Article:

Shan, J., Huang, Y. orcid.org/0000-0002-1220-6896, Jiang, W. et al. (2 more authors) (2025) GLoU-MiT: Lightweight Global-Local Mamba-Guided U-mix transformer for UAV-based pavement crack segmentation. Advanced Engineering Informatics, 65 (Part D). 103384. ISSN 1474-0346

https://doi.org/10.1016/j.aei.2025.103384

GLoU-MiT: Lightweight Global-Local Mamba-Guided U-Mix Transformer for UAV-based Pavement Crack Segmentation

Jinhuan Shan[a, b], Yue Huang[c], Wei Jiang[a, b, *], Dongdong Yuan[a, b], Feiyang Guo[a, b]

a. Key Laboratory for Special Area Highway Engineering of Ministry of Education, Chang'an University, Xi'an 710064, China

b. School of Highway, Chang'an University, Xi'an 710064, China

c. Institute for Transport Studies, University of Leeds, Leeds LS2 9JT, UK

Email: jhshan@chd.edu.cn (J. Shan), y.huang1@leeds.ac.uk (Y. Huang), jiangwei@chd.edu.cn (W. Jiang), ddy@chd.edu.cn (D. Yuan), 2024021063@chd.edu.cn (F. Guo)

*Corresponding author: Prof. Wei Jiang, jiangwei@chd.edu.cn

Abstract: The utility of Unmanned Aerial Vehicles (UAVs) for routine pavement distresses inspection has been increasingly recognized due to their efficiency, flexibility, safety, and low-cost automation. However, UAV-acquired high-altitude images present unique challenges for deep learning-based semantic segmentation models, such as minute crack details, blurred boundaries, and high levels of environmental noise. We propose GLoU-MiT, a lightweight segmentation model designed to address the difficulties of UAV-based pavement crack segmentation. Our model integrates a U-shaped Mix Transformer architecture for efficient hierarchical feature extraction, a Global-Local Mamba-Guided Skip Connection for improved feature alignment and computational efficiency, and a Boundary/Semantic Deep Supervision Refinement Module to enhance segmentation precision in complex scenarios. Extensive experiments on UAV-Crack500, CrackSC and Crack500 datasets demonstrate that GLoU-MiT effectively improves segmentation accuracy, particularly in low-contrast and complex background environments, making it a robust solution for UAV-based pavement crack inspection tasks. Furthermore, inference speed and energy consumption evaluations conducted on the Jetson Orin Nano (8GB) show that our model achieves an excellent balance between accuracy, energy efficiency, and speed. The code will be released at: https://github.com/SHAN-JH/GLoU-MiT.

Keywords: Pavement crack, Vision mamba, Vison Transformer, Semantic segmentation, Skip connection

1. Introduction

During the operation of pavement, early micro-cracks often emerge in the pavement structure under the dual influence of vehicular loads and climatic variations, as well as the underlying geological conditions [1–4]. Although these initial cracks may not significantly impact the usability of the pavement, their progression, especially under the combined effects of rainwater penetration and recurrent vehicular pressures, can rapidly evolve into various forms of distresses such as potholes, subsidence, and scouring, severely undermining the overall performance of asphalt pavements. Therefore, early detection and timely intervention of such pavement distresses are crucial, not only reducing the maintenance costs but also effectively extending the lifespan of the pavement [5]. However, the tasks of regular road inspection and accurate diagnosis of pavement distresses demand substantial manpower and financial resources from road maintenance departments.

With significant breakthroughs in computer science in recent years, especially the efficiency and efficacy demonstrated by artificial intelligence in handling labor-intensive and repetitive tasks, there are new avenues for achieving automated, high-precision, and cost-effective pavement disease detection [6–10]. This includes, but is not limited to, automated inspections using drones [11–13] or autonomous vehicles [14–16], and distresses identification using advanced algorithms like object detection and semantic segmentation [17–19].

Accurate maintenance decision-making relies heavily on high-quality disease data, which in turn depends on efficient automated detection technologies and precision image capturing strategies. For instance, drones, known for their flexibility and high level of automation, can work in conjunction with automated charging stations to facilitate continuous inspection operations [20]. However, there are three main challenges in semantic segmentation of pavement distresses like cracks using drone-captured images: class imbalance, irregularity of edges, and scene noise. The constraint of safe flying altitude results in a lower proportion of cracks in the captured images, thereby limiting the performance of detection algorithms. Additionally, pavement crack areas, characterized by random, sparse, and diverse pixel compositions along with uncommon textures and edges, further increase the difficulty of accurate crack segmentation. Most current detection algorithms and datasets are designed for close-range photographed images (Fig. 1 (b)), hence applying them directly to high-altitude drone images yields suboptimal results. Inappropriate choices of loss functions or network architecture might prevent the model from effectively capturing minute cracks, and in extreme cases, the model might predict all pixels as background, resulting in entirely black images. Moreover, the limitations in flying altitude mean that the captured images contain considerable noise and are heavily influenced by environmental factors such as changes in lighting, which makes it challenging to achieve effective segmentation results with limited training datasets (Fig. 1 (a)).

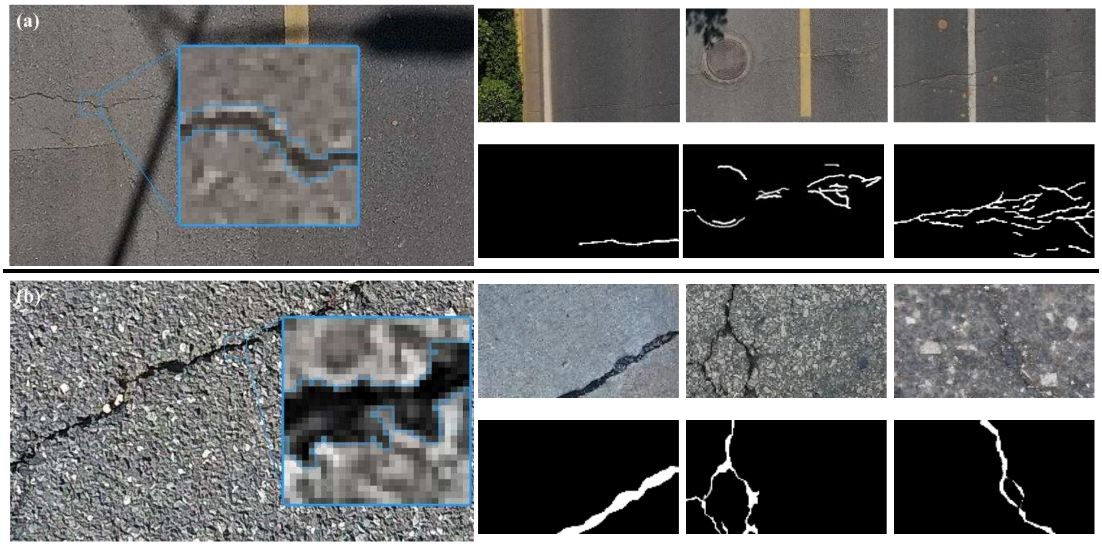


Fig. 1 (a) UAV-captured image of UAV-Crack500 dataset, and (b) Phone-captured image of Crack500 dataset

The introduction of Vision Transformers significantly enhances the model's capability to capture global contextual information, allowing it to transcend the limitations of a local perspective and greatly improve its ability to detect cracks in complex scenarios [21–24]. However, the quadratic computational complexity of Vision Transformers significantly restricts their deployment and application on edge devices. Mamba [25], as an efficient sequence modeling architecture, has attracted considerable attention after being introduced into the visual domain. With its linear computational complexity and robust long-range dependency modeling, Mamba has demonstrated outstanding performance in various visual tasks. Despite these advantages, Vision Mamba's application in crack semantic segmentation remains limited. Recent studies have explored different Mamba-based architectures for crack detection, aiming to balance efficiency and segmentation accuracy. ULNet [26] introduced a cross-visual Mamba feature extraction module and a frequency-domain feature extraction branch, enhancing fine crack detection while maintaining low computational costs. CrackMamba [27] leveraged VMambaV2 as the encoder and introduced a Snake Scan module, which reshapes crack feature sequences based on their natural development patterns, improving feature extraction for complex crack structures. MambaCrackNet [28] integrated Vision Mamba with depthwise separable residual convolutions, forming a hybrid CNN-Mamba segmentation network, which significantly improved crack detection accuracy while maintaining robustness to patch size and training sample variations. SCSegamba [29] further optimized Mamba-based segmentation by proposing a Structure-Aware Visual State Space module, which combines Gated Bottleneck Convolution (GBC) and a Structure-Aware Scanning Strategy (SASS) to enhance morphological feature modeling and semantic continuity of cracks, achieving high segmentation accuracy with only 2.8M parameters and demonstrating excellent real-world deployment performance.

Despite these advancements, existing studies indicate that Vision Mamba has not yet achieved performance on par with traditional Convolutional Neural Networks (CNNs) and Transformers in crack segmentation tasks [30]. One of the key reasons for this limitation is the unique nature of crack segmentation. Slender cracks in complex environments require both long-range dependency modeling to differentiate them from background textures and local feature extraction to precisely delineate their boundaries. Current Mamba-based models struggle to balance these two aspects effectively, leading to performance gaps compared to CNNs and Transformers. Therefore, further architectural improvements are necessary to fully unlock Mamba's potential for crack segmentation. Another limitation of Mamba-based crack segmentation models is their lack of evaluation on edge devices. This is primarily because Mamba relies on custom CUDA kernels for its scan operations, which are difficult to track and optimize via TorchScript. As a result, pure Mamba models exhibit slower inference speeds on resource-constrained devices, limiting their practicality for real-time crack detection and UAV-based inspections.

To address these challenges, we propose integrating Mamba with traditional CNN and Transformer models, leveraging their complementary strengths. CNNs excel at capturing local texture and edge details, while Transformers efficiently model global dependencies. By incorporating Mamba into this hybrid framework, our model can enhance long-range contextual understanding while preserving fine structural details, achieving a better balance between accuracy and computational efficiency in complex crack segmentation scenarios. We first introduce a U-shaped segmentation framework based on Mix Transformer, which leverages the hierarchical structure and efficient self-attention mechanisms of SegFormer for feature extraction, progressively upsampling and using skip connections to restore crack details. Next, a lightweight Global-Local Mamba-Guided Skip Connection based on Vision Mamba is employed to progressively filter out redundant information from the encoder, reducing the dimensionality of feature maps through direct addition operations, thereby lowering computational complexity in the decoder. Finally, a Boundary/Semantic Deep Supervision Refinement Module is integrated to refine crack boundaries and semantic information, enhancing the model's performance on UAV images. We conduct a comparative analysis of the commonly used close-range pavement defect datasets (Crack500 [29] and CrackSC [21]) and our collected UAV-based pavement defect dataset (UAV-Crack500 [30]). This comparison helps to understand the distributional differences between datasets and provides a theoretical foundation for model improvements.

Our contributions can be summarized as follows:

(1) We propose GLoU-MiT, a lightweight and efficient UAV-based pavement crack segmentation model. Built on a U-shaped Mix Transformer framework, it balances local and global feature extraction while reducing computational cost, making it suitable for edge deployment.

(2) We introduce a Global-Local Mamba-Guided Skip Connection to enhance feature alignment while reducing computational complexity. By progressively filtering redundant encoder details and directly adding feature maps instead of concatenation, this mechanism improves both efficiency and segmentation accuracy.

(3) To refine crack segmentation, particularly in low-contrast or complex backgrounds, we integrate a Boundary/Semantic Deep Supervision Refinement Module. This module enhances fine-grained crack boundary detection and semantic consistency, leading to improved F_1-score and Crack IoU, especially for thin and indistinct cracks.

(4) Extensive experiments on UAV-Crack500, CrackSC, and Crack500 datasets demonstrate moderate improvements in F_1-score and Crack IoU while maintaining high efficiency. Furthermore, inference speed and energy consumption evaluations on Jetson Orin Nano (8GB) confirm the model's practical deployment feasibility.

2. Related Works

To address the challenges of minute crack details, blurred boundaries, and high levels of environmental noise in crack segmentation tasks, deep learning model design has primarily focused on three key enhancements. First, multi-scale feature extraction and fusion techniques are employed to effectively capture cracks of varying sizes and intricate patterns[31–34]. Second, advanced attention mechanisms are integrated to highlight critical crack regions and suppress irrelevant background noise[35–38]. Third, boundary refinement strategies are developed to improve the precision of segmentation along crack edges, ensuring accurate delineation even in complex scenarios[39,40].

These design considerations have become essential for advancing the performance of deep learning models in crack segmentation. This section summarizes these foundational improvements while introducing the design philosophy of the proposed model.

2.1 Multi-scale Feature Extraction and Fusion

In Convolutional Neural Networks (CNNs), feature extraction transforms raw input data into higher-level representations, aiding subsequent higher-level tasks. This extraction is typically achieved through the convolution, pooling, and other basic modules of the backbone network, enabling understanding of higher-level semantics. Efficient backbone models are crucial for effective feature extraction and high-level semantic representation. Downsampling in backbones, though reducing parameter count and boosting robustness, also diminishes feature map dimensions. For semantic segmentation tasks, mapping high-level segmentation back to original image sizes without losing edge and detail information is essential. FCN [41] first combined features from different stages with transposed convolutional upsampling for end-to-end per-pixel semantic segmentation. U-Net [42], with skip connections, conjoins encoder and decoder stage feature maps, enabling the decoder to relearn details lost during encoding. DeepLabv3+ [43] utilized Atrous Spatial Pyramid Pooling to grasp multi-scale contextual information, merging high-level with low-level feature maps for better edge and detail detection. The skip connections in U-Net not only serve to obtain richer feature information during the decoding stage but also help alleviate the vanishing gradient problem during training, accelerate network convergence, and preserve detailed information in images. These functions of skip connections are what enable U-Net to perform exceptionally well in tasks such as medical image segmentation.

In this paper, we enhance the skip connections by integrating Local and Global Mamba into the skip connections at different layers, allowing the model to fully leverage the information from the encoder. This approach facilitates the extraction of effective local and global information, thereby improving the accuracy of crack segmentation in complex backgrounds.

2.2 Enhancements in Attention Mechanisms and Long-Range Dependency

In Convolutional Neural Networks (CNNs), a sequence of convolutional and nonlinear layers is employed for feature extraction from a global receptive field. However, this method traditionally treats all input regions uniformly, potentially leading to substantial noise in complex backgrounds and thereby impairing network efficacy. Deep learning has integrated attention mechanisms, inspired by the neural attention processes of the human brain, to address this issue. These mechanisms allow for differential weighting of features, focusing the network on more effective feature representations while inhibiting less discriminative ones. This approach effectively minimizes distractions from background noise and irrelevant areas, consequently augmenting model performance.

Attention mechanisms can be divided into categories like channel attention, spatial attention, and self-attention, depending on their area of focus. Squeeze-and-Excitation Networks (SENet) [44] apply global average pooling and fully connected layers to discern channel-wise feature dependencies. The Bottleneck Attention Module (BAM) [45] combines channel and spatial attentions, effectively enhancing feature extraction without augmenting network depth. The Convolutional Block Attention Module (CBAM) [46] integrates and decouples spatial and channel attentions, improving computational efficiency. Originating in natural language processing, self-attention mechanisms have been adeptly transposed to the realm of computer vision. These mechanisms, through queries, keys, and values, assign varying weights based on inter-feature relationships across different positions, thus aiding models in more effectively capturing contextual data within sequences. The Swin Transformer [47] incorporates a window-based self-attention mechanism, blending traditional CNN structures with attention strategies. This model leverages hierarchical attention mechanisms to assimilate both global and local image information, markedly enhancing its performance. Vision Mamba [48–50] is built upon a state space model (SSM) architecture, which has been adapted for vision tasks. This model incorporates a form of linear attention, making it efficient for processing high-resolution images and handling long-range dependencies.

In this paper, we introduce Global-Local Mamba-Guided Skip Connection. This approach leverages the long-range dependency capabilities and linear complexity of the Mamba model, thereby enhancing the model's spatial information perception efficiently without significantly increasing computational overhead. This strategy ensures that the model can effectively capture both local details and global context, improving overall performance in handling complex scenarios.

2.3 Refinement of Segmentation Edges

In the field of deep learning, particularly for semantic segmentation tasks, downsampling is crucial for extracting high-level semantic features. However, this process typically results in the loss of edge detail information. High-level semantic features, while effective for classifying categories, suffer from low resolution. In contrast, lower-level features, although higher in resolution and capable of generating sharp, detailed boundaries, also contain significant background noise. Thus, bridging the information flow between high and low-level features is essential for achieving precise edge segmentation. To effectively combine low and high-level information, focusing on edge information through specific convolutional architectures and loss functions is also crucial for improving boundary segmentation precision. Gated-SCNN [51] employs a dual-branch structure to separately process semantic and edge information, incorporating boundary loss to enhance edge definition in the prediction output. The Boundary-Aware Segmentation Network (BASNet) [52] combines a prediction network with a refinement module, using a hybrid loss function to capture predictions at various resolutions and post-refinement losses. Given that edge points often exhibit uncertainty in segmentation predictions (with confidence levels around 0.5), PointRend [53] identifies these uncertain points in coarse segmentation maps. It then re-predicts these points using a Multi-Layer Perceptron (MLP) that integrates both coarse and detailed features, thereby refining the edges. Deformable Convolution [54] introduces a learnable offset into its receptive field, enhancing the flexibility of convolution to align with actual boundary shapes more closely, thus improving feature extraction and boundary prediction capabilities. Additionally, the boundary loss [55] proposed by Hoel Kervadec et al. focuses on minimizing the interface area between segmentation boundaries and ground truth, thereby enhancing the network's capability in contour space prediction.

Inspired by deep supervision and refinement techniques, we effectively amalgamate boundary and semantic details from various refinement layers using a boundary/semantic fusion head, thereby substantially improving the performance of model to discern boundary and semantic nuances at different scales. This advancement not only bolsters crack segmentation accuracy but also systematically mitigates the interference caused by environmental noise pixels.

3. Proposed Architecture

3.1 Preliminaries

State Space Model (SSM): The state space model is a mathematical framework used to describe the representation of the current state and predict future states based on given inputs. Specifically, the model derives a predicted output function $y(t) \in \mathrm{R}$ from the continuous input function $x(t) \in \mathrm{R}$ and the hidden state representation $h(t) \in \mathrm{R}^N$, as shown in Equation (1).

$$\begin{cases} h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t) \\ y(t) = \mathbf{C}h(t) + \mathbf{D}x(t) \end{cases} \quad (1)$$

where $h'(t)$ is the time derivative of the state $h(t)$, indicating how the state evolves over time; $\mathbf{A} \in \mathrm{R}^{N\times N}$ is the state transition matrix, determines how the hidden state updates over time.; $\mathbf{B} \in \mathrm{R}^{N\times 1}$ is the input control matrix, defining how the input influences the state; $\mathbf{C} \in \mathrm{R}^{1\times N}$ is the output matrix, which translates the state to the output; $\mathbf{D} \in \mathrm{R}$ is the direct transmission matrix, representing the direct influence of the input on the output; N represents the latent state dimension. The former part of Equation (1) is referred to as the **state equation**, while the latter part is called the **output equation**. $\mathbf{D}x(t)$ directly influences the output $y(t)$ by bypassing the state variable $h(t)$, in a manner similar to a shortcut connection. Consequently, SSM further simplifies Equation (1) and omits $\mathbf{D}$ (or equivalently, sets $\mathbf{D=0}$), as shown in Equation (2).

$$\begin{cases} h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t) \\ y(t) = \mathbf{C}h(t) \end{cases} \tag{2}$$

Discretization: In Equation (2), the input is a continuous time-based signal. However, since real-world data is typically discrete, it is necessary to derive an equivalent equation in the discrete-time domain, as shown in Equation (3).

$$\begin{cases} h_k = \overline{\mathbf{A}}h_{k-1} + \overline{\mathbf{B}}x_k \\ y_k = \overline{\mathbf{C}}h_k \end{cases} \tag{3}$$

where h_k is the hidden state, representing the system's state at time step *k*; x_k is the input signal, representing the input provided to the system at time step *k*; y_k is the output signal, representing the output computed from the hidden state h_k; $\overline{\mathbf{A}}$ is the discrete state transition matrix; $\overline{\mathbf{B}}$ is the discrete input control matrix; $\overline{\mathbf{C}}$ is the discrete output matrix.

Mamba (S6 model) adopted the Zero-Order Hold (ZOH) discretization method to convert the continuous-time state equations into discrete form. The ZOH method assumes that the input remains constant within each sampling interval, effectively transforming the continuous-time system into a discrete-time system by holding the last known input value constant until the next sampling point.

(1) Discrete state transition matrix $\overline{\mathbf{A}}$

Assuming that the input signal remains constant within a time step Δ (ZOH method), i.e., $x(t) = x_k$ ($t \in [k\Delta, (k+1)\Delta]$). The solution for $h(t)$ of Equation (2) can be obtained using the matrix exponential method:

$$h(t) = e^{\mathbf{A}(t-k\Delta)} h_k + \int_0^{t-k\Delta} e^{\mathbf{A}\tau} \mathbf{B} x_k d\tau \tag{4}$$

where τ is integration variable representing continuous time within a single time step.

At $t = (k+1)\Delta$, i.e., after discretization to the next step:

$$h_{k+1} = e^{\mathbf{A}\Delta} h_k + (\int_0^{\Delta} e^{\mathbf{A}\tau} d\tau) \mathbf{B} x_k \tag{5}$$

Thus, according to Equation (3), the discrete transition matrix is:

$$\bar{\mathbf{A}} = e^{\mathbf{A}\Delta} \tag{6}$$

(2) Discrete input control matrix $\bar{\mathbf{B}}$

According to Equation (3) and (5), the discrete input control matrix is:

$$\bar{\mathbf{B}} = (\int_0^{\Delta} e^{\mathbf{A}\tau} d\tau) \mathbf{B} \tag{7}$$

In control theory, this integral has an analytical solution:

$$\int_0^{\Delta} e^{\mathbf{A}\tau} d\tau = \mathbf{A}^{-1}(e^{\mathbf{A}\Delta} - \mathbf{I}) \tag{8}$$

Thus, the discrete input matrix is:

$$\bar{\mathbf{B}} = \mathbf{A}^{-1}(e^{\mathbf{A}\Delta} - \mathbf{I})\mathbf{B} \tag{9}$$

where $\mathbf{I}$ represents the identity matrix. Using the first-order Taylor expansion for $e^{\mathbf{A}\Delta}$ ($e^{\mathbf{A}\Delta} \approx \mathbf{I} + \mathbf{A}\Delta$), matrix $\bar{\mathbf{B}}$ can be further simplified:

$$\bar{\mathbf{B}} = \mathbf{A}^{-1}(e^{\mathbf{A}\Delta} - \mathbf{I})\mathbf{B} \approx \mathbf{A}^{-1}(\mathbf{A}\Delta)\mathbf{B} = \Delta\mathbf{B} \tag{10}$$

(3) Discrete output matrix $\bar{\mathbf{C}}$

In a state space model, the temporal evolution of the system is primarily governed by the state equation, while the output equation serves as an instantaneous mapping and does not influence the state evolution. This equation merely describes how the current state $h(t)$ is mapped to the output $y(t)$, without involving differentiation with respect to t or any temporal accumulation effects. Therefore, it represents a static (instantaneous) linear transformation that does not contribute to the system's temporal dynamics. As a result, the output matrix remains unchanged after discretization:

$$\overline{\mathbf{C}} = \mathbf{C} \tag{11}$$

Initialization ($\mathbf{A}$): In Equation (2), if matrix $\mathbf{A}$ is initialized with random values during training, the model may struggle to achieve optimal results. This is because the next state is not only influenced by the current state but also by prior states. To address this, the High-order Polynomial Projection Operators (HiPPO) method is introduced, which produces a hidden state that effectively memorizes its history. This enhances the model's ability to handle long-range dependencies, allowing it to capture recent tokens efficiently while attenuating the influence of older tokens. Such a design helps the model maintain long-term memory while focusing more on recent information. Based on this, Mamba introduces two simplified initialization methods for both the complex and real cases [56], aiming to optimize the handling of long-range dependencies in different scenarios, as shown in Equation (12).

$$\mathbf{A}_{\text{init}} = \begin{cases} -\dfrac{1}{2} + ni, & \text{S4D-Lin} \\ -(n+1), & \text{S4D-Real} \end{cases} \tag{12}$$

where n is state dimension index of N, and i is imaginary unit ($i^2 = -1$).

Selection Mechanisms ($\mathbf{B}$, $\mathbf{C}$ and Δ): The SSM and S4 model (Structured SSM) exhibit limitations in certain key tasks due to their fixed linear time-invariant (LTI) nature. In these models, the entire historical state is compressed into a single representation, and the state evolution matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ remain static, meaning they cannot dynamically adjust to different inputs. This inherent limitation prevents the model from selectively focusing on or ignoring specific inputs, reducing its adaptability in complex tasks requiring contextual awareness. To overcome this limitation, Mamba (Selective SSM, S6 model) introduces an input-dependent selection mechanism, where

$\mathbf{B}$, $\mathbf{C}$, Δ are dynamically adjusted based on the input sequence, while $\mathbf{A}$ remains fixed. Specifically:

(1) Fixed parameters $\mathbf{A}$:

The matrix $\mathbf{A}$, which governs the evolution of the hidden state, is initialized (Equation (12)) and remains input-invariant throughout training. This design ensures stable memory retention and structured state dynamics, allowing the model to efficiently encode long-range dependencies while maintaining numerical stability. However, $\overline{\mathbf{A}}$ can still adapt indirectly through changes in Δ (Equation (6)).

(2) Input-dependent parameters $\mathbf{B}$, $\mathbf{C}$, Δ:

In contrast, $\mathbf{B}$, $\mathbf{C}$ and Δ are all input-dependent and adapt dynamically, as shown in Equation (13). This mechanism dynamically adjusts the weights over time based on the input, improving the model's efficiency in handling long-range dependencies.

$$\begin{cases} \mathbf{B} = S_{\mathbf{B}}(x) \\ \mathbf{C} = S_{\mathbf{C}}(x) \\ \Delta = \tau_{\Delta}(\mathbf{Parameter} + S_{\Delta}(x)) \end{cases} \tag{13}$$

where, $S_{\mathbf{B}}(x) = \mathbf{Linear}_N(x)$ and $S_{\mathbf{C}}(x) = \mathbf{Linear}_N(x)$ are fully connected layers that project the input to dimension N; τ_{Δ} represents the Softplus activation function, ensuring numerical stability; $\mathbf{Parameter}$ refers to a learnable bias term that is independent of the input sequence x; $S_{\Delta}(x) = \mathbf{Broadcast}_D(\mathbf{Linear}_1(x))$ is a linear transformation applied to x, followed by broadcasting to match the required shape D.

Visual State Space (VSS) Block: Originally, the Mamba model was primarily designed for processing sequential data and demonstrated remarkable capabilities in fields such as natural language processing. To extend its application to the visual domain, researchers have proposed extended models, such as Vim [57], VMamba [48], and LocalMamba [49]. These models transform two-dimensional image data into one-dimensional sequences along various directions, thereby enabling the S6 architecture to be directly applied to visual information processing. Specifically, the Vim model employs a bidirectional scanning mechanism to simultaneously extract contextual information with both forward and backward directions; VMamba introduces a four-directional cross-scanning strategy to comprehensively capture global features from the top, bottom, left, and right; and LocalMamba is designed with a localized scanning strategy, focusing on capturing fine-grained image details.

3.2 Comprehensive Architecture

The following section introduces the lightweight pavement crack segmentation model **GLoU-MiT** we designed, as shown in Fig. 2. The model consists of three main components: (1) a U-shaped segmentation framework based on Mix Transformer for hierarchical feature extraction; (2) a lightweight Global-Local Mamba-Guided Skip Connection based on Vision Mamba for enhanced feature alignment and computational efficiency; and (3) a Boundary/Semantic Deep Supervision Refinement Module to improve segmentation precision.

The U-shaped segmentation framework builds upon SegFormer's Mix Transformer, leveraging its hierarchical structure and efficient self-attention mechanisms to extract multi-scale features. Inspired by U-Net, the model progressively upsamples feature maps in the decoder and incorporates skip connections to restore crack details while maintaining computational efficiency. The detailed implementation of this module is described in Section 3.3.

Instead of direct concatenation, we introduce a Global-Local Mamba-Guided Skip Connection to enhance feature alignment and computational efficiency. This mechanism employs Local Mamba operations to refine fine-grained details and Global Mamba operations to capture long-range dependencies. Additionally, a Cascaded Gating Mechanism selectively filters redundant information while preserving key semantic features, ensuring an efficient fusion of encoder and decoder features. To further reduce computational cost, the skip connection applies channel reduction before feature processing and expands channels back after computation. The detailed implementation of this module is described in Section 3.4.

To enhance segmentation precision, particularly for narrow and low-contrast cracks, we incorporate a Boundary/Semantic Deep Supervision Refinement Module into the decoder. This module improves boundary detection and semantic consistency by leveraging multi-scale deep supervision, deformable convolutions, and attention mechanisms. By integrating both boundary-aware and semantic feature learning, the module enhances fine-grained segmentation. The detailed implementation of this module is described in Section 3.5.

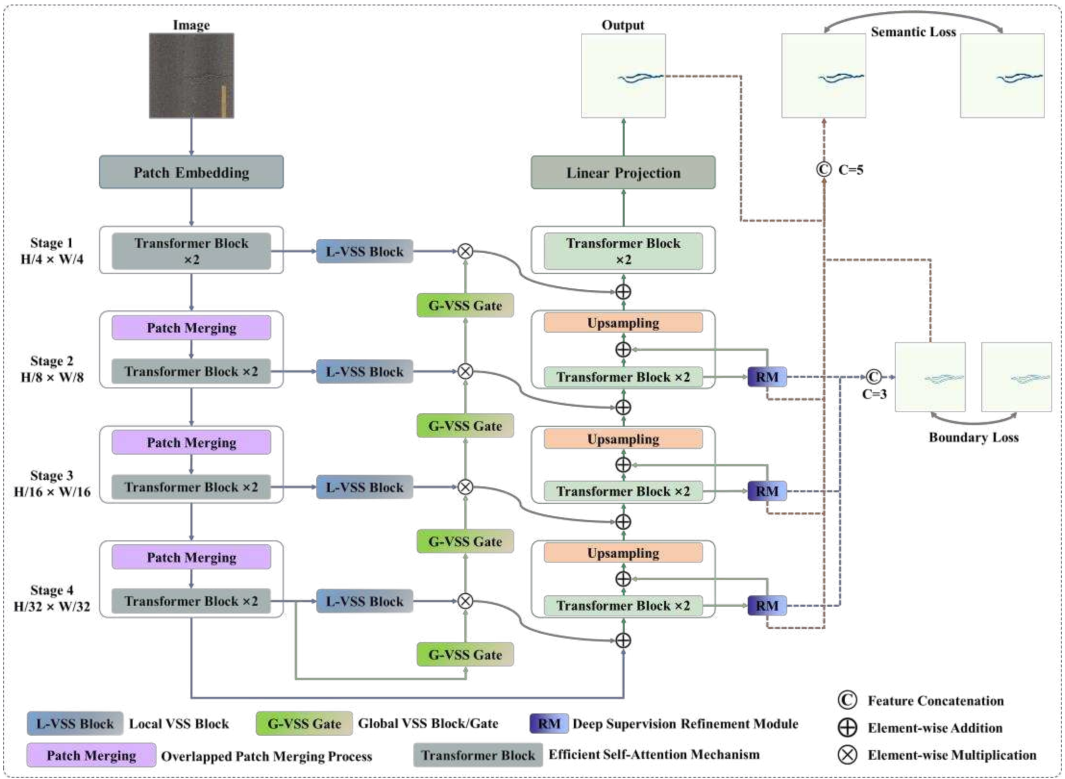


Fig. 2 Comprehensive Architecture

3.3 U-Mix Transformer

SegFormer introduces the Mix Transformer encoder, which effectively leverages Overlapped Patch Merging Process and Efficient Self-Attention Mechanism, combining the strengths of both CNNs and Transformers to capture local and global information. This significantly improves segmentation accuracy. The Efficient Self-Attention mechanism reduces the dimensionality of the Key and Query using convolution, which enhances computational efficiency. However, SegFormer's decoder directly concatenates feature maps from different hierarchical levels and decodes them through an All-MLP architecture. Although this approach reduces computational complexity, the results from concatenating feature maps of different levels tend to be suboptimal. This is because feature maps from different layers in the decoder inherently contain varied information, and directly processing them through MLP cannot fully exploit the features extracted at different levels.

Inspired by U-Net, we propose a symmetric **U-Mix Transformer** model (Fig. 3). Unlike U-Net, to improve computational efficiency, we do not concatenate feature maps at the same resolution. Instead, we directly add them together, followed by a Mix Transformer operation. This approach enhances efficiency while maintaining the ability to capture multi-level features effectively.

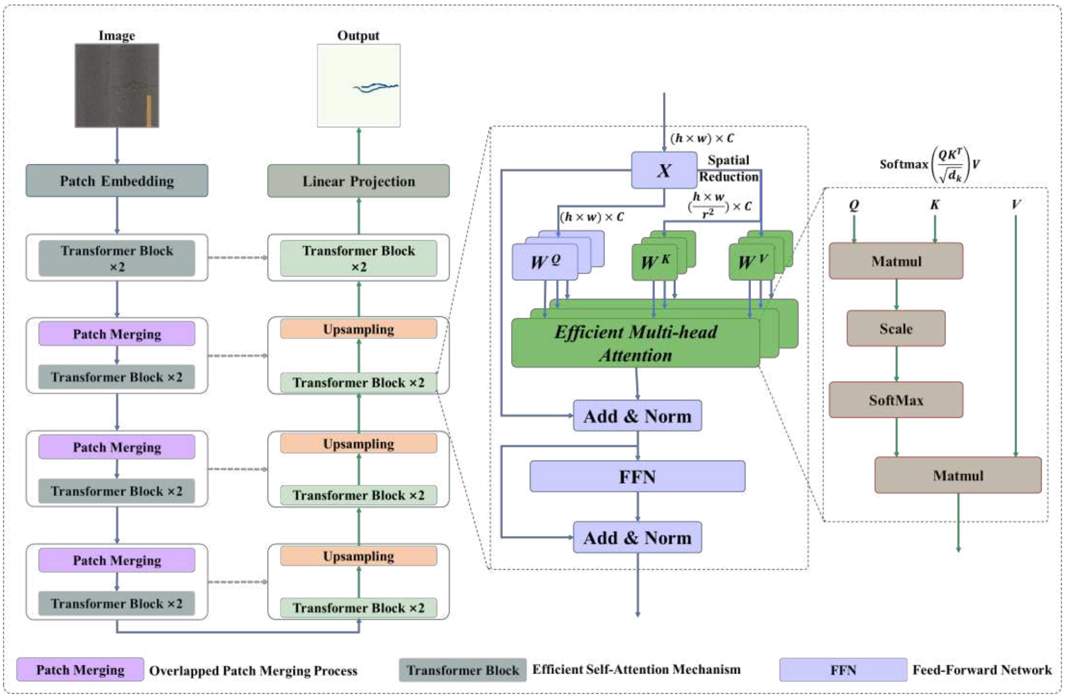


Fig. 3 U-Mix Transformer

3.4 Global-Local Mamba-Guided Skip Connection Module

Although directly adding feature maps of the same resolution but different levels—i.e., semantic-rich and detail-heavy maps—can reduce computational load, the differences in information, such as varying semantic and detail levels, may lead to suboptimal segmentation results. This is because redundant information from shallow layers can interfere with the deep semantic information. Like U-Net, the skip connection concatenates the encoder's feature maps with those of the decoder, but this increases the feature map dimensionality, leading to a quadratic increase in computation for Transformer modules. To address this issue, we propose the **Lightweight Global-Local Mamba-Guided Skip Connection Module** (Fig. 4).

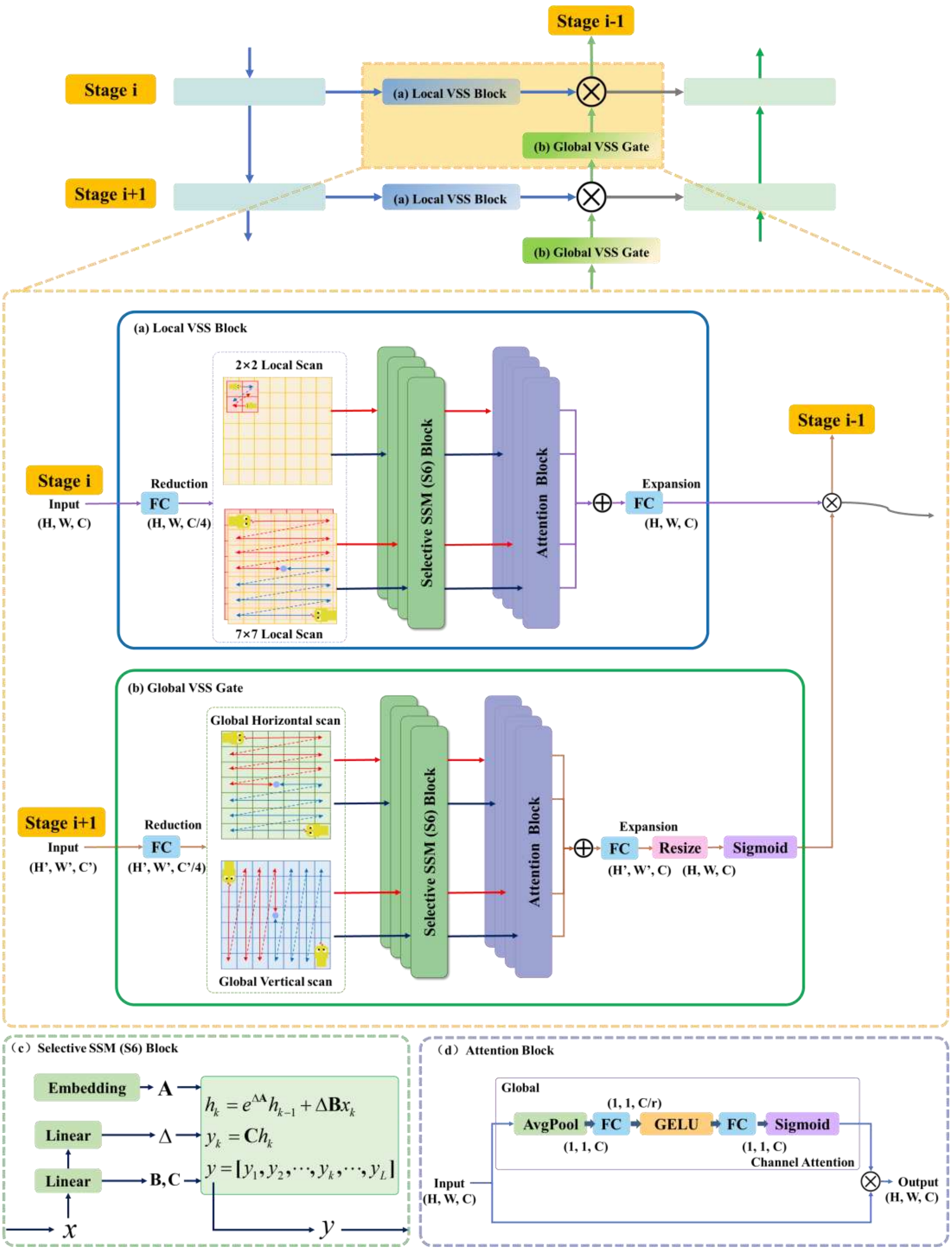


Fig. 4 Lightweight Global-Local Mamba-Guided Skip Connection Module

First, each feature map from different layers of the encoder is processed through local Mamba operations to extract detailed information from different levels.

Specifically, we introduced local horizontal scans with window sizes of 2 and 7 in Stage 1 and Stage 2 to construct Local VSS (L-VSS) Blocks (Fig. 4(a)). Since the window size of 2 inherently includes vertical scanning, we avoided adding additional vertical scans for the size 7 window, reducing computational cost while still capturing vertical dependencies.

To better fuse detailed information with deep semantic information, we propose a **Mamba-based Cascaded Gate Generator**. This module starts from the deep feature maps and progressively applies Global VSS (G-VSS) Blocks to extract global information, incorporating scans in four directions: horizontal (left to right and right to left) and vertical (top to bottom and bottom to top). A Sigmoid Gate is employed as a gating mechanism to control the flow of information at each layer. By dynamically adjusting the propagation of global features through the Sigmoid function, the model multiplies these with the extracted local features, ensuring that important global information is selectively retained while noise and irrelevant parts are suppressed.

Through this gating mechanism, the model effectively suppresses redundant information while maintaining global feature propagation. By cascading this process, the feature maps obtained from the encoder are better aligned with the information hierarchy in the decoder. As a result, the addition of these maps not only reduces computational load but also enhances the model's ability to capture fine details.

To manage computational costs, VSS Blocks are not directly inserted into the skip connections. Instead, we first apply channel reduction, decreasing the channels to 1/4 of their original number. The globally or locally scanned data is then passed into a Selective Scan State Space Models (S6) [25] for computation (Fig. 4(c)), facilitating effective global or local visual representation learning. Finally, the channels are expanded back to their original size. To further enhance feature representation and eliminate irrelevant information, we implement channel attention mechanism operations in LocalMamba [49]. This allows for weighted extraction of critical feature channels by dynamically adjusting their importance based on the input, ensuring that the model focuses on the most informative features while suppressing less relevant ones.

3.5 Deep Supervision Refinement Module

In our preliminary experiments, we observed that although the aforementioned structure improved segmentation accuracy for wider and more distinct cracks, its performance remained suboptimal when dealing with narrow cracks that resemble background pixels or are located in complex backgrounds. To address this issue, we propose the **Deep Supervision Refinement Module** (Fig. 5), which can be integrated into any network that requires semantic and boundary supervision. In this study, the module is inserted before the upsampling operation at each layer.

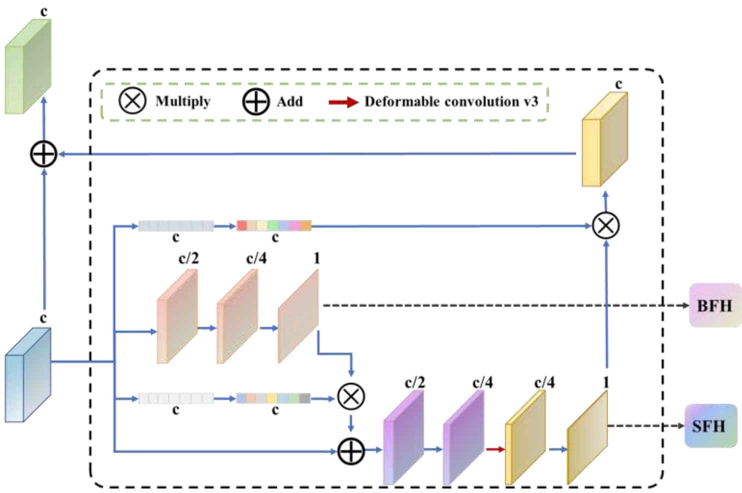


Fig. 5 Deep supervision refinement module

This module generates three outputs: a single-channel boundary feature map, a single-channel semantic feature map, and an upsampled feature map with the same number of channels as the input. First, the input feature map undergoes three convolutions to produce a single-channel boundary feature map, which is then sent directly to the boundary fusion head for further boundary refinement and boundary loss supervision. Additionally, channel boundary attention is computed on the input feature map to determine the boundary attention weights that need to be applied to each channel, and this result is added to the input feature map.

Next, the boundary-enhanced feature map is sent to the semantic calculation module, where two standard convolutions are applied to extract semantic features and reduce the channel dimensions. This is followed by a deformable convolution (Deformable Convolution v3) [42] to capture the non-uniform shapes of crack semantics. The output is a single-channel semantic feature map, which is passed to the semantic fusion head for further crack semantic feature extraction and semantic loss computation. Subsequently, semantic attention is calculated for the input feature map, weighted using the semantic feature map, and then added to the input feature map. Finally, the feature map is upsampled to seamlessly connect with the next layer's module.

The main idea behind this module is that feature maps after convolution and downsampling contain rich semantic information, but because their dimensions are smaller than the original image, directly upsampling them may result in inaccurate boundary information. By performing boundary and semantic supervision based on high-level semantic information, this module refines the boundary and semantic details at each layer. Additionally, a shortcut branch adds the supervised output back to the original feature map, preserving the original information while enhancing boundary and semantic channel supervision. This module can be integrated into any intermediate feature map calculation to supervise both boundary and semantic information. To capture boundary and semantic information at different scales, it is recommended to apply the module before each upsampling operation.

Furthermore, the boundary and semantic feature maps extracted at each layer are fed into the boundary and semantic fusion head for further refinement (Fig. 6). First, the boundary prediction result is upsampled to the original image size and concatenated, followed by a standard convolution layer and a deformable convolution layer, which refine the boundary segmentation information. The boundary-refined result is then concatenated with the semantic prediction result, passed through another standard convolution layer and deformable convolution layer, and the boundary information is used to constrain the semantic information, resulting in the final refined crack segmentation output.

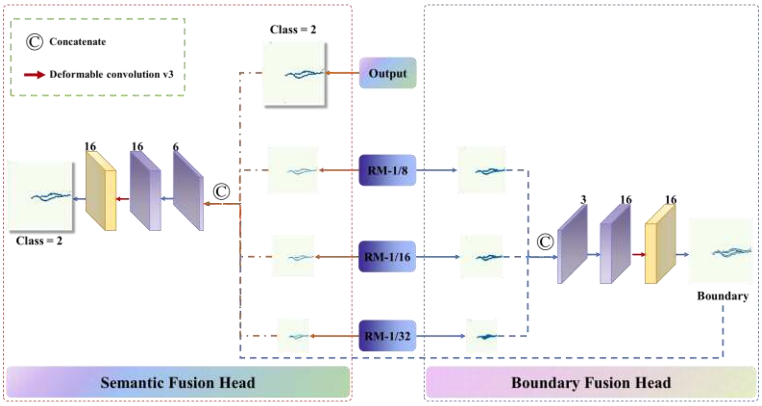


Fig. 6 Boundary and Semantic Fusion Head

4. Experimental Details

4.1 Datasets

To evaluate the versatility of data acquisition methods for pavement inspection and to validate the robustness and adaptability of the proposed algorithm, this study used three distinct datasets: our UAV-Crack500 [58], which is based on long-distance pavement distresses images captured by drones, the Crack500 [59], which is based on close-distance taken with cell phones, and the CrackSC [23], which features pavement cracks captured by cell phones in the presence of complex background noise.

For the UAV-Crack500 dataset, given urban scenario flight altitude restrictions, the drone flying altitude was set to 50 meters. To ensure high precision and efficient image data collection covering at least a width of three lanes in one direction, a camera with 4× zoom capabilities was used at a flying speed of 2.5 m/s. The collected images have a resolution of 2688 × 1512 pixels, corresponding to a ground coverage area of 16 m × 9 m, which equates to an actual size of 6 mm × 6 mm per pixel (Fig. 7). Due to the minor proportion that cracks occupy in drone images, directly training on full-resolution images results in the model being biased toward background predictions. To alleviate this issue and improve the proportion of crack pixels within each sample, we adopted a uniform, non-overlapping cropping strategy that divides each image into 16 equal-sized blocks of 672 × 378 pixels. These patches are then filtered to exclude those without visible cracks. A total of 500 image patches with representative and complex crack scenarios were meticulously selected and annotated. The selection process emphasized diversity and real-world complexity, including the presence of road markings, shadows, curbstones, trees, manhole covers, and road dividers (Fig. 1(a)). The dataset was randomly split into a training set (250 images), a validation set (50 images), and a test set (200 images). As depicted, the dataset presents elongated, fine-grained crack structures, where crack widths are often only a few pixels wide, embedded within complex urban environments.

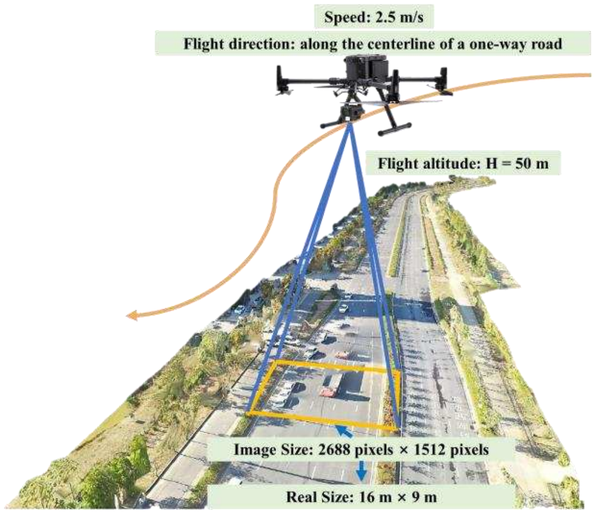


Fig. 7 Flight parameters for long-distance pavement monitoring

The Crack500 dataset comprises 500 high-resolution images of pavement captured at close range using a cell phone camera, each with dimensions of 2000 × 1500 pixels. For practicality in model training and evaluation, these images have been subdivided into 16 non-overlapping regions. The dataset is stratified into different subsets for the purposes of model development and performance assessment: 1896 regions are allocated for training, 348 for validation, and 1124 for testing. Visual inspection of the dataset reveals that the images feature relatively wide cracks, which occupy a more significant proportion of the image area compared to those in UAV-based datasets. Additionally, the images in the Crack500 dataset exhibit minimal variation in lighting conditions and are less affected by environmental noise (Fig. 1(b)).

The CrackSC dataset is a newly introduced pavement crack image dataset designed to address the challenges of detecting cracks in local roads with heavy shadows and dense crack formations, commonly found in low-maintenance areas. The dataset consists of 197 images of pavement surfaces collected using an iPhone 8 around Enoree Ave, Columbia, SC. Since the dataset is provided as a whole, we randomly split it into training, validation, and test sets in a 5:1:4 ratio to facilitate model evaluation.

4.2 Implementation Details

Our models were developed based on the MMSegmentation v1.2.0 framework [60], which provides a standardized and modular implementation for semantic segmentation. To ensure a fair and reliable comparison, all models, including our proposed model and the comparison models, were trained and evaluated under the same hardware and experimental settings. Specifically, all models were trained and inferred on an NVIDIA Tesla T4 GPU (16GB), with only the backbone and head components modified, while all other experimental settings remained identical. The backbone parameters were initialized using pre-trained weights from the official repository, ensuring that each model benefited from a strong initialization aligned with its architecture.

During training, we adopted a batch size of 16 and a learning rate of 6e-5, training each model for 30,000 iterations. To improve the generalization capability of the models, we applied data augmentation techniques, including flipping, rotation, color jittering, and random size cropping. The cropped images were then resized to 256 × 256 pixels before being used for training. Additionally, all models shared the same data preprocessing pipeline, training strategy, optimizer settings, loss function, learning rate schedule, and evaluation metrics, ensuring that the observed performance differences were solely attributed to the architectural variations rather than training discrepancies. To facilitate efficient model training, the study adopts AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $weight\ decay = 0.01$) as the optimizer of choice due to its effectiveness in handling sparse gradients and its adaptive learning rate capabilities, which are conducive for faster convergence. Additionally, a two-stage learning rate scheduling strategy was implemented. In the initial 1,500 iterations, a linear warm-up was applied, gradually increasing the learning rate from 1e-6 to the base learning rate 6e-5. Subsequently, a polynomial learning rate decay (power = 1.0) was adopted, ensuring a linear reduction in the learning rate from iteration 1,500 to 30,000, eventually reaching 0.0 at the end of training.

4.3 Loss Functions

The prevalent class imbalance, marked by a minor fraction of crack pixels compared to the overall image, severely hinders the capacity of models to discern crack features using a conventional Binary Cross-Entropy (BCE) loss function. This challenge is further pronounced in images obtained via UAVs, where the proportion of crack pixels is notably diminutive, leading to predictions with excessively fine or interrupted cracks, or in extreme cases, a complete bias towards background classification. To counteract this, Weighted Binary Cross-Entropy (*w*BCE) and Dice Loss have been advocated as effective loss functions to tackle the class imbalance issue.

The *w*BCE approach involves differential weighting for the positive (crack) and negative (background) classes, incentivizing the model to focus more on the sparsely represented crack class (Equation (14)). Although *w*BCE can mitigate the class imbalance problem, its effectiveness heavily depends on the weight parameter adjustment, often requiring extensive experimentation to determine the optimal setting for ensuring model stability and generalization.

$$\ell_{wBCE} = -\frac{1}{N}\sum_{i=0}^{N}(w_1 \cdot y_i \cdot \log(y_i) + w_0 \cdot (1-y_i) \cdot \log(1-y_i)) \tag{14}$$

where N represents the number of image pixels; w_1 represents the weight for the positive samples; w_0 represents the weight for the negative samples; y_i represents the actual probability of the positive samples; y_i represents the predicted probability of the positive samples. When $w_0 = w_1 = 1$, ℓ_{wBCE} reduces to the standard Binary Cross-Entropy (BCE) loss ℓ_{BCE}.

The Dice Loss effectively measures the degree of overlap between the predicted segmentation mask and the ground truth. It is inherently robust to class imbalance (Equation (15)). However, it is prone to gradient instability during training, especially when the intersection of the segmentation masks is minimal, potentially leading to oscillatory behaviors or convergence issues in the learning process.

$$\ell_{Dice} = 1 - \frac{2\sum_{i=0}^{N} y_i y_i}{\sum_{i=0}^{N} y_i^2 + \sum_{i=0}^{N} y_i^2} \tag{15}$$

To address the respective shortcomings of traditional BCE, *w*BCE, and Dice Loss, this study follows the approach in [61–65] and adopts a combined BCE and Dice loss strategy as the overall semantic loss function ℓ_s (Equation (16)). This method has been demonstrated to achieve better segmentation performance, particularly for fine crack detection.

$$\ell_s = \ell_{BCE} + \ell_{Dice} \tag{16}$$

Inspired by the CE2P model [66], we propose an approach to address boundary loss by first detecting the boundaries between different semantic regions through comparing the semantic categories of adjacent pixels in the segmentation map. These boundaries are then marked as edges. To further refine the edge information, a dilation operation is applied, which widens the edges and produces an expanded edge map (with an edge width of 4). This wider edge representation helps the model capture and learn complex semantic boundary information more effectively. Given that boundary pixels constitute a smaller proportion of the total pixels, we assign a weight of 20 (α=20) to the boundary BCE loss ℓ_b, following the empirical setting in the PIDNet model [67], before combining it with the semantic loss to form the final loss function (Equation (17)):

$$\ell = \ell_s + \alpha \ell_b \tag{17}$$

4.4 Evaluation Metrics

This study employs a suite of metrics for model evaluation, comprising pixel accuracy (PA), precision (Pr), recall (Re), F_1-score (F_1), and Crack Intersection over Union (IoU). Pixel accuracy quantifies the proportion of pixels correctly classified by the model relative to the total pixel count, serving as a measure of the model's overall classification efficacy (Equation (18)). Precision is defined as the ratio of true positive predictions to the total number of positive predictions made by the model, gauging the precision with which the model discerns positive cases (Equation (19)). Recall is the ratio of true positive predictions to the actual number of positive instances, evaluating the model's proficiency in detecting all positive cases (Equation (20)). The F_1-score, a weighted harmonic mean of precision and recall, balances the two metrics for a holistic performance assessment (Equation (21)). The Crack IoU metric assesses segmentation accuracy by calculating the ratio of the intersection to the union of the predicted and actual crack regions (Equation (22)).

$$\mathrm{PA} = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

$$\mathrm{Pr} = \frac{TP}{TP + FP} \tag{19}$$

$$\mathrm{Re} = \frac{TP}{TP + FN} \tag{20}$$

$$\mathrm{F}_1 = 2 \times \frac{\mathrm{Pr} \times \mathrm{Re}}{\mathrm{Pr} + \mathrm{Re}} \tag{21}$$

$$\mathrm{IoU} = \frac{TP}{TP + FP + FN} \tag{22}$$

where *TP* (True Positive) refers to the case when the actual class is positive and the model predicts it as positive; *TN* (True Negative) refers to the case when the actual class is negative and the model predicts it as negative; *FP* (False Positive) refers to the case when the actual class is negative but the model incorrectly predicts it as positive; *FN*

(False Negative) refers to the case when the actual class is positive but the model incorrectly predicts it as negative.

Params (the number of parameters) and FLOPs (floating point operations) are used in this paper as common metrics to evaluate the parameter complexity and computational complexity of the model. Params refers to the total number of trainable parameters in the model, typically including weights and biases, which are used to measure the model's storage requirements and memory consumption during training. FLOPs represent the number of floating-point operations required during a single inference, which reflects the computational resources needed by the model to process data.

Due to the unclear boundaries of cracks, along with the subjectivity and lack of repeatability in manual annotations, this study, following other research [68–70], also adopts a 2-pixel tolerance. To enhance the model's ability to accurately segment cracks, we use the approach where a prediction is considered positive if it falls within the 2-pixel dilated region of the ground truth.

4.5 Reference Evaluation on Jetson Orin Nano

This study systematically evaluates the deployment performance of deep learning models on the NVIDIA Jetson Orin Nano (8GB) platform (Fig. 8), which delivers 40 TOPS of AI computational power—an 80-fold increase compared to the NVIDIA Jetson Nano. Due to the limited support for ONNX and TensorRT operators in certain model implementations, we adopted two deployment strategies utilizing PyTorch Just-In-Time (JIT) at FP32 precision: Python-based PyTorch inference and C++-based LibTorch inference. The JIT compilation process converts dynamic models from Python environments into the TorchScript format, enabling cross-platform serialization and optimized execution. Each deployment approach offers unique advantages. Python-based PyTorch inference features short development cycles and flexible iterations, making it ideal for rapid prototyping and validation. Conversely, C++-based LibTorch inference is optimized for latency-sensitive applications and resource-constrained environments. Performance evaluation was conducted under a standardized protocol, which included setting the Jetson platform to maximum performance mode with the highest operating frequency. Each test consisted of a 10-sample warm-up phase followed by 50 inference samples. The evaluation metrics included Energy Per Sample (EPS) (Equation (23)), Inference Latency Per Sample (LPS) (Equation (24)), Throughput (Equation (25)), and the Energy Delay Product (EDP) (Equation (26)).

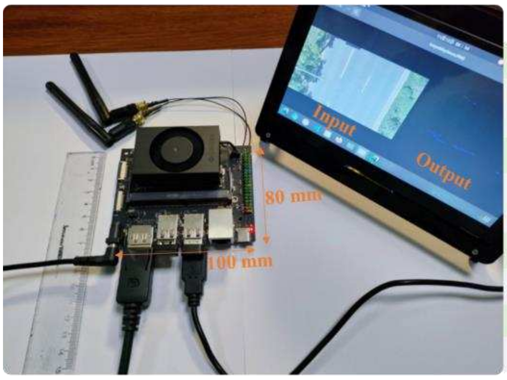
$$\mathrm{EPC} = \frac{\sum (V(t) \times I(t) \times \Delta t)}{N} \tag{23}$$

$$\mathrm{LPS} = \frac{\sum (T_{\text{inference\_end}} - T_{\text{inference\_start}})}{N} \tag{24}$$

$$\mathrm{Throught} = \frac{N}{T_{\text{total\_end}} - T_{\text{total\_start}}} \tag{25}$$

$$\mathrm{EDP} = \mathrm{EPS} \times \mathrm{LPS} \tag{26}$$

where $V(t)$ and $I(t)$ are real-time voltage and current measurements from the Jetson INA3221 sensor; N is the total number of samples; $T_{\text{inference\_end}}$ and $T_{\text{inference\_start}}$ represent the end and start times of the model inference phase only; $T_{\text{total\_end}}$ and $T_{\text{total\_start}}$ represent the end and start times of the complete pipeline of data preprocessing, model inference, and post-processing stages. During the training process, the model employed a crop size of (256, 256). To ensure consistency, the same crop size was applied during inference on edge devices. For test images with a resolution of (512, 512), we utilized a non-overlapping sliding window approach to divide the image into four patches. These patches were simultaneously input as a single batch for model inference. Finally, the prediction results were reassembled to restore the original image size, ensuring consistency and completeness of the output.



Jetson Orin Nano (8G) Specifications

AI Performance	40TOPS
GPU	1024-core NVIDIA Ampere architecture GPU with 32 Tensor Cores
GPU Frequency	625MHz (Max)
CPU	6-core Arm® Cortex®-A78AE v8.2 64-bit CPU, 1.5MB L2 + 4MB L3
CPU Frequency	1.5GHz (Max)
Memory	8GB 128-bit LPDDR5, 68GB/s
Storage	128GB NVMe Solid State Drive
Power	7W ~ 15W

Fig. 8 NVIDIA Jetson Orin Nano (8GB) platform and specifications

5. Results and Discussion

5.1 Quantitative Evaluation

To rigorously evaluate the effectiveness of our model, we conducted a comparative analysis against state-of-the-art semantic segmentation models. These included CNN-based models (U-Net [42], SegNeXt [71], RHACrackNet [72], TV-net (U-NetSmall) [58], CDS-Net [73]), Transformer-based models (SegFormer [74], U-MixFormer [75]), as well as advanced mamba-based models (LocalVMamba [49], Manba-UNet [50], SCSegamba [29]). Tables 1 to 3 summarize the performance comparison of state-of-the-art methods on UAV-Crack500, CrackSC, and Crack500 datasets. Table 4 presents a comparison of model parameters and FLOPs with a fixed input size of 3×512×512 during testing.

Table 1 Performance comparison with the state-of-the-art methods on UAV-Crack500

Method	PA	Pr	Re	F_1	IoU
U-Net	99.12	89.52	70.65	78.97	65.25
SegNeXt-T	99.21	91.25	75.11	82.40	70.06
RHACrackNet	99.11	87.96	72.44	79.45	65.91
TV-Net	99.12	87.42	73.23	79.70	66.25
CDS-Net	99.11	87.52	72.83	79.50	65.97
SegFormer-MiT-B0	**99.21**	90.51	74.57	81.77	69.16
U-MixFormer-MiT-B0	**99.22**	90.39	75.36	82.20	69.77
SCSegamba	99.00	**91.31**	64.31	75.47	60.60
LocalVMamba-T	99.16	89.56	73.09	80.49	67.36
Mamba-UNet	98.92	87.80	61.40	72.26	56.57
GLoU-MiT (Ours)	99.19	89.32	76.49	82.41	70.08
GLoU-MiT-DS (Ours)	99.20	88.82	**77.89**	**83.00**	**70.94**

Table 2 Performance comparison with the state-of-the-art methods on CrackSC

Method	PA	Pr	Re	F_1	IoU
U-Net	98.76	86.56	59.85	70.77	54.76
SegNeXt-T	98.63	88.53	51.28	64.94	48.08
RHACrackNet	98.70	87.02	56.78	68.72	52.35
TV-Net	98.76	86.62	60.49	71.24	55.32
CDS-Net	98.78	88.29	59.26	70.92	54.94
SegFormer-MiT-B0	98.81	**88.65**	61.91	72.90	57.36
U-MixFormer-MiT-B0	**98.84**	88.48	64.05	74.31	59.12
SCSegamba	98.64	88.29	51.70	65.21	48.38
LocalVMamba-T	98.72	87.44	57.42	69.32	53.04
Mamba-UNet	98.34	82.41	29.04	42.94	27.34
GLoU-MiT (Ours)	98.80	87.64	64.35	74.21	59.00
GLoU-MiT-DS (Ours)	98.82	87.71	**64.78**	**74.52**	**59.39**

Table 3 Performance comparison with the state-of-the-art methods on Crack500

Method	PA	Pr	Re	F_1	IoU
U-Net	97.48	**83.81**	72.57	77.79	63.65
SegNeXt-T	97.43	79.55	78.63	79.08	65.40
RHACrackNet	97.29	79.63	74.93	77.21	62.87
TV-Net	97.42	80.33	76.44	78.34	64.39
CDS-Net	97.47	82.45	74.28	78.15	64.14
SegFormer-MiT-B0	97.50	81.28	77.07	79.12	65.45
U-MixFormer-MiT-B0	97.56	81.21	78.46	79.81	66.40
SCSegamba	97.33	82.04	71.96	76.61	62.09
LocalVMamba-T	97.55	82.15	76.57	79.26	65.64
Mamba-UNet	97.41	81.20	75.02	77.98	63.91
GLoU-MiT (Ours)	97.59	80.79	**80.39**	**80.59**	**67.49**
GLoU-MiT-DS (Ours)	**97.62**	82.40	77.72	79.99	66.66

Table 4 Efficiency comparison

Method	Params (M)	Flops (G)
U-Net	29.0	203.0
SegNeXt-T	4.2	6.3
RHACrackNet	1.7	7.3
TV-Net	17.8	87.3
CDS-Net	7.2	48.5
SegFormer-MiT-B0	3.7	7.9
U-MixFormer-MiT-B0	6.4	5.2
SCSegamba	3.1	23.5
LocalVMamba-T	56.2	230.7
Mamba-UNet	19.2	1.5
GLoU-MiT (Ours)	6.6	6.5
GLoU-MiT-DS (Ours)	7.0	7.2

UAV-Crack500: This dataset consists of aerial images of pavement captured by drones. Due to the high altitude of capture, the images have relatively low resolution, with fine cracks and low contrast between the pavement and cracks, making it challenging to accurately segment the cracks. Although U-Net and Mamba-UNet perform well in medical imaging, their segmentation performance on this dataset is suboptimal. Our proposed GLoU-MiT model surpasses existing state-of-the-art models based on CNNs, transformers, and Mamba architecture, outperforming the advanced Vision Mamba model, LocalVMamba, with a 1.98% improvement in F_1-score and a 2.72% increase in Crack IoU. After incorporating the DS module, GLoU-MiT-DS achieves additional gains of 0.59% in F_1-score and 0.86% in Crack IoU with a slight increase of approximately 0.4M parameters and 0.7 GFLOPs.

CrackSC: The CrackSC dataset contains narrow cracks with significant environmental interference. Strong models like SegNeXt-T and Mamba-UNet perform worse than the traditional U-Net on this dataset. This is likely because U-Net applies convolutions and downsampling directly at higher resolutions, which is advantageous for detecting small cracks. However, U-Net's convolution and downsampling at the original resolution result in a substantial overhead in parameters and FLOPs. The lightweight model U-MixFormer, which is carefully designed, achieved promising results. In comparison, our GLoU-MiT model experienced a slight performance decrease of approximately 0.1% in both F_1 and IoU. However, after incorporating the DS supervision module, GLoU-MiT showed significant improvements, with F_1 and IoU increasing by 0.21% and 0.27%, respectively.

Crack500: In Crack500, the cracks occupy a larger portion of the image and the background is relatively simple, leading to smaller performance differences among various SOTA models. Under these circumstances, traditional U-Net is outperformed by more lightweight models. Our GLoU-MiT model achieves the best results, improving upon the advanced transformer model U-MixFormer by 0.78% in F_1 and 1.09% in IoU. However, after adding the DS supervision module, the segmentation performance decreased. This could be attributed to the edge supervision in the DS module, which is generated by dilating the edges by 4 pixels. This approach may have had a negative impact on the segmentation of clearly defined crack boundaries.

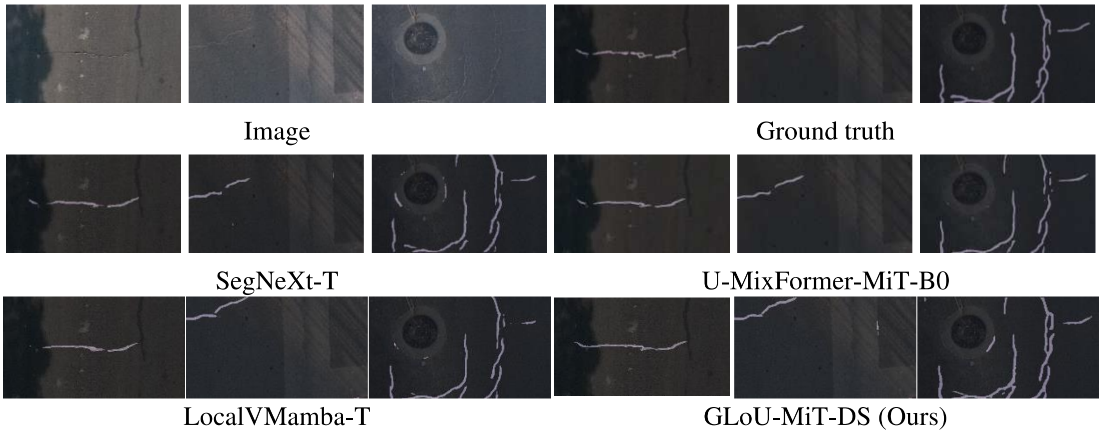


Fig. 9 Qualitative results on UAV-Crack500

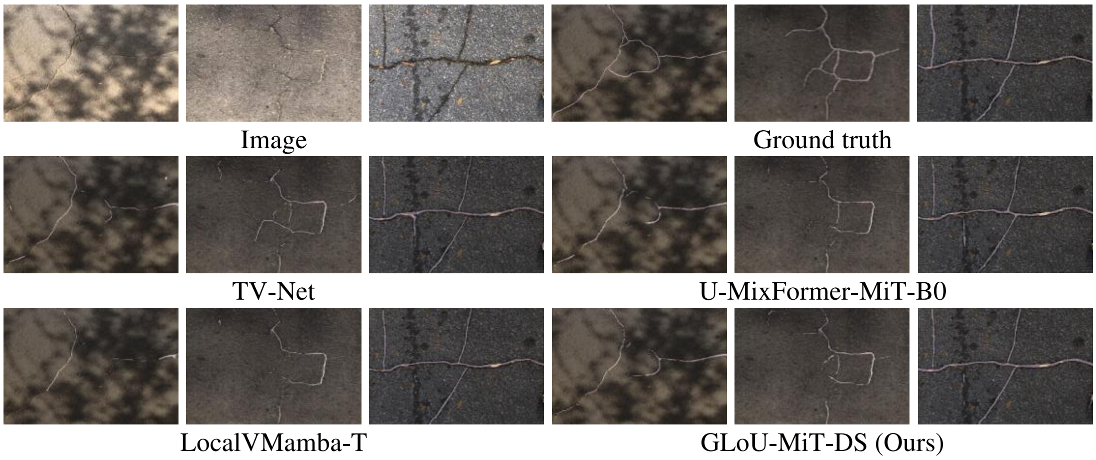


Fig. 10 Qualitative results on CrackSC

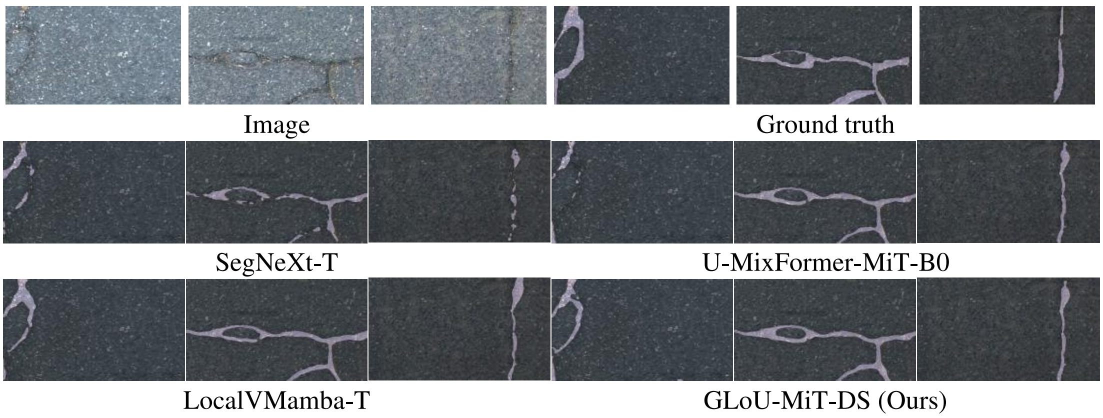


Fig. 11 Qualitative results on Crack500

5.2 Qualitative Evaluation

We selected three relatively challenging pavement images from each of the three datasets, and chose the best-performing models from the CNN, Transformer, and

Mamba network architectures for visualizing the segmentation results, as shown in Figs. 9-11. From these figures, it is evident that our model excels at distinguishing crack and pavement pixels in complex scenes. For narrow and elongated cracks, segmentation models often encounter discontinuities. However, our model performs better than other models in producing continuous crack segmentations, significantly improving segmentation accuracy.

5.3 Energy-Delay Performance Evaluation

As shown in Fig. 12, compiling the TorchScript model into C++ using LibTorch significantly improves inference performance and efficiency by eliminating the additional overhead and dynamic scheduling issues associated with the Python interpreter. The results in Fig. 12(a) demonstrate that LibTorch-based inference significantly reduces energy consumption per sample across most models, primarily due to its static thread management strategy, which efficiently utilizes hardware resources and minimizes thread switching and synchronization overhead.

However, for lightweight models such as SegFormer, U-MixFormer, and GLoU-MiT, where computational loads are lower, the CUDA cores on the Jetson device are not fully utilized. In these cases, the fixed threading model of LibTorch introduces task scheduling overhead, resulting in an EDP that is not always significantly lower than Python inference (Fig. 12(d)). Conversely, in heavier models like Mamba-UNet and LocalVMamba, where computational resources are fully leveraged, LibTorch's performance advantage becomes more pronounced.

Among all models, Mamba-UNet exhibits the highest latency and energy consumption, despite having lower FLOPs, due to the computationally expensive Mamba operations applied on high-resolution inputs. In contrast, SegFormer achieves a balance between inference speed and energy efficiency. Our proposed GLoU-MiT model demonstrates competitive inference speed and energy consumption compared to SegNeXt-T, but achieves higher segmentation accuracy across all datasets.

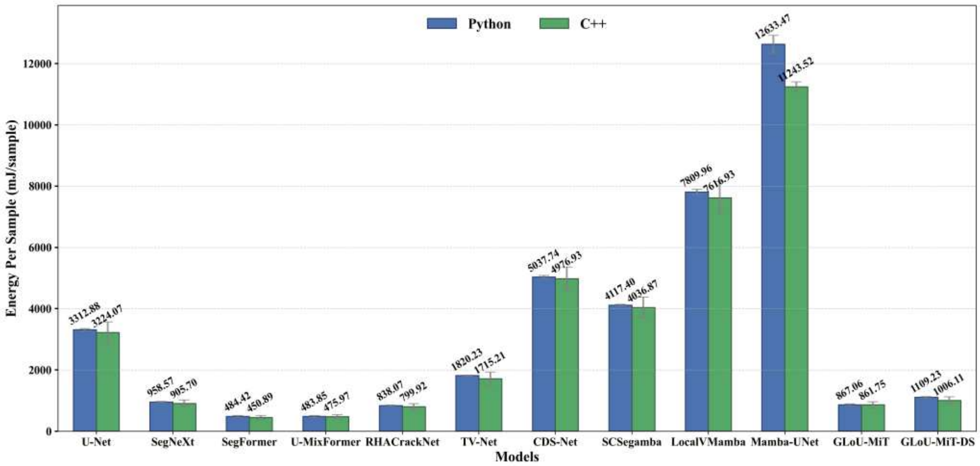
The Energy-Delay Product (EDP), a comprehensive measure of inference efficiency, shows that after compilation into C++, GLoU-MiT achieves an EDP of 8.06×10^4, which is 9.3% lower than SegNeXt-T (8.89×10^4), while improving the F_1-score on the Crack500 dataset by 1.51%. Although the EDP is slightly higher compared to U-MixFormer-MiT-B0 (2.56×10^4), GLoU-MiT improves the F_1-score on the Crack500 dataset by 0.78%, demonstrating a favorable trade-off between accuracy and efficiency.

With the incorporation of the Deep Supervision Refinement (DS) module, the EDP of GLoU-MiT-DS increases to 10.77×10^4, primarily due to the additional computation introduced by boundary and semantic supervision. However, this results in a notable improvement in segmentation performance, increasing the F_1-score on the UAV-Crack500 dataset to 83% (a 0.59% improvement over GLoU-MiT) and boosting the F_1-score on the CrackSC dataset from 74.21% to 74.52%.

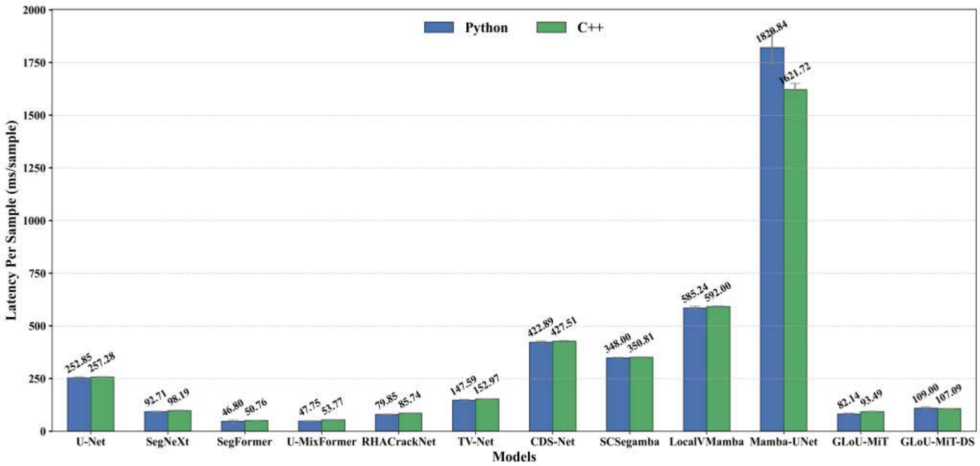
These findings suggest that while LibTorch inference generally reduces energy consumption, its efficiency gains vary depending on model complexity and workload distribution. Moreover, the DS module significantly enhances crack boundary segmentation, demonstrating the trade-off between accuracy and computational overhead in UAV-based crack detection scenarios.

Integrating the findings from Sections 5.1 to 5.3, it becomes evident that the number of parameters and FLOPs is not necessarily indicative of a model's real-world inference efficiency. Instead, inference performance is jointly determined by architectural design choices, the distribution and parallelizability of computation, and the ability to mitigate execution bottlenecks. For example, although Mamba-based models demonstrate favorable parameter efficiency, they exhibit suboptimal inference speed on edge devices. This discrepancy can be attributed to the current lack of mature GPU-level parallelization and hardware-specific optimization support for Mamba operators.

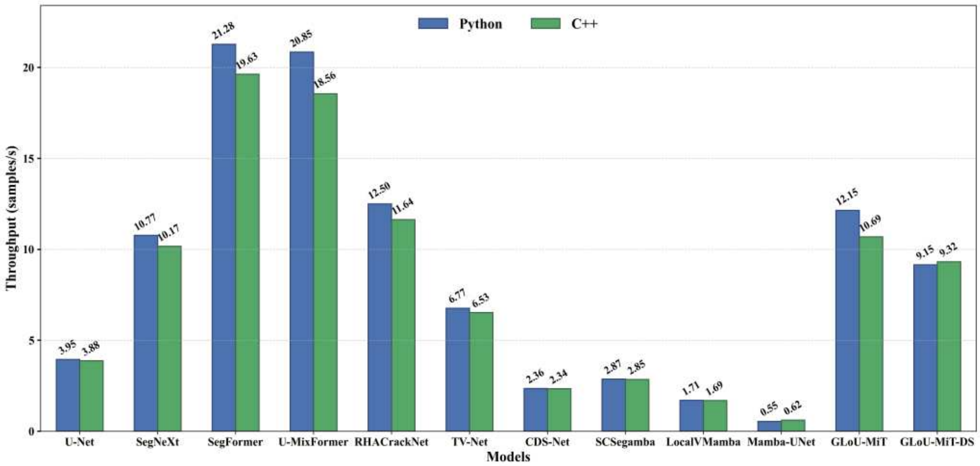
Therefore, the extensive deployment of Mamba modules is not advisable in latency-critical edge scenarios. Instead, our proposed architecture adopts a more pragmatic and effective approach by integrating Mamba modules within skip connections. This selective incorporation strategy enables the architecture to retain the modeling strengths of Mamba while mitigating its negative impact on inference latency. Furthermore, as Mamba currently lacks comprehensive GPU-oriented optimization and support strategies, future efforts to enhance inference speed may benefit from the integration of conventional model compression techniques, including quantization, pruning, and knowledge distillation.



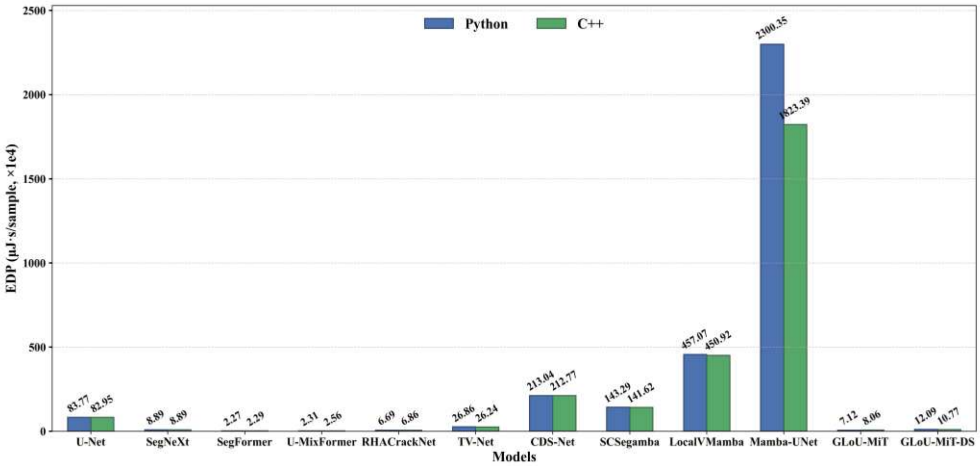
(a) Energy Per Sample (EPS)



(b) Latency Per Sample (LPS)



(c) Throughput



(d) Energy Delay Product (EDP)

Fig. 12 Energy-Delay efficiency of Python and C++ on Jetson Orin Nano (8G)

5.4 Ablation Studies

The carefully designed skip connections can effectively integrate low-level feature maps rich in detail with feature maps containing high-level semantic information. To validate the superiority of our model, we explored various designs for skip connections and proposed four different types (Fig.13):

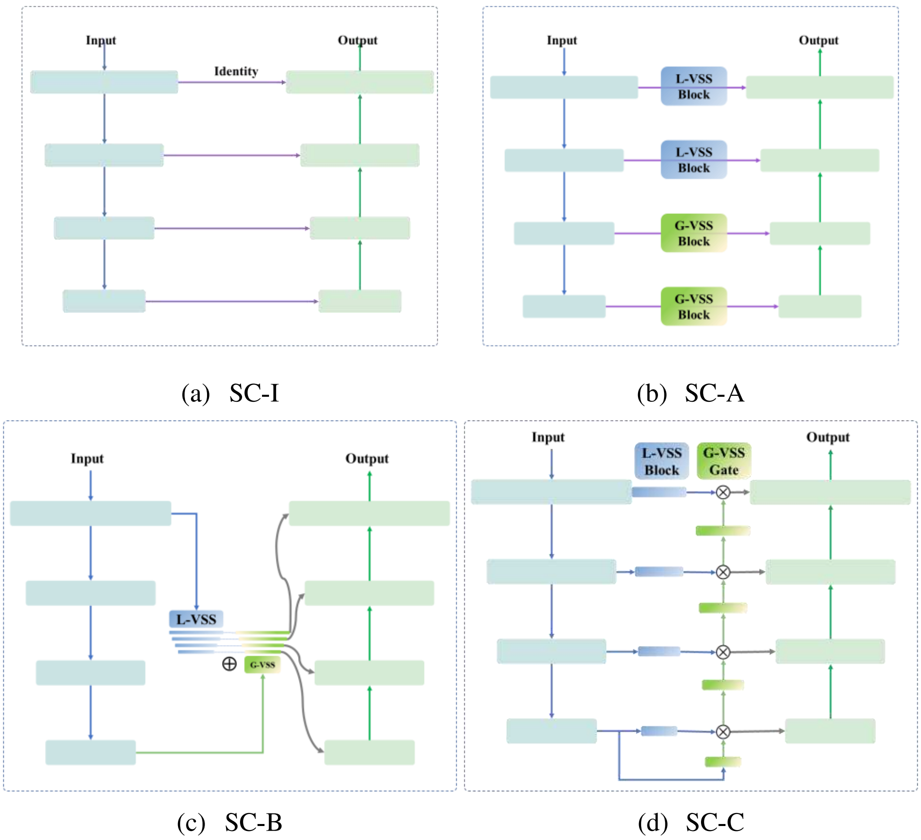
(1) **SC-I**: The feature maps generated by the encoder are directly added to the decoder's feature maps without any additional operations.

(2) **GLo-SC-A**: In the shallow layers (Stage 1 and Stage 2), we use a local VSS block to align the detailed feature maps with the semantic feature maps. In the deeper layers (Stage 3 and Stage 4), a global VSS block is used to align the

semantic-rich feature maps from the encoder with the semantic feature maps in the decoder.

(3) **GLo-SC-B**: In this design, the local VSS block is applied to the high-resolution layer (Stage 1) of the encoder to extract local features, while the global VSS block is used for the low-resolution layer (Stage 4) to extract global features. The detailed and semantic feature maps are added directly at corresponding levels.

(4) **GLo-SC-C**: As discussed in Section 3.4, we extract local features from each layer using local VSS blocks, with the low-resolution and semantically rich Layer 4 serving as the guiding layer. Through a progressive gating mechanism, we gradually control the amount of local feature information passed from each layer to its corresponding layer in the decoder.



(a) SC-I (b) SC-A (c) SC-B (d) SC-C

Fig. 13 Visualization of our proposed skip connection (SC) architectures

As shown in Table 5, the performance improves when the Global-Local Mamba skip connection (GLo-SC) modules A, B, and C are added, compared to the identity add method. This indicates that by extracting both global and local features from the skip connections, the semantic representation capability of the feature maps is enhanced, allowing them to align more effectively with the feature maps in the decoder. The GLo-SC-C module achieves the greatest improvement by innovatively integrating hierarchical global feature extraction with an adaptive gating mechanism. This design dynamically balances the incorporation of critical global context with the suppression of redundant or noisy information, thereby enhancing semantic richness and ensuring optimal fusion between encoder and decoder features. As a result, the model attains superior predictive accuracy and improves robustness in capturing fine-grained details.

Table 5 Ablation results on UAV-Crack500, CrackSC and Crack500

Model	Skip Connection				DS	Params (M)	Flops (G)	UAV-Crack500		CrackSC		Crack500	
	I	A	B	C				F_1	IoU	F_1	IoU	F_1	IoU
U-MiT	√					6.2	6.2	80.98	68.04	71.31	55.41	80.01	66.68
GLoU-MiT-A		√				6.4	6.3	81.65	68.99	71.98	56.22	80.11	66.82
GLoU-MiT-B			√			6.6	6.3	81.39	68.62	72.22	56.52	79.78	66.36
GLoU-MiT				√		6.6	6.5	82.41	70.08	74.21	59.00	**80.59**	**67.49**
U-MiT-DS	√				√	6.6	7.0	81.51	68.79	71.91	56.14	79.56	66.05
GLoU-MiT-DS-A		√			√	6.7	7.1	81.69	69.05	73.19	57.71	80.08	66.77
GLoU-MiT-DS-B			√		√	6.9	7.1	82.32	69.95	72.62	57.01	79.90	66.53
GLoU-MiT-DS				√	√	7.0	7.2	**83.00**	**70.94**	**74.52**	**59.39**	79.99	66.66

Furthermore, we applied deep supervision to the different skip connection models mentioned above by introducing a Boundary/Semantic Deep Supervision Refinement Module. The results show significant improvement, particularly for low-resolution, low-contrast datasets such as UAV-Crack500, and datasets with complex backgrounds such as CrackSC. This improvement is attributed to the addition of boundary and semantic supervision at each layer, which helps accelerate the learning of semantic information and enhances the representation of deeper features. As a result, the model can better understand and distinguish complex scenes or objects. However, for datasets where cracks are relatively obvious (e.g., Crack500), the addition of deep supervision leads to a decline in performance. This may be because the cracks are already prominent, and the base model structure is sufficient to capture the necessary features. In this case, deep supervision may over-constrain the learning of intermediate layers, thereby negatively affecting the overall performance.

5.5 Visualization of Activation Maps

To understand the impact of our designed skip connection modules on model performance, we employed LayerCAM [76] to visualize the class activation maps of two images from the publicly available CrackSC and Crack500 datasets at different stages of our proposed GLoU-MiT models and the SegFormer model. Specifically, we visualized the feature maps in stage 1 (high-resolution layer) of our designed model, with the visualization locations shown in Fig. 14. In this figure, F_1 represents the output of stage 1 in the encoder, F_2 is the output after the local VSS block, F_3 is the output from the previous layer followed by the global VSS block, F_4 is the output after applying sigmoid to the element-wise multiplication of F_2 and F_3, and F_5 is the final output of the decoder (corresponding to the MLP stage in SegFormer).

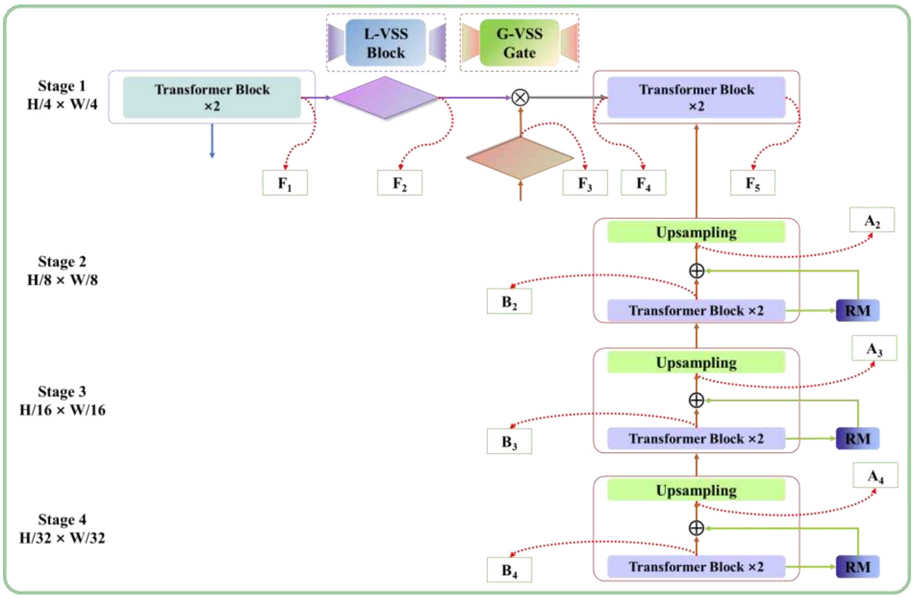
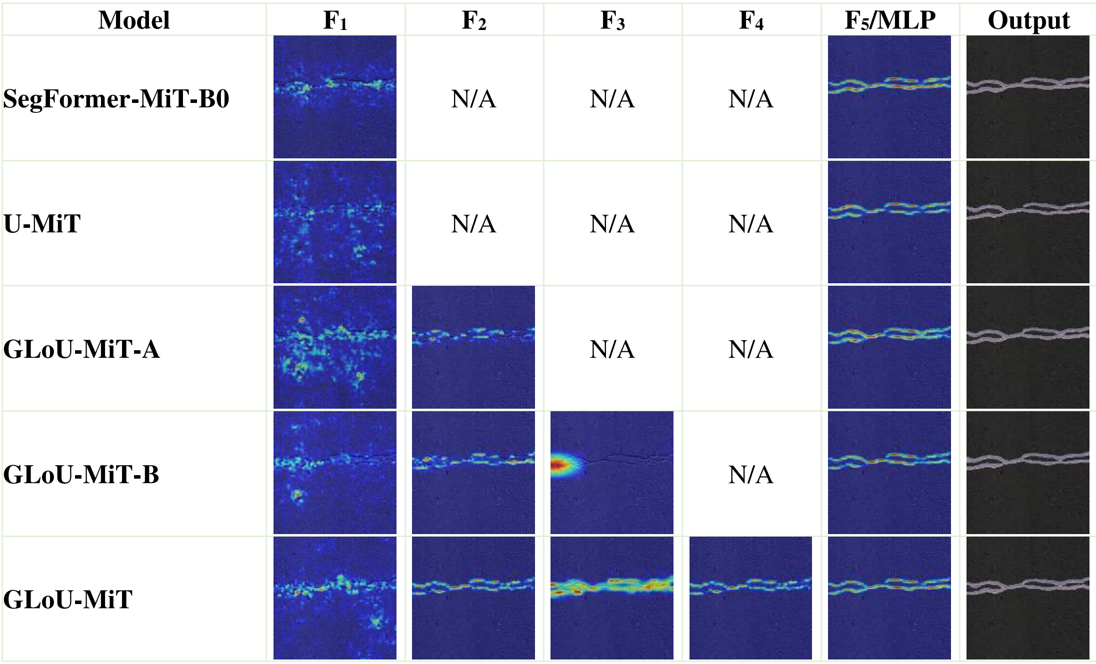
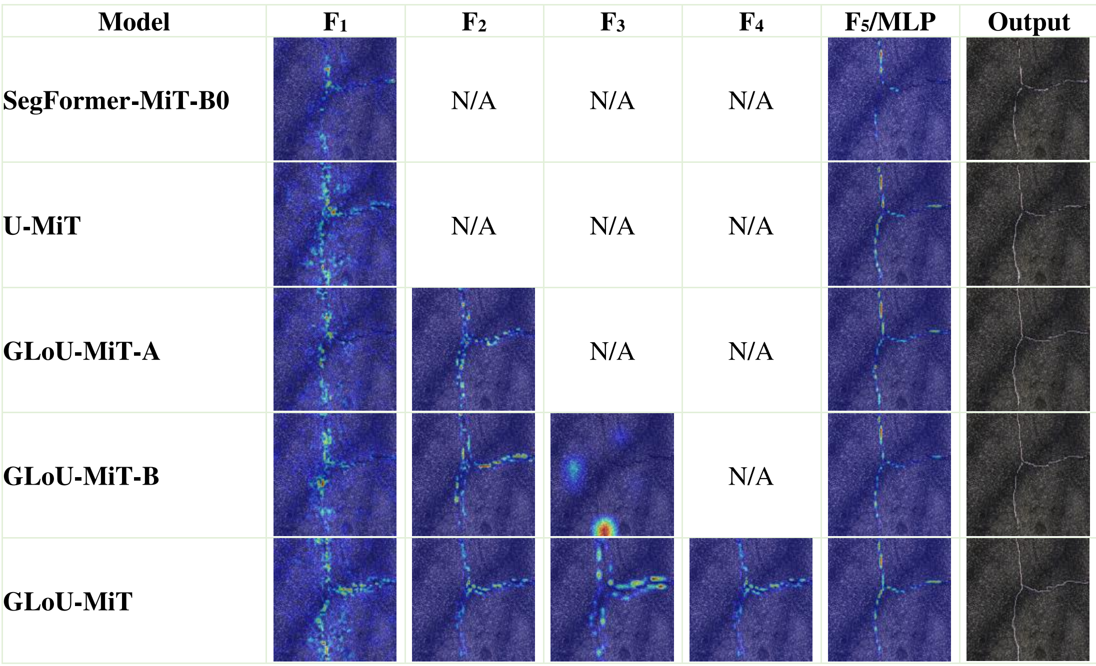


Fig.14 Diagram of LayerCAM visualization positions

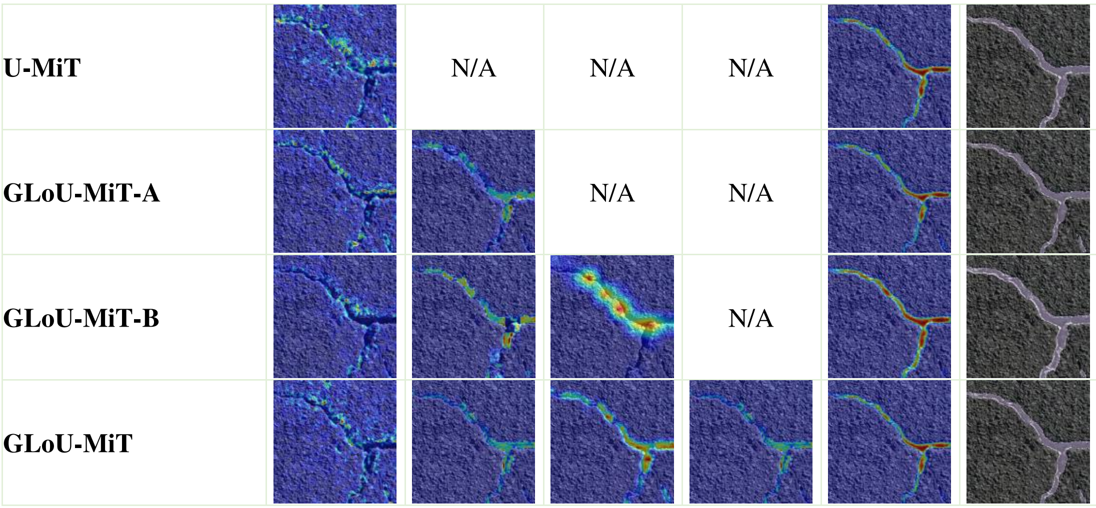
As can be seen from the Fig.15, the presence of environmental interference results in suboptimal activation of crack regions during the shallow stage (F_1), where only low-level features are extracted, leading to significant noise around the edges. After introducing the local VSS block, the activation maps F_2 become more focused around the crack regions. Compared to GLoU-MiT-B (SC-B) model that directly compute global features in stage 4, GLoU-MiT (SC-C), which gradually guides the feature extraction process through the global VSS gate, shows improved edge activation in its feature maps F_3. The feature maps F_4 processed by the global VSS gate reveals more comprehensive crack region activation. As a result, the final segmentation map demonstrates better continuity and improved performance in segmenting fine cracks in complex backgrounds.



(a) Image from the UAV-Crack500 dataset



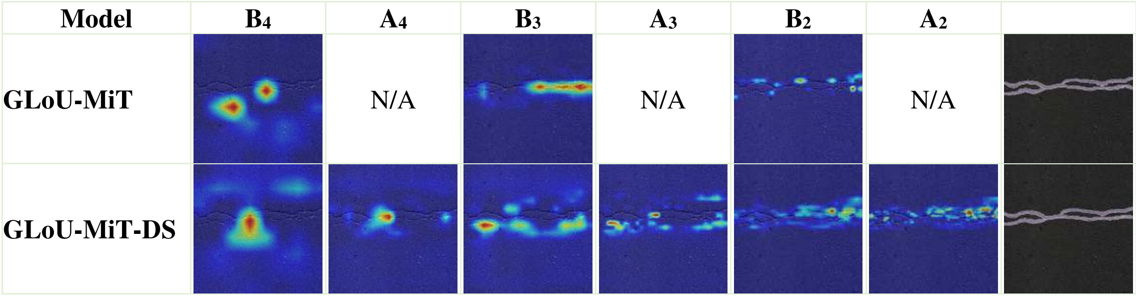
(b) Image from the CrackSC dataset



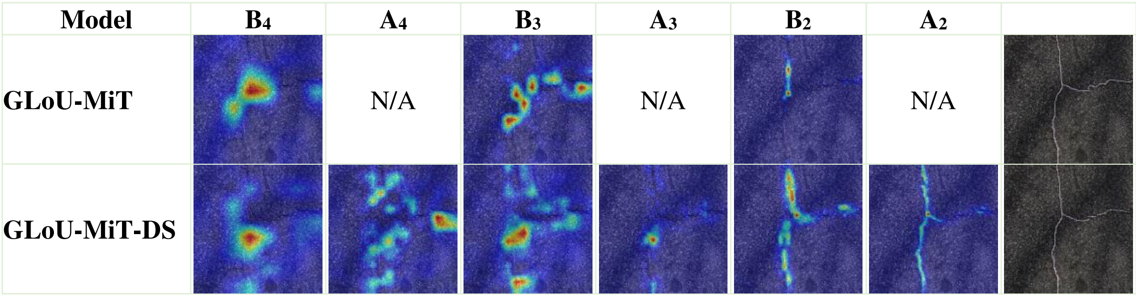
(c) Image from the Crack500 dataset

Fig.15 LayerCAM visualizations: Comparing of GLoU-MiT and SegFormer at corresponding layers

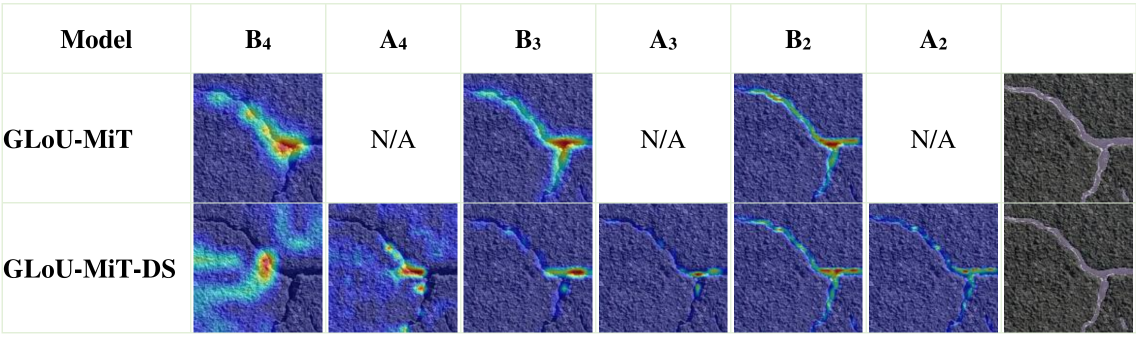
We further compare feature maps before and after the insertion of the DS module, as well as with and without the DS module. Specifically, we compare the LayerCAM visualizations across decoder Stage 2 to Stage 4 (Fig. 16). For the model with the Deep Supervision Refinement Module (GLoU-MiT-DS), the feature maps before DS insertion are denoted as B_k and those after insertion as A_k (where k indicates the stage) (Fig. 14). For the model without the DS module (GLoU-MiT), the corresponding feature maps are similarly denoted as B_k. The LayerCAM results indicate that, after DS insertion, the activation maps become more focused towards the crack centers and exhibit finer boundary delineation, which enhances the segmentation accuracy. Moreover, for datasets with finer cracks (e.g., CrackSC and UAVCrack500), the activation in the shallower B_2 layer increases with the DS module, indicating that the additional boundary and semantic supervision improves the overall crack detection. For datasets with more prominent cracks (e.g., Crack500), the activation regions post-DS insertion are more concentrated within the crack regions. These observations clearly demonstrate that the deep supervision (DS) module plays a crucial role in refining the boundary and semantic features, particularly for fine and narrow cracks that are challenging to detect.



(a) Image from the UAV-Crack500 dataset



(b) Image from the CrackSC dataset



(c) Image from the Crack500 dataset

Fig.16 LayerCAM Visualizations: Comparison of GLoU-MiT (without DS) and GLoU-MiT-DS (before and after DS insertion)

6. Conclusion and Future Research

In this paper, we present a novel lightweight pavement crack segmentation model, GLoU-MiT, designed to address the unique challenges posed by UAV-captured pavement images, such as low resolution, fine crack structures, and low contrast. The proposed model integrates three main components: a U-shaped segmentation framework based on Mix Transformer, a Global-Local Mamba-Guided Skip Connection mechanism, and a Deep Supervision Refinement Module. These innovations contribute to improved feature extraction, efficient computation, and precise segmentation of crack boundaries.

The U-Mix Transformer framework efficiently combines hierarchical feature extraction and attention mechanisms to enhance segmentation accuracy while reducing computational complexity. By replacing concatenation with direct addition in skip connections, the model achieves more effective multi-level feature fusion. The introduction of the Global-Local Mamba-Guided Skip Connection further improves semantic representation by dynamically filtering and fusing global and local features. Additionally, the deep supervision refinement module ensures accurate boundary and semantic supervision, particularly for fine and narrow cracks that are often difficult to detect. Comparative experiments on UAV-Crack500, CrackSC, and Crack500 datasets demonstrate that GLoU-MiT outperforms state-of-the-art CNN, Transformer, and Mamba-based models in terms of F_1-score and Crack IoU, particularly in complex scenarios with challenging crack structures.

Furthermore, while the absolute performance improvement of GLoU-MiT over existing models appears to be around 1% or less, this gain remains both theoretically and practically significant, particularly in UAV-based crack segmentation, where challenges such as low resolution, fine crack structures, and environmental noise make accurate detection inherently difficult. Even minor improvements in F_1-score and IoU can lead to more reliable crack identification, reduced false positives, and better decision-making in automated road maintenance, ultimately enhancing infrastructure monitoring efficiency. Although Mamba-based models excel in long-range feature modeling, their high computational cost limits their feasibility for real-time edge deployment. Our approach effectively balances segmentation accuracy, computational efficiency, and inference speed, making it better suited for UAV-based applications. Additionally, the incorporation of the DS module further enhances fine crack and boundary segmentation, reinforcing the practical advantages of our method in complex and challenging environments.

Looking ahead, future research will focus on further optimizing the inference speed of Mamba-based models, improving their computational efficiency to enhance the feasibility of real-time edge deployment on UAVs, ensuring faster and more efficient crack detection in real-world infrastructure monitoring.

CRediT author statement

Jinhuan Shan: Methodology, Software, Data Curation, Writing - Original Draft**.** **Yue Huang**: Conceptualization, Validation, Writing - Review & Editing**. Wei Jiang**: Conceptualization, Resources, Supervision, Funding acquisition. **Dongdong Yuan:** Writing - Review & Editing, Validation. **Feiyang Guo:** Software, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this study.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 52478432), Qin Chuangyuan “Scientist + Engineer” Team Construction Project of Shaanxi Province (Grant No. 2024QCY-KXJ-020), and the Scientific Research Project of Department of Transportation of Shaanxi Province (Grant No. 24-35K).

References

[1] M. Zheng, W. Chen, X. Ding, W. Zhang, S. Yu, Comprehensive Life Cycle Environmental Assessment of Preventive Maintenance Techniques for Asphalt Pavement, Sustainability 13 (2021) 4887. https://doi.org/10.3390/su13094887.

[2] T. Zhang, D. Wang, Y. Lu, ECSNet: An Accelerated Real-Time Image Segmentation CNN Architecture for Pavement Crack Detection, IEEE Transactions on Intelligent Transportation Systems (2023) 1–8. https://doi.org/10.1109/TITS.2023.3300312.

[3] Pixel-wise crack defect segmentation with dual-encoder fusion network, Constr. Build. Mater. 426 (2024) 136179. https://doi.org/10.1016/j.conbuildmat.2024.136179.

[4] A comparison study of semantic segmentation networks for crack detection in construction materials, Constr. Build. Mater. 414 (2024) 134950. https://doi.org/10.1016/j.conbuildmat.2024.134950.

[5] A. Ragnoli, M.R. De Blasiis, A. Di Benedetto, Pavement Distress Detection Methods: A Review, Infrastructures 3 (2018) 58. https://doi.org/10.3390/infrastructures3040058.

[6] Y. Li, P. Che, C. Liu, D. Wu, Y. Du, Cross-scene pavement distress detection by a novel transfer learning framework, Computer-Aided Civil and Infrastructure Engineering 36 (2021) 1398–1415. https://doi.org/10.1111/mice.12674.

[7] Z. Tong, T. Ma, W. Zhang, J. Huyan, Evidential transformer for pavement distress segmentation, Computer-Aided Civil and Infrastructure Engineering 38 (2023) 2317–2338. https://doi.org/10.1111/mice.13018.

[8] G. Zhu, J. Liu, Z. Fan, D. Yuan, P. Ma, M. Wang, W. Sheng, K.C.P. Wang, A lightweight encoder–decoder network for automatic pavement crack detection, Computer-Aided Civil and Infrastructure Engineering 39 (2023) 1743–1765. https://doi.org/10.1111/mice.13103.

[9] Z. Han, J. Tang, L. Hu, W. Jiang, A. Sha, Automated measurement of asphalt pavement rut depth using smartphone imaging, Autom. Constr. 174 (2025) 106124. https://doi.org/10.1016/j.autcon.2025.106124.

[10] J. Shan, W. Jiang, X. Feng, Bridging cross-domain and cross-resolution gaps for UAV-based pavement crack segmentation, Autom. Constr. 174 (2025) 106141. https://doi.org/10.1016/j.autcon.2025.106141.

[11] Y. Zhang, Z. Zuo, X. Xu, J. Wu, J. Zhu, H. Zhang, J. Wang, Y. Tian, Road damage detection using UAV images based on multi-level attention mechanism, Automation in Construction 144 (2022) 104613. https://doi.org/10.1016/j.autcon.2022.104613.

[12] M. Chai, G. Li, W. Ma, D. Chen, Q. Du, Y. Zhou, S. Qi, L. Tang, H. Jia, Damage characteristics of the Qinghai-Tibet Highway in permafrost regions based on UAV imagery, International Journal of Pavement Engineering 0 (2022) 1–12. https://doi.org/10.1080/10298436.2022.2038381.

[13] Xiao Jiang, Shanjun Mao, Mei Li, Hui Liu, Haoyuan Zhang, Shuwei Fang, Mingze Yuan, Chi Zhang, MFPA-Net: An efficient deep learning network for automatic ground fissures

extraction in UAV images of the coal mining area, Int. J. Appl. Earth Obs. Geoinf. (2022). https://doi.org/10.1016/j.jag.2022.103039.

[14] X. Lei, C. Liu, L. Li, G. Wang, Automated Pavement Distress Detection and Deterioration Analysis Using Street View Map, IEEE Access 8 (2020) 76163–76172. https://doi.org/10.1109/ACCESS.2020.2989028.

[15] L. Song, X. Wang, Faster region convolutional neural network for automated pavement distress detection, Road Materials and Pavement Design 22 (2021) 23–41. https://doi.org/10.1080/14680629.2019.1614969.

[16] Y. Du, N. Pan, Z. Xu, F. Deng, Y. Shen, H. Kang, Pavement distress detection and classification based on YOLO network, International Journal of Pavement Engineering 22 (2021) 1659–1672. https://doi.org/10.1080/10298436.2020.1714047.

[17] Z. Qu, C.-Y. Wang, S.-Y. Wang, F.-R. Ju, A Method of Hierarchical Feature Fusion and Connected Attention Architecture for Pavement Crack Detection, IEEE Transactions on Intelligent Transportation Systems 23 (2022) 16038–16047. https://doi.org/10.1109/TITS.2022.3147669.

[18] X. Sun, Y. Xie, L. Jiang, Y. Cao, B. Liu, DMA-Net: DeepLab With Multi-Scale Attention for Pavement Crack Segmentation, IEEE Transactions on Intelligent Transportation Systems 23 (2022) 18392–18403. https://doi.org/10.1109/TITS.2022.3158670.

[19] Q. Zhou, Z. Qu, S.-Y. Wang, K.-H. Bao, A Method of Potentially Promising Network for Crack Detection With Enhanced Convolution and Dynamic Feature Fusion, IEEE Transactions on Intelligent Transportation Systems 23 (2022) 18736–18745. https://doi.org/10.1109/TITS.2022.3154746.

[20] P.R.T. Peddinti, H. Puppala, B. Kim, Pavement Monitoring Using Unmanned Aerial Vehicles: An Overview, Journal of Transportation Engineering, Part B: Pavements 149 (2023) 03123002. https://doi.org/10.1061/JPEODX.PVENG-1291.

[21] H. Liu, J. Yang, X. Miao, C. Mertz, H. Kong, CrackFormer network for pavement crack segmentation, IEEE Trans. Intell. Transp. Syst. 24 (2023) 9240–9252. https://doi.org/10.1109/TITS.2023.3266776.

[22] J. Wang, Z. Zeng, P.K. Sharma, O. Alfarraj, A. Tolba, J. Zhang, L. Wang, Dual-path network combining CNN and transformer for pavement crack segmentation, Autom. Constr. 158 (2024) 105217. https://doi.org/10.1016/j.autcon.2023.105217.

[23] F. Guo, Y. Qian, J. Liu, H. Yu, Pavement crack detection based on transformer network, Autom. Constr. 145 (2023) 104646. https://doi.org/10.1016/j.autcon.2022.104646.

[24] J. Shan, Y. Huang, W. Jiang, DCUFormer: enhancing pavement crack segmentation in complex environments with dual-cross/upsampling attention, Expert Syst. Appl. 264 (2025) 125891. https://doi.org/10.1016/j.eswa.2024.125891.

[25] A. Gu, T. Dao, Mamba: Linear-Time Sequence Modeling with Selective State Spaces, (2024). http://arxiv.org/abs/2312.00752 (accessed August 25, 2024).

[26] Z. Zhang, B. Peng, T. Zhao, An ultra-lightweight network combining mamba and frequency-domain feature extraction for pavement tiny-crack segmentation, Expert Syst. Appl. 264 (2025) 125941. https://doi.org/10.1016/j.eswa.2024.125941.

[27] X. Zuo, Y. Sheng, J. Shen, Y. Shan, Topology-aware mamba for crack segmentation in structures, Autom. Constr. 168 (2024) 105845. https://doi.org/10.1016/j.autcon.2024.105845.

[28] C. Han, H. Yang, Y. Yang, Enhancing pixel-level crack segmentation with visual mamba and convolutional networks, Automation in Construction 168 (2024) 105770. https://doi.org/10.1016/j.autcon.2024.105770.

[29] H. Liu, C. Jia, F. Shi, X. Cheng, S. Chen, SCSegamba: lightweight structure-aware vision mamba for crack segmentation in structures, (2025). https://doi.org/10.48550/arXiv.2503.01113.

[30] Z. He, Y.-H. Wang, Mamba meets crack segmentation, (2024). https://doi.org/10.48550/arXiv.2407.15714.

[31] J. Zhong, J. Zhu, J. Huyan, T. Ma, W. Zhang, Multi-scale feature fusion network for pixel-level pavement distress detection, Automat Constr 141 (2022) 104436. https://doi.org/10.1016/j.autcon.2022.104436.

[32] Y. Wang, Z. He, X. Zeng, J. Zeng, Z. Cen, L. Qiu, X. Xu, Q. Zhuo, GGMNet: pavement-crack detection based on global context awareness and multi-scale fusion, Remote Sens. 16 (2024) 1797. https://doi.org/10.3390/rs16101797.

[33] S. Bai, L. Yang, Y. Liu, H. Yu, DMF-net: a dual-encoding multi-scale fusion network for pavement crack detection, IEEE Trans. Intell. Transp. Syst. 25 (2024) 5981–5996. https://doi.org/10.1109/TITS.2023.3331769.

[34] J. Dong, N. Wang, H. Fang, W. Guo, B. Li, K. Zhai, MFAFNet: an innovative crack intelligent segmentation method based on multi-layer feature association fusion network, Adv. Eng. Inf. 62 (2024) 102584. https://doi.org/10.1016/j.aei.2024.102584.

[35] Q. Zhou, Z. Qu, Y. Li, F. Ju, Tunnel Crack Detection With Linear Seam Based on Mixed Attention and Multiscale Feature Fusion, IEEE Trans. Instrum. Meas. (2022). https://doi.org/10.1109/tim.2022.3184351.

[36] J. Liang, X. Gu, D. Jiang, Q. Zhang, CNN-based network with multi-scale context feature and attention mechanism for automatic pavement crack segmentation, Autom. Constr. 164 (2024) 105482. https://doi.org/10.1016/j.autcon.2024.105482.

[37] M. Ma, L. Yang, Y. Liu, H. Yu, An attention-based progressive fusion network for pixelwise pavement crack detection, Measurement 226 (2024) 114159. https://doi.org/10.1016/j.measurement.2024.114159.

[38] A.M. Roy, J. Bhaduri, DenseSPH-YOLOv5: an automated damage detection model based on DenseNet and swin-transformer prediction head-enabled YOLOv5 with attention mechanism, Adv. Eng. Inf. 56 (2023) 102007. https://doi.org/10.1016/j.aei.2023.102007.

[39] Z. Al-Huda, B. Peng, R.N.A. Algburi, M.A. Al-antari, R. AL-Jarazi, D. Zhai, A hybrid deep

learning pavement crack semantic segmentation, Eng. Appl. Artif. Intell. 122 (2023) 106142. https://doi.org/10.1016/j.engappai.2023.106142.

[40] J.-M. Guo, H. Markoni, J.-D. Lee, BARNet: boundary aware refinement network for crack detection, IEEE Trans. Intell. Transp. Syst. 23 (2022) 7343–7358. https://doi.org/10.1109/TITS.2021.3069135.

[41] J. Long, E. Shelhamer, T. Darrell, Fully Convolutional Networks for Semantic Segmentation, (2015). https://doi.org/10.48550/arXiv.1411.4038.

[42] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in: N. Navab, J. Hornegger, W.M. Wells, A.F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015: pp. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28.

[43] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation, in: In Proceedings of the European Conference on Computer Vision (ECCV), Springer International Publishing, Cham, 2018: pp. 833–851. https://doi.org/10.1007/978-3-030-01234-2_49.

[44] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-Excitation Networks, (2019). https://doi.org/10.48550/arXiv.1709.01507.

[45] J. Park, S. Woo, J.-Y. Lee, I.S. Kweon, BAM: Bottleneck Attention Module, (2018). https://doi.org/10.48550/arXiv.1807.06514.

[46] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, CBAM: Convolutional Block Attention Module, in: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (Eds.), Computer Vision – ECCV 2018, Springer International Publishing, Cham, 2018: pp. 3–19. https://doi.org/10.1007/978-3-030-01234-2_1.

[47] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, (2021). https://doi.org/10.48550/arXiv.2103.14030.

[48] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, Y. Liu, VMamba: Visual State Space Model, (2024). http://arxiv.org/abs/2401.10166 (accessed August 15, 2024).

[49] T. Huang, X. Pei, S. You, F. Wang, C. Qian, C. Xu, LocalMamba: Visual State Space Model with Windowed Selective Scan, (2024). http://arxiv.org/abs/2403.09338 (accessed August 14, 2024).

[50] Z. Wang, J.-Q. Zheng, Y. Zhang, G. Cui, L. Li, Mamba-UNet: UNet-Like Pure Visual Mamba for Medical Image Segmentation, (2024). http://arxiv.org/abs/2402.05079 (accessed August 14, 2024).

[51] T. Takikawa, D. Acuna, V. Jampani, S. Fidler, Gated-SCNN: Gated Shape CNNs for Semantic Segmentation, (2019). https://doi.org/10.48550/arXiv.1907.05740.

[52] X. Qin, D.-P. Fan, C. Huang, C. Diagne, Z. Zhang, A.C. Sant'Anna, A. Suàrez, M. Jagersand, L. Shao, Boundary-Aware Segmentation Network for Mobile and Web Applications, (2021).

https://doi.org/10.48550/arXiv.2101.04704.

[53] A. Kirillov, Y. Wu, K. He, R. Girshick, PointRend: Image Segmentation as Rendering, (2020). https://doi.org/10.48550/arXiv.1912.08193.

[54] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, Y. Wei, Deformable Convolutional Networks, (2017). https://doi.org/10.48550/arXiv.1703.06211.

[55] H. Kervadec, J. Bouchtiba, C. Desrosiers, E. Granger, J. Dolz, I. Ben Ayed, Boundary loss for highly unbalanced segmentation, Medical Image Analysis 67 (2021) 101851. https://doi.org/10.1016/j.media.2020.101851.

[56] A. Gu, A. Gupta, K. Goel, C. Ré, On the parameterization and initialization of diagonal state space models, (2022). https://doi.org/10.48550/arXiv.2206.11893.

[57] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, X. Wang, Vision mamba: efficient visual representation learning with bidirectional state space model, (2024). http://arxiv.org/abs/2401.09417 (accessed October 12, 2024).

[58] J. Shan, W. Jiang, Y. Huang, D. Yuan, Y. Liu, Unmanned Aerial Vehicle (UAV)-Based Pavement Image Stitching Without Occlusion, Crack Semantic Segmentation, and Quantification, IEEE Trans. Intell. Transport. Syst. 25 (2024) 17038–17053. https://doi.org/10.1109/TITS.2024.3424525.

[59] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, H. Ling, Feature pyramid and hierarchical boosting network for pavement crack detection, IEEE Trans. Intell. Transp. Syst. 21 (2020) 1525–1535. https://doi.org/10.1109/TITS.2019.2910595.

[60] MMSegmentation Contributors, OpenMMLab semantic segmentation toolbox and benchmark, (2020). https://github.com/open-mmlab/mmsegmentation (accessed March 8, 2025).

[61] H. Zhang, A.A. Zhang, Z. Dong, A. He, Y. Liu, Y. Zhan, K.C.P. Wang, Robust semantic segmentation for automatic crack detection within pavement images using multi-mixing of global context and local image features, IEEE Trans. Intell. Transp. Syst. 25 (2024) 11282–11303. https://doi.org/10.1109/TITS.2024.3360263.

[62] H. Li, H. Zhang, H. Zhu, K. Gao, H. Liang, J. Yang, Automatic crack detection on concrete and asphalt surfaces using semantic segmentation network with hierarchical transformer, Eng. Struct. 307 (2024) 117903. https://doi.org/10.1016/j.engstruct.2024.117903.

[63] B. Liu, J. Kang, H. Guan, X. Zhi, Y. Yu, L. Ma, D. Peng, L. Xu, D. Wang, RTCNet: a novel real-time triple branch network for pavement crack semantic segmentation, Int. J. Appl. Earth Obs. Geoinf. 136 (2025) 104347. https://doi.org/10.1016/j.jag.2024.104347.

[64] M. Usman, H. Chen, M. Raza, A lightweight convolutional transformer architecture approach for crack segmentation in safety assessment, in: 2024 IEEE International Conference on Systems, Man, and Cybernetics (Smc), 2024: pp. 5195–5199. https://doi.org/10.1109/SMC54092.2024.10831651.

[65] W. Zheng, X. Jiang, Z. Fang, Y. Gao, TV-net: a structure-level feature fusion network based

on tensor voting for road crack segmentation, IEEE Trans. Intell. Transp. Syst. 25 (2024) 5743–5754. https://doi.org/10.1109/TITS.2023.3334266.

[66] T. Ruan, T. Liu, Z. Huang, Y. Wei, S. Wei, Y. Zhao, T. Huang, Devil in the details: towards accurate single and multiple human parsing, (2018). http://arxiv.org/abs/1809.05996 (accessed September 15, 2024).

[67] J. Xu, Z. Xiong, S.P. Bhattacharyya, PIDNet: a real-time semantic segmentation network inspired by PID controllers, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Vancouver, BC, Canada, 2023: pp. 19529–19539. https://doi.org/10.1109/CVPR52729.2023.01871.

[68] F. Panella, A. Lipani, J. Boehm, Semantic segmentation of cracks: data challenges and architecture, Autom. Constr. 135 (2022) 104110. https://doi.org/10.1016/j.autcon.2021.104110.

[69] Y. Zhang, J. Wu, Q. Li, X. Zhao, M. Tan, Beyond crack: fine-grained pavement defect segmentation using three-stream neural networks, IEEE Trans. Intell. Transp. Syst. 23 (2022) 14820–14832. https://doi.org/10.1109/TITS.2021.3134374.

[70] X. Weng, Y. Huang, W. Wang, Segment-based pavement crack quantification, Automation in Construction 105 (2019) 102819. https://doi.org/10.1016/j.autcon.2019.04.014.

[71] M.-H. Guo, C.-Z. Lu, Q. Hou, Z.-N. Liu, M.-M. Cheng, S.-M. Hu, SegNeXt: rethinking convolutional attention design for semantic segmentation, in: Proceedings of the 36th International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, 2022: pp. 1140–1156.

[72] G. Zhu, J. Liu, Z. Fan, D. Yuan, P. Ma, M. Wang, W. Sheng, K.C.P. Wang, A lightweight encoder–decoder network for automatic pavement crack detection, Computer-Aided Civil and Infrastructure Engineering 39 (2024) 1743–1765. https://doi.org/10.1111/mice.13103.

[73] Q. Tan, A. Li, L. Dong, W. Dong, X. Li, G. Shi, CDS-net: contextual difference sensitivity network for pixel-wise road crack detection, IEEE Trans. Circuits Syst. Video Technol. (2025) 1–1. https://doi.org/10.1109/TCSVT.2025.3529039.

[74] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J.M. Alvarez, P. Luo, SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., 2021: pp. 12077–12090. https://proceedings.neurips.cc/paper_files/paper/2021/hash/64f1f27bf1b4ec22924fd0acb550c235-Abstract.html (accessed July 25, 2024).

[75] S.-K. Yeom, J. von Klitzing, U-MixFormer: UNet-like Transformer with Mix-Attention for Efficient Semantic Segmentation, (2023). http://arxiv.org/abs/2312.06272 (accessed December 27, 2023).

[76] P.-T. Jiang, C.-B. Zhang, Q. Hou, M.-M. Cheng, Y. Wei, LayerCAM: exploring hierarchical class activation maps for localization, IEEE Trans. Image Process. 30 (2021) 5875–5888. https://doi.org/10.1109/TIP.2021.3089943.