

CSegNet-A Crack Segmentation Network Combining CNN and Transformer

Hao Dong

Science Island Branch of Graduate School University of Science and Technology of China

Yinlai Du

Anhui University

Dong Feng

Hefei Institutes of Physical Science

Qingyuan Hu

Science Island Branch of Graduate School University of Science and Technology of China

Mingzhu Zhou

China National Tobacco Quality Supervision and Test Center

Jun Xing

China National Tobacco Quality Supervision and Test Center

Long Zhang

Science Island Branch of Graduate School University of Science and Technology of China

Shu Wang

Hefei Institutes of Physical Science

Yong Liu

liuyong1s@163.com

Science Island Branch of Graduate School University of Science and Technology of China

Article

Keywords: Road Crack, Crack Segmentation, Transformer, CNN

Posted Date: February 20th, 2024

DOI: <https://doi.org/10.21203/rs.3.rs-3925781/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

CSegNet: A Crack Segmentation Network

Combining CNN and Transformer

Hao Dong^{a,b,c}, Yinlai Du^d, Dong Feng^b, Qingyuan Hu^{a,c}, Mingzhu Zhou^c, Jun Xing^c, Long Zhang^{a,b},
Shu Wang^{b*}, Yong Liu^{a,b*}

^a. Science Island Branch of Graduate School University of Science and Technology of China,
Hefei 230036, China.

^b. Hefei Institute of Physical Science, Chinese Academy of Sciences, Hefei 230036, China.

^c. China National Tobacco Quality Supervision and Test Center, Zhengzhou 450001, China

^d. School of Electronics and Information Engineering, Anhui University, Hefei 230601, China

*Corresponding author

Email: liuyong1s@163.com (Y.Liu.)

Email: wangshu@aiofm.ac.cn (S. Wang)

Abstract: Detecting cracks from images plays a crucial role in road maintenance. Road cracks exhibit significant diversity and complexity in terms of shape, size, texture, and road images may contain various noises and interferences such as lighting variations, shadows, and different appearances due to varying perspectives and scales. To address these challenges, we constructed a comprehensive dataset called the Comprehensive Road Crack Dataset (CRCrack Dataset), which encompasses various crack characteristics. In this study, we propose a road crack segmentation network called CSegNet (Crack Segmentation Network), which combines convolutional neural networks (CNNs) and Transformers. The network adopts an encoder-decoder framework, like DeepLab V3+. In the encoder, leveraging the flexibility of Transformers in modeling long-term dependencies and the ability of CNNs to capture local contextual information through local receptive fields, weight sharing, and spatial subsampling, we design a ResNeXTR (ResNeXt-Transformer) feature extraction module as the backbone network to enhance the feature extraction capability for road

crack images. To reduce the computational cost in self-attention computation of transformer, we introduce an average pooling layer to downsample the dimensions of the encoded features. In the decoder, to focus on the key information of road cracks under diverse environmental conditions and interferences, we combine the Efficient Channel Attention Module (ECAM) and the Spatial Attention Module (SAM) to design an Efficient Convolutional Block Attention Module (ECBAM) attention module to further optimize feature representation. Additionally, we employ the ReLU activation function, SGD gradient descent, and a hybrid loss function of Binary Cross Entropy with Logits to accelerate convergence speed and improve segmentation accuracy. Through comparative experiments on the CRCrack dataset, the results demonstrate that our proposed method outperforms classic networks such as U-Net and DeepLab V3+ in terms of IoU, Dice, and AUROC evaluation metrics. It exhibits good adaptability to ground crack images from different sources, providing a basis for estimating the degree of road damage.

Key Words: Road Crack, Crack Segmentation, Transformer, CNN

1. Introduction

In the field of industrial defect detection, the identification and analysis of road cracks are crucial components. Road cracks not only represent potential safety hazards but can also lead to traffic accidents and high maintenance costs. Therefore, it is necessary to take timely measures to detect and repair road cracks. In traditional detection methods, personnel primarily rely on visual inspection to identify cracks on the road. While this method can accurately locate cracks to some extent, it is labor-intensive and time-consuming, especially when dealing with long-distance road inspections.

With the advancement of computer vision technology, machine learning and image processing methods have been employed for automated detection and recognition of road cracks [1][2]. Li et al. [2] utilized the adjacent difference histogram method to identify regions containing cracks, assuming that pixels belonging to crack regions are

generally darker than other pixels. Li et al. [3] employed two thresholding techniques and adaptive iterative thresholding to detect cracks on airport runway surfaces. Kapela et al. [3] utilized Hough Transform Features (HTF) and Local Binary Patterns (LBP) to extract edge orientations and texture features of cracks. Quan et al. [5] proposed a crack detection method based on modified gray-level histogram probability weighting factors, which enhanced the accuracy of object extraction and addressed issues related to prominent peaks, effectively handling various types of road crack image processing problems. Huan et al. [6] proposed an improved Canny operator for road crack detection, where they replaced the original Gaussian filter and Otsu's maximum interclass variance thresholding algorithm with morphological filtering, achieving adaptive determination of dual thresholds. Amhaz et al. [7] proposed an algorithm for automatic detection of two-dimensional road cracks based on minimum path localization.

With the advancement of deep learning techniques, particularly the emergence of Convolutional Neural Networks (CNN) [8], new methods have emerged in the field of crack detection. These methods primarily involve the analysis of captured images using CNN, which can automatically learn features from the images without the need for manual feature design or selection. This enables them to achieve higher accuracy and robustness when dealing with complex scenes or different types of cracks. Currently, several researchers have employed classification networks and object detection networks [9], [10], [11] for crack detection research. However, these approaches do not directly extract cracks at the pixel level and require the combination of threshold algorithms, edge algorithms, and other image processing techniques to accomplish pixel-level crack extraction. On the other hand, crack detection algorithms based on segmentation networks can segment the crack at the pixel level, allowing for the calculation of parameters such as crack length, width, area, etc., which provide a basis for estimating the degree of road damage. Eisenbach et al. [12] proposed a road crack dataset for training deep learning networks and conducted an initial evaluation of road crack detection techniques. Zhang et al. [13] proposed CrackNet, a CNN-based approach that detects cracks at the pixel level. CrackNet does not include any pooling layers, and the feature map size remains constant across all layers. Subsequently, in

2018, Zhang et al. [14] proposed an improved version called CrackNet II. Compared to CrackNet, CrackNet II incorporates trainable parameters in all hidden layers, further enhancing crack detection performance. In 2020, Chen et al. [15] proposed a rotation-invariant FCN called ARF-Crack, which requires fewer network parameters. Nhung et al. [16] proposed a two-stage CNN-based method for pixel-level road crack detection and segmentation. Moreover, the encoder-decoder architecture has been widely used in various road crack detection methods. Li et al. [17] proposed a Convolutional Encoder-Decoder Network (CedNet) for crack detection in images. DeepLabv3+ has gained popularity in the field of crack detection due to its powerful segmentation performance. Ji et al. [18] were the first to apply DeepLabv3+ to road crack detection. They validated the effectiveness of DeepLabv3+ in detecting single cracks, multiple cracks, and crossed cracks, outperforming FCN [19] and DeepCrack [20]. Sun et al. [21] proposed a multi-scale attention module in the decoder of DeepLabv3+ to generate attention masks and dynamically allocate weights between high-level and low-level feature maps to address the shortcomings of DeepLabv3+ in capturing crack details. In 2022, Cai et al. [22] utilized MobileNetV2 as the backbone feature extraction network for DeepLabv3+ to significantly reduce model computation while improving computational speed. They also introduced attention mechanism modules into the feature extraction network and decoder to further optimize the model's target recognition and segmentation accuracy.

CNN can capture rich local contextual information of road cracks through local receptive fields, weight sharing, and spatial sub-sampling. However, due to the significant diversity and complexity of actual crack shapes, sizes, textures, and the susceptibility of crack images to various noise, interferences, as well as variations in shooting angles and scales, achieving precise crack segmentation remains a challenging problem. Recently, the Transformer, a new paradigm in self-attention mechanisms, has gained wide attention from researchers in the field of natural language processing and has been successfully extended to other domains such as image processing and video processing, demonstrating excellent performance [23][24]. Therefore, in this paper, we propose a hybrid network called CSegNet that combines Transformer and CNN to

improve the segmentation accuracy of road crack detection without increasing the parameter size. The main contributions of this paper are as follows:

- (1) Leveraging the flexibility of Transformer in modeling long-range dependencies and considering the image characteristics of road cracks, we design a backbone network called ResNeXTR by integrating ResNeXt50 and Swin Transformer.
- (2) We further focus on crack features within the road by designing an efficient attention mechanism called Efficient Convolutional Block Attention Module (ECBAM). Additionally, we employ the ReLU activation function, the SGD gradient descent method, and the Binary Cross Entropy with Logits loss function to accelerate convergence and enhance segmentation accuracy.
- (3) Finally, we construct the CRCrack Dataset, a comprehensive dataset for joint crack detection, to train the CSegNet network and achieve precise segmentation of road cracks.

2. Methods

The CSegNet network model adopts the basic encoder-decoder architecture of DeepLab V3+, as shown in Figure 1.

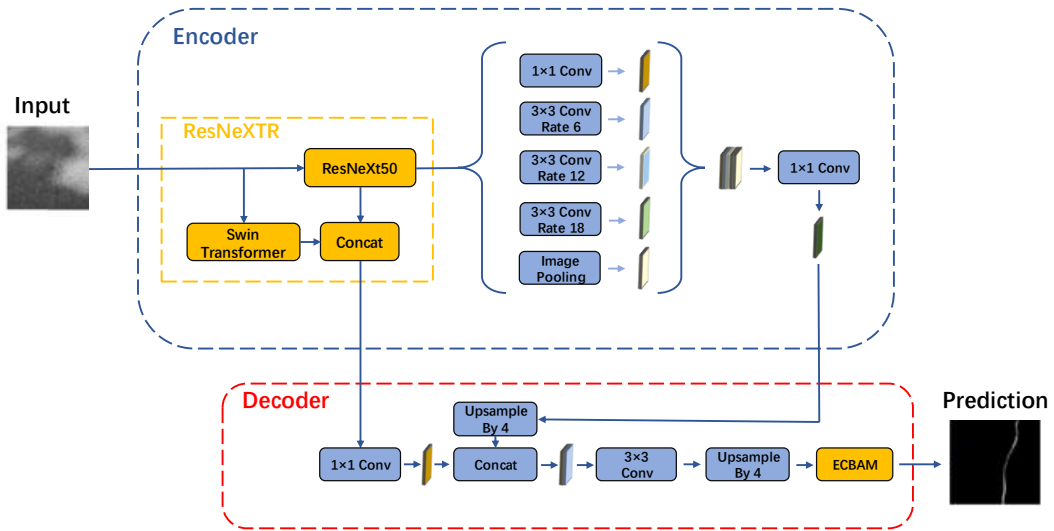


Figure 1 The structure of the proposed CSegNet

The encoder consists of a backbone feature extraction network and an Atrous Spatial Pyramid Pooling (ASPP) module. As the Transformer exhibits flexibility in modeling long-range dependencies, while CNN can capture local contextual

information through local receptive fields, weight sharing, and spatial sub-sampling, we designed a ResNeXt-Transformer (ResNeXTR) feature extraction module as the backbone network to enhance the feature extraction capability of the input image. In the ResNeXTR feature extraction module, we employed the efficient ResNeXt50 network for the CNN part and the Swin Transformer network based on self-attention mechanisms for the Transformer part. To reduce the computational cost of self-attention calculations, we introduced average pooling to downsample the dimensions after encoding. After the image is passed through the encoder, to further enrich the feature information and enhance the feature extraction capability of the model, we concatenated the shallow features from ResNeXt50 with the output of Swin Transformer as shallow outputs, which are then input to the decoder for fusion. The deep feature maps from ResNeXt50 are processed by the ASPP module, which includes a 1×1 convolution, global average pooling, and three 3×3 dilated convolutions with dilation rates of 6, 12, and 18. This structure enables the extraction of rich contextual information. The fused and scaled feature maps are then input to the decoder.

The decoder maintains the structure of DeepLab V3+, but before the classification output, we designed the ECBAM attention module to strengthen long-range interaction between low-level feature maps without significantly increasing the parameter size, thus further optimizing feature representation. The decoder takes the multi-scale semantic feature maps from the ASPP module and performs a 4x upsampling. Then, the features are fused with shallow semantic information extracted from the ResNeXTR module. After a 3×3 convolution and 4x upsampling, the features are passed through the ECBAM attention module. Finally, the semantic segmentation results are obtained.

A. ResNeXTR

The ResNeXTR feature extraction module used for feature extraction in the encoder module is depicted in Figure 2. We employ the ResNeXt50 network as the CNN component in the ResNeXTR feature extraction module, allowing the network to learn a wider range of diverse local features from crack images. To balance efficiency and performance, we utilize the Swin Transformer network based on self-attention

mechanisms as the Transformer component in the ResNeXTR feature extraction module. This enables the network to capture long-range dependencies in crack images and effectively supplement global contextual information. The ResNeXt50 network provides rich local features for the input image, while the Swin Transformer network based on self-attention mechanisms complements the input image with global contextual information. By concatenating the output features of these two networks, we construct a feature representation with rich expressive power for crack detection and segmentation.

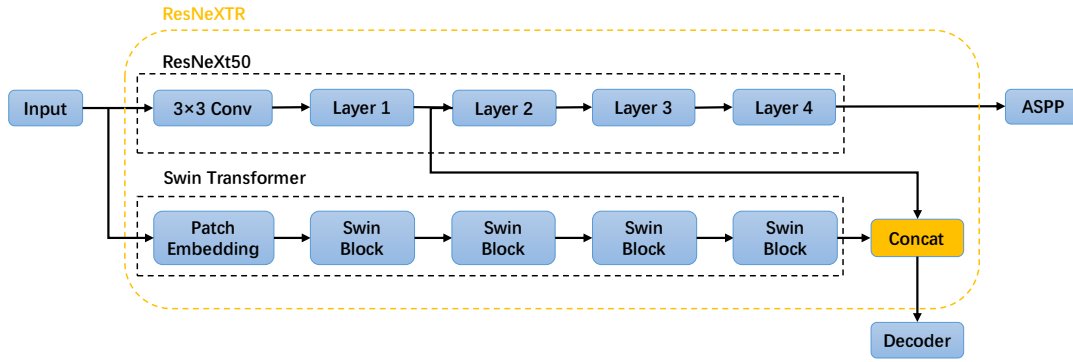


Figure 2 Structure of ResNeXTR

1) ResNeXt

ResNeXt50 consists of multiple residual blocks, which are unique to the ResNet network. These residual blocks are distributed in four stages following a configuration of [3, 4, 6, 3]. Through these residual blocks, the ResNeXt50 network performs in-depth feature analysis on the input image, generating multi-scale feature maps that ensure accurate and rich semantic representation of cracks in the input image.

The architecture with residual connections has been widely used by researchers in various studies. It has been shown that residual connections effectively address the problem of gradient vanishing, prevent a decrease in model accuracy, and improve the performance of neural networks. This skip connection approach ensures that data flows through the network without interference. The schematic diagram of residual connections is shown in Figure 3.

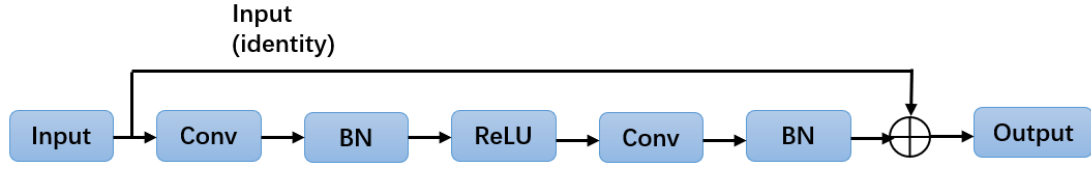


Figure 3 Structure of residual blocks

Using the ResNeXt50 network is based on several considerations. Firstly, ResNeXt50 introduces ‘cardinality’ as a new dimension alongside depth and width, which increases the model's representational capacity without significantly increasing its complexity. Secondly, the modular design of ResNeXt50 makes the model more flexible, easily scalable, and efficient to train. Additionally, ResNeXt50 outperforms the initial encoder network of DeepLab V3+, Xception, on the ImageNet dataset and demonstrates excellent generalization capabilities across multiple tasks and datasets. Therefore, incorporating ResNeXt50 as part of the feature extraction network can improve model performance, enhance stability, and robustness.

During actual training, we first utilize the pre-trained ResNeXt50 network on the publicly available ImageNet dataset [27] for initial feature extraction from input images.

2) Transformer with Average Pooling

In addition to the ResNeXt module mentioned earlier, we have also introduced the Swin Transformer, which is pre-trained on the ImageNet dataset, to analyze and construct detailed feature maps for the shallow features of the input image. Swin Transformer is an efficient visual Transformer network architecture. Unlike traditional Transformer networks, Swin Transformer processes images using window-based self-attention mechanisms and introduces local information exchange to reduce computational complexity. In our model, the self-attention mechanism of Swin Transformer helps capture more local and global details, providing a finer feature representation for generating semantic labels for cracks.

In practical implementation, we have observed that the original attention computation still involves relatively high computational complexity, placing demands on the hardware. To reduce the computational burden of the model, we have made improvements to the basic attention computation, as shown in Figure 4, to reduce the computational cost while preserving important information.

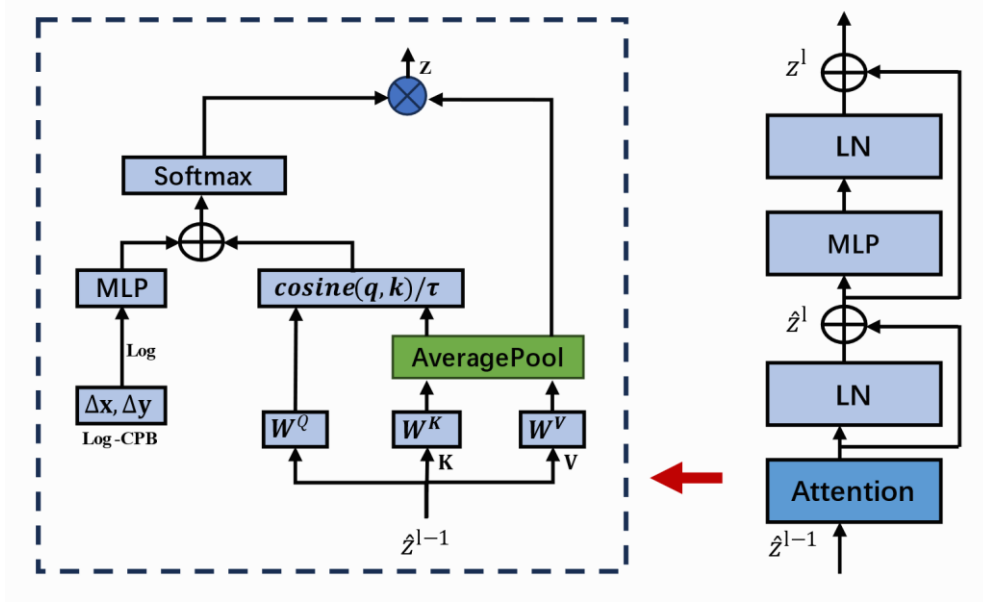


Figure 4 Encoder Structure of Swin Transformer

To reduce the computational complexity of self-attention calculations, an average pooling layer, denoted as P_S , is introduced to downsample the dimensionality of the encoded features. Furthermore, pointwise convolution is utilized to further reduce the feature dimensionality. The specific computation process of self-attention, SA , is as follows:

$$SA(X) = \text{Attention}(X \cdot W^Q, P_S(X \cdot W^K), P_S(X \cdot W^V)) \quad (1)$$

Where W^Q , W^K and W^V represent the linear layers used for encoding. *Attention* refers to the attention computation method, which can be expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\text{SIM}(Q, K))V$$

$\text{SIM}(Q, K)$ represents the similarity calculation between Q and K , using cosine similarity as follows:

$$\text{Sim}(q_i, k_j) = \frac{\cos(q_i, k_j)}{\tau} + B_{ij} \quad (2)$$

Where B_{ij} denotes the relative positional bias between pixel i and pixel j , and τ is a learnable scaling parameter.

B is obtained through a continuous relative positional bias approach, where rather than directly optimizing parameterized biases, a small-scale sub-network $\mathcal{G}$ is employed to update biases based on relative coordinates:

$$B(\Delta x, \Delta y) = \mathcal{G}(\Delta x, \Delta y) \quad (3)$$

Here, $\mathcal{G}$ is a small network, such as a 2-layer MLP with ReLU activation by default.

The Transformer block consists of self-attention (SA) and MLP modules, and the computational flow is as follows:

$$z^l = \text{LN}(\text{SA}(z^{l-1})) \quad (4)$$

$$\hat{z}^l = \text{Concat}(\hat{z}^l, z^{l-1}) \quad (5)$$

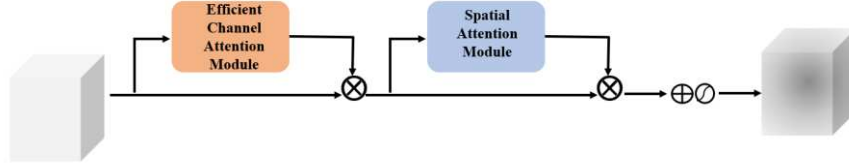
$$\ddot{z}^l = \text{LN}\left(\text{MLP}(\hat{z}^l)\right) \quad (6)$$

$$z^l = \text{Concat}(\ddot{z}^l, \hat{z}^l) \quad (7)$$

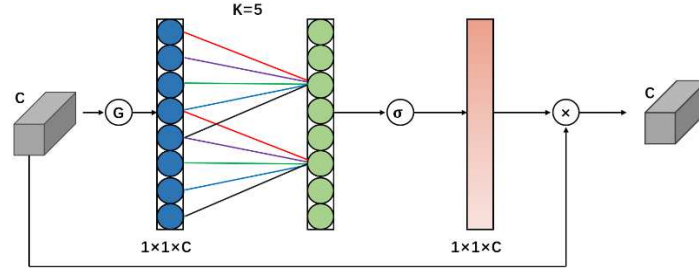
Where $\hat{z}^l$ represents the output of SA after LN (layer normalization), $\ddot{z}^l$ represents the output of MLP after LN, and z^l represents the final output of the Transformer block.

B. ECBAM (Efficient Convolutional Block Attention Module)

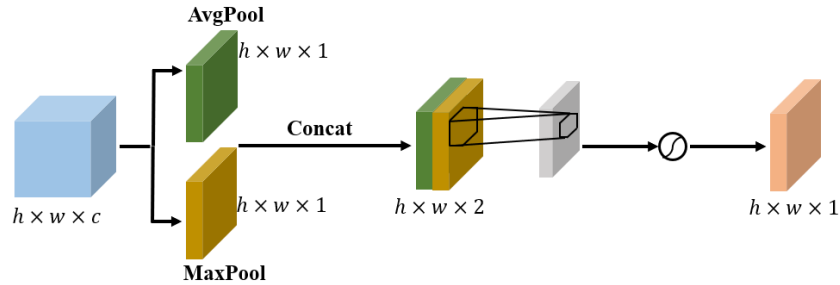
To focus on the key information of road cracks amidst various noise and complex backgrounds, while suppressing irrelevant information from different channels, we designed an Efficient Convolutional Block Attention Module (ECBAM) attention mechanism module in the output part of the Decoder, as shown in Figure 5(a).



(a) Efficient Convolutional Block Attention Module



(b) Efficient Channel Attention Module



(c) Spatial Attention Module

Figure 5 Structure of attention modules.

In the field of deep learning, attention mechanisms have become an important

component, especially in CNNs. ECAM (Efficient Channel Attention Mechanism), as shown in Figure 5(b), provides an efficient channel attention method by assigning weights to each channel, enhancing important channels and reducing relatively unimportant ones, thereby improving model performance. Compared to other attention mechanisms, ECAM has the advantage of using one-dimensional convolution on globally average-pooled features, making it simple and efficient. However, ECAM only considers channel attention while ignoring spatial information.

CBAM (Convolutional Block Attention Module) is a lightweight and versatile attention module that combines both spatial and channel attention modules in a sequential manner to extract crucial features [28]. As shown in Figure 5(c), SAM (Spatial Attention Module) proposed in this article enables the model to focus more on spatial features that are deemed more important, while reducing the number of model parameters and computational complexity. In this work, we combine ECAM and SAM to design an ECBAM attention module, which integrates both channel and spatial information.

To reduce the increase in model parameter count caused by adding the Hybrid Attention Mechanism module, ECBAM first inputs the feature map into the ECAM, which has a lower parameter count. The ECAM efficiently achieves local cross-channel interactions using 1D convolution. Then, the feature map is passed through the SAM, which allows the model to focus more on the crucial pixel regions for crack detection and reduces attention on non-pertinent pixel regions. This design allows ECBAM to strike a balance between capturing important channel-wise interactions and emphasizing key pixel areas relevant to crack detection while minimizing the impact on the overall model parameter count.

C. Loss Function

During the training and validation phases of the model, we employ a combination loss function consisting of BCEWithLogitsLoss (Binary Cross-Entropy with Logits Loss) and Dice loss.

BCEWithLogitsLoss is commonly used for binary classification problems. It

combines the Sigmoid activation function and binary cross-entropy loss. When applied to crack detection and segmentation tasks, using BCEWithLogitsLoss provides explicit feedback to the model, enabling more accurate crack detection and segmentation. The definition of BCEWithLogitsLoss is given by the following equation:

$$L_{BCEwithLogits} = \frac{1}{N} \sum_{i=1}^N (y_i \cdot \log(\sigma(p_i)) + (1 - y_i) \cdot \log(1 - \sigma(p_i))) \quad (8)$$

In the equation, N represents the total number of samples. y_i represents the ground truth label of the i -th sample, where 1 indicates that the pixel belongs to a crack, and 0 indicates that the pixel does not belong to a crack. p_i represents the raw output of the model's prediction for the i -th pixel. $\sigma(p_i)$ represents the result obtained by applying the sigmoid function to the raw output of the model's prediction for the i -th pixel. The sigmoid function squashes the output into the range of $[0, 1]$, representing the predicted probability that the pixel belongs to a crack.

Another loss function is the Dice loss function. Unlike BCEWithLogitsLoss, the Dice loss function measures the overlap between the labels and predictions. In crack detection and segmentation, the Dice loss function quantifies the similarity between the predicted and ground truth labels. Let X denote the ground truth segmentation image with true labels, and Y denote the predicted segmentation image with pixel classes. The Dice loss function is defined as shown in Equation (9) and Equation (10):

$$L_{Dice} = 1 - Dice\ Coefficient \quad (9)$$

$$Dice\ Coefficient = \frac{2|X \cap Y|}{|X| + |Y|} \quad (10)$$

Here, $|X \cap Y|$ represents the number of elements in the intersection between X and Y , and $|X|$ and $|Y|$ represent the number of elements in X and Y , respectively.

The final loss function used by the model is represented by the following equation:

$$L = 0.5L_{BCEwithlogits} + 0.5L_{Dice} \quad (11)$$

This equation combines the BCEWithLogits loss and the Dice loss with equal weights (0.5 each). The combination of these two loss functions helps to balance the accuracy of crack detection and the precision of boundary delineation in the segmentation task.

3. Experiments and results

3.1 Datasets

We constructed a comprehensive road crack dataset called CRCrack using multiple open-source datasets for model training, validation, and testing. It consists of 9,503 training images and 1,483 testing images. Specifically, the dataset is composed of 10 sub-datasets: Crack500 [29], CrackForest [30], GAPs384 [31], CrackTree [32], AEL [33], Sylvie Chambon [34], Volker [35], Rissbilder [35], Eugen Muller [36], and DeepCrack [37].

Crack500 is a large-scale dataset used for road and pavement crack detection, containing 500 high-resolution images along with corresponding crack annotations. CrackForest is a widely used benchmark dataset for crack detection in road images, comprising high-resolution images from real-world scenarios and pixel-level crack annotations. GAPs384 is a dataset focused on German asphalt pavement gap detection, which addresses comparability issues in the field of road surface defects by providing standardized, high-quality, and large-scale data. CrackTree is a dataset that includes various types of crack patterns and scenes, aimed at evaluating crack detection and classification algorithms. AEL is a dataset consisting of crack images captured under various environmental and lighting conditions, serving as an evaluation benchmark for robust crack detection algorithms. Sylvie is a dataset specifically designed for crack detection in structural cracks, featuring various building materials and crack types. Volker is a dataset dedicated to evaluating crack detection in concrete cracks, including challenges encountered in practical applications. Rissbilder is a dataset containing various crack morphologies and types, used for evaluating and comparing crack detection algorithms. Eugen is a dataset focused on road and bridge crack detection, encompassing various road types and crack scenarios. DeepCrack is a dataset specifically designed for deep learning-based crack detection, providing high-resolution images with precise annotations.

It should be noted that some subsets of datasets, such as a portion of the CRACK500 dataset, were included in the CRCrack dataset through cropping. The use

of multiple sources of data helps improve the robustness and applicability of the proposed model for more accurate crack detection. During the training phase of the model, the 9,503 training images are randomly divided in a 0.85:0.15 ratio, where 85% of the images are used for model training and the remaining 15% are used for model validation.

3.2 Experiment settings

In learning stage, the SGD optimizer is used with a default learning rate of 0.001. If the validation loss continues to increase for 10 consecutive epochs, the learning rate is reduced by a factor of 0.1. Additionally, the batch size for training is set to 4, and the batch size for testing is set to 8.

A deep learning environment was set up on Ubuntu 22.04.2 LTS. The hardware configuration includes an Intel® Xeon(R) E5-2678v3 CPU @ 2.50GHz × 48 and an NVIDIA GeForce RTX 3080 GPU. The Python version used in the deep learning environment is 3.10, PyTorch version is 1.12.1, OpenCV version is 4.6, CUDA version is 11.6, and CUDNN version is 8.2.

To comprehensively evaluate the performance of the algorithm in the segmentation task, three commonly used evaluation metrics were considered: Intersection over Union (IoU), Dice coefficient, and Area Under the ROC Curve (AUROC). These evaluation metrics are widely used for assessing image segmentation performance. The formulas for calculating these evaluation metrics are shown in Equations (12)-(14).

$$IoU = \frac{A \cup B}{A \cap B} \quad (12)$$

$$Dice = 2 * \frac{(A \cap B)}{(|A| + |B|)} \quad (13)$$

$$AUROC = \int_0^1 TPR(t) dt \quad (14)$$

In Equations (12) to (14), A represents the pixel region predicted as cracks, B represents the pixel region of the actual cracks, $\cap$ represents the number of pixels that overlap between A and B , $\cup$ represents the total number of pixels in A and B , and $TPR(t)$ is the True Positive Rate at threshold t , where t ranges from 0 to 1.

These metrics are used to assess the spatial consistency between the predicted results and the ground truth. IoU, also known as the Jaccard coefficient, calculates the ratio of the intersection to the union between the predicted pixel region of cracks and the actual pixel region of cracks. A higher IoU value indicates a higher degree of overlap between the predicted results and the ground truth, indicating better model performance. Dice coefficient is a commonly used metric for evaluating binary segmentation problems. Like IoU, Dice coefficient measures the similarity between the predicted results and the ground truth. Similarly, a higher Dice coefficient indicates higher accuracy in the predictions. AUROC is a statistical measure used to evaluate the performance of classifiers at different decision thresholds. It is based on the Receiver Operating Characteristic (ROC) curve, which represents the trade-off between true positive rate (TPR) and false positive rate (FPR). Specifically, TPR is the proportion of positive samples correctly predicted as positive, while FPR is the proportion of negative samples incorrectly predicted as positive. A higher AUROC value indicates better performance in the segmentation task (which can be viewed as a binary classification task between crack and background regions).

IoU, Dice coefficient, and AUROC collectively assess the performance of image segmentation models. IoU intuitively measures the degree of intersection between true positives and predicted positives. Dice coefficient further emphasizes the balance between positive and negative samples. AUROC, as a classic evaluation tool, demonstrates the model's classification ability at various possible thresholds. Therefore, by considering these three metrics in the image segmentation evaluation process, a more comprehensive and in-depth understanding of the model's actual performance and potential strengths and weaknesses can be obtained.

3.3 Results

3.3.1 Ablation experiment

We investigated the impact of adding different modules to the baseline model on crack segmentation results. DeepLab V3+ was used as the baseline model. Then the encoder of DeepLab V3+ was replaced as a pretrained ResNeXt50 network or

ResNeXTR module respectively. Finally, ECBAM was added into the decoder. The corresponding experimental results are shown in Table 1.

Table 1. Results of ablation experiments

Baseline Model	ResNeXt50	Swin-Transformer	ECBAM	IoU	Dice	AUROC
				0.3550	0.4583	0.7469
DeepLabv3+	√			0.5160	0.6421	0.8689
	√	√		0.5672	0.6921	0.9133
	√	√	√	0.5844	0.7122	0.9209

From Table 1, it can be observed that the model achieved the best detection performance when combining the shallow features of the ResNeXt50 network with the outputs of Swin Transformer and using the overall deep features of ResNeXt50 as the encoder, while adding the ECBAM attention mechanism to the original Decoder part of DeepLab V3+.

Figure 6 presents the partial crack segmentation results obtained using the method proposed in this paper on the CRCrack dataset. It can be observed that our model is capable of accurately predicting the crack locations, regardless of the simplicity of the background (as shown in the first column) or the complexity of the background (as shown in the second, fourth, sixth, and tenth columns). Furthermore, our model performs well even in challenging scenarios with severe environmental interferences, such as high lighting conditions (as shown in the third column), fine cracks (as shown in the fifth and eighth columns), zebra crossings (as shown in the ninth column), and cases with no cracks present (as shown in the seventh column).

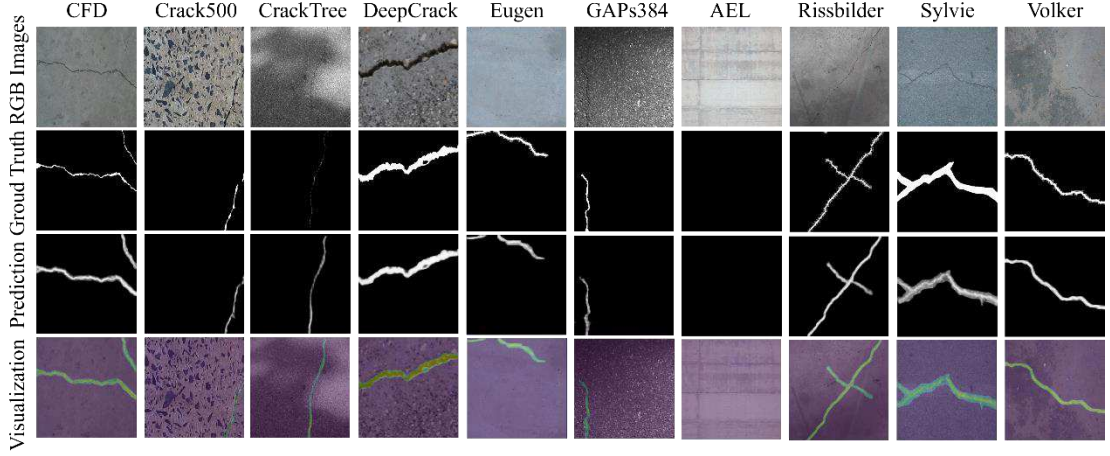


Figure 6 Results on the comprehensive dataset

3.3.2 Comparison with SOTA methods

To evaluate the performance of our model, we conducted comparative experiments with classical models such as Otsu, U-Net, U-Net++[38], and DeepLab V3+ on the comprehensive large dataset CRCrack, as well as the smaller datasets Crack500, CrackForest, and GAPs384, as shown in Table 2.

Table 2. Quantitative evaluation of different methods

Methods	Evaluate Metrics		
	IoU	Dice	AUROC
Ostu	0.0226	0.04342	0.3612
U-Net	0.4625	0.5855	0.8056
U-Net++	0.4930	0.6166	0.8212
DeepLabv3+	0.3550	0.4583	0.7469
Ours	0.5844	0.7122	0.9209

The Otsu method is an automatic thresholding technique that selects a threshold by maximizing the variance between two classes (foreground and background). In the case of crack segmentation, the Otsu method can distinguish between crack (foreground) and non-crack regions (background). U-Net, U-Net++, and DeepLab V3+ are encoder-decoder architectures with symmetric structures.

From Table 2, it can be observed that our proposed model outperforms the compared methods in terms of crack segmentation performance, as evaluated by the IoU, Dice, and AUROC metrics, on the constructed dataset. Additionally, our method also demonstrates superior performance on the Crack500, CrackForest, and GAPs384

datasets compared to other methods, as shown in Table 3.

Table 3. The crack segmentation performance of various models on the Crack500, CrackForest, and GAPs384 datasets

Methods	Crack500			CrackForest			GAPs384		
	IoU	Dice	AUROC	IoU	Dice	AUROC	IoU	Dice	AUROC
Ostu	0.0189	0.0364	0.3325	0.0058	0.0115	0.2605	0.0070	0.0138	0.3782
U-Net	0.4500	0.6183	0.6885	0.1248	0.2180	0.9177	0.2563	0.3754	0.7223
U-Net++	0.4725	0.6254	0.7167	0.1534	0.2627	0.9403	0.3438	0.4879	0.8032
DeepLab V3+	0.5197	0.6671	0.7796	0.1767	0.2961	0.8836	0.3005	0.4364	0.7398
Ours	0.5737	0.6977	0.8225	0.4036	0.5677	0.9712	0.4107	0.5645	0.8761

The complexity analysis of deep learning-based methods is presented in Table 4. It includes measurements of Floating Point Operations per Second (FLOPS) (in giga (G) units) and the number of parameters (in million (M) units). For all previous deep learning methods and the proposed method, the measured tensor input size is $4 \times 448 \times 448 \times 3$. Table 4 demonstrates the trade-off of the proposed method, where the parameter count of our proposed model is smaller than that of all compared methods, including method D. Additionally, our proposed model only introduces a marginal increase in computational complexity compared to the DeepLab V3+ network.

Table 4. FLOPS and number of parameters of each method

Methods	FLOPS(G)	Param(M)
U-Net	763.57	78.10
U-Net++	427.57	91.60
DeepLab V3+	531.05	39.76
Ours	552.52	39.23

4. discussions

Figure 7 illustrates the challenging task of crack detection, including complex crack textures, uneven lighting variations, and various forms of noise and interference such as shadows.

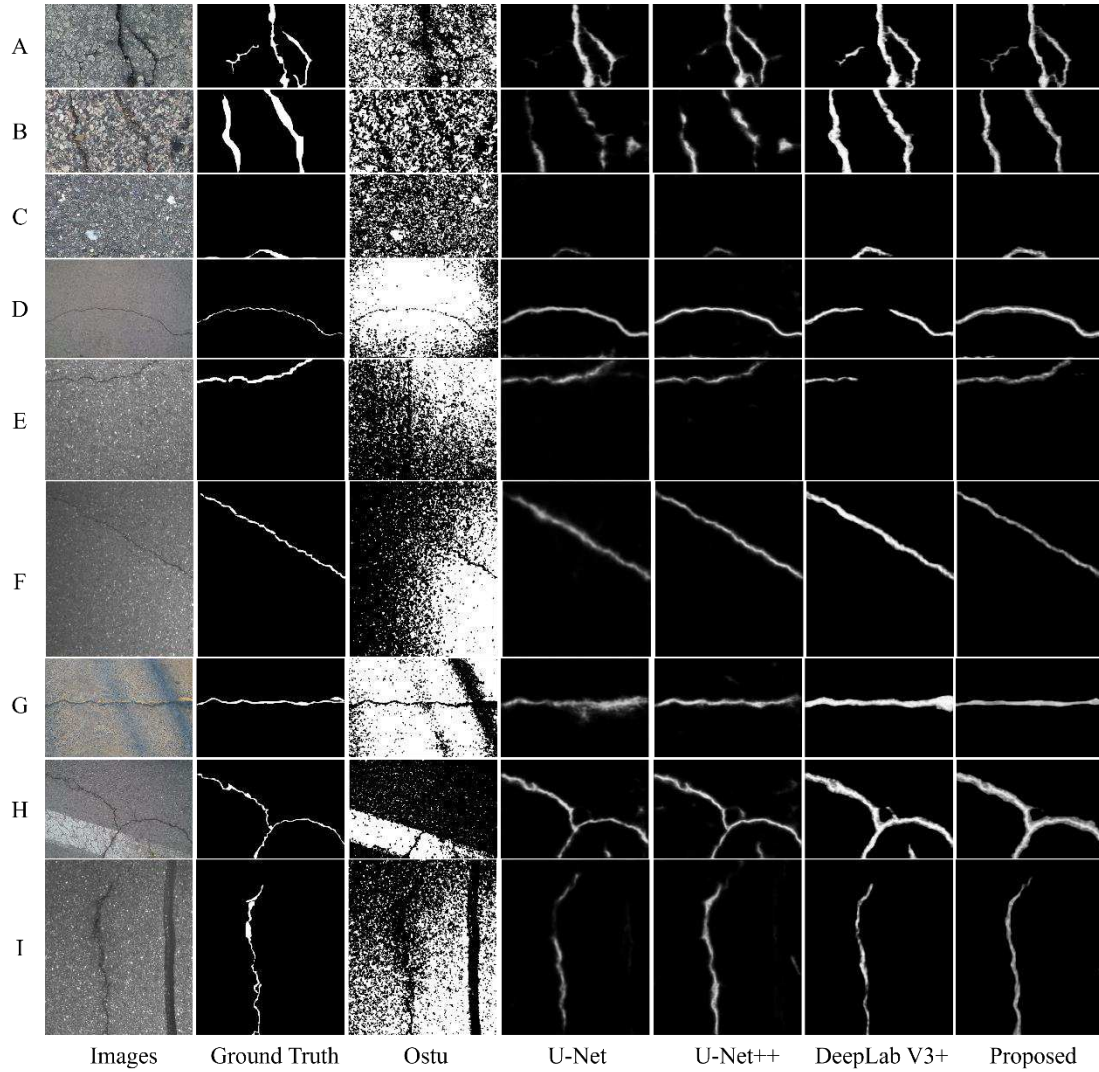


Figure 7 Challenging cases, including (A-C) rough surface, (D-F) fine crack, (G) shadow, (H)road marking, (I) oil stain

The images in rows A, B, and C are from the Crack500 dataset, where numerous noisy and interfering elements can be observed. The images in row D,E and F are from the CrackForest dataset, which lacks background interference but presents the challenge of detecting and locating fine cracks. However, our proposed method can accurately locate cracks, although the detected crack width may be slightly larger than the actual width. In the case of the lighting conditions shown in row G, H, and I, where the images are from the GAPs384 dataset, the proposed method can accurately detect cracks while mitigating the interference caused by shadows and sunlight. Simultaneously, for the images with zebra crossings shown in row H, the proposed algorithm can locate crack positions, but there is still an issue of the detected crack

regions being wider than the actual cracks.

5. Conclusions

This paper aims to address the challenges of inaccurate and low-precision road crack segmentation in the presence of complex backgrounds. To this end, we propose an advanced road crack segmentation algorithm called CSegNet. Our approach leverages the ResNeXt-Transformer (ResNeXTR) feature extraction module as the backbone network, effectively enhancing the capability to extract meaningful features from road crack images. Additionally, we propose the ECBAM attention mechanism, which combines ECAM and SAM, to further optimize feature representations. By optimizing feature extraction and representation, our proposed method achieves an IoU of 58.44%, Dice coefficient of 71.22%, and AUROC of 92.09% on the comprehensive large dataset CRCrack. Comparative experiments with other methods demonstrate the superior performance of our approach in road crack segmentation under complex background conditions. This method effectively improves the accuracy of intelligent road damage assessment, providing a more reliable basis for estimating the extent of road damage.

References

- [1] Oliveira H, Correia P L. CrackIT—An image processing toolbox for crack detection and characterization[C]//2014 IEEE international conference on image processing (ICIP). IEEE, 2014: 798-802.
- [2] Li Q, Liu X. Novel approach to pavement image segmentation based on neighboring difference histogram method[C]//2008 Congress on Image and Signal Processing. IEEE, 2008, 2: 792-796.
- [3] Peng L, Chao W, Shuangmiao L, et al. Research on crack detection method of airport runway based on twice-threshold segmentation[C]//2015 Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC). IEEE, 2015: 1716-1720.
- [4] Kapela R, Śniatała P, Turkot A, et al. Asphalt surfaced pavement cracks detection based on histograms of oriented gradients[C]//2015 22nd International Conference Mixed Design of Integrated Circuits & Systems (MIXDES). IEEE, 2015: 579-584.
- [5] Quan Y, Sun J, Zhang Y, et al. The method of the road surface crack detection by the improved Otsu threshold[C]//2019 IEEE International Conference on Mechatronics and Automation (ICMA). IEEE, 2019: 1615-1620.
- [6] Xu H, Li Z, Jiang Y, et al. Pavement crack detection based on OpenCV and improved Canny operator[J]. Computer engineering and design, 2014, 35(12).
- [7] Amhaz R, Chambon S, Idier J, et al. Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection[J]. IEEE Transactions on Intelligent Transportation Systems, 2016, 17(10): 2718-2729.
- [8] O'Shea K, Nash R. An introduction to convolutional neural networks[J]. arXiv preprint arXiv:1511.08458, 2015.
- [9] Xu X, Zhao M, Shi P, et al. Crack detection and comparison study based on faster R-CNN and mask R-CNN[J]. Sensors, 2022, 22(3): 1215.
- [10] Zhang R, Shi Y, Yu X. Pavement crack detection based on deep learning[C]//2021 33rd Chinese Control and Decision Conference (CCDC). IEEE, 2021: 7367-7372.
- [11] Hou Y, Shi H, Chen N, et al. Vision image monitoring on transportation

- infrastructures: a lightweight transfer learning approach[J]. IEEE Transactions on Intelligent Transportation Systems, 2022.
- [12] Eisenbach M, Stricker R, Seichter D, et al. How to get pavement distress detection ready for deep learning? A systematic approach[C]//2017 international joint conference on neural networks (IJCNN). IEEE, 2017: 2039-2047.
- [13] Zhang A, Wang K C P, Li B, et al. Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network[J]. Computer-Aided Civil and Infrastructure Engineering, 2017, 32(10): 805-819.
- [14] Zhang A, Wang K C P, Fei Y, et al. Deep learning–based fully automated pavement crack detection on 3D asphalt surfaces with an improved CrackNet[J]. Journal of Computing in Civil Engineering, 2018, 32(5): 04018041.
- [15] Chen F C, Jahanshahi M R. ARF-Crack: rotation invariant deep fully convolutional network for pixel-level crack detection[J]. Machine Vision and Applications, 2020, 31(6): 47.
- [16] Nguyen N H T, Perry S, Bone D, et al. Two-stage convolutional neural network for road crack detection and segmentation[J]. Expert Systems with Applications, 2021, 186: 115718.
- [17] Li S, Zhao X. Automatic crack detection and measurement of concrete structure using convolutional encoder-decoder network[J]. IEEE Access, 2020, 8: 134602-134618.
- [18] Ji A, Xue X, Wang Y, et al. An integrated approach to automatic pixel-level crack detection and quantification of asphalt pavement[J]. Automation in Construction, 2020, 114: 103176.
- [19] Yang X, Li H, Yu Y, et al. Automatic pixel-level crack detection and measurement using fully convolutional network[J]. Computer-Aided Civil and Infrastructure Engineering, 2018, 33(12): 1090-1109.
- [20] Liu Y, Yao J, Lu X, et al. DeepCrack: A deep hierarchical feature learning architecture for crack segmentation[J]. Neurocomputing, 2019, 338: 139-153.
- [21] Sun X, Xie Y, Jiang L, et al. DMA-Net: DeepLab with multi-scale attention for pavement crack segmentation[J]. IEEE Transactions on Intelligent Transportation

Systems, 2022, 23(10): 18392-18403.

- [22]Cai M, Yi X, Wang G, et al. Image segmentation method for sweetgum leaf spots based on an improved DeeplabV3+ network[J]. Forests, 2022, 13(12): 2095.
- [23]Dosovitskiy, A., et al., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. 2020.
- [24]Arnab, A., et al., ViViT: A Video Vision Transformer. 2021.
- [25]Agarap A F. Deep learning using rectified linear units (relu)[J]. arXiv preprint arXiv:1803.08375, 2018.
- [26]Goyal P, Dollár P, Girshick R, et al. Accurate, large minibatch sgd: Training imagenet in 1 hour[J]. arXiv preprint arXiv:1706.02677, 2017.
- [27]Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database[C]. 2009 IEEE conference on computer vision and pattern recognition, 2009: 248-255.
- [28]Woo S , Park J , Lee J Y , et al. CBAM: Convolutional Block Attention Module[C]// European Conference on Computer Vision. Springer, Cham, 2018.
- [29]Yang F, Zhang L, Yu S, et al. Feature pyramid and hierarchical boosting network for pavement crack detection[J]. IEEE Transactions on Intelligent Transportation Systems, 2019, 21(4): 1525-1535.
- [30]Shi Y, Cui L, Qi Z, et al. Automatic road crack detection using random structured forests[J]. IEEE Transactions on Intelligent Transportation Systems, 2016, 17(12): 3434-3445.
- [31]Eisenbach M, Stricker R, Seichter D, et al. How to get pavement distress detection ready for deep learning? A systematic approach[C]. 2017 international joint conference on neural networks (IJCNN), 2017: 2039-2047.
- [32]Zou Q, Cao Y, Li Q, et al. CrackTree: Automatic crack detection from pavement images[J]. Pattern Recognition Letters, 2012, 33(3): 227-238.
- [33]Amhaz R, Chambon S, Idier J, et al. Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection[J]. IEEE Transactions on Intelligent Transportation Systems, 2016, 17(10): 2718-2729.
- [34]Chambo, S : Aiglern

[https://www.irit.fr/Sylvie.Chambon/Crack/Detection/\[EB/OL\]](https://www.irit.fr/Sylvie.Chambon/Crack/Detection/[EB/OL]).

- [35] Pak M, Kim S. Crack detection using fully convolutional network in wall-climbing robot[C]. Advances in Computer Science and Ubiquitous Computing: CSA-CUTE 2019, 2021: 267-272.
- [36] Ham S, Bae S, Kim H, et al. Training a semantic segmentation model for cracks in the concrete lining of tunnel[J]. Journal of Korean Tunnelling and Underground Space Association, 2021, 23(6): 549-558.
- [37] Liu Y, Yao J, Lu X, et al. DeepCrack: A deep hierarchical feature learning architecture for crack segmentation[J]. Neurocomputing, 2019, 338: 139-153.
- [38] Zhou Z, Siddiquee M, Tajbakhsh N, et al. A nested U-Net architecture for medical image segmentation (2018)[J]. arXiv preprint arXiv:1807.10165.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- [CRCrack.rar](#)