scientific reports

Check for updates

OPEN

PUNet: a lightweight parallel U-Net architecture integrating Mamba–CNN for high-precision image segmentation

Zhaoyan Xie[1]✉, Xiaowei Li[1], Hongyao Ma[2], Sihao Wu[1] & Dayou Cui[1]

Real-time high-precision image segmentation on mobile and edge devices remains challenging due to the limited ability of traditional convolutional networks to model long-range dependencies and their high computational cost at high resolutions. We propose PUNet, a lightweight parallel U-Net variant that integrates depthwise separable convolutions (DSConv) with a structured state-space Mamba module in a dual-path encoder. The core component, the parallel structured state-space encoder, employs two branches to efficiently capture local spatial features (via DSConv) and model global semantic dependencies (via the visual Mamba layer). At the same time, a squeeze-and-excitation skip connection adaptively fuses multi-scale features. With only 0.26 M parameters and linear computational complexity, PUNet enables real-time inference on resource-constrained platforms. Experiments on the CamVid and CRACK500 datasets demonstrate superior performance, achieving validation Dice scores of 0.9208 and 0.7902, and mean Intersection-over-Union of 0.8643 and 0.6612, respectively, significantly outperforming other lightweight models.

Keywords Mamba, CNN, Image segmentation, Lightweight architecture

Mobile terminals have become indispensable in intelligent transportation[1], road inspection[2], and infrastructure monitoring[3]. These edge devices often operate under dynamic illumination and unstable motion, requiring real-time, high-precision image segmentation for tasks such as pavement crack detection[2] and traffic sign recognition[3,4], thereby supporting collision avoidance, autonomous navigation, and rapid emergency response[5,6]. Traditional image processing methods rely on hand-crafted texture, color, and shape features[6,7], which lack robustness under direct sunlight, shadow occlusion, and multi-angle viewpoints[3,8], and thus fail to meet the dual requirements of accuracy and real-time performance in practical applications[9,10].

With the advent of deep learning, end-to-end convolutional neural network (CNN) segmentation models such as U-Net[11], FCN[12], and DeepLabv3+[13] have significantly improved segmentation accuracy by automatically learning multi-scale features, becoming the dominant solutions in both academia and industry[14]. Early CNNs stacked multiple convolution and pooling layers to extract hierarchical representations; even with dilated convolutions to enlarge the receptive field[13], the intrinsic locality of convolution kernels hinders the capture of long-range dependencies, leading to blurred object boundaries and missed small targets in complex scenes[15]. To address these issues, U-Net and its variants (e.g., U-Net++[15], Attention U-Net[16]) employ a symmetric encoder–decoder architecture with skip connections, effectively fusing high-level semantics with low-level details and demonstrating outstanding performance in medical imaging and remote sensing segmentation tasks[17]. Nevertheless, these architectures still depend on convolution operations, limiting their global modeling capability[18]. Their computational cost grows rapidly with input resolution—for instance, DeepLabv3+ incurs over 100G FLOPs on 1024 $\times$1024 images[13], making it challenging to balance lightweight design and high accuracy[19].

Transformer-based segmentation frameworks, such as SegFormer[20], Swin Transformer[21], and Swin UNETR[22], leverage the self-attention mechanisms for global context modeling and hierarchical feature fusion, outperforming traditional CNN models on various benchmarks. To address boundary blurring, the boundary-aware feature propagation module introduces semantic boundary supervision, proving the key role of combining global and local information for improved segmentation accuracy[23]. Hybrid architectures like TransUNet[24], SwinUNet[18], and TransBTS[25] split images into patches for self-attention, significantly enhancing

[1]Shandong Jiaotong University, Haitang Road 5001, Jinan 250357, China. [2]College of Computer Science, Beijing University of Technology, Beijing 100124, China. ✉email: 215036@sdjtu.edu.cn

segmentation of complex structures in tumor boundary delineation and remote sensing applications[25]. However, the quadratic time complexity of the self-attention mechanism results in a significant increase in computational cost as the resolution grows[26]. This makes the computation resource-intensive. Additionally, the large number of parameters in models (for instance, some advanced architectures like SwinUNet have a substantial parameter count[18]) renders their deployment on uncrewed aerial vehicles (UAVs) and embedded devices unfeasible[8,19].

State-space models (SSMs) represented by Mamba offer linear complexity O(L) (where L is the sequence length) and efficient long-range dependency modeling, providing a novel path to overcome traditional bottlenecks[27]. By replacing self-attention with structured state-space operations, Mamba maintains global modeling capability while reducing computational cost to scale linearly with input size, enabling lightweight design[28]. Recent works such as UltraLight VM-UNet[29], VM-UNet[30], and LightM-UNet[19] integrate Mamba into U-Net by substituting convolutional blocks or encoder modules[31], achieving substantial parameter reduction alongside accuracy gains in medical image segmentation[32]. Despite these advances, existing Mamba–CNN fusion schemes still suffer from insufficient feature interaction[25], coarse lightweight design[29], and limited generalizability across diverse scenarios.

CNNs excel at extracting local details[11], Transformers enable powerful global context modeling[23], and Mamba provides low-cost long-range dependency capture[28]. The primary challenge resides in effectively integrating these complementary advantages to construct an image segmentation model that is both lightweight and highly accurate, tailored for edge-computing scenarios in intelligent driving and other resource-constrained mobile environments. The main contributions of this paper are as follows:

- We propose PUNet, a hybrid architecture that integrates the Mamba sequence model with lightweight convolutional modules to enhance long-range dependency modeling while harmonizing local detail extraction and global semantic representation, thereby improving robustness in complex segmentation scenarios.
- We propose the parallel structured state-space encoder (PSSSE), which adopts a dual-path architecture to capture both local and global contextual information. The first path utilizes depthwise separable convolutions (DSConv) for efficient extraction of local spatial features. The second path incorporates the visual Mamba layer (VMLayer), leveraging its state-space formulation to model long-range dependencies across sequences. To facilitate effective feature fusion, a squeeze-and-excitation skip connection (SE-Skip Connection) is employed to adaptively recalibrate and integrate multi-resolution features through channel-wise attention, enabling simultaneous enhancement of fine-grained details and holistic semantic context.
- With just 0.26 M parameters, PUNet drastically reduces computational overhead, enabling real-time deployment on resource-constrained platforms such as embedded systems and UAVs. Experimental results on CRACK500 and CamVid demonstrate its superior cross-domain generalization and lightweight segmentation performance.

Related work

Image segmentation is a fundamental task in computer vision, aiming to assign a semantic label to each pixel. The proposed PUNet model is a lightweight segmentation network inspired by three primary research directions: U-Net architectures, lightweight CNNs, and vision models based on SSMs.

U-Net and its variants

The U-Net[11] is a classic encoder-decoder architecture widely recognized for its symmetrical structure and skip connections, which effectively fuse multi-scale features. Its success has led to numerous variants. For example, Attention U-Net[16] introduced attention gates to filter and suppress irrelevant feature responses, while Recurrent Residual U-Net (R2U-Net)[33] improved feature extraction with recurrent residual convolutional units. These models primarily rely on convolutional operations to capture local features. In contrast, our PUNet model adopts a novel parallel dual-branch encoder that integrates both DSConv for efficient local feature extraction and Mamba blocks for modeling global dependencies. This parallel design allows our model to simultaneously process local and global information, distinguishing it from traditional U-Net variants that are predominantly CNN-based.

Lightweight convolutional neural networks

The development of lightweight CNNs is crucial for real-world deployment on resource-constrained devices. Techniques such as depthwise separable convolutions (e.g., MobileNet[34]) and grouped convolutions (e.g., ShuffleNet[35]) are commonly used to reduce computational costs. While effective, these methods often struggle to capture long-range dependencies efficiently. Recent advancements in lightweight CNNs and hybrids have explored other strategies, such as robust feature downsampling[36], lightweight group attention[37], and efficient feature decoupling[38] to enrich feature representation without a heavy computational burden. More recently, hybrid architectures have emerged, combining CNNs with novel modules. For example, LightM-UNet[19] integrates vision state-space (VSS) modules into a U-Net backbone to enhance its ability to model global contexts.

A significant distinction of our work lies in the design of the VMLayer, which is a key component of PUNet's parallel encoder. Unlike the VSS module in LightM-UNet, which may face high computational costs when processing high-dimensional features, our VMLayer employs a "divide-and-conquer" strategy. It divides the overall feature channels into multiple subchannels, processes them independently with Mamba modules, and then fuses them using a channel attention mechanism. This approach drastically reduces the number of parameters and computational complexity while maintaining segmentation accuracy. Therefore, our VMLayer offers a more efficient and effective solution to balance performance and model size compared to other lightweight Mamba-based models.

Visual models with state space models

Inspired by the success of Mamba[27] in natural language processing, researchers have explored its application in computer vision. VMamba[28] was one of the first works to adapt the Mamba block for 2D images. U-Mamba[32] further integrated Mamba with CNNs in a U-Net architecture for biomedical image segmentation, typically by embedding Mamba blocks within the residual connections of the encoder.

Our PUNet diverges from these hybrid approaches in its architectural philosophy. Instead of a sequential or substitution-based integration of Mamba and CNNs, we propose a parallel architecture where a Mamba branch and a CNN branch operate concurrently. This design allows the CNN to specialize in capturing fine-grained local features, while the Mamba branch efficiently models global semantic dependencies, enabling a truly complementary collaboration. Furthermore, our model introduces a unique squeeze-and-excitation skip connection to refine the fusion of multi-scale features, ensuring better semantic alignment and detail preservation. These distinct architectural choices position PUNet as a novel and highly efficient solution in the landscape of Mamba-based vision models.

Methods

In this section, we present PUNet, comprising a six-stage encoder and a six-stage decoder for bottom-up feature extraction and top-down reconstruction, respectively. As shown in Fig. 1, each encoder stage applies the PSSSE with 3×3 convolutions to downsample spatial dimensions and expand channels, effectively capturing rich semantic features. Decoder stages 1–5 employ the VMLayer to upsample features and reduce channels, while stage 6 reuses PSSSE to align feature dimensions with the input and fuse multi-scale information, producing the final fine-grained segmentation output.

Depthwise separable convolution (DSConv)

To maintain the receptive field size while reducing both the number of parameters and computational cost, most convolutions in PUNet adopt DSConv[39]. Let $X \in \mathbb{R}^{B\times C\times H\times W}$ denote the input tensor, where B is the batch size, C the number of input channels, and H, W the spatial dimensions. The DSConv module proceeds in three stages.

First, a depthwise convolution applies an independent $K \times K$ filter to each channel:

$$Z_{b,c,i,j} = \sum_{u=1}^{K}\sum_{v=1}^{K} W^{\mathrm{dw}}_{c,u,v}\, X_{b,c,i+u-\lfloor K/2\rfloor,\, j+v-\lfloor K/2\rfloor}, \tag{1}$$

where $W^{\mathrm{dw}} \in \mathbb{R}^{C\times K\times K}$ and zero-padding of size $\lfloor K/2\rfloor$ preserves the spatial dimensions (H, W).

Next, a pointwise convolution with a 1×1 kernel fuses information across channels:

$$Y_{b,m,i,j} = \sum_{c=1}^{C} W^{\mathrm{pw}}_{m,c}\, Z_{b,c,i,j}, \tag{2}$$

where $W^{\mathrm{pw}} \in \mathbb{R}^{C_{\mathrm{out}}\times C}$ and m indexes the C_{out} output channels.

Finally, batch normalization followed by a ReLU activation introduces nonlinearity:

$$F^{\mathrm{DSC}} = \mathrm{ReLU}\big(\mathrm{BN}(Y)\big), \tag{3}$$

where $\mathrm{BN}(\cdot)$ standardizes each channel to zero mean and unit variance, and $\mathrm{ReLU}(x) = \max(0, x)$.

Visual Mamba layer (VMLayer)

Traditional convolutional architectures such as U-Net rely heavily on local receptive fields, limiting their ability to capture long-range semantic dependencies. As is shown in Fig. 2, this constraint weakens global context modeling, especially in complex visual scenes. As an efficient variant of the structured state space model (SSM), Mamba provides linear-time processing of long sequences, with computational complexity $\mathcal{O}(N)$, where N denotes the sequence length. However, when directly applied to high-dimensional image features (e.g., $C = 256$),

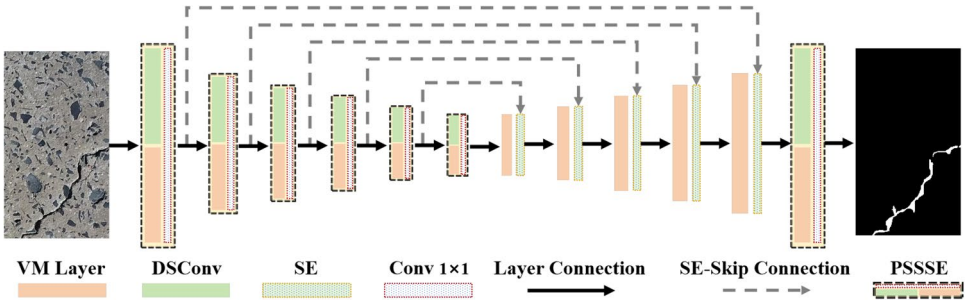


Fig. 1. Overall architecture of the PUNet model.

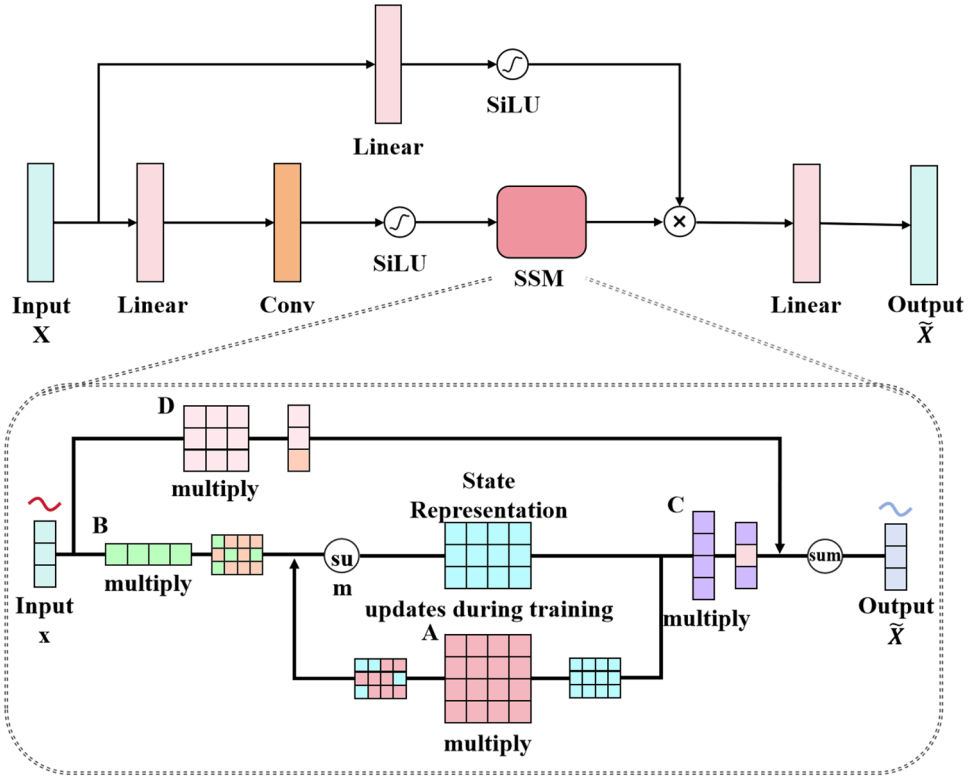


Fig. 2. Mamba module.

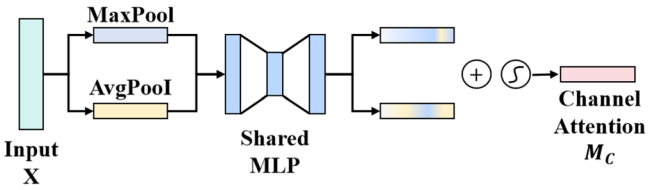


Fig. 3. Channel attention mechanism module.

the channel-wise computation cost remains significant. To overcome this limitation, the VMLayer employs a divide-and-conquer strategy, as illustrated in Fig. 4. It begins by dividing the input feature channels into several low-dimensional subspaces, enabling independent processing of each subspace using the Mamba module. To enhance global contextual understanding across channels, a lightweight channel attention mechanism (CAM)[40] is subsequently applied, as illustrated in Fig. 3. This facilitates efficient modeling of cross-channel dependencies and effective feature integration, while maintaining a low computational footprint. In modern deep learning architectures, efficient feature processing is crucial for enhancing model performance. The following describes a comprehensive process for processing an input feature tensor, which combines sequence reshaping, normalization, module—based transformation, pooling, attention mechanism, and projection operations to generate a refined feature output.

We start with an input feature tensor $X \in \mathbb{R}^{B\times C\times H\times W}$, where B represents the batch size, which indicates the number of samples processed simultaneously during training or inference. C denotes the number of channels in the feature tensor, where each channel can capture different types of semantic information. H and W represent the height and width of the feature map, respectively, defining its spatial dimensions.

The first step is to reshape the input feature tensor X into a sequence representation:

$$X_\flat = \text{reshape}(X) \in \mathbb{R}^{B\times N\times C}, \quad N = H \times W \tag{4}$$

The parameter N is the product of the height H and the width W of the feature map. By reshaping, we transform the two-dimensional spatial information of the feature map into a one-dimensional sequence of length N. This reshaping operation is beneficial for subsequent processing that is more suitable for sequence—based models.

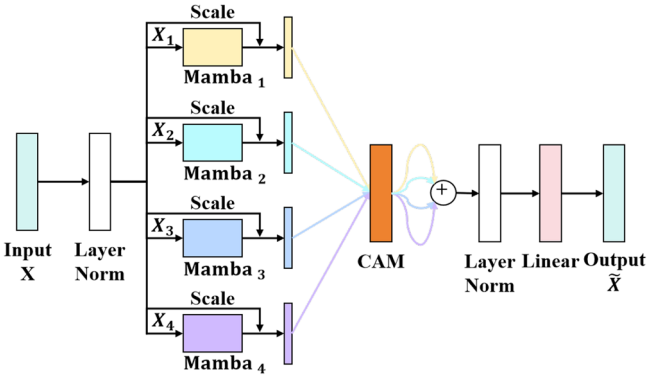


Fig. 4. Visual Mamba layer module.

After reshaping, layer normalization is applied to $X_\flat$ to stabilize the training process. Layer normalization standardizes the features within each sample across all channels, which helps in faster convergence and better generalization. Then, the normalized feature is equally partitioned along the channel dimension into four groups.

$$[X_1, X_2, X_3, X_4] = \text{chunk}(\text{LayerNorm}(X_\flat), 4), \quad X_k \in \mathbb{R}^{B\times N\times \frac{C}{4}} \tag{5}$$

The chunk function divides the feature tensor into four non-overlapping sub-tensors along the channel dimension. Each sub-feature X_k has a reduced number of channels ($\frac{C}{4}$), allowing for independent processing and potentially capturing different aspects of the input features.

Each sub-feature X_k is independently processed by the Mamba module, which is designed to model long-range dependencies in sequences efficiently. After the Mamba module, a learnable residual connection is introduced:

$$X'_k = \text{Mamba}(X_k) + \alpha X_k \tag{6}$$

The scalar α is a learnable parameter. During the training process, the model adjusts the value of α to balance the contribution of the original sub-feature X_k and the transformed sub-feature $\text{Mamba}(X_k)$. The residual connection helps in mitigating the vanishing gradient problem and enables the network to learn more effectively.

The processed sub-sequences X'_k are then reshaped back to the spatial domain:

$$\widetilde{X}_k = \text{reshape}(X'_k) \in \mathbb{R}^{B\times \frac{C}{4}\times H\times W} \tag{7}$$

This operation restores the two-dimensional spatial structure of the feature map, preparing it for subsequent spatial-based operations.

For each $\widetilde{X}_k$, we apply both global average pooling (GAP) and global max pooling (GMP) across the spatial dimensions:

$$z_k^{\text{avg}} = \text{GAP}(\widetilde{X}_k) \in \mathbb{R}^{B\times \frac{C}{4}}, \quad z_k^{\max} = \text{GMP}(\widetilde{X}_k) \in \mathbb{R}^{B\times \frac{C}{4}} \tag{8}$$

Global average pooling computes the average value of each channel throughout the spatial extent of the feature map, while global max pooling selects the maximum value. These two pooling operations capture different types of global information from the feature map, providing complementary information for subsequent processing.

Both z_k^{avg} and $z_k^{\max}$ are passed through a two-layer bottleneck multi-layer perceptron (MLP). The first layer of the MLP reduces the dimensionality by a factor of r, and the second layer restores it:

$$\text{MLP}(z) = W_2(\text{ReLU}(W_1 z)), \quad W_1 \in \mathbb{R}^{\frac{C}{4r}\times \frac{C}{4}}, \quad W_2 \in \mathbb{R}^{\frac{C}{4}\times \frac{C}{4r}} \tag{9}$$

The weight matrix W_1 of size $\frac{C}{4r} \times \frac{C}{4}$ maps the input features to a lower-dimensional space, which acts as a bottleneck. The ReLU activation function introduces non-linearity, allowing the network to learn complex relationships. The weight matrix W_2 of size $\frac{C}{4} \times \frac{C}{4r}$ then maps the features back to the original dimensionality.

The channel attention weights are computed by aggregating the two MLP outputs with a sigmoid activation:

$$s_k = \sigma\left(\text{MLP}(z_k^{\text{avg}}) + \text{MLP}(z_k^{\max})\right) \in \mathbb{R}^{B\times \frac{C}{4}} \tag{10}$$

The sigmoid function squashes the aggregated output to the range $[0, 1]$, producing attention weights that indicate the importance of each channel.

The attention vector s_k is reshaped into a spatial attention map:

$$S_k = \text{reshape}(s_k) \in \mathbb{R}^{B\times \frac{C}{4}\times H\times W} \tag{11}$$

and then used to perform element-wise multiplication with the original feature map $\widetilde{X}_k$:

$$\widehat{X}_k = S_k \odot \widetilde{X}_k, \quad k = 1, 2, 3, 4 \tag{12}$$

This operation weights the feature map according to the importance of each channel, enhancing the discriminative power of the features.

The attention-weighted features $\widehat{X}_k$ are flattened and concatenated across channel groups. Then, a linear projection is applied, followed by reshaping to recover the original spatial dimension:

$$F^{\mathrm{VML}} = \mathrm{reshape}\left(\mathrm{Proj}\left(\mathrm{concat}_{k=1}^{4}\,\mathrm{flatten}(\widehat{X}_k)\right)\right) \tag{13}$$

The resulting feature F^{VML} has the same shape as the input feature tensor X but contains more refined and discriminative information, which can be further used in downstream tasks such as classification or segmentation.

Squeeze-and-excitation skip connection (SE-skip connection)

In standard U-Net architectures, skip connections simply concatenate or add encoder and decoder features, ignoring the varying importance of individual channels. This often propagates irrelevant or noisy channels into the decoder, leading to the loss of boundary and fine-detail information. We therefore integrate a squeeze-and-excitation (SE)[41] module into the skip connection to perform adaptive channel-wise recalibration on the encoder features before fusion, as illustrated in Fig. 5. The SE module suppresses uninformative channels and highlights critical ones, thereby enhancing semantic alignment and detail preservation across network stages.

Let the output of the i-th encoder stage be denoted as $S^i \in \mathbb{R}^{B\times C_s\times H_s\times W_s}$, where B signifies the batch size, representing the number of samples processed concurrently; C_s stands for the number of channels in the encoder's output, which carry diverse feature information; and H_s and W_s respectively denote the height and width of the spatial dimensions of this output feature map. Concurrently, the upsampled input fed into the corresponding decoder stage is expressed as $G^i \in \mathbb{R}^{B\times C_g\times H_g\times W_g}$, where C_g indicates the number of channels in the decoder's input, which may differ from C_s depending on the network architecture; and H_g and W_g respectively represent the height and width of the spatial dimensions of this upsampled feature map.

First, a spatial alignment operation is carried out:

$$U^i = \mathrm{Interpolate}\big(G^i;\ \mathrm{size} = (H_s, W_s)\big) \in \mathbb{R}^{B\times C_g\times H_s\times W_s} \tag{14}$$

This interpolation step ensures that the decoder input G^i has the same spatial dimensions as the encoder output S^i, facilitating subsequent feature fusion.

Subsequently, the SE (squeeze-and-excitation) re-calibration process unfolds in three distinct steps:

A global average pooling operation is applied across the spatial dimensions of the encoder output S^i. This operation aggregates the spatial information within each channel, effectively summarizing the global context, which is defined as:

$$z = \mathrm{GAP}(S^i) \quad z_{b,c} = \frac{1}{H_s W_s}\sum_{x=1}^{H_s}\sum_{y=1}^{W_s} S^i_{b,c,x,y} \ \in\ \mathbb{R}^{B\times C_s} \tag{15}$$

Here, for each batch element b and channel c, the value $z_{b,c}$ is the average of all spatial elements in the corresponding channel of the encoder output.

Two fully connected layers with a reduction ratio r and a sigmoid activation function are employed to capture channel-wise dependencies. The first fully connected layer W_1 maps the input z to a lower dimensional space, followed by a ReLU activation function to introduce non-linearity. The second fully connected layer W_2 then maps the output back to the original dimensionality. Finally, a sigmoid function σ is applied to obtain the channel attention weights. The equations are as follows:

$$e = W_2\big(\mathrm{ReLU}(W_1\, z)\big) \quad s = \sigma(e) \ \in\ \mathbb{R}^{B\times C_s} \tag{16}$$

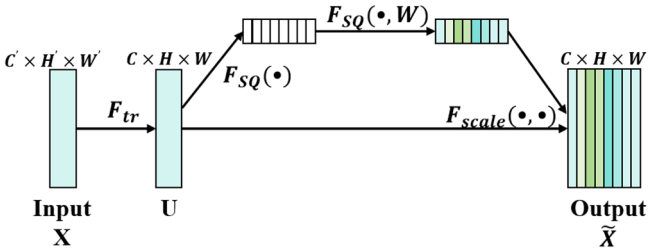


Fig. 5. Squeeze-and-excitation skip connection module.

where $W_1 \in \mathbb{R}^{\frac{C_s}{r} \times C_s}$ and $W_2 \in \mathbb{R}^{C_s \times \frac{C_s}{r}}$. The sigmoid function σ squashes the output values to the range [0, 1], indicating the importance of each channel.

The encoder output S^i is calibrated channel-wise using the attention weights s. Each element in S^i is multiplied by the corresponding attention weight, effectively emphasizing or suppressing the channels based on their importance. The re-calibrated encoder output $\widetilde{S}^i$ is computed as:

$$\widetilde{S}^i_{b,c,x,y} = s_{b,c} \cdot S^i_{b,c,x,y} \quad \widetilde{S}^i \in \mathbb{R}^{B \times C_s \times H_s \times W_s} \tag{17}$$

The re-calibrated encoder features $\widetilde{S}^i$ are then fused with the aligned decoder features U^i through an element wise addition operation:

$$F^i = U^i + \widetilde{S}^i \in \mathbb{R}^{B \times \max(C_g, C_s) \times H_s \times W_s} \tag{18}$$

This fusion step combines the refined encoder information with the decoder input, leveraging the complementary nature of the two feature maps.

Optionally, to further enhance the quality of the fused features F^i, we apply a sequence of operations including DSConv, batch normalization, and ReLU activation. The refined output is given by

$$F^i_{\text{out}} = \text{ReLU}\big(\text{BN}(\text{DSConv}(F^i))\big) \tag{19}$$

By performing channel re-calibration before the fusion process, the SE-Skip Connection mechanism can adaptively highlight the encoder information that is relevant to the decoder while suppressing noise. This results in more accurate boundary delineation and better preservation of fine-grained details in the final output, which is beneficial for various computer vision tasks such as image segmentation.

Parallel structured state space encoder (PSSSE)

Single-path feature extractors often face challenges in balancing local details—such as object boundaries—and global context—such as scene semantics. To address this, we introduce the Parallel Structured State Space Encoder (PSSSE), which utilizes two parallel branches—the local branch and the global branch—to capture complementary information and dynamically fuse them through channel attention, as depicted in Fig. 6.

We start with an input tensor $X \in \mathbb{R}^{B \times C \times H \times W}$, where (B) stands for the batch size, (C) represents the number of input channels, and $(H \times W)$ defines the spatial resolution.

The local branch employs a series of DSConv to efficiently extract fine-grained local spatial features such as edges, textures, and object boundaries. This operation is critical for maintaining high-resolution details with a compact computational footprint.

$$F^{(1)} = \text{ReLU}\big(\text{BN}(\text{DSC}(X))\big) \tag{20}$$

The global branch leverages the visual Mamba layer (VMLayer) to model long-range semantic dependencies across the entire image. VMLayer's "divide-and-conquer" strategy allows for efficient capture of global context with linear complexity, making it highly suitable for lightweight architectures.

$$F^{(2)} = F^{\text{PVML}}(X) \tag{21}$$

The core strength of this parallel design is the synergy between the two branches. The DSConv branch provides high-resolution, local geometric details, while the VMLayer branch offers low-resolution, high-level global semantics. This complementary information is then intelligently combined.

To effectively harness this feature diversity, each branch's output is processed by a squeeze-and-excitation (SE) module:

$$\widetilde{F}^{(1)} = \text{SE}\big(F^{(1)}\big), \quad \widetilde{F}^{(2)} = \text{SE}\big(F^{(2)}\big) \tag{22}$$

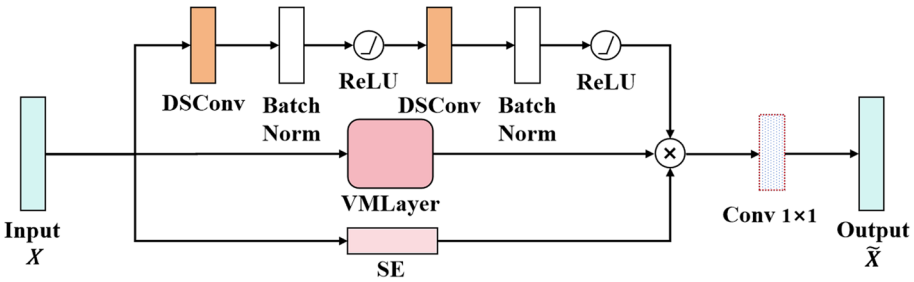


Fig. 6. Parallel structured state space encoder module.

where the SE module adaptively recalibrates channel-wise features. This step is crucial as it dynamically weights channels based on their relevance to the segmentation task, highlighting the most discriminative information from both the local and global branches while suppressing noise.

The attention-enhanced features from both branches are then concatenated and fused via a 1×1 convolution:

$$F^{\mathrm{PSSSE}} = \mathrm{Conv}_{1\times1}\left([\widetilde{F}^{(1)}, \widetilde{F}^{(2)}]\right) \tag{23}$$

This fusion step combines the refined local details and enhanced global context, creating a more comprehensive feature representation that is robust to both fine-grained variations and large-scale semantic ambiguities.

By cascading PSSSE with downsampling operations—halving the spatial resolution and doubling the channel dimension at each level—we construct a multi-level encoder that progressively refines features from fine-grained local details to high-level global semantics. This hierarchical design yields a rich yet efficient feature representation, critical for precise segmentation in resource-constrained scenarios.

Through the integrated design of VMLayer for global dependency modeling, SE-Skip Connection for cross-layer feature alignment, and PSSSE for parallel feature extraction, PUNet achieves a lightweight architecture with only 0.26 M total parameters. This design effectively balances local detail preservation and global context modeling, providing a reproducible and scalable solution for high-precision semantic segmentation tasks across diverse edge computing environments.

Experimental

Datasets

To comprehensively evaluate the model's performance, particularly its adaptability to both general scene understanding and specific defect detection, we selected two representative datasets: CamVid and CRACK500.

The CamVid dataset[42] consists of 701 driving-view images with an original resolution of 960×720 pixels. It provides 32-class semantic annotations for urban road scenes under various lighting conditions. Following the official split, 367 images were used for training, 101 for validation, and 233 for testing. All images were resized to 672×896 pixels to meet the model's input requirements. For data augmentation, we applied random scaling (0.5 to 2.0), horizontal flipping, and normalized the RGB channels using a fixed mean and standard deviation to promote model convergence.

The CRACK500 dataset[43] comprises 500 high-resolution asphalt pavement images (2000×1500 pixels) with sub-pixel accurate binary crack masks. To address data scarcity, we cropped the images into non-overlapping sub-blocks of 480×360 pixels, retaining only regions with at least 1000 crack pixels. This resulted in 3368 sub-images. During augmentation, we employed techniques such as random rotation ($\pm$ 15○), Gaussian blur (0.5 to 1.5), and contrast adjustment (0.8 to 1.2 times). We also normalized pixel values to the range [0, 1] and binarized the masks for consistent input.

For all experiments, including those with our proposed model and baselines like DeepLabv3 and SwinUNet, we applied a consistent preprocessing strategy. To meet the training requirements of models like SwinUNet and Mamba, which are typically optimized for square inputs, we resized the CRACK500 sub-images to 224×224 pixels. This step is crucial because Transformer-based architectures, such as Swin Transformer, rely on patch partitioning and window-based self-attention. They divide the input image into fixed-size, non-overlapping patches (e.g., 4×4 pixels), and the input resolution must be an integer multiple of the patch size to avoid artifacts from padding or truncation. The CamVid dataset was maintained at a size of 672×896 pixels. Notably, all benchmarked models were trained and evaluated using these same image dimensions (224×224 for CRACK500 and 672×896 for CamVid), ensuring a fair and direct comparison of their performance. Key dataset parameters are summarized in Table 1, validating the model's performance across diverse scenarios.

Implementation details

All experiments were conducted on an NVIDIA RTX 3090 GPU, with the software environment configured as CUDA 11.8 and Python 3.8, utilizing PyTorch 1.13.0 as the deep-learning framework. The Adam optimizer was employed with parameters: initial learning rate 1×10^{-3}, momentum coefficients $\beta_1 = 0.9$ and $\beta_2 = 0.999$, epsilon 1×10^{-8}, and weight decay 5×10^{-4}. A cosine annealing scheduler (CosineAnnealingLR) was applied to adjust the learning rate, with the maximum number of epochs set as $T_{\max} = 1000$. Data processing was implemented using the CamVidDataSet class. Training data underwent random scaling and mirroring augmentations, followed by normalization with fixed channel means $[0.485, 0.456, 0.406]$. Data loaders were configured with a batch size of 8 to balance computational efficiency and gradient estimation stability.

Evaluation metrics

In autonomous driving scene understanding and road crack detection, we use mean Dice (mDice) and mean Intersection over Union (mIoU) as primary segmentation metrics:

Dataset	Image count (train/validation/test)	Resolution (original/processed)
CamVid	367/101/233	960×720 to 672×896
Crack 500	2694/337/337	2000×1500 to 224×224

Table 1. Details of the two datasets used.

Models	Train Dice	Train IoU	Train Acc	Val Dice	Val IoU	Val Acc	Params (M)	FLOPs (MMac)
PUNet	0.9021	0.8357	0.9649	0.9208	0.8643	0.9726	0.26	237.30
U-Net[11]	0.8777	0.7968	0.9548	0.8540	0.9151	0.9699	31.04	41,929.98
Deeplabv3[13]	0.8949	0.8252	0.9616	0.8558	0.9146	0.9705	39.64	31,412.47
SegNet[44]	0.8199	0.7231	0.9384	0.7932	0.8694	0.9587	29.45	31,021.06
FCN[12]	0.8984	0.8303	0.9631	0.9155	0.8573	0.9720	32.95	26,577.78
Swin-Unet[22]	0.7712	0.8552	0.9510	0.8178	0.8846	0.9657	27.17	5,914.31
ENet[45]	0.8641	0.7804	0.9494	0.8319	0.8989	0.9648	0.36	431.49
LR-ASPP[46]	0.8709	0.7921	0.9549	0.8244	0.8909	0.9660	3.22	3,072.17
LightMUNet[19]	0.6734	0.7759	0.9271	0.7573	0.8352	0.9537	0.38	514.58
UltraLight VM-UNet[29]	0.7170	0.6829	0.9174	0.7723	0.7625	0.9425	0.05	55.18

Table 2. Performance comparison of PUNet and other models on the CamVid dataset.

Models	Train Dice	Train IoU	Train Acc	Val Dice	Val IoU	Val Acc	Params (M)	FLOPs (MMac)
PUNet	0.8173	0.7015	0.9789	0.7902	0.6612	0.9789	0.26	3,887.61
U-Net[11]	0.7641	0.6337	0.9728	0.7692	0.6374	0.9728	31.04	503,159.71
Deeplabv3[13]	0.9288	0.8757	0.9928	0.7981	0.6714	0.9928	39.64	376,943.81
SegNet[44]	0.7940	0.6703	0.9759	0.7786	0.6468	0.9759	29.45	372,252.73
FCN[12]	0.9160	0.8538	0.9914	0.7942	0.6659	0.9914	32.95	318,933.41
Swin-Unet[22]	0.8050	0.6849	0.9776	0.7766	0.6433	0.9776	27.17	53,228.82
ENet[45]	0.8364	0.7294	0.9819	0.7841	0.6524	0.9819	0.36	5,177.86
LR-ASPP[46]	0.8833	0.7997	0.9875	0.7970	0.6692	0.9875	3.22	5,864.37
LightMUNet[19]	0.7455	0.6085	0.9697	0.7547	0.6171	0.9697	0.38	6,174.68
UltraLight VM-UNet[29]	0.1195	0.0646	0.0646	0.1116	0.0600	0.0646	0.05	661.95

Table 3. Performance comparison of PUNet and other models on the CRACK500 Dataset.

$$\text{Dice} = \frac{2|P \cap T|}{|P| + |T|} \tag{24}$$

$$\text{IoU} = \frac{|P \cap T|}{|P \cup T|} \tag{25}$$

where P and T denote the predicted and ground-truth masks, respectively. Both metrics range in $[0, 1]$, with higher values indicating better overlap. IoU is generally more stringent when overlapping areas are small (hence often lower than Dice), while mDice is more robust to class imbalance; as overlap increases, the two converge.

During training, we monitor both Dice and mIoU. Dice mitigates background dominance and improves segmentation of petite or thin structures (e.g., fine cracks), and together with mIoU provides comprehensive feedback for sharper boundaries and more apparent class separation. For the final evaluation, in addition to mDice and mIoU, accuracy (Acc) is also reported. Among these, mIoU, owing to its robustness across object categories and scales, is widely regarded as the standard metric in semantic segmentation.

Experimental results and analysis

To comprehensively evaluate PUNet's accuracy and efficiency, we conducted comparative experiments on two benchmarks: CamVid (multi-class scene segmentation) and CRACK500 (road crack detection). The results are summarized in Tables 2 and 3. Tables 2 and 3 compare PUNet with several representative models, including traditional convolutional networks (e.g., FCN[12], DeepLabv3[13], SegNet[44], U-Net[11]), lightweight architectures (e.g., ENet[45], LightM-UNet[19], LR-ASPP[46], UltraLight VM-UNet[29]), and Transformer-based models (e.g., Swin-Unet[22]), evaluated by Dice, IoU, accuracy (Acc), Params (Million,M), and FLOPs (measured in million multiply-accumulate operations, MMac).

Performance on the CamVid dataset

Table 2 shows that PUNet achieves a validation Dice of 0.9208, IoU of 0.8643, and accuracy of 0.9726. Although its validation IoU is slightly lower than that of DeepLabv3 (0.9146) and ENet (0.8989), PUNet leads in both Dice and accuracy while using only 0.26 M parameters—an order of magnitude fewer than DeepLabv3 (39.64 M) and slightly fewer than ENet (0.36 M). More importantly, PUNet requires just 237.3 million multiply-accumulate operations (MMac FLOPs), drastically lower than DeepLabv3's 31,412.47 MMac and ENet's 431.49 MMac. This significant reduction in computational complexity, combined with a compact parameter size, highlights PUNet's capability to deliver high-precision multi-class segmentation efficiently, making it highly suitable for deployment on resource-constrained edge devices.

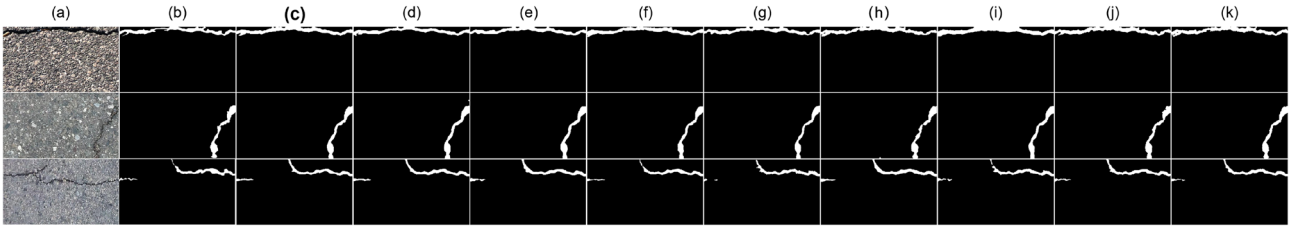


Fig. 7. Visualization results of different models on CRACK500. (**a**) Original crack images. (**b**) Ground truth of crack regions. (**c**) Results of PUNet. (**d**) Results of DeeplabV3. (**e**) Results of ENet. (**f**) Results of FCN. (**g**) Results of LightM-UNet. (**h**) Results of LR-ASPP. (**i**) Results of Swin-Unet. (**j**) Results of SegNet. (**k**) Results of UNet. PUNet is our proposed model, and the red markers indicate the cases of segmentation failure.

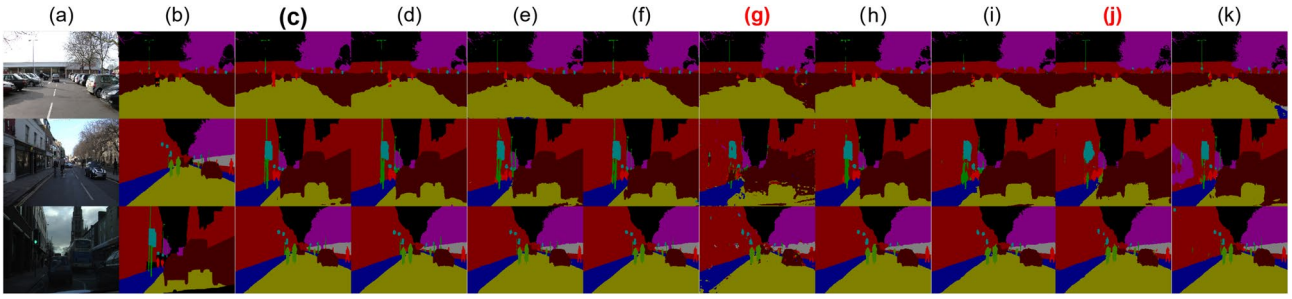


Fig. 8. Visualization results of different models on CamVid. (**a**) Original crack images. (**b**) Ground truth of crack regions. (**c**) Results of PUNet. (**d**) Results of DeeplabV3. (**e**) Results of ENet. (**f**) Results of FCN. (**g**) Results of LightM-UNet. (**h**) Results of LR-ASPP. (**i**) Results of Swin-Unet. (**j**) Results of SegNet. (**k**) Results of UNet. PUNet is our proposed model, and the red markers indicate the cases of segmentation failure.

Performance on the CRACK500 dataset

As reported in Table 3, large models such as DeepLabv3 achieve very high training metrics (Dice 0.9288, IoU 0.8757) but experience a notable drop in validation IoU to 0.6714, indicating potential overfitting. Moreover, these large models incur extremely high computational costs, with DeepLabv3 requiring 376,943.81 million multiply-accumulate operations (MMac FLOPs), which limits their practicality in resource-constrained environments. At the other extreme, the UltraLight VM-UNet is exceptionally lightweight and efficient, consuming only 661.95 MMac FLOPs, but achieves a poor validation Dice of 0.1116 due to insufficient feature extraction capacity.

In contrast, PUNet strikes an effective balance between accuracy and efficiency: with just 0.26 M parameters and 3887.61 MMac FLOPs—significantly lower than most competing models—it attains a validation Dice of 0.7902, IoU of 0.6612, and accuracy of 0.9789. This performance surpasses ENet (validation Dice 0.7841, 5177.86 MMac FLOPs) and LightM-UNet (validation Dice 0.7547, 6,174.68 MMac FLOPs), demonstrating robust accuracy while maintaining low computational cost, which is critical for practical deployment in resource-limited scenarios such as UAV-based inspections.

Segmentation result visualization

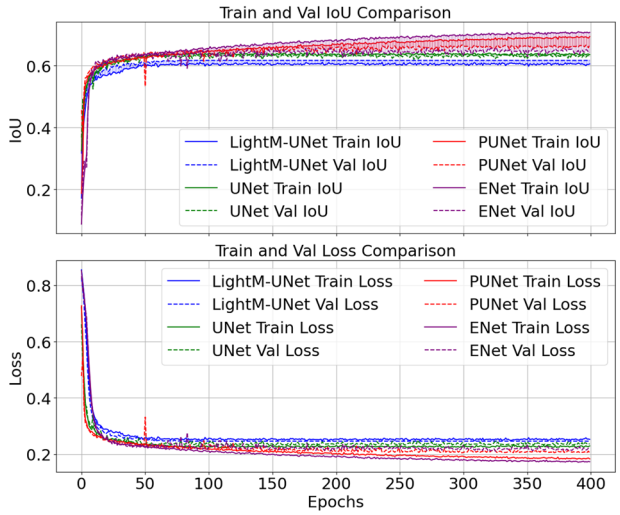
Figures 7 and 8 present qualitative segmentation outputs on the CamVid and CRACK500 datasets for Deeplabv3, ENet, FCN, LightM-UNet, LR-ASPP, Swin-Unet, SegNet, UltraLight VM-UNet, U-Net, and the proposed PUNet. Traditional single-path feature extractors struggle to model fine-grained boundary details and global scene context jointly. In contrast, PUNet employs parallel *local* and *global* branches to respectively capture spatial details and high-level semantic context. These multi-scale features are then fused via an attention mechanism to achieve complementary enhancement.

As observed in Fig. 8, this parallel design markedly improves pixel-level classification accuracy and preserves object contour integrity. PUNet consistently outperforms other models in regions of semantic ambiguity and along complex boundaries, demonstrating its superior feature representation capability in challenging scenarios.

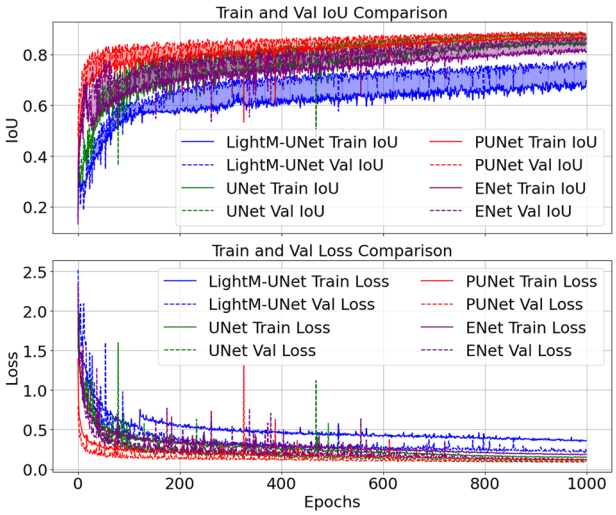
Training process analysis

Figure 9 shows the training loss and IoU curves of PUNet, UNet, LightM-UNet, and ENet in the Crack500 and CamVid datasets, respectively. These models are selected for comparison based on their architectural relevance or lightweight characteristics: PUNet shares a U-shaped structure with UNet, adopts Mamba-like modules similar to LightM-UNet, and ENet serves as a strong lightweight baseline.

As illustrated in Fig. 9a, PUNet exhibits rapid convergence and steady accuracy improvement on the Crack500 dataset, significantly outperforming the other three models. The loss curves of LightM-UNet and UNet tend to stabilize around the 50th epoch, with their IoU curves following a similar trend, whereas PUNet and ENet continue to optimize. After approximately 150 epochs, PUNet achieves a notably higher validation IoU



(a) IoU and Loss Comparison of Different Models during Training and Validation on the CRACK500



(b) IoU and Loss Comparison of Different Models during Training and Validation on the CamVid

Fig. 9. IoU and loss comparison of different models on different datasets.

Config	DSConv	VMLayer	SE-skip	PSSSE	Crack500			CamVid		
					mDice	mIoU	Acc	mDice	mIoU	Acc
I					0.7592	0.6274	0.9677	0.8942	0.8400	0.9697
II	✓				0.7557	0.6245	0.9655	0.8910	0.8341	0.9671
III	✓	✓			0.7621	0.6307	0.9693	0.9024	0.8432	0.9704
IV	✓	✓	✓		0.7757	0.6386	0.9779	0.9130	0.8517	0.9712
V	✓	✓	✓	✓	0.7902	0.6612	0.9789	0.9208	0.8643	0.9726

Table 4. Ablation study results: impact of different module combinations on model performance.

compared to ENet. Although ENet reaches the highest training accuracy, its validation IoU plateaus after 250 epochs, and a widening gap between training and validation IoU indicates the onset of overfitting.

Figure 9b presents the results on the CamVid dataset. Due to the complexity of multi-class segmentation, all models experience larger fluctuations during training. Nevertheless, PUNet maintains the most stable training dynamics, achieving continuous improvement with minimal overfitting. In contrast, both ENet and LightM-UNet suffer performance degradation in this setting, especially LightM-UNet, whose segmentation quality deteriorates significantly, as further confirmed by the visualization results in Fig. 8.

In summary, PUNet demonstrates efficient convergence and robust generalization across diverse datasets, validating the effectiveness and stability of its architectural design.

Ablation study

To further analyze the contribution of each key component in PUNet, we conduct a series of ablation experiments based on a simplified six-layer U-Net architecture (Config I) with channel sizes set to {16, 24, 32, 64, 128, 256}. We progressively integrate DSConv, the proposed VMLayer, SE-Skip connections, and PSSSE module. The detailed experimental results on Crack500 and CamVid datasets are summarized in Table 4.

Starting from the baseline model, a standard six-layer U-Net (Config I), we progressively integrate our proposed components. Replacing the standard convolutions in Config I with DSConv forms Config II, which slightly decreases the performance on both datasets (e.g., mDice on Crack500 drops from 0.7592 to 0.7557). This is because standard convolutions inherently perform both spatial feature extraction and cross-channel information fusion in a single operation. In contrast, DSConv separates these two processes: the depthwise convolution only performs spatial filtering within each channel, and the subsequent pointwise convolution handles cross-channel interaction. This separation may weaken the synergistic expressive power of features, leading to a less sufficient extraction of fine local details and cross-channel correlations, thereby resulting in a minor performance compromise.

Building upon Config II, we introduce the VMLayer into the encoder's architecture, as detailed in Fig. 1, to form Config III. This effectively remedies the performance drop, yielding a notable improvement compared to Config II (e.g., mDice rises from 0.7557 to 0.7621 on Crack500 and from 0.8910 to 0.9024 on CamVid). This confirms that the VMLayer, with its powerful ability to model long-range global dependencies, provides a strong semantic context that compensates for the weaker local features from the DSConv branch. This synergy

demonstrates that our parallel design is not merely an addition of components, but a deliberate architectural choice where the VMLayer's global context effectively "fills in the gaps" left by the DSConv's efficiency-driven local feature extraction.

We then extend Config III by replacing the standard skip connections with our proposed SE-Skip connections, creating Config IV, which further boosts the model's ability to capture cross-layer semantic information. This leads to a significant gain in validation metrics (e.g., Crack500 mDice increases from 0.7621 to 0.7757 and CamVid mDice from 0.9024 to 0.9130).

Finally, we integrate all proposed components, replacing the individual DSConv and VMLayer modules with the unified PSSSE, which encapsulates the parallel branches and SE-Skip connections for a cohesive and lightweight architecture (Config V). This configuration brings the most substantial improvement across all metrics, achieving the best performance with mDice scores of 0.7902 and 0.9208 on Crack500 and CamVid respectively. This demonstrates that the PSSSE module effectively captures hierarchical contextual dependencies and complements the lightweight backbone.

Overall, these ablation results validate the effectiveness and synergy of each proposed component in PUNet, collectively contributing to its strong segmentation performance while maintaining a remarkably compact model size.

Conclusion

This work introduces PUNet, a lightweight segmentation network tailored for resource-constrained environments, which integrates a depthwise separable convolution branch and a Mamba-based sequence modeling branch in a parallel encoder to balance local detail extraction and global context modeling via channel attention. The SE-Skip connection enhances cross-layer feature alignment, improving segmentation accuracy in complex scenes. Despite its compact size and linear computational complexity, PUNet outperforms traditional CNNs, lightweight architectures, and Transformer-based models on both CamVid dataset and CRACK500 dataset, verifying its practicality and robustness on edge devices. A limitation of the current work is its focus on specific application domains. Future work will validate its generalization ability across more diverse domains, explore multi-modal inputs, further improve the model, and test its real-time segmentation performance on mobile devices in real-world scenarios.

Data availability

The datasets used in this study are openly available. The CamVid dataset can be accessed at http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/, and the CRACK500 dataset is available at https://universe.roboflow.com/uni-of-birmingham/crack500-instance-segmentation.

Received: 9 May 2025; Accepted: 1 October 2025
Published online: 06 November 2025

References

1. Xing, Y. et al. Emg-yolo: Road crack detection algorithm for edge computing devices. *Front. Neurorobot.* **18**, 1423738 (2024).
2. Duan, Z., Liu, J., Ling, X., Zhang, J. & Liu, Z. Ernet: A rapid road crack detection method using low-altitude UAV remote sensing images. *Remote Sens.* **16**, 1741 (2024).
3. Ayachi, R., Afif, M., Said, Y. & Abdelali, A. B. Traffic sign recognition for smart vehicles based on lightweight CNN implementation on mobile devices. In *2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, 12–18 (IEEE, 2022).
4. Szilágyi, D., Sziroczák, D., Rohács, D. & Fendrik, Á. Feasibility of a drone-based road network inspection system. *Eng. Proc.* **79**, 76 (2024).
5. Tang, W., Wu, Z., Wang, W., Pan, Y. & Gan, W. VM-UNET++ research on crack image segmentation based on improved VM-UNET. *Sci. Rep.* **15**, 8938 (2025).
6. Gong, T., Zhu, L., Yu, F. R. & Tang, T. Edge intelligence in intelligent transportation systems: A survey. *IEEE Trans. Intell. Transp. Syst.* **24**, 8919–8944 (2023).
7. Bento, N. et al. Comparing handcrafted features and deep neural representations for domain generalization in human activity recognition. *Sensors* **22**, 7324 (2022).
8. Nanni, L., Ghidoni, S. & Brahnam, S. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognit.* **71**, 158–172 (2017).
9. Anari, S., Sadeghi, S., Sheikhi, G., Ranjbarzadeh, R. & Bendechache, M. Explainable attention based breast tumor segmentation using a combination of UNet, ResNet, DenseNet, and EfficientNet models. *Sci. Rep.* **15**, 1027 (2025).
10. Chen, Y., Yang, T.-J., Emer, J. & Sze, V. Understanding the limitations of existing energy-efficient design approaches for deep neural networks. *Energy* **2**, L3 (2018).
11. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III*, Vol. 18, 234–241 (Springer, 2015).
12. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440 (2015).
13. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F. & Adam, H. Encoder–decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 801–818 (2018).
14. Liu, R., Pu, W., Nan, H. & Zou, Y. Retina image segmentation using the three-path Unet model. *Sci. Rep.* **13**, 22579 (2023).
15. Zhou, Z., Rahman Siddiquee, M. M., Tajbakhsh, N. & Liang, J. Unet++: A nested u-net architecture for medical image segmentation. In *Deep learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, proceedings*, Vol. 4, 3–11 (Springer, 2018).
16. Oktay, O. et al. Attention U-net: Learning where to look for the pancreas. arXiv preprint arXiv:1804.03999 (2018).
17. Li, G. et al. A landslide area segmentation method based on an improved Unet. *Sci. Rep.* **15**, 11852 (2025).
18. Cao, H. et al. Swin-unet: Unet-like pure transformer for medical image segmentation. In *European Conference on Computer Vision*, 205–218 (Springer, 2022).

19. Liao, W. et al. Lightm-unet: Mamba assists in lightweight unet for medical image segmentation. arxiv 2024. arXiv preprint arXiv:2403.05246 (2024).
20. Xie, E. et al. Segformer: Simple and efficient design for semantic segmentation with transformers. *Adv. Neural. Inf. Process. Syst.* **34**, 12077–12090 (2021).
21. Liu, Z. et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10012–10022 (2021).
22. Hatamizadeh, A. et al. Swin unetr: Swin transformers for semantic segmentation of brain tumors in MRI images. In *International MICCAI Brainlesion Workshop*, 272–284 (Springer, 2021).
23. Ding, H., Jiang, X., Liu, A. Q., Thalmann, N. M. & Wang, G. Boundary-aware feature propagation for scene segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer vision*, 6819–6829 (2019).
24. Chen, J. et al. Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306 (2021).
25. Su, C., Luo, X., Li, S., Chen, L. & Wang, J. VMKLA-UNet: vision Mamba with KAN linear attention U-Net. *Sci. Rep.* **15**, 13258 (2025).
26. Vaswani, A. et al. Attention is all you need. In *Advances in Neural Information Processing Systems*, Vol. 30 (2017).
27. Gu, A. & Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752 (2023).
28. Liu, Y. et al. Vmamba: Visual state space model. *Adv. Neural. Inf. Process. Syst.* **37**, 103031–103063 (2024).
29. Wu, R., Liu, Y., Liang, P. & Chang, Q. Ultralight VM-UNET: Parallel vision mamba significantly reduces parameters for skin lesion segmentation. arXiv preprint arXiv:2403.20035 (2024).
30. Ruan, J., Li, J. & Xiang, S. VM-UNET: Vision mamba UNet for medical image segmentation. arXiv preprint arXiv:2402.02491 (2024).
31. Zhang, H. et al. A survey on visual Mamba. *Appl. Sci.* **14**, 5683 (2024).
32. Ma, J., Li, F. & Wang, B. U-Mamba: Enhancing long-range dependency for biomedical image segmentation. arXiv preprint arXiv:2401.04722 (2024).
33. Alom, M., Hasan, M., Yakopcic, C., Taha, T. & Asari, V. Recurrent residual convolutional neural network based on U-Net for medical image segmentation. arXiv preprint arXiv:1802.06955 (2018).
34. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. & Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4510–4520 (2018).
35. Zhang, X., Zhou, X., Lin, M. & Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6848–6856 (2018).
36. Lu, W., Chen, S.-B., Tang, J., Ding, C. H. & Luo, B. A robust feature downsampling module for remote-sensing visual tasks. *IEEE Trans. Geosci. Remote Sens.* **61**, 1–12 (2023).
37. Lu, W., Chen, S.-B., Ding, C. H., Tang, J. & Luo, B. Lwganet: A lightweight group attention backbone for remote sensing visual tasks. arXiv preprint arXiv:2501.10040 (2025).
38. Lu, W., Chen, S.-B., Shu, Q.-L., Tang, J. & Luo, B. Decouplenet: A lightweight backbone network with efficient feature decoupling for remote sensing visual tasks. *IEEE Trans. Geosci. Remote Sens.* **62**, 1–13. https://doi.org/10.1109/TGRS.2024.3465496 (2024).
39. Howard, A. G. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017).
40. Woo, S., Park, J., Lee, J.-Y. & Kweon, I. S. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19 (2018).
41. Hu, J., Shen, L. & Sun, G. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7132–7141 (2018).
42. Brostow, G. J., Fauqueur, J. & Cipolla, R. Semantic object classes in video: A high-definition ground truth database. *Pattern Recogn. Lett.* **30**, 88–97 (2009).
43. Zhang, L., Yang, F., Zhang, Y. D. & Zhu, Y. J. Road crack detection using deep convolutional neural network. In *2016 IEEE International Conference on Image Processing (ICIP)*, 3708–3712 (IEEE, 2016).
44. Badrinarayanan, V., Kendall, A. & Cipolla, R. A deep convolutional encoder–decoder architecture for image segmentation. In *Segnet* (2016).
45. Paszke, A., Chaurasia, A., Kim, S. & Culurciello, E. A deep neural network architecture for real-time semantic segmentation. In *Enet* (2016).
46. Howard, A. et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1314–1324 (2019).

Author contributions

Z.X. and H.M. conceived and supervised the study. Z.X. designed the PUNet architecture and drafted the manuscript. X.L. implemented the model and conducted experiments on CamVid and CRACK500 datasets and secured funding and oversaw project administration. H.M. performed formal analysis and visualization of results. S.W. carried out validation, ablation studies, and contributed to manuscript revision. D.C. collected, preprocessed, augmented, and managed the experimental data, and visualized the model results. All authors reviewed and approved the final manuscript.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Z.X.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.