IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 November 2025, accepted 10 December 2025, date of publication 22 December 2025,
date of current version 2 January 2026.

Digital Object Identifier 10.1109/ACCESS.2025.3646927

RESEARCH ARTICLE

BiSegUNet: An Efficient Binary Convolution Network With Multi-Scale Feature Refinement for Semantic Segmentation

HASSAN KHAN[1], SUNBAL IFTIKHAR[2], STEVEN DAVY[2], (Member, IEEE), AND JOHN G. BRESLIN[1], (Senior Member, IEEE)
[1]Data Science Institute, University of Galway, Galway, H91 TK33 Ireland
[2]Technological University (TU) Dublin, Ireland

Corresponding author: Hassan Khan (h.khan5@universityofgalway.ie)

This work was supported by Taighde Éireann-Research Ireland under Grant 21/FFP-A/9174 (SustAIn), Grant 21/RC/10303_P2 (VistaMilk), and Grant 12/RC/2289_P2 (Insight).

ABSTRACT Semantic segmentation is critical for applications like autonomous driving, medical imaging, and urban monitoring. However, existing state-of-the-art models often require high computational and memory resources, making them unsuitable for real-time and resource-constrained environments. This paper identifies the gap of feature refinement and multi-scale representation while balancing computational efficiency with segmentation accuracy, particularly for binary neural networks in semantic segmentation. We propose BiSegUNet, a novel lightweight semantic segmentation architecture. It incorporates a Capacity Block with dense connections, multi-branch convolutions, and attention mechanisms to enhance feature refinement and multi-scale context aggregation. Additionally, the architecture integrates a novel bottleneck design combining binary, grouped, and dilated convolutions for real-time performance without significant accuracy loss. Extensive evaluations on Cityscapes, PASCAL VOC, and ADE20K datasets demonstrate that BiSegUNet achieves competitive performance with up to 19× reduction in FLOPs and 6.5× memory savings compared to full-precision networks. It also outperforms comparable lightweight models, achieving 75.1% mIoU and 158 FPS inference speed at 1024×2048 and 512×1024 resolution respectively on Cityscapes dataset. These results highlight its scalability and potential for deployment in real-world applications like autonomous vehicles and edge computing, offering a promising solution for efficient semantic segmentation in constrained environments.

INDEX TERMS Binary neural network (BNN), real-time semantic segmentation, computational efficiency.

I. INTRODUCTION

Semantic segmentation, a fundamental task in computer vision, aims to assign a class label to every pixel in an image, enabling applications in autonomous driving, medical imaging, and environmental monitoring. The development of accurate and robust semantic segmentation models has been a primary focus of research in recent years, with architectures such as UNet [1], DeepLabv3+ [2], and PSPNet [3] pushing the boundaries of segmentation performance. However, these models often rely on computationally intensive convolutional operations, which limit their deployment in real-time or resource-constrained environments such as mobile devices and embedded systems.

The demand for computational efficiency has driven the exploration of various optimization techniques in deep learning. One notable approach is *quantization*, which reduces the precision of weights and activations to lower bit-widths, significantly reducing memory and computation requirements while maintaining accuracy [4], [5]. Another promising strategy is *knowledge distillation*, where a smaller network (student) learns from a larger, pre-trained network (teacher) to achieve competitive performance with fewer parameters [6]. Additionally, *neural*

The associate editor coordinating the review of this manuscript and approving it for publication was Tai Fei.

architecture search (NAS) has been employed to automatically discover efficient network architectures tailored for specific tasks [7], [8].

In the field of computer vision, lightweight architectures like MobileNets [9], ShuffleNet [10], BiSeNet [11], BiseNet-v2 [12], EdgeNet [13], MSCFNet [14], and LETNet [15] have demonstrated remarkable efficiency for classification and segmentation tasks. These models employ depthwise separable convolutions and channel shuffling, respectively, to reduce computational overhead. Such innovations have inspired adaptations for semantic segmentation, including MobileNetV3-based DeepLab [16] and ShuffleSeg [17]. For semantic segmentation specifically, efforts have focused on balancing accuracy and efficiency. Techniques such as the use of depthwise separable convolutions in lightweight backbones [18] and region-based pooling mechanisms [3] have been pivotal in reducing computational complexity. Additionally, real-time segmentation networks like ERFNet [19], ESPNet [20] and ESPNET-v2 [21] have introduced novel designs to achieve high-speed performance on edge devices. This paper introduces **BiSegUNet**, a novel *binary convolution-based neural architecture for semantic segmentation*, designed to address these limitations by incorporating a *Capacity Block*. The proposed capacity block combines dense connections and multi-branch convolutions with attention mechanisms, enhancing feature refinement and multi-scale representation while preserving computational efficiency. Positioned between the encoder and decoder stages, the Capacity block refines encoder outputs before they are integrated as skip connections into the decoder, facilitating improved context aggregation and feature alignment. To further enhance global context representation, the architecture integrates *Atrous Spatial Pyramid Pooling (ASPP)* and an *Attention Network Part (ANP)* in the bottleneck layer. These components capture multi-scale spatial information and apply channel-wise attention, respectively, ensuring robust performance across diverse segmentation scenarios. Figure 1 illustrates the evaluation of the proposed architecture, BiSegUNet, in comparison with state-of-the-art semantic segmentation models on the Cityscapes dataset. BiSegUNet achieves a compelling balance among these three factors, with significantly reduced FLOPs and high FPS, BiSegUNet outperforms many lightweight models while maintaining competitive accuracy. Compared to models like DeepLab-V3+, which achieve high mIOU but require substantial computational resources, BiSegUNet offers a sustainable alternative for real-time applications in resource-constrained environments.

We evaluate the proposed architecture on benchmark datasets, including Cityscapes [22], PASCAL VOC [23], and ADE20K [24], demonstrating its ability to achieve state-of-the-art efficiency while maintaining competitive accuracy. Ablation studies reveal the critical role of binary convolutions and the capacity block in balancing computational efficiency and segmentation performance.

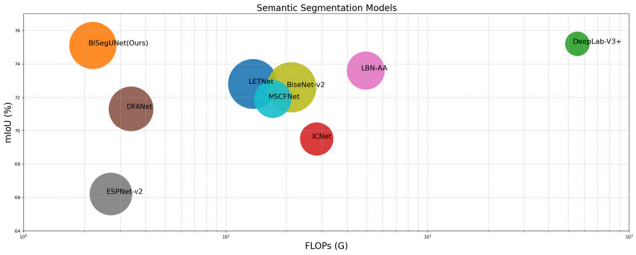

FIGURE 1. Evaluation of accuracy, computational complexity, and inference speed on the Cityscapes dataset. The circle radius represents the inference speed, with larger circles indicating faster performance. Our method is shown in orange.

The main contributions of this work are as follows:

- We propose a computationally efficient semantic segmentation architecture based on binary convolutions, achieving up to 19× FLOPs reduction and 6.5× memory savings compared to full-precision networks. When compared to BiseNet-v2, BiSegUNet achieves a 2.03× reduction in FLOPs and a marginally faster inference speed (1.01×), while delivering a 2.5% higher mIoU, demonstrating a superior balance between accuracy and efficiency.
- We introduce the Capacity Block, which combines dense connections, multi-branch convolutions, and attention mechanisms to enhance feature representation, refine encoder outputs, and improve segmentation performance.
- We leverage a novel bottleneck design that integrates grouped, binary, and dilated convolutions, reducing computational complexity and enabling real-time performance without significant accuracy loss.
- We systematically evaluate the effectiveness of the architecture through extensive experiments on Cityscapes and ADE20K, highlighting its generalization capability, scalability, and applicability to resource-constrained scenarios such as autonomous driving and edge computing.

The proposed Capacity Block combines dense connectivity, multi-branch convolutions, and channel attention in a form tailored to efficient binary segmentation. It fuses three parallel convolutional transformations such as binary, dilated, and grouped convolutions afterwards applies a learned channel-wise attention to the concatenated outputs, thereby capturing both fine spatial detail and broader contextual information within a lightweight module. By inserting Capacity Blocks at encoder–decoder skip connections, the network densifies information flow and injects multi-scale context early into decoder features, which is distinct from typical ASPP or SE/CBAM modules that are usually applied only at the bottleneck or a single scale. This design directly addresses feature degradation in binary networks, since binary and grouped branches keep computation low while attention re-weights informative channels, and it yields a substantial improvement in mIoU over a baseline without Capacity Blocks, as confirmed by our ablation results.

The remainder of the paper is organized as follows: Section II reviews related work on semantic segmentation and binary neural networks. Section III describes the proposed method, including the capacity block and bottleneck layer. Section IV provides the implementation details. Section V presents experimental results and discussion. Section VI presents the ablation study. Finally, Section VII concludes the paper and outlines future directions.

II. RELATED WORK

The field of semantic segmentation has witnessed significant progress due to advancements in deep learning. However, the computational demands of state-of-the-art segmentation architectures limit their deployment in resource-constrained environments. This section reviews related works focusing on **binary neural networks (BNNs), efficient architectures,** and the challenges of applying BNNs to segmentation tasks.

A. NETWORK QUANTIZATION FOR EFFICIENCY

Network quantization has emerged as a powerful approach for reducing the computational and memory footprint of deep neural networks. By lowering the precision of weights and activations, quantized networks enable deployment on resource-constrained devices such as edge and mobile platforms. Quantization techniques, such as those proposed by Jacob et al. [4], allow neural networks to operate with integer arithmetic while maintaining competitive accuracy. A recent survey by Gholami et al. [5] categorizes advancements in quantization strategies, including post-training quantization (PTQ), quantization-aware training (QAT), and mixed-precision quantization, highlighting their effectiveness in image classification. Quantization has been particularly successful in classification tasks, where reduced precision offers significant gains. Works such as PACT [25], QIL [26], and HAQ [27] explore various adaptive quantization strategies to balance accuracy and efficiency. These approaches leverage loss-aware quantization and search techniques to optimize bit-width allocation for each layer, achieving state-of-the-art results on classification benchmarks like ImageNet. Despite these successes, extending quantization to tasks like semantic segmentation remains challenging due to the need to preserve spatial granularity and contextual features. Existing segmentation architectures such as MobileNetV3 [16] and ESPNet [20] employ quantized backbones for efficient processing but often rely on floating-point operations in decoder stages to retain accuracy. Our work bridges this gap by integrating binary operations throughout the architecture for segmentation tasks.

Binary Neural Networks (BNNs) take quantization to the extreme by constraining weights and activations to 1-bit, enabling highly efficient models with low memory requirements. Early works like BinaryConnect [28] and BinaryNet [29] introduced binarized weights for classification, reducing model size while maintaining competitive accuracy. XNOR-Net [30] further optimized binary convolutions with approximations of floating-point operations, enabling large-scale image classification with significant computational savings. Recent advancements in BNNs, such as Bi-Real Net [31] and MeliusNet [32], improve representational capacity through shortcuts and multi-scale learning, achieving improved accuracy on datasets like CIFAR-10 and ImageNet. Group-Net [33] demonstrates that structured binary convolutions can bridge the gap between accuracy and efficiency. While these efforts have been largely confined to classification tasks, they establish a foundation for extending binary methods to more complex domains.

While BNNs have proven effective in classification, their application to semantic segmentation is still limited. Unlike classification, segmentation requires pixel-level precision, which is difficult to achieve with binary operations due to their inherently coarse representations. Initial attempts, such as Binary DAD-Net [34], explored domain-specific segmentation tasks but were restricted by accuracy constraints. Cellular BNNs [35] represent one of the few attempts to apply binary operations to segmentation tasks, incorporating cellular automata-like structures to refine feature maps. However, these approaches often fall short in capturing global context and multi-scale features, which are critical for segmentation. Structured BNNs [33] address this by employing parallel binary convolutions, but their computational overhead limits their practical deployment.

B. EFFICIENT ARCHITECTURES FOR SEMANTIC SEGMENTATION

Efforts to design efficient architectures for segmentation have predominantly focused on compact networks and separable convolutions. MobileNetV3 [16] combines depthwise separable convolutions with an optimized backbone, while BiSeNet [11] employs a bilateral network structure to separate spatial and semantic processing. ShuffleNet [10] and ESPNet [20] aim to minimize computational costs by rethinking kernel design. However, these architectures rely on floating-point operations, which still incur higher computational costs compared to BNNs.

Another area of research focuses on neural architecture search (NAS), as explored in Auto-DeepLab [36] and recent advancements like FBNet [37]. These methods automate the design of segmentation models to optimize efficiency and performance. Despite these innovations, achieving a balance between fine-grained accuracy and computational constraints remains an open challenge.

The loss of precision in BNNs, due to extreme quantization, poses significant challenges in capturing fine-grained details. Prior methods like Structured BNNs [33] attempt to address this by introducing parallel binary structures, but they fall short of achieving competitive segmentation performance.

Our work, BiSegUNet, addresses these challenges by leveraging binary convolutions and proposing capacity blocks. The integration of multi-scale feature refinement and attention mechanisms ensures the preservation of spatial

granularity, while binary convolutions drastically reduce computational overhead. To our knowledge, BiSegUNet represents comprehensive applications of BNNs for general semantic segmentation, achieving a balance between accuracy and efficiency.

III. PROPOSED METHOD

In this section, we present the *BiSegUNet* architecture, a novel encoder-decoder framework designed to leverage binary convolutions, capacity-enhancing intermediate modules, and multi-scale attention mechanisms for semantic segmentation. As illustrated in Fig. 2, BiSegUNet integrates: (1) a Multi-Stream Gateway Block for enriched initial features, (2) a hierarchical encoder for progressive semantic abstraction, (3) Capacity Blocks supported by Atrous Spatial Pyramid Pooling (ASPP) and an Attention Network Part (ANP) at the bottleneck to model global context, and (4) a decoder that reconstructs a high-resolution segmentation map. Crucially, the output of the ANP, residing within the bottleneck layers, is fused at the first decoder level along with the corresponding capacity block output from the encoder, ensuring a seamless integration of global context and local detail early in the decoding process.The summarized layer details of architecture is presented in Table 1.

A. MULTI-STREAM GATEWAY BLOCK

The input to BiSegUNet is a high-resolution image $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$. The Multi-Stream Gateway Block applies parallel convolutional transformations to produce an enriched initial feature map $\mathbf{F}_0 \in \mathbb{R}^{H\times W\times C}$. Let $\mathcal{C}(\cdot)$ represent a generic convolutional operation (including nonlinearity and normalization). Then:

$$\mathbf{F}_0 = \mathcal{F}(\mathbf{I}) = \mathcal{C}_1(\mathbf{I}) \oplus \mathcal{C}_2(\mathbf{I}) \oplus \cdots \oplus \mathcal{C}_M(\mathbf{I}), \tag{1}$$

where $\oplus$ denotes concatenation along the channel dimension and each $\mathcal{C}_m$ represents a parallel convolutional stream. Incorporating multiple kernels and features ensures a diverse set of initial representations.

B. ENCODER WITH HIERARCHICAL FEATURE EXTRACTION

The encoder transforms $\mathbf{F}_0$ into a hierarchy of feature maps $\{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_L\}$, progressively extracting higher-level semantics. For encoder stage l:

$$\mathbf{F}_l = \mathcal{E}_l(\mathbf{F}_{l-1}), \tag{2}$$

where $\mathcal{E}_l$ typically involves convolutions and downsampling. This process yields increasingly abstract representations while reducing spatial resolution. Intermediate encoder outputs $\mathbf{F}_l$ are later integrated into the decoder through skip connections, preserving fine-grained spatial information.

C. CAPACITY BLOCKS AND BOTTLENECK LAYER

1) CAPACITY BLOCK CONSTRUCTION

Positioned between encoder and decoder stages Fig. 3, each Capacity Block refines intermediate encoder features. Let $\mathbf{F}_{enc} \in \mathbb{R}^{H'\times W'\times C'}$ be the input from the encoder:

a: BINARY CONVOLUTIONS

Let $\mathcal{B}(\cdot)$ denote a binary convolution (weights $\in \{-1, +1\}$). Two sequential 3×3 binary convolutions and a parallel single 3×3 binary convolution yield:

$$\mathbf{B}_1 = \mathcal{B}_1(\mathcal{B}_2(\mathbf{F}_{enc})), \quad \mathbf{B}_2 = \mathcal{B}_3(\mathbf{F}_{enc}). \tag{3}$$

b: DILATED CONVOLUTION

A 7×7 dilated convolution with $d = 2$ expands the receptive field:

$$\mathbf{D} = \mathcal{D}_2(\mathbf{F}_{enc}), \tag{4}$$

where:

$$\mathcal{D}_2(\mathbf{F}_{enc})(x, y) = \sum_{(i,j)\in\Omega} w_{i,j}\mathbf{F}_{enc}(x + 2i, y + 2j). \tag{5}$$

c: GROUPED CONVOLUTION

A 3×3 grouped convolution $\mathcal{G}(\cdot)$, splitting channels into subsets:

$$\mathbf{G} = \mathcal{G}(\mathbf{F}_{enc}). \tag{6}$$

Concatenating these outputs:

$$\mathbf{F}_{cap} = [\mathbf{B}_1 \oplus \mathbf{B}_2 \oplus \mathbf{D} \oplus \mathbf{G}]. \tag{7}$$

2) CHANNEL ATTENTION MECHANISM

A channel attention mechanism refines $\mathbf{F}_{cap}$ by emphasizing informative channels. Compute a global descriptor:

$$z_c = \frac{1}{H'W'}\sum_{x=1}^{H'}\sum_{y=1}^{W'} f_c(x, y), \tag{8}$$

for each channel c. Let $\delta(\cdot)$ be a nonlinear activation (e.g., ReLU) and $\sigma(\cdot)$ be a sigmoid. After passing $[z_1, z_2, \ldots, z_{C_{cap}}]$ through a two-layer MLP:

$$\mathbf{w} = \sigma(W_2\delta(W_1\mathbf{z})), \tag{9}$$

the attended feature map is:

$$\mathbf{F}_{att}(x, y, c) = w_c \cdot \mathbf{F}_{cap}(x, y, c). \tag{10}$$

3) ASPP AND ANP BOTTLENECK

At the bottleneck, Atrous Spatial Pyramid Pooling (ASPP) captures multi-scale context:

$$\mathbf{F}_{aspp} = \bigoplus_r \mathcal{D}_r(\mathbf{F}_{att}), \tag{11}$$

where r are chosen dilation rates. The Attention Network Part (ANP) applies self-attention:

$$\mathbf{Q} = \mathbf{F}_{aspp}W_Q, \quad \mathbf{K} = \mathbf{F}_{aspp}W_K, \quad \mathbf{V} = \mathbf{F}_{aspp}W_V. \tag{12}$$

The attention matrix:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right), \tag{13}$$

yields the bottleneck output:

$$\mathbf{F}_{bottleneck} = \mathbf{A}\mathbf{V}. \tag{14}$$

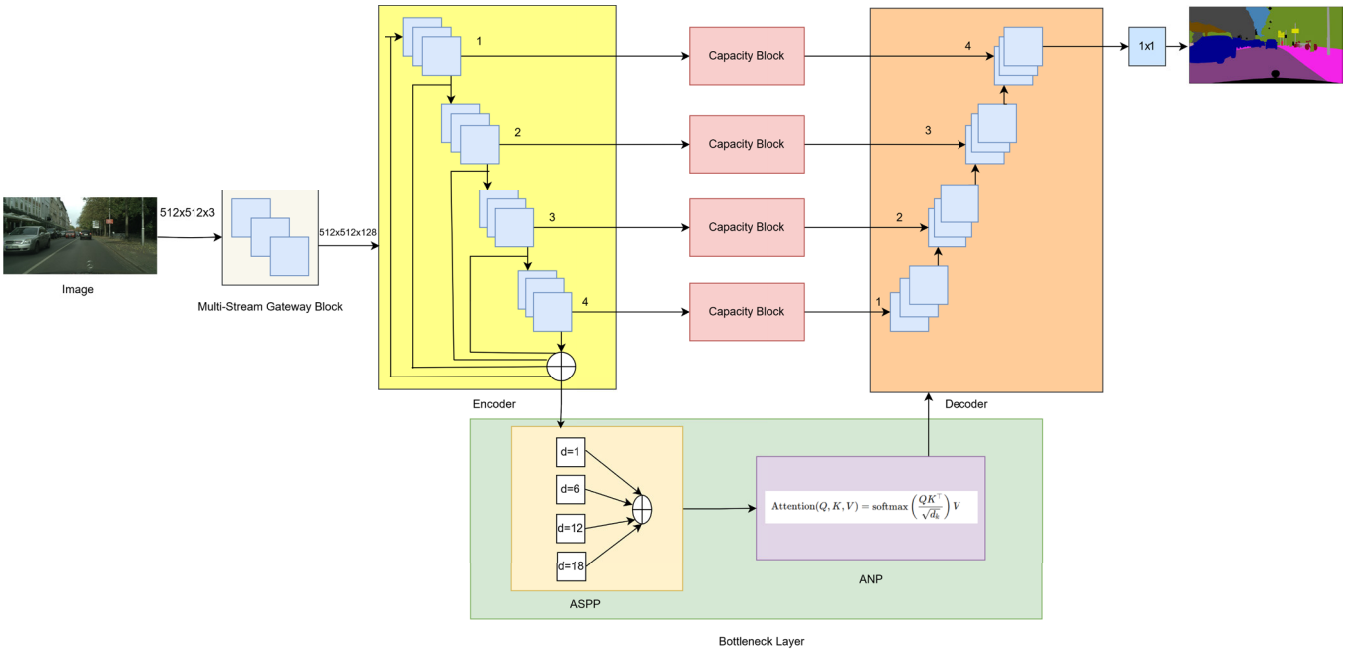

FIGURE 2. Overview of the proposed BiSegUNet architecture. The network begins with a Multi-Stream Gateway Block for enriched initial features, followed by a hierarchical encoder to capture semantic abstractions. Capacity blocks, combined with ASPP and ANP at the bottleneck, provide global context and refined features. A symmetric decoder with skip connections then reconstructs accurate, context-aware segmentation masks.

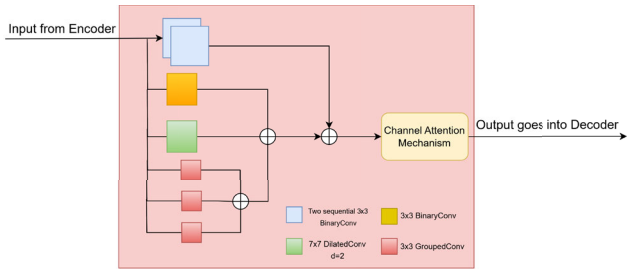

FIGURE 3. Illustration of the capacity block. Parallel binary, dilated, and grouped convolutions extract diverse features, which are merged and reweighted by channel attention. This produces contextually rich, discriminative representations that aid the decoder in producing precise segmentations.

D. DECODER AND FINAL PREDICTION

The output of the ANP ($\mathbf{F}_{bottleneck}$) is integrated at the *first decoder level*, together with the skip connection from the Capacity Block at the top encoder stage. Let $\mathbf{F}^1_{dec}$ denote the feature map at the first decoder level. We fuse:

$$\mathbf{F}^1_{dec} = \mathcal{H}(\mathbf{F}_{bottleneck}, \mathbf{F}^{(top)}_{att}), \tag{15}$$

where $\mathbf{F}^{(top)}_{att}$ is the attended feature from the top encoder stage's Capacity Block, and $\mathcal{H}(\cdot)$ denotes a fusion operation (e.g., convolution + nonlinearity).

Subsequent decoder stages progressively upsample and integrate intermediate skip connections:

$$\mathbf{F}^l_{dec} = \mathcal{U}(\mathbf{F}^{l+1}_{dec}) \oplus \mathcal{H}'(\mathbf{F}_l, \mathbf{F}^{(l)}_{att}), \tag{16}$$

for $l = 2, \ldots, L$, where $\mathcal{H}'(\cdot)$ and $\mathcal{U}(\cdot)$ are analogous fusion and upsampling operations, and $\mathbf{F}^{(l)}_{att}$ are the Capacity Block outputs at intermediate encoder levels. A final 1×1 convolution maps the final decoder output to the segmentation classes:

$$\hat{\mathbf{Y}} = \mathcal{C}_{1\times1}(\mathbf{F}^L_{dec}). \tag{17}$$

IV. IMPLEMENTATION DETAILS

The proposed BiSegUNet model is trained over a total of 180 epochs on 3 A4000 GPU (16GB), transitioning from an initial float-precision regime to a fully binarized configuration. This approach ensures stable convergence, robust feature learning, and efficient inference on embedded hardware (NVIDIA AGX ORIN). During the initial phase (first 50 epochs), we train the network in floating-point (FP32) precision using a standard segmentation objective (a combination of cross-entropy and dice loss to capture both pixel and region level [38]) and the Adam optimizer.

A. LOSS FUNCTION

In this paper, we employ a hybrid loss function combining cross-entropy loss and Dice loss to optimize BiSegUNet for semantic segmentation. This combination effectively balances pixel-level accuracy and region-level overlap, which is essential for the high-performance segmentation tasks that BiSegUNet addresses. Additionally, to encourage binary weight learning while maintaining model efficiency, we incorporate Binary Weight Regularization as a regularization technique. The total loss function is composed of two main components: cross-entropy loss and Dice loss.

TABLE 1. Summary of the BiSegUNet architecture.

Stage	Input Size	Operation	Output Size
Input Layer	$512 \times 512 \times 3$	Normalize pixel values.	$512 \times 512 \times 3$
Grouped Stem Block	$512 \times 512 \times 3$	3×3 BinaryConv2D, strides=1, output channels=[32, 64, 128].	$512 \times 512 \times 128$
Encoder Stage 1	$512 \times 512 \times 128$	3×3 BinaryConv2D, strides=2, output channels=256.	$256 \times 256 \times 256$
Encoder Stage 2	$256 \times 256 \times 256$	3×3 BinaryConv2D, strides=2, output channels=512.	$128 \times 128 \times 512$
Encoder Stage 3	$128 \times 128 \times 512$	3×3 BinaryConv2D, strides=2, output channels=1024.	$64 \times 64 \times 1024$
Encoder Stage 4	$64 \times 64 \times 1024$	3×3 BinaryConv2D, strides=2, output channels=2048.	$32 \times 32 \times 2048$
Capacity Block	$H \times W \times C$ (Encoder Output)	Two 3×3 BinaryConv2D (output: $0.5C$), concatenation, channel-wise attention.	$H \times W \times C$
Bottleneck Layer	$32 \times 32 \times 2048$ (concatenated encoder outputs)	ASPP with dilations [1, 6, 12, 18], followed by Attention Neural Process (ANP).	$32 \times 32 \times 1024$
Decoder Stage 1	$32 \times 32 \times 1024$ + Capacity Block ($32 \times 32 \times 2048$)	Upsampling and addition of skip connection from Encoder Stage 4.	$64 \times 64 \times 1024$
Decoder Stage 2	$64 \times 64 \times 1024$ + Capacity Block ($64 \times 64 \times 1024$)	Upsampling and addition of skip connection from Encoder Stage 3.	$128 \times 128 \times 512$
Decoder Stage 3	$128 \times 128 \times 512$ + Capacity Block ($128 \times 128 \times 512$)	Upsampling and addition of skip connection from Encoder Stage 2.	$256 \times 256 \times 256$
Decoder Stage 4	$256 \times 256 \times 256$ + Capacity Block ($256 \times 256 \times 256$)	Upsampling and addition of skip connection from Encoder Stage 1.	$512 \times 512 \times 128$
Capacity Block	$H \times W \times C$ (Decoder Output)	Multi-branch processing (3×3 BinaryConv2D, dilated conv, grouped conv), fusion with attention.	$H \times W \times C$
Output Layer	$512 \times 512 \times 128$	1×1 Conv2D to reduce the number of classes.	$512 \times 512 \times num\_classes$

The cross-entropy loss captures pixel-level classification accuracy, commonly used in segmentation tasks to penalize incorrect class predictions at the pixel level. For each pixel i and class c, the cross-entropy loss is defined as:

$$\mathcal{L}_{\text{CE}} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) \tag{18}$$

where y_c is the ground truth label for class c, $\hat{y}_c$ is the predicted probability for class c, and C is the total number of classes. This loss ensures that the model learns to classify each pixel accurately.

The Dice loss is designed to capture region-level overlap between predicted and ground truth regions, which is particularly useful in cases of class imbalance (e.g., small object regions). The Dice coefficient for a binary class is defined as:

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum_{i=1}^{N} \hat{y}_i y_i}{\sum_{i=1}^{N} \hat{y}_i + \sum_{i=1}^{N} y_i} \tag{19}$$

where $\hat{y}_i$ is the predicted value for pixel i, and y_i is the ground truth label. For multi-class segmentation, Dice loss is computed for each class and averaged. This loss encourages the network to focus on region-level coherence, making it especially useful in segmentation tasks with highly imbalanced classes.

The final combined loss is a weighted sum of both cross-entropy and Dice loss:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{CE}} + \lambda_2 \mathcal{L}_{\text{Dice}} \tag{20}$$

where λ_1 and λ_2 are hyperparameters balancing the importance of pixel-level and region-level accuracy. The combination of these losses ensures that the model learns both fine-grained pixel information and global spatial patterns, leading to more accurate segmentation results. Rectified Linear Units (ReLU) [39] serve as activation functions:

$$\delta(z) = \max(0, z), \tag{21}$$

ensuring stable gradients and well-conditioned representations. Batch Normalization (BN) and He-initialization further stabilize early training and enhance convergence. After establishing a strong float-precision baseline, we introduce binarization over subsequent epochs (50–150). Learning rate follows a cosine-annealing schedule [40] in this phase starting from an initial value η_0. Rather than converting all layers

at once, subsets of convolutional layers are incrementally binarized using:

$$\mathbf{W}^{(bin)} = \text{sign}(\mathbf{W}^{(fp)}), \tag{22}$$

and a straight-through estimator [41] to approximate gradients through the sign function. This phased approach mitigates performance degradation and allows the model to adapt gradually. Each binarized layer employs a learnable scaling factor α:

$$\mathbf{W}^{(scaled)} = \alpha \mathbf{W}^{(bin)}, \quad \alpha > 0. \tag{23}$$

B. BINARY WEIGHT REGULARIZATION

Binary neural networks (BNNs) require the model's weights to be binarized ($\{-1, +1\}$). However, during training, the weights are continuous, and binarization occurs during the forward pass. To ensure that the learned weights remain close to binary values, we apply Binary Weight Regularization. This technique adds a penalty term to the loss function that encourages the network to keep its weights as close to binary values as possible. The binary weight regularization term is defined as the L1 norm of the difference between the real-valued weights and their binarized counterparts:

$$\mathcal{L}_{\text{bin}} = \sum_{l} \|\mathbf{W}^{(l)} - \text{sign}(\mathbf{W}^{(l)})\|_1 \tag{24}$$

where $\mathbf{W}^{(l)}$ represents the weights of layer l, and $\text{sign}(\mathbf{W}^{(l)})$ is the binarized version of these weights (either -1 or $+1$). The regularization term penalizes deviations from binary values, encouraging the model to learn weights that are close to binary, preserving the efficiency benefits of binary convolutions. The total loss function, combining the cross-entropy, Dice loss, and binary weight regularization, is formulated as:

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{total}} + \lambda_3 \mathcal{L}_{\text{bin}} \tag{25}$$

where λ_3 is a hyperparameter controlling the contribution of the binary weight regularization term. This regularization technique ensures that the binary model does not drift too far from the desired binary weights, making the binary convolutions efficient in terms of memory and computational cost while maintaining segmentation performance. Optimizing α jointly with the weights preserves a suitable dynamic range, improving stability and segmentation accuracy under binary constraints. We apply spatial and photometric augmentations (random crops, flips, color perturbations) to enhance robustness. Light regularization such as label smoothing:

$$\tilde{y}_c = (1 - \epsilon) y_c + \frac{\epsilon}{C}, \tag{26}$$

discourages overconfidence and promotes flatter minima, which benefit the binary regime. In the middle phase, we adopt a cosine-annealing schedule with periodic warm restarts [42]. Each restart temporarily increases the learning rate, enabling the optimizer to escape suboptimal minima and encouraging stable improvement until training completes at

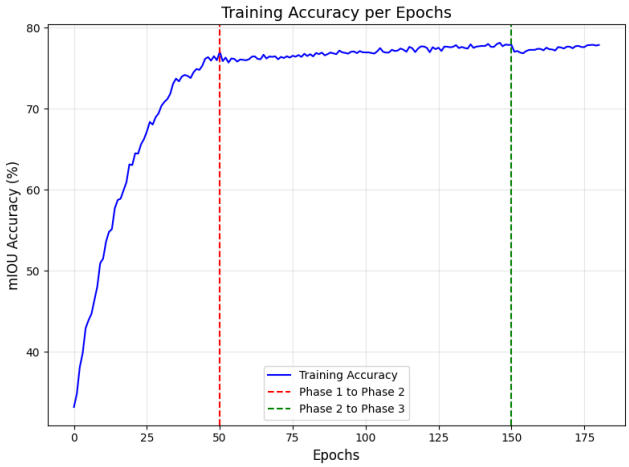

FIGURE 4. This figure shows the training accuracy of the BiSegUNet model, measured by mIoU, across epochs. The red dashed line marks the transition to binary weights at epoch 50, and the green dashed line indicates the shift to the fully binarized phase at epoch 150. Accuracy stabilizes after the transition to binary weights.

180 epochs. In the final stage (last 30 epochs), the model operates fully with binarized weights and ReLU activations, refining its representations and achieving stable convergence. The final model is then exported to the NVIDIA AGX ORIN, leveraging efficient binary operations for real-time, low-latency inference.

V. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we present the experimental results obtained from evaluating the **BiSegUNet** model on three widely used benchmark datasets for semantic segmentation: **PASCAL VOC, Cityscapes, and ADE20K**. We discuss the performance of the model in terms of accuracy, segmentation quality, and computational efficiency.

A. DATASETS AND DATA PREPARATION

For the experiments, we used the following datasets:

1) PASCAL VOC

The PASCAL VOC dataset consists of 1,464 training images, 1,449 validation images, and 1,456 test images, with 20 foreground object classes and one background class. To improve the performance of segmentation models, we augmented the original dataset by incorporating additional annotations from the Semantic Boundaries Dataset (SBD) [43]. This augmentation resulted in a total of 10,582 training images, providing more diverse and detailed segmentation annotations. The evaluation metric used for performance assessment was the mean intersection-over-union (mIoU), averaged across the 21 classes (20 foreground classes and the background class). This metric is widely used to evaluate segmentation models and provides a clear measure of pixel-wise segmentation accuracy.

Algorithm 1 Training Procedure of BiSegUNet

Training data $\mathcal{D}$, combined loss function $\mathcal{L}_{\text{total}}$, learning rate η, epochs E, optimizer $\mathcal{O}$. Trained BiSegUNet model.

Forward Process:

Initialize input $I \in \mathbb{R}^{H\times W\times 3}$.

Extract multi-scale features using **Multi-Stream Gateway Block**:

$$F_0 = \text{Concat}(C_1(I), C_2(I), \ldots, C_M(I))$$

Generate hierarchical features with **Hierarchical Encoder**:

$$F_l = E_l(F_{l-1}), \quad l = 1, \ldots, L$$

Refine features with **Capacity Block**:

$$F_{\text{cap}} = \text{Att}(\text{Concat}(\text{DilConv}, \text{GroupConv}, \text{BinConv}))$$

Apply ASPP and ANP for context aggregation:

$$F_{\text{ASPP}} = \text{Concat}\big(\text{DilConv}_r(F_{\text{cap}})\big), \ r \in \{1, 6, 12, 18\}$$

$$F_{\text{ANP}} = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V,$$

$$Q, K, V = W_Q F_{\text{ASPP}}, W_K F_{\text{ASPP}}, W_V F_{\text{ASPP}}$$

Decode and integrate skip connections:

$$F_{\text{dec}} = \text{Upsample}(F_{\text{ANP}}) \oplus H(F_{\text{cap}}, F_{\text{enc}})$$

Output segmentation map $\hat{Y}$ using pixel-wise classification:

$$\hat{Y} = \text{Conv}_{1\times 1}(F_{\text{dec}})$$

Backward Process:

Compute loss:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{CE}} + \lambda_2 \mathcal{L}_{\text{Dice}}$$

Update weights using gradient descent:

$$W \leftarrow W - \eta \frac{\partial \mathcal{L}_{\text{total}}}{\partial W}$$

Segmentation map $\hat{Y}$.

2) CITYSCAPES

The Cityscapes dataset is designed for urban scene understanding, containing fine-grained pixel-level annotations of 5,000 images, split into 2,975 training images, 500 validation images, and 1,525 test images. This dataset is known for its large-scale street view images captured in multiple cities. The model's performance on this dataset is evaluated using mIoU over the 19 foreground object classes.

3) ADE20K

ADE20K is a large-scale dataset for semantic segmentation, containing 20,210 images with dense pixel-level annotations across 150 object categories. This dataset provides a more challenging evaluation due to its wide range of object classes and diverse environments. As with the other datasets, the model performance on ADE20K is evaluated using mIoU.

The proposed BiSegUNet architecture is evaluated using both performance and efficiency metrics to demonstrate its effectiveness and suitability for real-time semantic segmentation in resource-constrained environments.

B. PERFORMANCE METRICS

The proposed BiSegUNet architecture is evaluated using both performance and efficiency metrics to demonstrate its effectiveness and suitability for real-time semantic segmentation in resource-constrained environments. **Mean Intersection over Union (mIoU):** This metric evaluates segmentation accuracy by calculating the overlap between the predicted and ground truth regions across all classes. It serves as the primary measure of segmentation quality. **Dice Score:** The Dice coefficient measures the similarity between the predicted and ground truth regions, emphasizing how well the model segments objects. **Pixel Accuracy:** This metric quantifies the percentage of correctly classified pixels, providing an overall assessment of segmentation accuracy.

C. EFFICIENCY METRICS

FLOPs: Floating Point Operations (FLOPs) represent the computational complexity of the model during inference, reflecting its processing demand. **Memory Usage:** This metric indicates the memory required to store model parameters and intermediate computations, highlighting the model's resource footprint. **Speed (FPS):** Frames Per Second (FPS) measures the inference speed, demonstrating the model's capability for real-time applications. **Parameters:** This metric reflects the total number of trainable parameters in the model, which directly impacts storage requirements and model size.

These metrics collectively assess the balance between segmentation accuracy and computational efficiency, illustrating the effectiveness of BiSegUNet for scalable and sustainable semantic segmentation tasks.

D. TRAINING RESULTS

This section presents the experimental results of the **BiSegUNet** model trained on standard segmentation dataset **Cityscapes**. We focus on the evaluation of training accuracy and learning rate scheduling.

As shown in Fig. 4, the accuracy of the training is evaluated using mean intersection-over-union (mIoU). The model exhibits rapid improvement in accuracy during the early epochs when training with float precision, as indicated by the steep rise in accuracy. The transition from float precision to binary weights at epoch 50 results in a stable increase in accuracy, highlighting that the model maintains robust performance even with binary weights. By epoch 150, the accuracy plateaus, indicating that the model has fully adapted to the binary weights and achieved optimal performance. This smooth progression confirms that the BiSegUNet model is capable of retaining high accuracy while transitioning to a more efficient binary weight representation. The learning rate schedule follows a cosine annealing strategy with warm

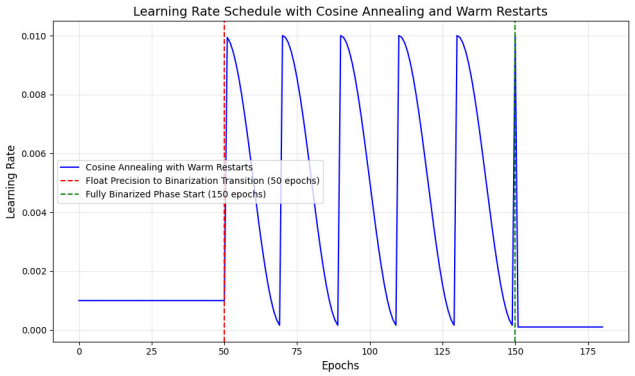

FIGURE 5. This figure illustrates the learning rate schedule using cosine annealing with warm restarts. The red dashed line marks the transition from float precision to binary weights at epoch 50, and the green dashed line indicates the start of the fully binarized phase at epoch 150.

TABLE 2. Accuracy metrics for BiSegUNet across datasets.

Dataset	mIoU (%)↑	Dice Score (%)↑	Pixel Accuracy (%)↑
Cityscapes	75.1	89.2	92.5
ADE20K	41.5	67.8	76.3
PASCAL VOC	72.3	82.3	90.4

restarts, as shown in Fig. 5. Initially, the learning rate was reduced using cosine annealing, enabling faster convergence during early epochs. The transition from float precision to binary weights occurs at epoch 50, marked by the red dashed line, allowing the model to benefit from high precision early on and smoothly shift to a binary representation. At epoch 150, indicated by the green dashed line, the model enters the fully binarized phase, where all weights are binary, optimizing computational efficiency without significantly impacting performance.The use of warm restarts in the learning rate schedule helps the model avoid local minima, improving its ability to explore the solution space. The training process is effectively adapted to the varying precision phases, ensuring that the model benefits from high precision in the early stages and efficient binary operations in the later stages.

E. DISCUSSION

The primary objective of BiSegUNet is to address the computational challenges in semantic segmentation by delivering high efficiency and competitive accuracy, particularly for real-time applications in resource-constrained environments. The Capacity Block, a central component of the architecture, facilitates multi-scale feature refinement and context aggregation through a bottleneck design. This block combines binary convolutions, grouped convolutions, and dilated convolutions to process features efficiently. Additionally, the integration of attention mechanisms further enhances global and local feature alignment, improving segmentation precision. Together, these components enable BiSegUNet to achieve a favorable trade-off between computational efficiency and segmentation performance.Unlike many state-of-the-art (SOTA) models that rely on large computational budgets, BiSegUNet achieves comparable performance with significantly reduced resource requirements, positioning itself as a lightweight yet effective solution. The analysis presented in Table 2 shows significant variations in segmentation performance metrics across three datasets evaluated with BiSegUNet. Cityscapes demonstrates robust segmentation accuracy with a mean Intersection over Union (mIoU) of 75.1% and a Dice score of 89.2%, showcasing the model's proficiency in structured object classification. In contrast, ADE20K presents challenges with a lower mIoU of 41.5% and a Dice score of 67.8%, reflecting difficulties in segmenting complex environmental scenes. PASCAL VOC achieves strong performance metrics with an mIoU of 72.3% and a Dice score of 82.3%, highlighting the effectiveness of BiSegUNet in urban scene segmentation tasks. The fixed input size ensures consistency in evaluating the model's segmentation capabilities across datasets, underscoring its adaptability and reliability in diverse real-world applications. The results presented in Table 3 provide a detailed evaluation of BiSegUNet on the Cityscapes dataset, highlighting the trade-offs between accuracy and computational efficiency across different image sizes and precision methods. The binarized version of BiSegUNet achieves substantial efficiency gains compared to its FP32 counterpart. Specifically, at a resolution of 512×1024, the binarized model reduces FLOPs by approximately $\sim$ **19**× and memory usage by $\sim$**6.5**. For 1024×2048 resolution, FLOPs are reduced by $\sim$ **14**×, and memory usage is reduced by $\sim$ **4.9**×. Additionally, the binarized model achieves significant speed improvements, being approximately $\sim$ **7**× faster at 512×1024 and $\sim$ **4.7**× faster at 1024×2048 resolution. Despite these substantial gains in efficiency, the binarized model retains competitive accuracy, with only a minor decrease in mIoU: **4.1%** at 512×1024 and **6.1%** at 1024×2048 resolution. These results underscore the capability of the binarized BiSegUNet to achieve an optimal balance between computational efficiency and segmentation performance. The models in Table 4 provides a comprehensive evaluation of BiSegUNet and state-of-the-art (SOTA) methods on the **Cityscapes dataset**, categorized into **large**, **medium**, and **small-sized models** based on their computational complexity (FLOPs), memory usage, and number of parameters:

- **Large Models:** Parameters exceeding **50M** and FLOPs range of **300 G**. These models prioritize accuracy (e.g., *Lawin Transformer*: **84.4% mIoU**) but have limited applicability in real-time scenarios due to their computational demands.
- **Medium Models:** Parameters between **7M and 30M** and FLOPs between **3G and 300G**, achieving a balance between accuracy and efficiency (e.g., *ICNet*: **69.5% mIoU**).

TABLE 3. Performance and efficiency metrics for BiSegUNet on cityscapes with different image sizes and precision methods.

Dataset	Image Size	Precision	mIoU (%)↑	Dice Score (%)↑	Pixel Accuracy (%)↑	FLOPs (G)↓	Memory (MB)↓	Speed (FPS)↑	Parameters (M)↓
Cityscapes	512×1024	FP32	73.1	88.5	93.0	194.7	7.8	22.8	18.5
	512×1024	Binarized	69.0	81.2	88.0	10.4	1.2	158	4.3
	1024×2048	FP32	81.2	93.0	95.5	390.3	11.5	11.8	23.4
	1024×2048	Binarized	75.1	85.2	91.8	27.5	2.3	56	6.7

TABLE 4. Comparison of BiSegUNet with SOTA methods for semantic segmentation on the cityscapes dataset.

Method	Year	Resolution	Parameters (M)↓	FLOPs (G)↓	Speed (FPS)↑	mIoU (%)↑	Backbone
Large Size Models							
PSPNet [3]	2017	713×713	250.8	412.2	0.78	81.2	ResNet-101
DeepLab-v3+ [2]	2018	-	**15.4**	555.4	**8.4**	75.2	Xception
DenseASPP [45]	2018	512×512	35.7	632.9	-	80.6	DenseNet
SETR-PUP [46]	2021	768×768	318.3	-	0.50	82.2	ViT-Large
SegFormer-B5 [47]	2021	1024×2048	84.7	1447.6	2.5	84.0	MiT-B5
Lawin Transformer [48]	2022	1024×1024	-	1797	-	**84.4**	Swin-L Transformer
BiSegUNet (Ours FP32)	-	1024x2048	23.4	**390.3**	11.8	81.2	No
Medium Size Models							
SegNet [49]	2017	640x360	29.50	286.0	17	57.0	VGG-16
ICNet [50]	2018	1024×2048	26.5	28.3	30	69.5	PSPNet-50
DFANet [51]	2019	1024×1024	7.8	**3.4**	100	71.3	Xception
STDC1-50 [52]	2021	512×1024	8.4	-	87	71.9	No
HSBNet [53]	2021	512×1024	12.1	-	**124**	73.1	ResNet-34
LBN-AA [54]	2021	448×896	**6.2**	49.5	51	73.6	No
FPANet [55]	2022	512×1024	14.1	-	-	72.0	No
SegFormer-B0 [47]†	2021	512×1024	13.2	22.8	86	70.2	MiT-B0
SegFormer-B0 [47]†	2021	1024x2048	13.2	60.2	34	**75.9**	MiT-B0
SSFormer-T [56]†	2022	512×1024	9.1	18.6	98	69.8	No
SSFormer-T [56]†	2022	1024x2048	9.1	49.8	39	73.4	No
BiSegUNet (Ours FP32)	-	512x1024	18.5	194.7	22.8	73.1	No
Small Size Models							
ENet [18]	2016	512x1024	**0.36**	3.8	135	58.3	No
ESPNet [20]	2018	512×1024	**0.36**	-	113	60.3	ESPNet
ESPNet-v2 [21]†	2019	512×1024	1.2	**3.1**	112	64.3	ESPNet-v2
ESPNet-v2 [21]†	2019	1024x2048	1.2	8.1	77	66.8	ESPNet-v2
LEDNet [57]	2019	512x1024	0.94	-	40	70.6	No
BiSeNet-v2 [12]†	2021	512×1024	3.4	9.5	**160**	69.0	Xception
BiSeNet-v2 [12]†	2021	1024x2048	3.4	25.5	62	73.1	Xception
EdgeNet [13]	2021	512×1024	-	-	31	71.0	No
FBSNet [58]	2022	512x1024	0.62	9.7	90	70.9	No
MSCFNet [14]	2022	512×1024	1.15	17.1	50	71.9	Lightweight CNN
AFFormer-T [59]†	2023	512×1024	3.5	17.7	104	69.3	No
AFFormer-T [59]†	2023	1024x2048	3.5	46.5	41	**76.3**	No
LETNet [15]	2024	512×1024	0.95	13.6	150	72.8	No
BiSegUNet (Ours Binarized)†	-	512x1024	4.3	10.4	158	69.0	No
BiSegUNet (Ours Binarized)†	-	1024x2048	6.7	27.5	56	75.1	No

† Evaluate on same inference setting (single value, No − TTA).

- **Small Models:** Parameters less than **7M** and FLOPs below **30G**, designed for real-time applications. *BiSegUNet* falls into this category, demonstrating superior mIoU and speed compared to its peers.

On the Cityscapes dataset (Table 4), the proposed BiSegUNet achieves an impressive mean Intersection over Union (mIoU) of **75.1%** in its binarized configuration at a resolution of **1024**×2048, outperforming other lightweight models such as *LETNet* (**72.8**), *ESPNet-v2* (**66.2%**) and *BiseNet-v2* (**72.6%**). This is achieved while maintaining a significantly lower computational complexity, with a FLOP count of only **27.5 G**. Furthermore, the architecture demonstrates exceptional inference speed, achieving **158 FPS** at a resolution of **512**×1024, which is substantially higher than many comparable lightweight models. Under the single-scale, no-TTA protocol on Cityscapes, the binarized BiSegUNet shows a consistently better accuracy–efficiency trade-off when compared to both compact CNNs and lightweight Transformers. At a resolution of 512 × 1024, it outperforms ESPNet-v2

TABLE 5. Per-class IoU (%) results on the cityscapes test set. "Avg" represents the average results of all these categories.

Methods	Avg	Bic	Bus	Bui	Car	Fen	Mot	Pol	Per	Rid	Roa	Sid	Sky	Tru	Tra	TLi	Ter	TSi	Veg	Wal
SegNet [49]	57.0	51.9	43.1	84.0	89.3	29.0	35.8	35.1	62.8	42.8	96.4	73.2	91.8	38.1	44.1	39.8	63.8	45.1	87.0	28.4
ENet [18]	58.3	55.4	50.5	75.0	90.6	33.2	38.8	43.4	65.5	38.4	96.3	74.2	90.6	36.9	48.1	34.1	61.4	44.0	88.6	32.2
ESPNet [20]	63.0	57.2	52.5	76.2	92.3	36.1	41.8	45.0	67.0	40.9	97.0	77.5	92.6	38.1	50.1	35.6	63.2	46.3	90.8	35.0
ESPNet-v2 [21]	66.2	59.9	65.9	88.8	91.8	42.1	44.2	49.3	72.9	53.1	97.3	78.6	93.3	53.0	53.2	52.6	66.8	60.0	90.5	43.5
ICNet [50]	69.5	70.5	72.7	89.7	92.6	48.9	53.6	61.5	74.6	56.1	97.1	79.2	93.5	51.3	51.3	60.4	68.3	63.4	91.5	43.2
LEDNet [57]	70.6	**71.6**	64.0	91.6	90.9	49.9	44.4	62.8	76.2	53.7	98.1	79.5	94.9	**64.4**	52.7	61.3	61.2	72.8	92.6	47.7
FBSNet [58]	70.9	70.1	56.0	91.5	93.9	53.5	56.2	62.5	82.5	**63.8**	98.0	83.2	94.4	50.5	37.6	**67.6**	70.5	71.5	**92.7**	50.9
EdgeNet [13]	71.0	67.7	60.9	91.6	94.3	50.6	55.3	62.6	80.4	61.1	98.1	83.1	94.9	50.0	52.5	67.2	69.7	71.4	92.4	45.4
MSCFNet [14]	71.9	70.2	66.1	91.0	94.1	52.5	57.6	61.2	**82.7**	62.7	97.7	82.8	94.3	50.9	51.9	67.1	70.2	71.4	92.3	49.0
LETNet [15]	72.8	69.3	72.4	91.6	**94.4**	53.7	56.1	61.0	82.3	61.7	**98.2**	83.6	94.9	55.0	57.0	66.7	**70.5**	70.5	92.5	50.9
BiSegUNet (Ours)	75.1	70.0	**74.6**	**92.5**	91.6	**55.7**	**62.0**	**83.6**	64.7	62.9	96.8	**86.2**	**95.1**	62.4	**57.3**	57.8	68.9	**74.3**	**92.7**	**51.6**

TABLE 6. Comparison of BiSegUNet with SOTA methods for semantic segmentation on the ADE20K dataset.

Model	Year	Resolution	Parameters (M)↓	FLOPs (G)↓	mIoU (%)↑
SegFormer-B4 [47]	2021	512×512	84.7	183.3	**51.8**
SeMask [60]	2021	512x512	35	40	43.36
SSFormer [56]	2022	512x512	87.5	91.1	47.7
AFFormer [59]	2023	512X512	**3**	4.6	41.8
CGRSeg [61]	2024	1024x512	35.7	14.9	48.3
BiSegUNet (Ours)	-	512x512	4.3	**2.6**	41.5

and BiSeNet-v2 in mean Intersection over Union (mIoU), while maintaining superior throughput and significantly lower FLOPs compared to SSFormer-T, SegFormer-B0, and AFFormer-T, which achieve only marginal mIoU improvements at considerably higher computational expenses. At a resolution of 1024 × 2048, BiSegUNet's mean Intersection over Union (mIoU) approaches that of lightweight Transformers. However, it is significantly faster and more computationally efficient, achieving practical real-time performance on the target device, whereas ESPNet-v2 and BiSeNet-v2 show lower accuracy. The results demonstrate that BiSegUNet significantly outperforms previous lightweight CNNs and reduces the accuracy disparity with lightweight Transformers, all while avoiding their associated FLOP and latency drawbacks.

This improvement is attributed to the proposed **Capacity Block** and binary design, which facilitates multi-scale, attention-guided refinement at minimal cost, positioning it as an optimal solution for edge or latency-sensitive applications.

The binarized configuration of BiSegUNet exhibits a minor reduction in accuracy compared to its FP32 counterpart, with a **4.1%** decrease in mIoU observed at a resolution of **512**×**1024**. While such a trade-off is expected in binarized architectures, it may present challenges for applications requiring highly precise segmentation. On the ADE20K dataset (Table 6), BiSegUNet achieves a competitive mIoU of **41.5%**. Although BiSegUNet generalizes well to structured outdoor datasets, its performance on ADE20K remains limited compared to state-of-the-art models. On ADE20K with 150 classes, BiSegUNet attains 41.5% mIoU, which is substantially below heavier architectures such as SegFormer-B4. This gap is largely attributable to the dataset's high scene complexity, dense object co-occurrence, and abundance of small or thin structures, where our lightweight binary network, primarily designed for efficient urban-scene segmentation, struggles to capture fine details and long-range context. In particular, we frequently observe misclassification or omission of small indoor objects and ambiguous boundaries between visually similar classes (e.g., wall versus door), indicating insufficient representational capacity under binary quantization. Furthermore, unlike several leading methods on ADE20K, our model does not benefit from large-scale ImageNet pre-training or additional data. Overall, these results highlight an expected trade-off in which BiSegUNet offers strong efficiency and competitive accuracy on street-scene benchmarks but incurs a noticeable performance penalty on highly heterogeneous, fine-grained datasets such as ADE20K. BiSegUNet demonstrates strong generalization across datasets with varying complexities. While it performs robustly on structured datasets like Cityscapes, the moderate mIoU on ADE20K highlights potential areas for improvement in handling extreme intra-class variability. The architecture remains robust to varying input resolutions, with consistent performance observed from **512**×1024 to **1024**×2048. Table 5 provides a breakdown of per-class IoU for BiSegUNet and SOTA methods, providing insights into the model's performance across diverse semantic categories. The average IoU (mIoU) represents the overall segmentation performance. BiSegUNet achieves the highest overall mIoU (**75.1%**) among small-sized models. It performs exceptionally well in classes requiring finer boundary delineation, such as *sidewalk* (**86.2%**) and *building* (**92.5%**).

TABLE 7. Cityscapes accuracy and efficiency comparison of our Capacity block vs. standard multi-scale + attention module.

Variant	mIoU (%) ↑	GFLOPs ↓
Baseline (no Capacity Block)	64.4	21.4
Multi-scale + **SE**	68.9	25.7
Multi-scale + **CBAM**	71.2	28.8
Capacity Block (proposed)	**75.1**	**27.5**

TABLE 8. Per-branch ablation of the capacity block on the cityscapes dataset. Each result represents the mean mIoU (%) ± standard deviation over three runs (validation set, single-scale 1024 × 2048). "No binary", "No dilated", and "No grouped" indicate that the corresponding branch is removed from every capacity block, while retaining the remaining branches.

Variant	mIoU (%) ↑	GFLOPs ↓
Baseline (no Capacity Block)	64.4 ± 0.3	21.4
No Binary conv branch	73.1 ± 0.3	22.4
No Dilated conv branch	72.5 ± 0.4	24.3
No Grouped conv branch	70.3 ± 0.2	26.6
Capacity Block (proposed)	**75.1 ± 0.1**	**27.5**

TABLE 9. Impact of the number of capacity blocks on cityscapes segmentation performance. "# Blocks used" denotes how many decoder stages (including the final output stage) employ a capacity block, starting from the deepest/highest-level features. Results are mean mIoU (%) ± std over multiple runs.

#Blocks (placement)	mIoU (%) ↑	GFLOPs ↓
0 (none)	64.4 ± 0.3	21.4
1 (bottleneck only)	68.4 ± 0.1	23.4
2 (top-2 decoder stages)	71.3 ± 0.2	25.3
3 (top-3 decoder stages)	73.4 ± 0.2	26.9
4 (all decoder stages)	**75.1 ± 0.1**	**27.5**

However, minor challenges are observed in handling dynamic objects like *truck* (**62.4%**) and *bicycle* (**70.0%**), where complex details and high variability pose segmentation difficulties. This per-class analysis highlights BiSegUNet's ability to generalize across most categories while identifying potential areas for improvement. Figure 6 and Figure 7 provide a qualitative comparison of segmentation results for BiSegUNet and several state-of-the-art (SOTA) models on the *Cityscapes* and *PASCAL VOC* datasets. *BiSegUNet* consistently produces sharper and more accurate segmentation boundaries compared to lightweight models such as *LEDNet* and *BiseNet-v2*. On the other hand, compared to larger models like *DeepLab-v3+*, *BiSegUNet* achieves visually comparable results, despite operating at significantly lower computational costs. This demonstrates the architecture's efficiency in balancing accuracy and resource constraints, as well as its potential for scalable and practical deployment in real-world applications such as autonomous driving, urban monitoring, and edge computing.

VI. ABLATION STUDY

We systematically evaluate the contributions of the **Capacity Block** and **attention mechanisms** to BiSegUNet's performance, starting from a baseline configuration and progressively incorporating these components. The results, as visualized in Figure 8, show the significance of these architectural choices in enhancing segmentation quality.We begin with a base configuration of BiSegUNet, without the Capacity Block, and observe the progression as the Capacity Block and attention mechanisms are sequentially introduced. In the absence of the Capacity Block (Figure 8a, the feature maps exhibit poor refinement with indistinct boundaries and weak spatial context aggregation. This results in limited segmentation accuracy and a lack of contextual understanding, as the model struggles to capture multi-scale features effectively. The output is coarse, underlining the necessity of multi-scale processing. Introducing the Capacity Block into the architecture leads to a significant improvement in feature refinement. The feature maps (Figure 8b display clearer boundaries and more distinct representations of objects. The multi-scale aggregation enabled by the Capacity Block enhances the model's ability to capture both global and local details, substantially improving spatial and contextual understanding. When attention mechanisms are added, the feature maps (Figure 8c reach their highest quality. By prioritizing critical spatial and channel features, attention mechanisms further amplify the precision and contextual richness of the feature representations. The analysis evaluates the baseline model, without the Capacity Block, which achieves the lowest mIoU, highlighting its limitations in capturing multi-scale features. Adding the Capacity Block leads to a significant **2.7%** improvement in mIoU, demonstrating its crucial role in feature refinement and multi-scale context aggregation. Incorporating attention mechanisms further enhances mIoU by **3.6%**, emphasizing their effectiveness in prioritizing critical features and complementing the Capacity Block to improve global context understanding. This progression underscores the importance of these components in achieving robust segmentation performance. The results reveal the importance of each component of BiSegUNet in achieving optimal segmentation performance. The Capacity Block serves as the foundation for multi-scale refinement, while attention mechanisms provide an additional layer of precision and contextual understanding. The results validate the architectural design, with measurable improvements at each step highlighting the association between the Capacity Block and attention mechanisms. This structured progression of feature and performance enhancements underscores BiSegUNet's capability to balance efficiency and accuracy effectively. To validate the effectiveness of the proposed Capacity Block, we perform an ablation study by replacing it with a standard multi-scale and attention module. One variant employs a multi-scale convolutional module, incorporating parallel dilated convolutions similar to ASPP, followed by a Squeeze-and-Excitation (SE) block. In another variant, the same module is followed by a CBAM attention block. All the other components, such as the decoder and bottleneck ASPP/ANP, remain unchanged.

Table 7 illustrates that both alternatives yield a lower mIoU on the Cityscapes dataset in comparison to our Capacity Block. The SE-based multi-scale module achieves approximately 68.9% mIoU, while CBAM attains around

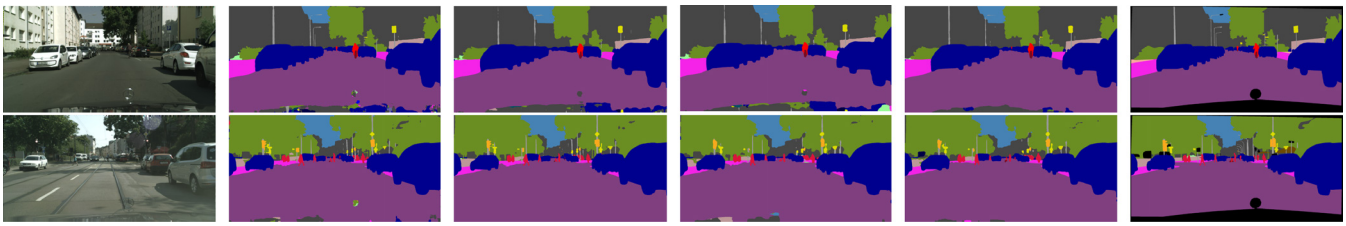

FIGURE 6. Visual comparative results on the PASCAL-VOC dataset. From left to right: Input image, segmentation results from our BiSegUNet, DeepLab-v3+ [2], LEDNet [56], BiseNet-v2 [12], and ground truth.

FIGURE 7. Visual comparative results on the PASCAL-VOC dataset. From left to right: Input image, segmentation results from our BiSegUNet, DeepLab-v3+ [2], LEDNet [56], BiseNet-v2 [12], and ground truth.

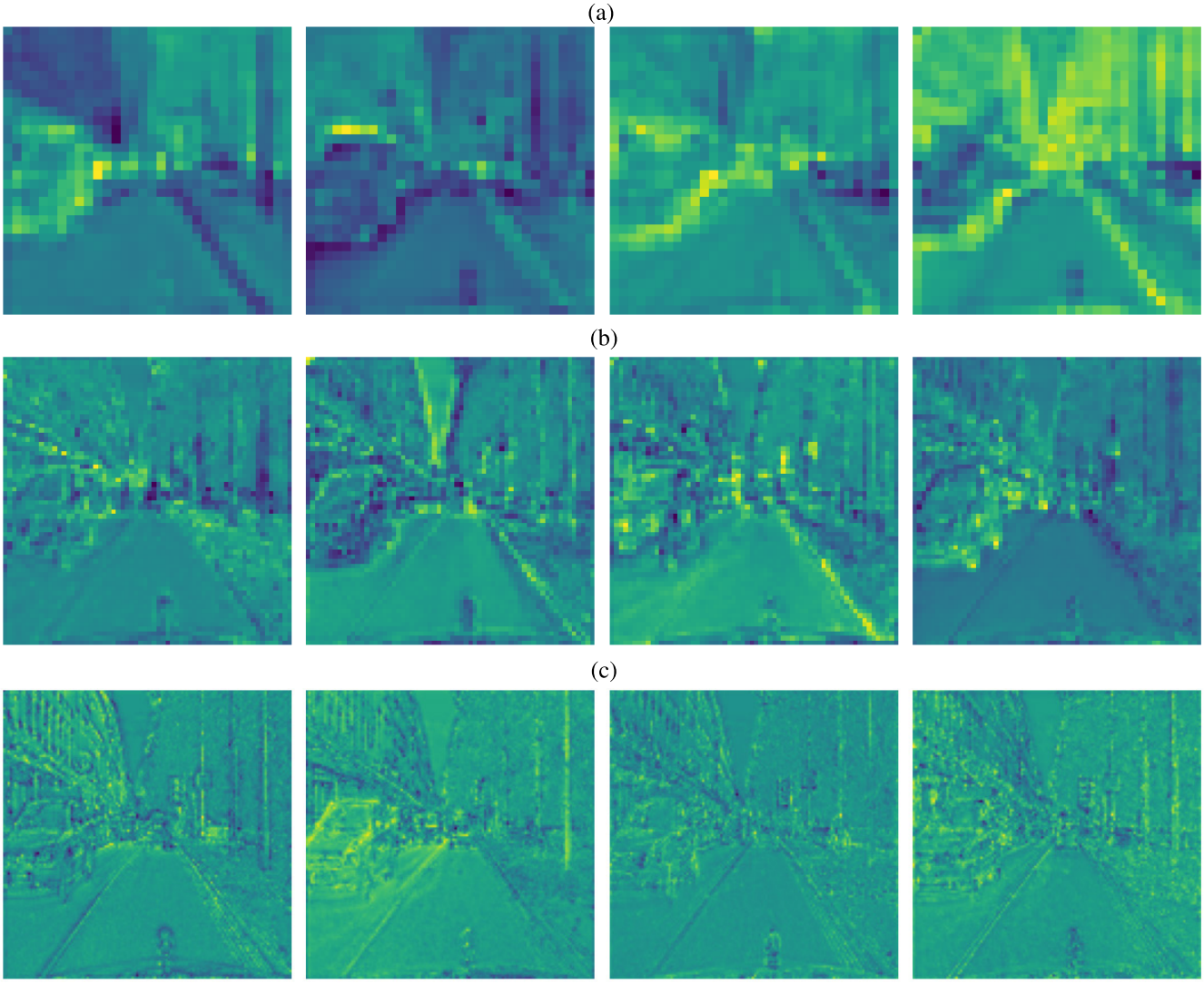

FIGURE 8. Comparison of feature maps at different stages of BiSegUNet: (a) without Capacity Block, (b) with Capacity Block, and (c) with Capacity Block and Attention. The evolution of feature representation highlights the contribution of these components.

71.2%, compared to 75.1% with the Capacity Block. This indicates that although standard attention modules enhance multi-scale feature fusion, our Capacity Block provides an additional performance improvement due to its more

TABLE 10. Training hyperparameters and experimental setup for BiSegUNet.

Hyperparameter	Value / Description
Training epochs	**180** (full training schedule)
Batch size (total)	**16** (e.g., 16 images per iteration, distributed across 3 GPUs)
Optimizer	**Adam** (β_1=0.9, β_2=0.999)
Initial learning rate	$\mathbf{1\times10^{-3}}$ (0.001)
Learning rate schedule	**Cosine annealing** with warm restarts at epoch 50 and 150 (reset when switching precision phases)
Weight decay	None (no L2 regularization applied to weights)
Loss function	Hybrid Cross-Entropy + Dice loss
Binary weight regularization	Yes (L1 penalty with coefficient $\lambda_3 \approx 1 \times 10^{-4}$ to encourage weights ±1)
Binarization schedule	Float training for 50 epochs, then incremental binarization of conv layers from epoch 50–150, fully binary by epoch 150–180
Activation function	ReLU throughout (including binary layers)
Normalization	BatchNorm after each convolution (momentum 0.1)
Pre-training	None (training from scratch)

cohesive design. The Capacity Block's dense multi-branch convolutional processing, coupled with integrated attention mechanisms, facilitates superior feature refinement compared to the addition of an SE or CBAM module to a standard multi-scale block. A detailed ablation study is conducted to quantify the contribution of each branch within the Capacity Block. The Capacity Block consists of three parallel branches, binary convolution, dilated convolution, and grouped convolution. The outputs of these branches are concatenated and re-weighted using channel attention. Each branch is systematically removed, and the model is retrained, with the mean mIoU and standard deviation reported over 3 runs on the Cityscapes dataset. Table 8 presents a summary of the results. The removal of any component from the Capacity Block results in performance degradation, indicating that each branch plays a vital role in the overall effectiveness. Removing the dilated convolution branch significantly affects performance, resulting in a mean Intersection over Union (mIoU) decrease of approximately 2.6%. This underscores the importance of multi-scale context. The binary convolution branch and the grouped convolution branch each offer unique advantages, with their removal resulting in approximately 2% and 4.8% performance declines, respectively. Even with the removal of any single branch, the model continues to outperform the baseline lacking a Capacity Block, emphasizing that all three types of convolutions collectively enhance feature representation. Following that, we examine the distribution and quantity of Capacity Blocks within the network. In the standard BiSegUNet architecture, a Capacity Block is incorporated at each decoder stage, following the addition of each encoder–decoder skip connection. We investigate the impact of reducing the number of Capacity Blocks on performance scalability. Table 9 presents the mean Intersection over Union (mIoU) results corresponding to varying quantities of Capacity Blocks. For instance, "1 block" indicates the utilization of a single Capacity Block at the decoder bottleneck, while "2 blocks" signifies the application of Capacity Blocks at the two deepest levels of the decoder, and so forth. A clear trend shows that increasing the number of Capacity Blocks consistently enhances accuracy, even with diminishing returns. The initial Capacity Block (located at the encoder bottleneck) produces the most significant improvement, increasing from 64.4% to approximately 68.4% mIoU. The implementation of Capacity Blocks across two decoder levels increases the mean Intersection over Union (mIoU) to approximately 71.3%. Increasing the number of blocks to 3 or 4, by inserting them at shallower decoder stages, results in smaller improvements, reaching approximately 75.1% with 4 blocks. This saturating trend indicates that the majority of the advantage arises from the refinement of high-level features (global context) in the deeper layers, whereas additional blocks at shallower layers primarily adjust local details. The final design, incorporating Capacity Blocks at all skip connections and the decoder output, was selected to optimize accuracy. However, this study suggests a potential trade-off: utilizing fewer Capacity Blocks (e.g., only at the top 1–2 levels) may reduce computational demands while still achieving a significant portion of the accuracy enhancement.

VII. CONCLUSION

In this paper, we proposed **BiSegUNet**, a lightweight real-time semantic segmentation network that combines an innovative **Capacity Block** and a **bottleneck design** with binary convolutions to achieve efficient multi-scale feature refinement. The inclusion of **attention mechanisms** enhances global context understanding, enabling the model to balance accuracy and computational efficiency. Extensive evaluations on the *Cityscapes* and *ADE20K* datasets demonstrate BiSegUNet's ability to achieve competitive segmentation performance. On *Cityscapes*, it achieved an **mIoU of 75.1%** at **1024×2048 resolution** with a high inference speed of **158 FPS** at **512×1024 resolution**, outperforming many lightweight models. These results highlight BiSegUNet's scalability, efficiency, and potential for real-world applications such as autonomous driving and edge computing, making it an effective solution for resource-constrained environments.

The Capacity Block is the central architectural innovation of BiSegUNet, providing an efficient mechanism

for multi-scale feature refinement under binary constraints. Through its combination of parallel convolutions and channel attention, BiSegUNet attains strong segmentation accuracy on Cityscapes while substantially reducing model size and computational cost. Our experiments show that this module is more effective than replacing it with conventional ASPP with SE or CBAM designs at comparable computational budgets, indicating that carefully structured multi-branch attention can partly offset the representational loss introduced by binary weights. The Capacity Block therefore offers a practical template for enriching lightweight networks and suggests a general direction for future research on attention-augmented, multi-scale architectures for efficient semantic segmentation.

A. LIMITATIONS AND FUTURE WORK

BiSegUNet, while effective, faces limitations such as a slight accuracy reduction in its binarized configuration compared to full-precision networks and challenges with dynamic, complex object classes in datasets like *ADE20K*. These limitations highlight the need for improved feature representation and generalization. Future work will aim to enhance robustness in complex scenarios by integrating advanced attention mechanisms and hybrid precision techniques to balance accuracy and efficiency. Expanding BiSegUNet to domain-specific tasks like medical imaging and aerial surveillance will showcase its adaptability, while deployment in real-world scenarios and incorporating unsupervised learning will enhance generalization.

APPENDIX A
TRAINING HYPERPARAMETERS

For clarity and reproducibility, Table 10 summarizes the key training hyperparameters and settings used in our experiments.

REFERENCES

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

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[3] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6230–6239.

[4] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2704–2713.

[5] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," in *Low-Power Computer Vision*, 2022, pp. 291–326.

[6] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[7] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

[8] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2815–2823.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[10] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[11] C. Yu, J. Wang, C. Peng, C. Gao, and G. Yu, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 325–341.

[12] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "BiSeNet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, no. 11, pp. 3051–3068, Nov. 2021.

[13] H.-Y. Han, Y.-C. Chen, P.-Y. Hsiao, and L.-C. Fu, "Using channel-wise attention for deep CNN based real-time semantic segmentation with class-aware edge information," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 1041–1051, Feb. 2021.

[14] G. Gao, G. Xu, Y. Yu, J. Xie, J. Yang, and D. Yue, "MSCFNet: A lightweight network with multi-scale context fusion for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25489–25499, Dec. 2022.

[15] G. Xu, J. Li, G. Gao, H. Lu, J. Yang, and D. Yue, "Lightweight real-time semantic segmentation network with efficient transformer and CNN," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 15897–15906, Dec. 2023.

[16] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[17] M. Gamal, M. Siam, and M. Abdel-Razek, "ShuffleSeg: Real-time semantic segmentation network," 2018, *arXiv:1803.03816*.

[18] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*.

[19] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized ConvNet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Jan. 2018.

[20] S. Mehta, M. Rastegari, A. Caspi, L. G. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 561–580.

[21] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNetv2: A light-weight, power efficient, and general purpose convolutional neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9190–9200.

[22] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[23] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[24] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5122–5130.

[25] J. Choi, Z. Wang, S. Venkataramani, P. I-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: Parameterized clipping activation for quantized neural networks," 2018, *arXiv:1805.06085*.

[26] S. Jung, C. Son, S. Lee, J. Son, J.-J. Han, Y. Kwak, S. J. Hwang, and C. Choi, "Learning to quantize deep networks by optimizing quantization intervals with task loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4345–4354.

[27] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, "HAQ: hardware-aware automated quantization with mixed precision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8612–8620.

[28] M. Courbariaux, Y. Bengio, and J. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1–18.

[29] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 4107–4115.

[30] M. Rastegari, V. Ordóñez, J. Redmon, and A. Farhadi, "XNOR-Net: Imagenet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 525–542.

[31] Z. Liu, B. Wu, W. Luo, X. Yang, W. Liu, and K. Cheng, "Bi-real Net: Enhancing the performance of 1-bit CNNs with improved representational capability and advanced training algorithm," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 722–737.

[32] J. Bethge, C. Bartz, H. Yang, Y. Chen, and C. Meinel, "MeliusNet: Can binary neural networks achieve MobileNet-level accuracy?" 2020, *arXiv:2001.05936*.

[33] B. Zhuang, C. Shen, M. Tan, L. Liu, and I. Reid, "Structured binary neural networks for accurate image classification and semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 413–422.

[34] A. Frickenstein, M.-R. Vemparala, J. Mayr, N.-S. Nagaraja, C. Unger, F. Tombari, and W. Stechele, "Binary DAD-Net: Binarized driveable area detection network for autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 2295–2301.

[35] X. Zhou, R. Ding, Y. Wang, W. Wei, and H. Liu, "Cellular binary neural network for accurate image classification and semantic segmentation," *IEEE Trans. Multimedia*, vol. 25, pp. 8064–8075, 2023.

[36] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei, "Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 82–92.

[37] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, "FBNet: Hardware-aware efficient ConvNet design via differentiable neural architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10726–10734.

[38] R. Azad, M. Heidary, K. Yilmaz, M. Hüttemann, S. Karimijafarbigloo, Y. Wu, A. Schmeink, and D. Merhof, "Loss functions in the era of semantic segmentation: A survey and outlook," 2023, *arXiv:2312.05391*.

[39] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML-10)*, 2010, pp. 807–814.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[41] P. Yin, J. Lyu, S. Zhang, S. Osher, Y. Qi, and J. Xin, "Understanding straight-through estimator in training activation quantized neural nets," 2019, *arXiv:1903.05662*.

[42] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," 2016, *arXiv:1608.03983*.

[43] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. Int. Conf. Comput. Vis.*, Mali, Nov. 2011, pp. 991–998.

[44] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3684–3692.

[45] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. S. Torr, and L. Zhang, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6881–6890.

[46] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Álvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 12077–12090.

[47] H. Yan, C. Zhang, and M. Wu, "Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention," 2022, *arXiv:2201.01615*.

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

[49] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 418–434.

[50] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9514–9523.

[51] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking BiSeNet for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9716–9725.

[52] G. Li, L. Li, and J. Zhang, "Hierarchical semantic broadcasting network for real-time semantic segmentation," *IEEE Signal Process. Lett.*, vol. 29, pp. 309–313, 2022.

[53] G. Dong, Y. Yan, C. Shen, and H. Wang, "Real-time high-performance semantic image segmentation of urban street scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3258–3274, Jun. 2021.

[54] Y. Wu, J. Jiang, Z. Huang, and Y. Tian, "FPANet: Feature pyramid aggregation network for real-time semantic segmentation," *Int. J. Speech Technol.*, vol. 52, no. 3, pp. 3319–3336, Feb. 2022.

[55] W. Shi, J. Xu, and P. Gao, "SSformer: A lightweight transformer for semantic segmentation," in *Proc. IEEE 24th Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2022, pp. 1–5.

[56] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, "Lednet: A lightweight encoder–decoder network for real-time semantic segmentation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 1860–1864.

[57] G. Gao, G. Xu, J. Li, Y. Yu, H. Lu, and J. Yang, "FBSNet: A fast bilateral symmetrical network for real-time semantic segmentation," *IEEE Trans. Multimedia*, vol. 25, pp. 3273–3283, 2023.

[58] B. Dong, P. Wang, and F. Wang, "Head-free lightweight semantic segmentation with linear transformer," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 1, pp. 516–524.

[59] J. Jain, A. Singh, N. Orlov, Z. Huang, J. Li, S. Walton, and H. Shi, "SeMask: Semantically masked transformers for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2023, pp. 752–761.

[60] Z. Ni, X. Chen, Y. Zhai, Y. Tang, and Y. Wang, "Context-guided spatial feature reconstruction for efficient semantic segmentation," 2024, *arXiv:2405.06228*.

HASSAN KHAN received the Bachelor of Science degree in electronics engineering and the Master of Science degree in electrical engineering from the Capital University of Science and Technology (CUST), Islamabad, in 2017 and 2020, respectively. He is currently pursuing the Ph.D. degree in artificial intelligence with the University of Galway. Previously, he was a Research Assistant with CUST, focusing on machine learning applications for wireless communication networks and energy-efficient systems. His research interests include AI, deep learning, optimization models, and sustainability. He received the Dean's Honor Awards for academic excellence from the CUST.

SUNBAL IFTIKHAR received the Bachelor of Science degree (Hons.) in electronics engineering from Mohammad Ali Jinnah University (MAJU), Islamabad, in 2017, and the Master of Science (by research) degree (Hons.) in communication networks and machine learning from the Capital University of Science and Technology (CUST), Islamabad, in 2020. She is currently pursuing the Ph.D. degree in sustainability of AI models. Since 2020, she has been a Research Assistant with the Department of Electrical Engineering, CUST, where she contributed to various projects. She is a Ph.D. candidate in artificial intelligence with Technological University Dublin (TU Dublin). She has hands-on experience with Sun Microsystems' SunSPOT testbed, contributing to her solid foundation in research and development. She has conducted significant research in optimizing 6G wireless communication channels using deep learning for path loss prediction and machine learning-driven optimization of indoor 5G infrastructure deployment. She focuses on sustainable and distributed training of large language models (LLMs).

STEVEN DAVY (Member, IEEE) received the B.A. Mod degree (Hons.) in computer science from Trinity College Dublin and the Ph.D. degree from Waterford Institute of Technology. He is currently an Experienced Researcher and Leader in the field of sustainable digital technologies. He is also the Director of the Centre for Sustainable Digital Technologies, TU Dublin. He has over two decades of experience in academia and industry. His research focuses on policy-based network management, edge computing, and autonomous systems. He has made significant contributions to the field, including developing the first formal model of the policy continuum and pioneering work on Edge-as-a-Service. He has led multiple high-profile European research projects and has a strong track record in securing research funding. Throughout his career, he has demonstrated a commitment to bridging the gap between academic research and industry applications. He has been instrumental in developing innovative services for businesses and has successfully commercialized research outcomes, including the creation of a spin-out company. As a leader, he has played a crucial role in developing research teams and mentoring early-career researchers. He actively contributes to the broader research community through his roles as a reviewer for funding agencies and academic journals.

JOHN G. BRESLIN (Senior Member, IEEE) received the B.E. and Ph.D. degrees. He is currently a Personal Professor of the electronic engineering with the College of Science and Engineering, University of Galway, where he is also the Director of the TechInnovate/AgInnovate Programmes. He has taught electronic engineering, computer science, innovation and entrepreneurship topics over the past 25 years. He is associated with two Taighde Éireann-Research Ireland Centres. He is also a Principal Investigator at Insight (Data Analytics) and a Funded Investigator at VistaMilk (AgTech), and a Principal Investigator on the EDIH Data2Sustain. He has jointly written over 300 peer-reviewed academic publications, including books on *The Social Semantic Web* and *Social Semantic Web Mining*. He co-created the SIOC framework, implemented in 100 of applications (by Yahoo, Boeing, and Vodafone) on at least 65 000 websites with 35 million data instances.