$$L_{Mask}^k = \delta\left(BN\left(\otimes_{3\times3}\left(\Gamma\left(X_1^k, X_2^k, X_3^k\right)\right)\right)\right), \tag{6}$$

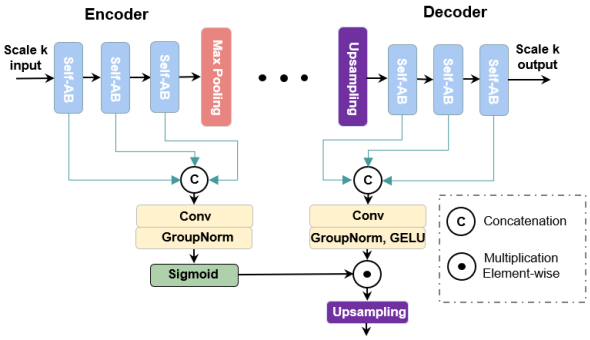

Fig. 6. The scaling-attention block.

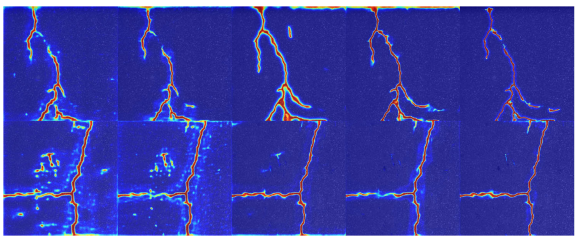

Fig. 7. From left to right: the scaling-attention maps from stage 1 to stage 5, respectively.

where $\Gamma(\cdot)$ denotes a tensor concatenation operation, and $\otimes_{3\times3}(\cdot)$ denotes a 3×3 convolution operation and followed by a BatchNorm BN, and $\delta(\cdot)$ is a Sigmoid activation function. Subsequently, side output of the k^{th} stage is predicted by the scaling-attention mechanism as follows,

$$S_{side}^k = L_{Mask}^k \odot BN\left(\otimes_{3\times3}\left(\Gamma\left(Y_1^k, Y_2^k, Y_3^k\right)\right)\right), \tag{7}$$

where $\odot$ denotes an element-wise multiplication operation. The scaling-attention maps at each stage are visualised as Fig. 7, which is an attention coefficient mask from high-level features. Around the semantic crack, there is a stronger response. From the output features of different stages, we can see a coarse to fine semantic crack response, which can be used to refine more crisp boundary.

After the feature in each stage is upsampled in order to make its dimension the same as that of the input image, we get five predicted results $S_{side}^k, k = 1, 2, \cdots, 5$, which are concatenated and fused to generate the final output S_{fuse} like the HED [30], RCF [32] and DeepCrack [17] etc. Finally, all sides and fused output are fully supervised by the crack ground truth labels.

E. Loss Function

The balanced weight cross-entropy loss function used in the CrackFormer network [28] is adopted for training.

IV. Implementation Details

A. Definition Details

In Table I, we introduced the detailed definition of our CrackFormer-II model. Specifically, we listed the detailed definition of each block about its input size ("In. S." in short), output size ("Out. S." in short), input channel ("In. Ch." in

TABLE I
CRACKFORMER-II DEFINITION DETAILS

Stage	Scale	Block Name	In. S.	Out. S.	In. Ch.	Out. Ch.	Paras (K, r)
Encoder	Scale 1	Conv3×3	$H \times W$	$H \times W$	3	64	-
		Trans-EB	$H \times W$	$H \times W$	64	64	(7, 4)
		Max-Pool	$H \times W$	$\frac{H}{2} \times \frac{W}{2}$	-	-	-
	Scale 2	Trans-EB	$\frac{H}{2} \times \frac{W}{2}$	$\frac{H}{2} \times \frac{W}{2}$	64	128	(7, 4)
		Trans-EB	$\frac{H}{2} \times \frac{W}{2}$	$\frac{H}{2} \times \frac{W}{2}$	128	128	(7, 4)
		Max-Pool	$\frac{H}{2} \times \frac{W}{2}$	$\frac{H}{4} \times \frac{W}{4}$	-	-	-
	Scale 3	Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	128	256	(7, 4)
		Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	256	256	(7, 4)
		Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	256	256	(7, 4)
		Max-Pool	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{8} \times \frac{W}{8}$	-	-	-
	Scale 4	Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{8} \times \frac{W}{8}$	256	512	(7, 4)
		Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{8} \times \frac{W}{8}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{8} \times \frac{W}{8}$	512	512	(7, 4)
		Max-Pool	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{16} \times \frac{W}{16}$	-	-	-
	Scale 5	Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
		Max-Pool	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{32}$	-	-	-
Decoder	Scale 5	Max-UnPool	$\frac{H}{32} \times \frac{W}{32}$	$\frac{H}{16} \times \frac{W}{16}$	-	-	-
		Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{16} \times \frac{W}{16}$	512	512	(7, 4)
	Scale 4	Max-UnPool	$\frac{H}{16} \times \frac{W}{16}$	$\frac{H}{8} \times \frac{W}{8}$	-	-	-
		Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{8} \times \frac{W}{8}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{8} \times \frac{W}{8}$	512	512	(7, 4)
		Trans-EB	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{16} \times \frac{W}{16}$	512	256	(7, 4)
	Scale 3	Max-UnPool	$\frac{H}{8} \times \frac{W}{8}$	$\frac{H}{4} \times \frac{W}{4}$	-	-	-
		Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	256	256	(7, 4)
		Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	256	256	(7, 4)
		Trans-EB	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{4} \times \frac{W}{4}$	256	128	(7, 4)
	Scale 2	Max-UnPool	$\frac{H}{4} \times \frac{W}{4}$	$\frac{H}{2} \times \frac{W}{2}$	-	-	-
		Trans-EB	$\frac{H}{2} \times \frac{W}{2}$	$\frac{H}{2} \times \frac{W}{2}$	128	128	(7, 4)
		Trans-EB	$\frac{H}{2} \times \frac{W}{2}$	$\frac{H}{2} \times \frac{W}{2}$	128	64	(7, 4)
	Scale 1	Max-UnPool	$\frac{H}{2} \times \frac{W}{2}$	$H \times W$	-	-	-
		Trans-EB	$H \times W$	$H \times W$	64	64	(7, 4)
		Trans-EB	$H \times W$	$H \times W$	64	64	(7, 4)
Fusion	Scale 5	Scaling-Fusion	$\frac{H}{16} \times \frac{W}{16}$	$H \times W$	512+512	64	-
	Scale 4	Scaling-Fusion	$\frac{H}{8} \times \frac{W}{8}$	$H \times W$	512+256	64	-
	Scale 3	Scaling-Fusion	$\frac{H}{4} \times \frac{W}{4}$	$H \times W$	256+128	64	-
	Scale 2	Scaling-Fusion	$\frac{H}{2} \times \frac{W}{2}$	$H \times W$	128+64	64	-
	Scale 1	Scaling-Fusion	$H \times W$	$H \times W$	64+64	64	-
Concat		Conv3×3	$H \times W$	$H \times W$	64	5	-
Final		Conv1×1	$H \times W$	$H \times W$	5	1	-

short), output channel ("Out. Ch." in short) and corresponding parameters at each scale in the encoder, decoder and fusion stage, supposing that the size of the input image is $H \times W$. Different with the previous version, the channel reduction ratio r is set to 4 according to the bottleneck pipeline, which has been shown effective to reduce parameters while maintaining performance. The relative position index K is set to 7 in local self-attention layer.

B. Data Augmentation

The training set was conducted data augmentation by clipping, flipping, and rotation operations. To avoid generating same images after rotating and flipping transformations, the rotating angles are set as 19, 23, 90 degrees and the croppings are with inner oriented rectangles. The images are horizontally flipped and finally applied by two Gamma corrections (0.3030, 0.6060) to reduce the influence of brightness. In the end, each training set was expanded by 228 times of the original samples.

C. Training & Validation Parameters

To improve the robustness of the model, the images in the training set hold its original dimension and have not been resized. The *BatchSize* in the experiment is set to 1, and the *Shuffle* strategy is set True. We choose the *Stochastic Gradient Descent (SGD)* as optimizer and set the MOMENTUM to 0.9.

Due to the data augmentation, the total training epoch is set to 500 and the initial learning rate is set to 1e-3. We adopt the *StepLR* strategy to adjust the learning rate at epoch 20, 50 and 100. At each epoch milestone, the learning rate will decay 1/10 times of the previous one.

V. EXPERIMENTS

A. Datasets

Our model is trained and evaluated on four public benchmarks, the CrackTree260, CrackLS315, Stone331 and DeepCrack537. These datasets cover diverse types of cracks, which perhaps appear in asphalt or concrete pavement and are with bare type, rough type or dirty type. These datasets cover fine-level cracks (the width of most cracks only 1 pixel or few ones, mainly in CrackTree260, CrackLS315, Stone331) and coarse-level cracks (the width of cracks reaching 180 pixels, mainly in DeepCrack537).

The CrackTree260 [9] contains 260 road pavement images. These pavement images are captured by an area-array camera under visible-light illumination, and the size of each sample is 800×600. 200 samples are chosen for training, 20 samples for validation and 40 samples for testing.

The CrackLS315 [17] contains 315 images of asphalt pavement captured under laser illumination by a line-array camera. Each image has a size of 512×512. Among them, 265 samples are selected for training, and the remaining 10 samples for validation and 40 samples for testing.

The Stone331 [47] contains 331 images of the stone surface, captured by an area-array camera under visible-light illumination. Original image size is 1024×1024, because of the irregularity of cutting surface, original images are center-cropped to 512×512 clipped samples. 261 images of them are chosen for training, 20 for validation and 50 for testing.

The DeepCrack537 [25] contains 537 pictures, each of them has a size of 544×384 pixels. The pictures contain various cracks. Compared with the CrackLS315 data set, the cracks in this data set are significantly diverse, for instance with top-down view, or with tilted view, and with concrete and asphalt surface, with wide and thin crack, and some of them are even partly occluded. We choose 300 of them for training, 237 of them are used as the testing set.

B. Performance Metrics

The performance metrics of $Precision$ (abbre. as PR) and $Recall$ (abbre. as RE) are calculated as $PR = \frac{TP}{TP+FP}$ and $RE = \frac{TP}{TP+FN}$ for binary classification tasks.

Specifically, for each image, PR and RE can be calculated by comparing the detected cracks against the human annotated ground truth. Then, the F-measure ($\frac{2 \cdot PR \cdot RE}{PR+RE}$) can be computed as an overall metric for performance evaluation. Specifically, Precision (P), Recall (R) and three different F-measure-based metrics are employed in the evaluation, the best F-measure on the data set for a fixed threshold - Optimal Dataset Scale

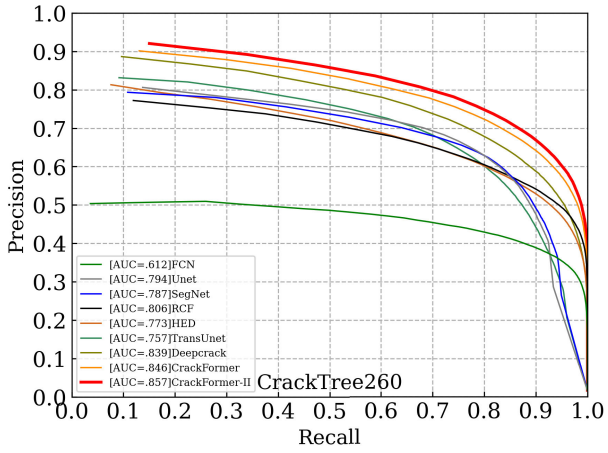

Fig. 8. PR curves on the CrackTree260.

(ODS), the aggregate F-measure on the data set for the best threshold on each image - Optimal Image Scale (OIS), the average precision (AP) [17], [28] and the mean intersection over union (mIoU) [48]. Moreover, the efficiency in terms of frames per second (FPS) is also used for inference speed evaluation, and all models are evaluated on Nvidia RTX TITAN GPUs.

C. Comparison With the SOTA Methods

To evaluate our model's performance, some classical models, such as the SE [49], FCN [50], HED [30], RCF [32], SegNet [23], SRN [51], UNet [24], FPHBN [18], TransUNet [38], DeepCrack [17] are compared with ours on crack detection task. The SE [49] is a classical method based on random decision forest used for edge detection. The HED [30] is a model based on the VGG16, whose feature maps are generated at each stage of the VGG16 and aggregated for multi-stage fusion. The RCF [32] and SRN [51] are similar with the HED, which is an extension of the HED. The SegNet [23] and UNet [24] are encoder and decoder architecture with symmetrical structures. The DeepCrack [17] is an extension to the SegNet for crack detection.

1) The Results on the CrackTree260: The CrackTree260 is a thin crack dataset labeled with a single pixel width or extremely tiny edges. On the asphalt surface and under visible-light illumination, the crack exhibits extreme weak contrast between the "crack" and "non-crack" pixels.

From the precision-recall curves and area under curve (AUC) values in Fig. 8, CrackFormer-II is the highest among all compared methods. Since the output of SE [49] is a binary map, we cannot compute the PR curve and just list the ODS, OIS and AP value in related tables. From the statistical performance in Tab. II, it can be seen that our proposed transformer models: CrackFormer, CrackFormer-II outperform the compared SOTA CNN methods on the CrackTree260, and CrackFormer achieved F-measure with 0.881 on ODS, 0.883 on OIS, 0.887 on mIoU and 0.896 on AP, and 0.892 on Precision and 0.870 on Recall, and CrackFormer-II further achieved F-measure with 0.897 on ODS, 0.912 on OIS, 0.896 on mIoU and 0.906 on AP, and 0.914 on Precision and 0.881 on Recall, respectively. We obtain a gain of 4.5% on ODS, 4.8% on OIS, 3.1% on mIoU and 3.1% on AP, respectively, compared with the DeepCrack. Meanwhile, the CrackFormer-II works at 16 FPS with an 800×600 image as input, which is comparable to most CNN models. Visualized results in Fig. 9 show that the CrackFormer-II's results (seen as the third row) are more continuous and crisp than the compared deep learning models. The crack profile shows that the CrackFormer-II can achieve higher prediction accuracy even for cracks with one-pixel width or tiny edge.

TABLE II
PERFORMANCE ON THE CRACKTREE260

Model	P	R	ODS	OIS	mIoU	AP	FPS
SE [49]	0.689	0.637	0.662	0.673	-	0.683	5
FPHBN [18]	0.553	0.485	0.517	0.579	-	-	12
FCN8s [50]	0.519	0.418	0.463	0.520	0.612	0.401	5
TransUNet [38]	0.797	0.746	0.771	0.781	0.803	0.776	11
SRN [51]	0.801	0.748	0.774	0.781	-	0.779	18
HED [30]	0.837	0.796	0.816	0.820	0.836	0.831	21
SegNet [23]	0.851	0.837	0.844	0.851	0.858	0.862	10
UNet [24]	0.860	0.834	0.847	0.832	0.861	0.869	11
RCF [32]	0.869	0.845	0.857	0.863	0.868	0.861	18
DeepCrack [17]	0.871	0.834	0.852	0.864	0.865	0.875	12
CrackFormer [28]	0.892	0.870	0.881	0.883	0.887	0.896	12
CrackFormer-II	**0.914**	**0.881**	**0.897**	**0.912**	**0.896**	**0.906**	16

TABLE III
PERFORMANCE ON THE CRACKLS315

Model	P	R	ODS	OIS	mIoU	AP	FPS
SE [49]	0.502	0.423	0.459	0.521	-	0.495	8
FCN8s [50]	0.515	0.426	0.466	0.472	0.615	0.325	8
UNet [24]	0.703	0.643	0.672	0.703	0.736	0.740	15
SRN [51]	0.769	0.742	0.755	0.789	-	0.795	20
SegNet [23]	0.782	0.741	0.761	0.780	0.796	0.780	14
HED [30]	0.778	0.748	0.763	0.798	0.798	0.829	26
RCF [32]	0.802	0.773	0.788	0.816	0.815	0.829	22
TransUNet [38]	0.871	0.852	0.861	0.859	0.861	0.881	15
DeepCrack [17]	0.869	0.837	0.853	0.867	0.866	0.877	16
CrackFormer [28]	0.884	0.858	0.871	0.879	0.884	**0.883**	15
CrackFormer-II	**0.912**	**0.863**	**0.887**	**0.908**	**0.895**	0.865	20

2) The Results on the CrackLS315: The images of this dataset are captured under laser illumination. The training on this dataset is more difficult than on the other datasets because of the extreme low contrast. The precision-recall curves are shown in Fig. 10 and it can be seen that CrackFormer-II achieved the highest AUC value.

It can be seen from Tab. III that the CrackFormer-II achieves the best performance on the CrackLS315. CrackFormer obtains a gain of 1.8% on ODS, 1.2% on OIS, 1.8% on mIoU, 0.6% on AP, and CrackFormer-II obtains a gain of 5.5% on ODS, 3.0% on OIS, 2.9% on mIoU, respectively, compared with the DeepCrack. Moreover, CrackFormer-II achieves higher Precision and Recall performance than most previous models. On CrackLS315, CrackFormer-II works at 20 FPS with an 512×512 image as input, which is comparable to most classical CNN models. The ODS of the HED, SRN, SegNet and UNet, is 14.5%, 15.3%, 14.7% and 23.6% lower than the

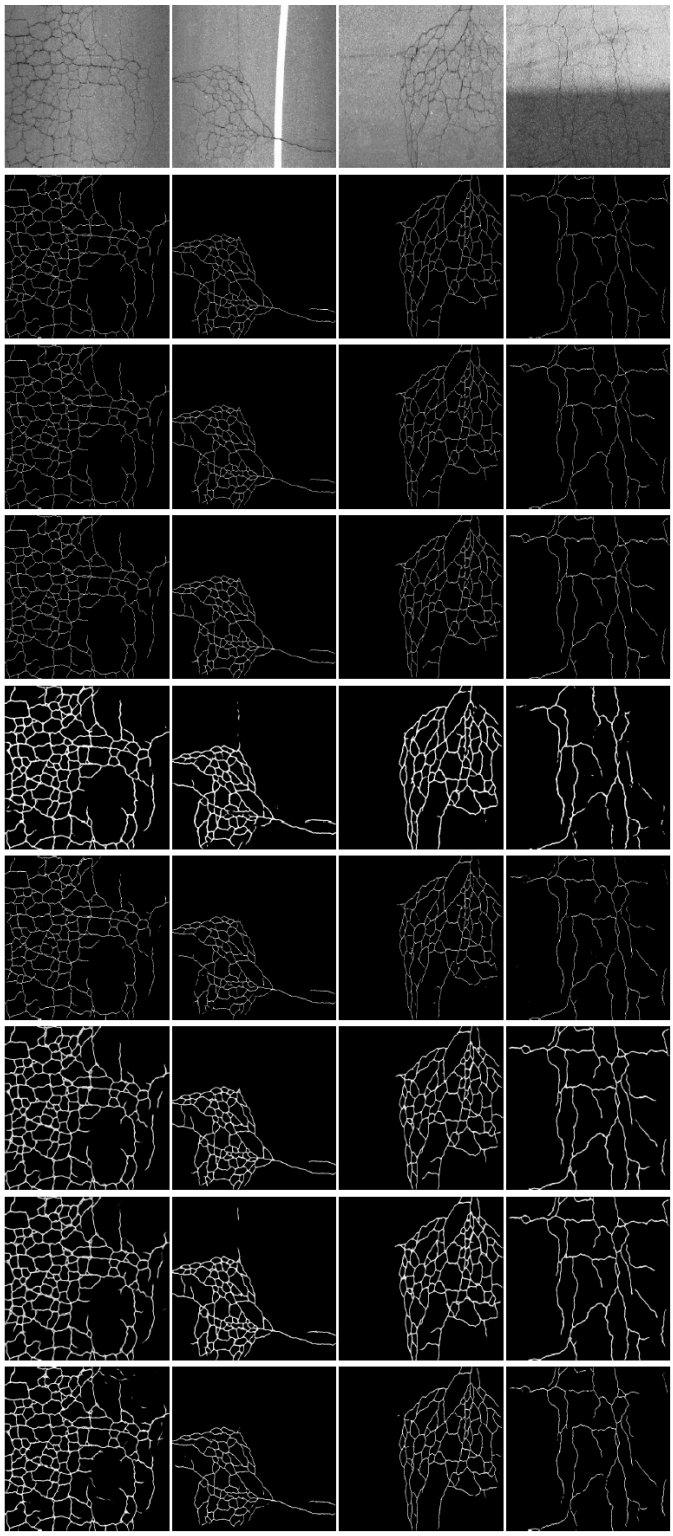

Fig. 9. Results on the CrackTree260. From the top row to the bottom, showing the original image, ground truth and inference results from our method, DeepCrack [17], SegNet [23], TransUNet [38], RCF [32], HED [30] and Unet [24].

CrackFormer-II, respectively. Compared with the method SE, the CrackFormer-II obtains an improvement of 42.8% in terms of ODS. The HED, SRN, RCF and SegNet show comparable results, while the CrackFormer-II has better performance than these methods. Visualized results in Fig. 11 (seen as the third row) show that the CrackFormer-II can predict more detailed thin crack from low contrast asphalt pavements. From Fig. 11, we note that CrackFormer-II has more accurate and continuous prediction results on the thin dataset than the other methods.

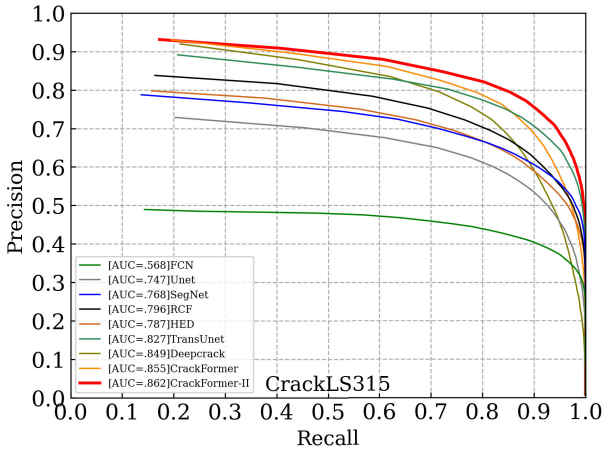

Fig. 10. PR curves on the CrackLS315.

3) The Results on the Stone331: This dataset is from stone cutting surface and its smooth surface makes the crack texture too weak to be observed even by human eyes. The visualized results in Fig. 12 (seen as the third row) show that the CrackFormer-II can predict the most continuous and complete crack detection results. It can be seen from precision-recall curves in Fig. 13, the CrackFormer-II outperforms the other compared methods and achieves the highest AUC value.

From statistical performance in Tab. IV, the CrackFormer-II performs best on the Precision and Recall metrics, and achieves an ODS of 0.912, 0.887 OIS, 0.897 on mIoU and 0.906 AP, respectively, on the test dataset. The CrackFormer-II obtains a gain of 3.1% on ODS, 3.7% on OIS, 2.1% on mIoU and 2.8% on AP, respectively, compared with the DeepCrack. Moreover, the CrackFormer-II obtains a gain of 1.0% on ODS, 0.27% on OIS and 0.12% on AP, respectively, and works at higher FPS compared with the CrackFormer. Compared with the mainstream deep learning models, it outperforms by 11.8%, 12.3%, 15.5% and 17.7% on ODS over the SegNet, RCF, UNet and SRN respectively. Compared with traditional method SE, the CrackFormer-II obtains an improvement of 35.5% in terms of ODS.

4) The Results on the DeepCrack Benchmark: This dataset is created from crack images with manually annotated segmentations in various scenes and scales to universally represent the characteristics of cracks, which consists of bare type, rough type and dirty type, and covers two major scenes, asphalt and concrete, which are commonly used in man-made buildings. The width of cracks is in ranges from 1 pixel to 180 ones in the database. The visualized inference results in Fig. 14 (seen as the third row) show that the CrackFormer-II can predict the most continuous and complete crack detection results. It can be seen from precision-recall curves in Fig. 15, the CrackFormer-II outperforms the other compared methods.

From statistical performance in Tab. V, the CrackFormer-II achieves an ODS of 0.869, 0.881 OIS and 0.947 AP, respectively, on the DeepCrack537 testing dataset. The

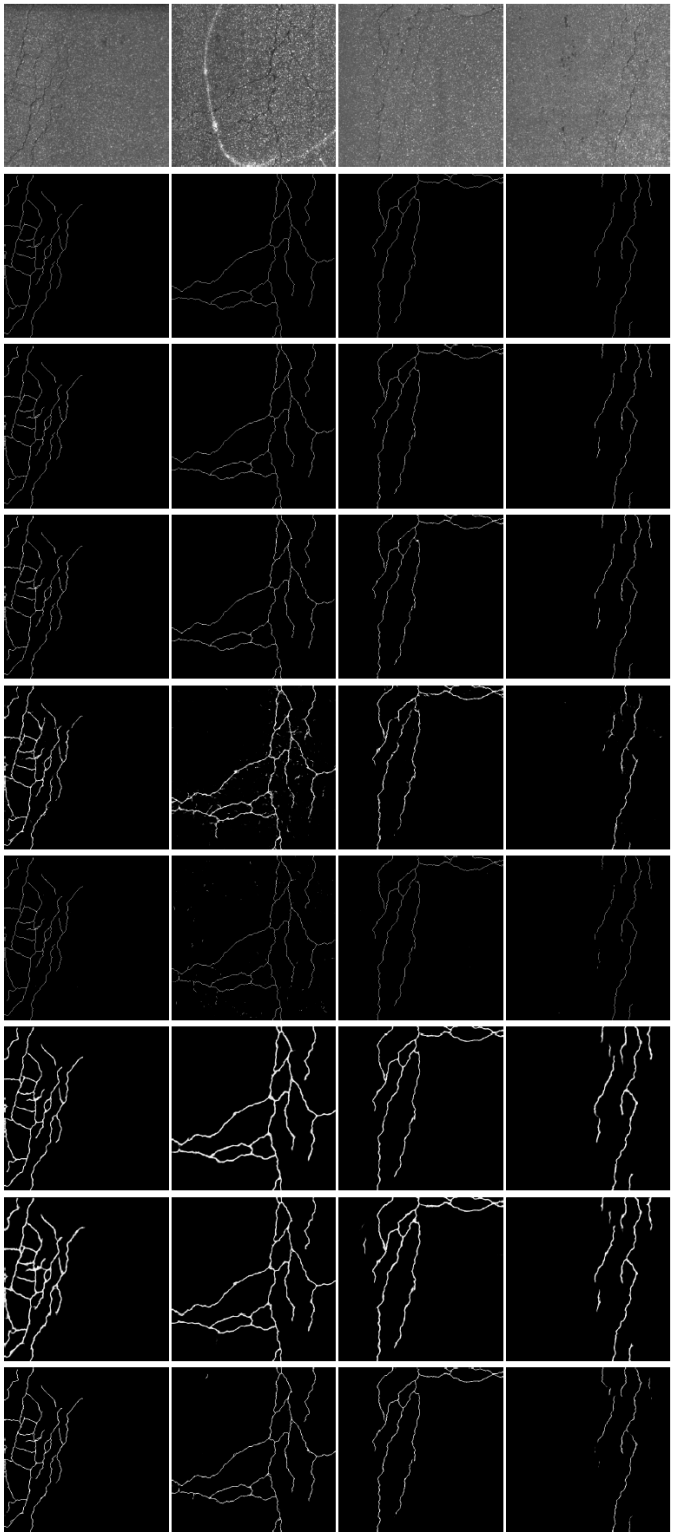

Fig. 11. Results on the CrackLS315. From the top row to the bottom, showing the original image, ground truth and inference results from our method, DeepCrack [17], SegNet [23], TransUNet [38], RCF [32], HED [30] and Unet [24].

CrackFormer-II obtains a gain of 2.2% on ODS, 1.1% on OIS, 1.8% on mIoU and 1.1% on AP, respectively, compared with the DeepCrack. And CrackFormer-II works at 21 FPS with an 544×384 image as input, which is potentially applicable in practical scenes. Compared with the mainstream deep learning models, it outperforms by 2.2%, 2.9%, 3.5%, 4.5% and 5.4% on F-measure on ODS over the DeepCrack, SegNet, RCF, HED and UNet, respectively.

TABLE IV
PERFORMANCE ON THE STONE331

Model	P	R	ODS	OIS	mIoU	AP	FPS
SE [49]	0.623	0.503	0.557	0.623	-	0.605	8
FCN8s [50]	0.699	0.664	0.681	0.653	0.645	0.548	8
HED [30]	0.732	0.706	0.719	0.763	0.749	0.758	26
SRN [51]	0.778	0.697	0.735	0.776	-	0.741	20
UNet [24]	0.793	0.724	0.757	0.776	0.778	0.809	15
RCF [32]	0.791	0.787	0.789	0.829	0.821	0.820	22
SegNet [23]	0.806	0.782	0.794	0.815	0.837	0.831	14
TransUNet [38]	0.882	0.856	0.869	0.871	0.871	0.863	15
DeepCrack [17]	0.861	0.851	0.856	0.875	0.876	0.878	16
CrackFormer [28]	0.891	0.863	0.877	0.885	0.885	0.894	15
CrackFormer-II	**0.902**	**0.872**	**0.887**	**0.912**	**0.897**	**0.906**	20

TABLE V
PERFORMANCE ON THE DEEPCRACK BENCHMARK

Model	P	R	ODS	OIS	mIoU	AP	FPS
FCN8s [50]	0.829	0.796	0.812	0.817	0.833	0.936	9
UNet [24]	0.841	0.791	0.815	0.839	0.837	0.942	16
SRN [51]	0.835	0.806	0.820	0.825	-	-	24
HED [30]	0.839	0.810	0.824	0.847	0.840	0.949	28
RCF [32]	0.845	0.823	0.834	0.863	0.843	0.938	24
SegNet [23]	0.857	0.824	0.840	0.852	0.851	**0.956**	15
DeepCrack [17]	0.876	0.819	0.847	0.870	0.861	0.936	17
CrackFormer-II	**0.905**	**0.837**	**0.876**	**0.883**	**0.879**	0.947	21

TABLE VI
ABLATION STUDY ON THE CRACKLS315

Model	LFF	LSA	Scal-FU	MSF	ODS ↑	OIS ↑
SegNet					0.761	0.780
DeepCrack				✓	0.853	0.867
-		✓			0.859	0.869
-			✓		0.858	0.869
-		✓		✓	0.864	0.870
-			✓	✓	0.862	0.872
CrackFormer		✓	✓	✓	0.871	0.879
CrackFormer-II	✓	✓	✓	✓	0.908	0.897

D. Ablation Study

To further check the gain of each module of our model, ablation study is done on the CrackLS315. The experimental results are shown in Tab. VI. We first select the SegNet as the baseline. After the conv 3×3 is replaced by the LSA 0.9% and 0.5% on ODS and OIS, respectively, indicating that the model works better on multi-scale fusion architecture as well.

Moreover, the LSA and Scal-FU (Scaling Fusion) modules further achieve a gain of $0.7\% - 0.9\%$ and $0.7\% - 0.9\%$ on ODS and OIS, respectively, indicating that the two attention mechanisms are compatible in crack detection task.

Finally, the CrackFormer-II upgraded by the LFF further obtained a gain of 3.7% and 1.8% on ODS and OIS, respectively, indicating the LFF is valuable for crack detection task.

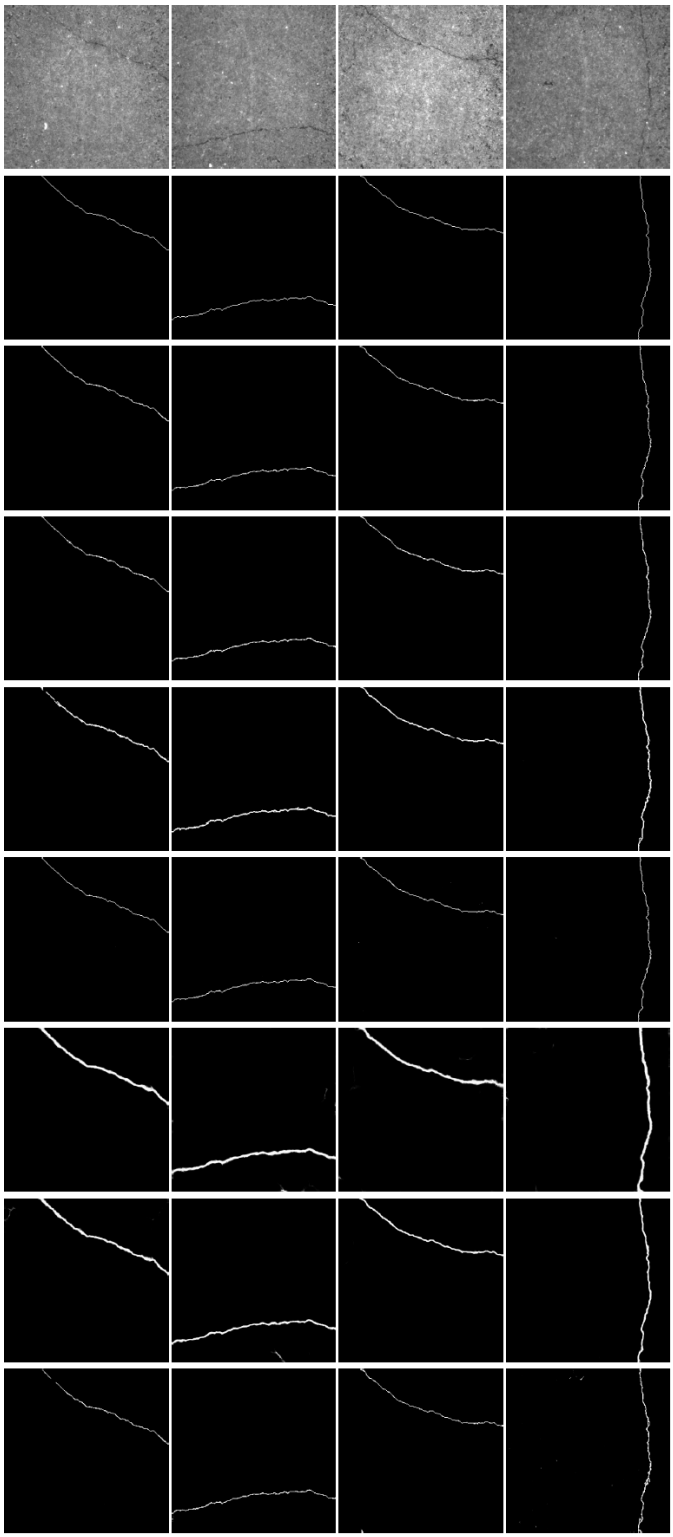

Fig. 12. Results on the Stone331. From the top row to the bottom, showing the original image, ground truth and inference results from our method, DeepCrack [17], SegNet [23], TransUNet [38], RCF [32], HED [30] and Unet [24].

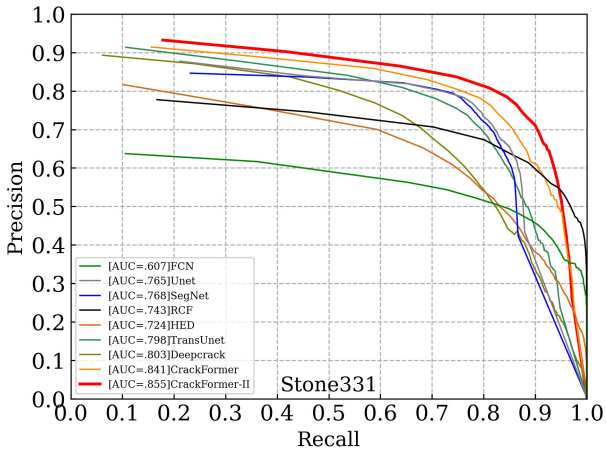

Fig. 13. PR curves on the Stone331.

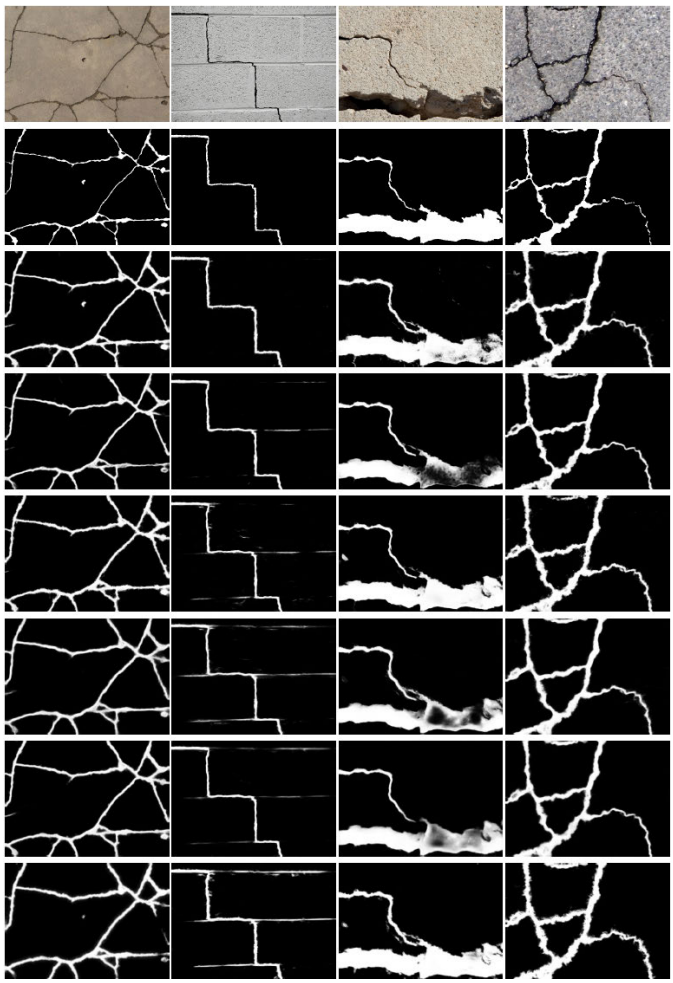

Fig. 14. Results on the DeepCrack537. From the top row to the bottom, showing the original image, ground truth and inference results from our method, DeepCrack [17], SegNet [23], RCF [32], HED [30] and Unet [24].

E. Hyper-Parameters Analysis

Scaling the hyper-parameters of deep learning models aims to balance model's parameter and computational burden with its performance. The hyper-parameter scaling analysis to search the balance of CrackFormer-II's performance and its parameters is conducted in our work. In CrackFormer-II, two hyper-parameters are essential to us, one is the relative position index (marked as "K" in Table I) in local self-attention layer, which can be selected from 23, 15, 11 to 7, and the other is the channel reduction ratio (marked as "r" in Table I), which can

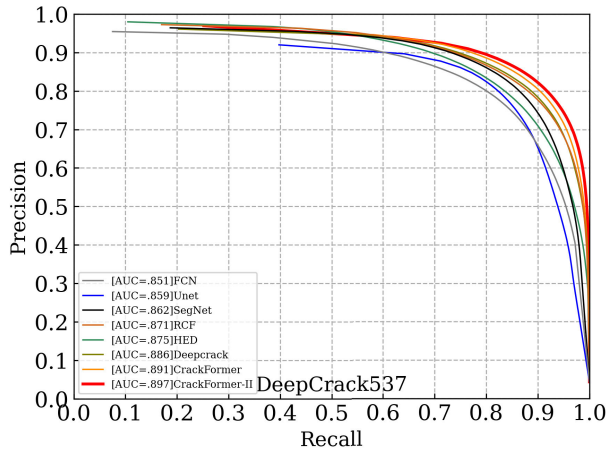

Fig. 15. PR curves on the DeepCrack537.

TABLE VII
HYPER-PARAMETERS ANALYSIS ON THE CRACKLS315

K	r	ODS ↑	OIS ↑	FLOPs ↓	Paras ↓
23	1	0.916	0.904	137.8G	13.5M
15	1	0.912	0.899	137.8G	13.36M
11	1	0.905	0.894	137.8G	13.32M
7	1	0.913	0.900	137.8G	13.29M
23	4	0.912	0.899	82.7G	5.16M
15	4	0.905	0.894	82.7G	5.03M
11	4	0.901	0.889	82.7G	4.99M
7	**4**	**0.908**	**0.897**	**82.7G**	**4.96M**

be selected from 1 and 4 in the Trans-EB bottleneck module. The hyper-parameters analysis results can be summarized in Table VII.

On the CrackFormer-II, we are scaling the model by selecting different hyper-parameters, such as relative position indexes and channel reduction ratios to balance the ODS F-measure performance and learnable parameters/computational burden, Finally, In the released version of CrackFormer-II, the relative position index and channel reduction ratio are recommended to use 7 and 4 respectively, where we can obtain an acceptable performance with lower computational burden and fewer parameters.

F. Efficiency Analysis

The FPS evaluation on the compared models are shown from Tab. II to Tab. V with different inference image sizes (600 × 800, 512 × 512, 512 × 512 and 544 × 384, respectively). It can be seen that the CrackFormer-II is a more parameter-efficient model after channel scaling. Specifically, the CrackFormer achieves higher accuracy than the the DeepCrack [17] with 8.1× fewer FLOPs and 4.2× fewer parameters, and CrackFormer-II achieves higher accuracy than the the DeepCrack [17] with 6.7× fewer FLOPs and 6.2× fewer parameters. Compared to other classical models, such as UNet [24], SRN [51], HED [30], RCF [32] and SegNet [23], the CrackFormer-II achieves a higher ODS value, 2× to 3× faster in average and 4× to 6× fewer parameters. Compared with the CrackFormer model [28], CrackFormer-II achieves higher mIoU performance, and higher FPS as well, but needs fewer parameters.

TABLE VIII
THE PERFORMANCE EFFICIENCY ANALYSIS ON STONE331 DATASET

Model	mIoU↑	FLOPs↓	Params↓	FPS↑
FCN8s [50]	0.645	190.1G	134.3M	8
UNet [24]	0.778	219.6G	34.5M	15
HED [30]	0.749	80.3G	14.7M	26
RCF [32]	0.821	102.7G	14.8M	22
SegNet [23]	0.837	170.1G	29.5M	14
DeepCrack [17]	0.876	547.4G	30.9M	16
CrackFormer [28]	0.885	67.2G	7.35M	15
CrackFormer-II	0.897	82.7G	4.96M	20

TABLE IX
MULTI-THE SCALE ANALYSIS ON THE THREE DATASETS

Scale	CrackTree260		CrackLS315		Stone331	
	ODS↑	OIS↑	ODS ↑	OIS↑	ODS↑	OIS↑
S1	0.680	0.702	0.648	0.671	0.760	0.771
S2	0.709	0.723	0.691	0.632	0.769	0.775
S3	0.740	0.742	0.746	0.652	0.779	0.812
S4	0.756	0.761	0.755	0.661	0.796	0.821
S5	0.799	0.801	0.761	0.665	0.809	0.815
S1+S2	0.768	0.772	0.735	0.742	0.815	0.820
S1+S2+S3	0.802	0.818	0.809	0.821	0.821	0.823
S1+S2+S3+S4	0.854	0.857	0.828	0.835	0.851	0.867
Fused	**0.881**	**0.883**	**0.871**	**0.879**	**0.877**	**0.883**

G. Multi-Scale Analysis

The multi-scale fusion scheme has proven to be an effective way in both Transformer-based models to enhance crack detection performance [2]. In fact, because crack images exhibit different characteristics at different scales. At a large-scale stage, crack detection is reliable, but its localization is poor and may miss thin cracks. At a small-scale stage, details are preserved, but detection suffers a lot from clutters in background texture. Therefore, we quantitatively analyze output of different-scale stage and scale-wise fusion performance in CrackFormer model on the three fine-grained datasets. The statistical results are shown in Tab. IX. Overall, the F-measure on ODS and OIS values increase step by step from stage S1 to S5, and we obtain a 9.4% ODS gain in average. This means that the output of the CracnFormer and CrackFormer-II from coarse to fine scale (stage) gradually matches the true scale of this kind of thin crack benchmark. From the viewpoint of multi scale fusion, it can be found that the incremental fusion experiments from S1+S2 to S1+S2+S3+S4, or even to all scale fusion (S1+S2+S3+S4+S5) could increase the ODS and OIS values over the output of each single scale. Moreover, the final fused results can further obtain the ODS gain over the finest scale (S5) by 8.7% in average.

VI. CONCLUSION

The proposed CrackFormer-II aims at detecting challenging pavement cracks. We derive our model from the SegNet basic architecture and novel attention mechanisms. The proposed

self-attention modules are embedded in the encoder-decoder blocks, where the 1×1 convolutional kernels are adopted for extracting contextual information across feature-channels, and efficient positional embedding to capture large receptive field spatial contextual information for long range interactions. The proposed scaling-attention modules combine output from the corresponding encoder and decoder, and enable to obtain crisp crack boundary. On four classical crack segmentation benchmark datasets, the CrackTree260, CrackLS315, Stone331 and DeepCrack537, we can obtain pixel-level crack segmentation accuracy on these fine and coarse-level datasets and achieve SOTA performance.

REFERENCES

[1] A. Mohan and S. Poobal, "Crack detection using image processing: A critical review and analysis," *Alexandria Eng. J.*, vol. 57, no. 2, pp. 787–798, 2017.

[2] 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.

[3] Y. Noh, D. Koo, Y.-M. Kang, D. Park, and D. Lee, "Automatic crack detection on concrete images using segmentation via fuzzy C-means clustering," in *Proc. Int. Conf. Appl. Syst. Innov. (ICASI)*, May 2017, pp. 877–880.

[4] T. H. Dinh, Q. P. Ha, and H. M. La, "Computer vision-based method for concrete crack detection," in *Proc. 14th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Nov. 2016, pp. 1–6.

[5] Y. Sato, Y. Bao, and Y. Koya, "Crack detection on concrete surfaces using V-shaped features," *World Comput. Sci. Inf. Technol. J.*, vol. 8, no. 1, pp. 1–6, 2018.

[6] P. Prasanna et al., "Automated crack detection on concrete bridges," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 591–599, Apr. 2014.

[7] R. S. Adhikari, O. Moselhi, and A. Bagchi, "Image-based retrieval of concrete crack properties for bridge inspection," *Autom. Construct.*, vol. 39, pp. 180–194, Apr. 2014.

[8] 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.

[9] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, "CrackTree: Automatic crack detection from pavement images," *Pattern Recognit. Lett.*, vol. 33, no. 3, pp. 227–238, 2012.

[10] 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.

[11] T. S. Nguyen, S. Begot, F. Duculty, and M. Avila, "Free-form anisotropy: A new method for crack detection on pavement surface images," in *Proc. 18th IEEE Int. Conf. Image Process.*, Sep. 2011, pp. 1069–1072.

[12] R. Amhaz, S. Chambon, J. Idier, and V. Baltazart, "A new minimal path selection algorithm for automatic crack detection on pavement images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 788–792.

[13] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, and H. Omata, "Road damage detection and classification using deep neural networks with smartphone images," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 33, no. 12, pp. 1127–1141, Jun. 2018.

[14] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, May 2017.

[15] A. Alfarrarjeh, D. Trivedi, S. H. Kim, and C. Shahabi, "A deep learning approach for road damage detection from smartphone images," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 5201–5204.

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

[17] 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.

[18] F. Yang, L. Zhang, S. Yu, D. V. 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.

[19] Y. Fei et al., "Pixel-level cracking detection on 3D asphalt pavement images through deep-learning-based CrackNet-V," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 273–284, Jan. 2020.

[20] J. Ji, L. Wu, Z. Chen, J. Yu, P. Lin, and S. Cheng, "Automated pixel-level surface crack detection using U-Net," in *Proc. Int. Conf. Multi-Disciplinary Trends Artif. Intell. (MIWAI)*, 2018, pp. 69–78.

[21] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual U-Net," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 5, pp. 749–753, May 2018.

[22] T. M. Quan, D. G. C. Hildebrand, and W. Jeong, "FusionNet: A deep fully residual convolutional neural network for image segmentation in connectomics," *Frontiers Comput. Sci.*, vol. 3, May 2021, Art. no. 613981.

[23] 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. 2015.

[24] 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.*, 2015, pp. 234–241.

[25] 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.

[26] F. Long, Z. Qiu, Y. Pan, T. Yao, J. Luo, and T. Mei, "Stand-alone inter-frame attention in video models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 3192–3201.

[27] I. Bello, B. Zoph, Q. Le, A. Vaswani, and J. Shlens, "Attention augmented convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3286–3295.

[28] H. Liu, X. Miao, C. Mertz, C. Xu, and H. Kong, "CrackFormer: Transformer network for fine-grained crack detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3783–3792.

[29] G. Suh and Y.-J. Cha, "Deep faster R-CNN-based automated detection and localization of multiple types of damage," in *Proc. SPIE*, vol. 10598, pp. 197–204, Mar. 2018.

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

[31] W. Shen, K. Zhao, Y. Jiang, Y. Wang, X. Bai, and A. Yuille, "DeepSkeleton: Learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images," *IEEE Trans. Image Process.*, vol. 26, no. 11, pp. 5298–5311, Nov. 2017.

[32] 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.

[33] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[34] D. Zhang, H. Zhang, J. Tang, M. Wang, X. Hua, and Q. Sun, "Feature pyramid transformer," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 323–339.

[35] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 568–578.

[36] J. Guo et al., "CMT: Convolutional neural networks meet vision transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 12175–12185.

[37] O. Petit, N. Thome, C. Rambour, L. Themyr, and T. Collins, "U-Net transformer: Self and cross attention for medical image segmentation," in *Proc. Int. Workshop Mach. Learn. Med. Imag.*, 2021, pp. 267–276.

[38] J. Chen et al., "TransUNet: Transformers make strong encoders for medical image segmentation," 2021, *arXiv:2102.04306*.

[39] Y. Sha, Y. Zhang, X. Ji, and L. Hu, "Transformer-UNet: Raw image processing with UNet," 2021, *arXiv:2109.08417*.

[40] A. Lin, B. Chen, J. Xu, Z. Zhang, and G. Lu, "DS-TransUNet: Dual Swin transformer U-Net for medical image segmentation," 2021, *arXiv:2106.06716*.

[41] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1623–1637, Mar. 2022.

[42] W. Wang et al., "PVT v2: Improved baselines with pyramid vision transformer," 2021, *arXiv:2106.13797*.

[43] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," 2021, *arXiv:2103.13413*.

[44] I. Bello, "LambdaNetworks: Modeling long-range interactions without attention," in *Proc. ICLR*, 2021, pp. 1–31.

[45] Y. Dong, J.-B. Cordonnier, and A. Loukas, "Attention is not all you need: Pure attention loses rank doubly exponentially with depth," 2021, *arXiv:2103.03404*.
[46] O. Oktay et al., "Attention U-Net: Learning where to look for the pancreas," 2018, *arXiv:1804.03999*.
[47] J. König, M. Jenkins, M. Mannion, P. Barrie, and G. Morison, "Optimized deep encoder–decoder methods for crack segmentation," 2020, *arXiv:2008.06266*.
[48] Y. Zhang, J. Wu, Q. Li, X. Zhao, and M. Tan, "Beyond crack: Fine-grained pavement defect segmentation using three-stream neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14820–14832, Sep. 2022.
[49] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 8, pp. 1558–1570, Aug. 2015.
[50] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.
[51] W. Ke, J. Chen, J. Jiao, G. Zhao, and Q. Ye, "SRN: Side-output residual network for object symmetry detection in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1068–1076.
[52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.

Huajun Liu (Member, IEEE) received the Ph.D. degree from the School of Computer Science, Nanjing University of Science and Technology, in 2007. He is currently with the School of Computer Science and Engineering, Nanjing University of Science and Technology. He worked as a Visiting Scholar and a Post-Doctoral Fellow with the Robotics Institute, Carnegie Mellon University, from 2018 to 2020. His research interests include computer vision, information fusion, and deep learning.

Jing Yang is currently pursuing the master's degree with the School of Computer Science and Engineering, Nanjing University of Science and Technology. His research interests include computer vision, especially on image binary segmentation and representation learning for images.

Xiangyu Miao received the master's degree from the School of Computer Science and Engineering, Nanjing University of Science and Technology. His research interests include deep learning and computer vision.

Christoph Mertz received the Ph.D. degree in physics from Arizona State University. After continuing his nuclear physics research for another year, he joined the CMU Robotics Institute as a Research Staff, where he developed surround sensing and object detection systems for intelligent vehicles and search and rescue robots. He worked on projects sponsored by ARL, DOT, FTA, PennDOT, NIJ, DARPA, NSF, and several companies. His current research focuses are on how edge computing can be used to update HD maps, monitor infrastructure, and count traffic on a scale as large as a whole city. He is also developing nighttime perception for off-road robots. Besides his full-time position with CMU, he has been the Co-Founder and a Chief Scientist with RoadBotics. The startup has recently been acquired by Michelin, where he continues this part-time position as a Chief Scientist.

Hui Kong (Member, IEEE) received the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, in 2007. He is currently with the State Key Laboratory of Internet of Things for Smart City (SKL-IOTSC), the Department of Electromechanical Engineering (EME), and the Department of Computer and Information Science (CIS), University of Macau (UM). Before that, he was affiliated with NJUST, MIT, The Ohio State University, Ecole Normale Superieure (ENS), Paris, France. His research areas are sensing and perception for autonomous driving, mobile robotics, SLAM, and multiview geometry in computer vision. His research has been supported by the Science and Technology Development Fund of Macau (FDCT), the National Natural Science Foundation of China (NSFC), and several companies including Huawei Technology, Horizon Robotics, and Amy Robotics. He serves as an Associate Editor for the *International Journal of Computer Vision* (IJCV) and the *International Journal of AI and Autonomous Systems* (AIAS).