HARD: Hardware-Aware lightweight Real-time semantic segmentation model Deployable from Edge to GPU

YoungWook Kwon[1*[0009−0009−0157−3023]], WanSoo Kim[1*[0009−0009−4176−6057]], and HyunJin Kim[1†[0000−0001−5017−3995]]

Dept. of Electronics and Electrical Engineering, Dankook University, 152, Jukjeon-ro, Suji-gu, Yongin-si, 16890, Gyeonggi-do, Republic of Korea
kyw96@naver.com, dhkstn115@naver.com, hyunjin2.kim@gmail.com

Abstract. The two-branch model ensures high performance in semantic segmentation. However, the additional branch causes the fusion between high-resolution and low-resolution contexts to corrupt the surrounding context and increases the computational overhead. Existing methods with many parameters and high computational costs are not well-suited for the low-power devices used in applications like autonomous driving and robotics. This study proposes a robust semantic segmentation architecture with any kind of device, from GPUs to edge devices. We introduce five variants called HARD. HARD achieves fast inference speeds while maintaining good performance on any kind of device. Notably, the proposed Dual Atrous Pooling Module (DAP) can effectively fuse contexts of variable resolutions without decreasing inference speed. Besides, a lightweight decoder named Serialized Atrous Module (SA) is proposed to extract global context. The proposed models are evaluated on both GPU and embedded computing devices from NVIDIA and ARM Cortex-M CPU. In experiments on Cityscapes, CamVid, and COCO-Stuff datasets, the proposed variants of HARDs achieve 73.8, 76.3, and 41.0 mIoU, which outperform existing SOTA models.

Keywords: Real-time Semantic Segmentation · Embedded Device · MicroProcessor.

1 Introduction

Semantic segmentation, which assigns a semantic class mask to each pixel of an input image, is significant in applications such as autonomous driving, medical image processing, mobile applications, and many other fields. Recently, semantic segmentation methods have focused on integrating diverse spatial and positional contexts within images. Thus, the pyramid pooling module (PPM) [54] and self-attention [41] have been used to extract contextual information. However, these methods require significant computational overhead. Many recent

*These authors contributed equally to this work.
†Corresponding Author

real-time semantic segmentation architectures employ bilateral architectures to rapidly extract high-quality contextual information. BiSeNet [51], DDRNet [16], SeaFormer [42], AFFormer [8], and SCTNet [48] propose the bilateral architecture that separates spatially informative features from high-level contextual information early in the layer for parallel processing. However, due to their computational overhead, these designs can only perform real-time processing on GPUs. Therefore, the state-of-the-art (SOTA) existing real-time segmentation models are not suitable for IoT solutions. There has been much research on lightweight segmentation networks with a reduced number of parameters for minimizing memory usage. Although lightweight models such as ENet [32], ESPNet [29], FastSCNN [34], and MiniNet [1] can be deployed on NVIDIA embedded computing devices, they suffered from significant performance degradation. Therefore, there are still challenges in applying the existing studies to autonomous driving solutions, smart manufacturing, personalized medicine, etc. Therefore, we proposed HARD, an architecture applicable to both edge devices and GPUs. The proposed HARD employs the Dual Atrous Pooling (DAP) module to extract long-range context. Besides, the Serialized Atrous (SA) module is proposed to perform real-time semantic segmentation by serially extracting contextual information.

The main contributions of HARD are summarized as follows:

1. HARD is designed to be deployed on diverse device solutions, ranging from MCUs to GPUs.
2. DAP and SA modules are proposed to minimize computational costs, thereby enabling real-time segmentation.
3. This paper demonstrates the robustness of the proposed model using the Cityscapes, CamVid, and COCO-stuff-10k datasets and conducts experiments on GPU, embedded computing board, and MCU.

HARD has faster inference speed and higher accuracy than existing real-time semantic segmentation models. In the Cityscapes dataset, the HARD-GPU achieved 73.8 mIoU with an inference speed of 315 FPS, outperforming existing SOTA models. HARD-Edge on ARM Cortex-M presented 33 FPS, so it is the first acceptable semantic segmentation model on MCUs, as far as we know.

2 Related Works

2.1 Lightweight Semantic Segmentation

ENet [32] and FastSCNN [34] enhanced both performance and inference speed through a lightweight bottleneck structure that effectively extracts contextual information during downsampling with small parameters. They were resulted in parameter redundancy and the loss of significant local details, negatively impacting performance. ERFNet [35] and FDDWNet [22] designed models using 1-D convolutions to reduce the computational overhead. LiteHRNet [52] addressed computational bottlenecks from 1×1 convolutions with conditional

channel weighting. Although the 1×1 convolutions reduced the number of parameters, the excessive usage of 1×1 convolutions for performance enhancement increased unnecessary computational overhead. LEDNet [45], FBSNet [12], LETNet [40], EdgeNet[10], DFANet [20], and FPENet [23] enhanced performance by integrating Convolutional Neural Networks (CNNs) with attention mechanisms to fusion local and global contexts. SGCPNet [14] and ADSCNet [43] proposed spatial-detail guided context propagation and Dense Dilated Convolution Connections (DDCC) modules to prevent the loss of context information when resolution decreased. ESPNet [29] and ESPNet-v2 [30] reduced the number of parameters and computational costs by decomposing convolutions into 1×1 and dilation convolutions. CGNet [46] utilized Context Guided (CG) blocks to train local features and their surrounding context. The CGNet was designed to reduce parameters and memory usage. Similar to Inceptionv2 [39], CFPNet [28] focused on encoding context at multiple resolutions using a channel pyramid feature module for real-time segmentation. MiniNetv2 [2] designed fast models suitable for CPU environments through multi-dilation depth-wise convolutions. FSSNet [17] employed a ResNet [15] backbone with continuous factorized blocks to extract low-level features and adopted continuous dilated blocks to ensure a wide scale of receptive fields.

2.2 Real-Time Semantic Segmentation

FCN [25], U-Net [36], and RefineNet [21], Deeplabv3+ [5] have been conducted to achieve high performance in semantic segmentation tasks. Besides, many studies have focused on enhancing real-time processing for these tasks. For instance, ICNet [53] employed a multi-resolution branch to enhance network inference speed without accuracy degradation. PSPNet [54] utilized a Pyramid Pooling Module (PPM) to aggregate global context, while SFNet [18] proposed a flow alignment module to enhance feature representation. BiseNet [51] and BiseNetv2 [50] adopted a bilateral segmentation network structure to balance accuracy and inference speed. BiseNet improved the training of contextually varying feature representations by dividing it into spatial and context paths. BiseNetv2 adopted Detail and Semantic branches, successfully merging their features through an Aggregate layer. STDC [11] improved the inefficiencies of bilateral structure in BiseNet and proposed a single-stream approach for learning spatial information with a new Detail Aggregation module. Inspired by HRNet [44], DDRNet [16] proposed a network structure with two branches that are dependent on each other. They also proposed Deep Aggregation Pyramid Pooling Module (DAPPM) that combined feature aggregation and pyramid pooling. PP-Liteseg [33], demonstrating a robust encoder-decoder architecture in semantic segmentation, introduces a Flexible and Lightweight Decoder (FLD). PIDNet [47] designs a network with a three-branch structure, enabling diverse feature representations for enhanced performance. SCTNet [48] adopted a single-branch architecture for fast inference speeds and incorporated a Transformer block during the training phase to optimize performance trade-offs. While many studies

have considered real-time inference in semantic segmentation, their executions are focused only on GPUs.

3 Proposed Method

Real-time semantic segmentation is challenging on embedded computing devices and autonomous vehicles. Our research aims to design new models that could be applied to various types of hardware, including GPUs, embedded devices, and edge devices. In this section, we propose two modules and a training methodology that can be adjusted for the target device. Consequently, we introduce five variants called HARD. The proposed models are designed to minimize the computational overhead for their target devices. The overall explanation of HARD is presented in Figs. 4 and 5.

3.1 Dual-Atrous Pooling (DAP) Module

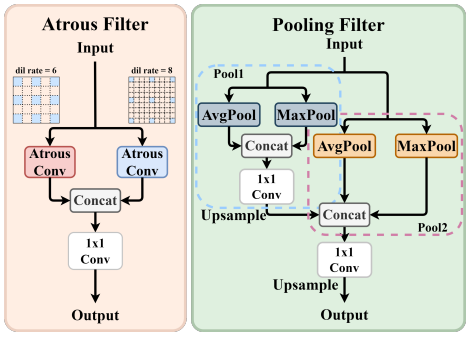

Fig. 1: Architecture of Atrous and Pooling filters.

Real-time segmentation models such as DDRNet [16] and STDC [11] processed encoders and decoders in parallel, maintaining variable high-resolution feature maps. These bilateral structures are designed to utilize the parallel processing of GPUs. However, they had difficulty in achieving real-time inference on other hardware platforms due to their high computational overhead. In contrast, SCTNet [48] adopted a unilateral structure for inference to achieve better performance than existing models with small latency. Nevertheless, SCTNet also employed a bilateral structure to extract multiple feature information during the training process. To address the above weakness, the proposed HARD-GPU adopts the proposed DAP module to enhance both performance and latency. Fig. 1 shows the structure of the Atrous and Pooling filters. DAP Module is

applied over the input x as follows:

$$\text{AtrousFilter}(x_i) = \text{Conv}_{1\times1}(Concat(\sum_k x_{i+6\times k} \cdot w_k, \sum_k x_{i+8\times k} \cdot w_k)). \quad (1)$$

$$\text{Pool1}(x_i) = \text{Conv}_{1\times1}(Concat(\frac{1}{n_1}\sum_{i=1}^{n_1} x_i, \lim_{p\to\infty}(\frac{1}{n_1}\sum_{i=1}^{n_1} x_i^p)^{\frac{1}{p}})). \quad (2)$$

$$\text{Pool2}(x_i) = Concat(\text{Pool1}(x_i), \frac{1}{n_2}\sum_{i=1}^{n_2} x_i, \lim_{p\to\infty}(\frac{1}{n_2}\sum_{i=1}^{n_2} x_i^p)^{\frac{1}{p}}). \quad (3)$$

$$\text{PoolingFilter}(x_i) = \text{Conv}_{1\times1}(\text{Pool2}). \quad (4)$$

Firstly, the Atrous filter extracts context information through two convolutions with different dilation sizes and then compresses the two pieces of information using 1×1 convolution. Channels are then expanded to match the final channels of the encoder through two consecutive dilated convolutions. Because pooling can extract global context more effectively than convolution, existing segmentation models have utilized the pyramid pooling modules such as DAPPM. Motivated by the above existing idea, the proposed Pooling filter performs average and maximum pooling in parallel, extracting the global context and then interpolating it back to the same shape as the input feature map. The extracted context performs two convolutions and average pooling to extract both local and global context. Then, it is expanded to match the final channels of the encoder. When the feature map is $\frac{1}{8}$, the DAP module is performed. To effectively extract context information, the operation continues through to the final layer. The context information extracted from the DAP module is concatenated with the encoder output. Fig. 4 shows a DAP module consisting of Atrous and Pooling filters. In our experiments, the proposed DAP shows a 4 mIoU performance improvement on the Cityscapes dataset. Therefore, we conclude that the DAP is a suitable context information extraction module for real-time semantic segmentation.

3.2 Serialized Atrous (SA) Module

The PPM and DAPPM decoder structures are designed for real-time segmentation that handles features at variable resolutions in parallel. However, in commercial embedded computing boards such as NVIDIA Jetson GPUs with limited CUDA cores, real-time segmentation having more than 30 FPS is impossible. To address the above limitations, we propose the Atrous Inverted Bottleneck (AIVB) module. Fig. 2 shows the proposed AIVB module, including two 1×1 convolutions and three depth-wise separable atrous convolutions arranged in a serialized structure. The AIVB module utilizes three different sizes of dilated

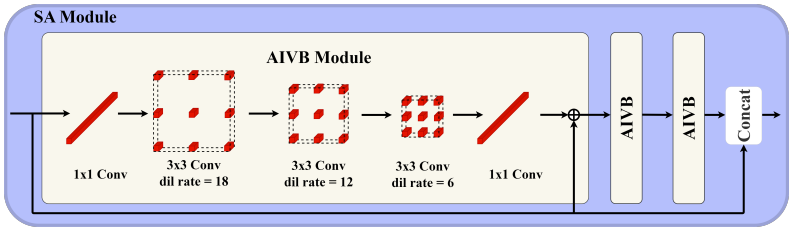

Fig. 2: Architecture of Serialized Atrous (SA) module and Atrous Inverted Bottleneck (AIVB) module.

(a) Image (b) GT (c) Concat (d) No concat

Fig. 3: Effects of concatenation after Atrous convolution on segmentation mask output.

kernels for all depth-wise convolutions. The atrous depth-wise convolution used by the AIVB module is represented as follows:

$$y_c[i] = \sum_k x_c[i + d \times k] \cdot w_c[k]. \tag{5}$$

In Eq. 5, d represents the dilation rate and c is the channel-dependent value, respectively. The Serialized Atrous (SA) module repeats the AIVB module three times with decreasing dilation rates to extract long-range context information using decreasing dilation rates. Recently, segmentation models have been extracting global context through an attention mechanism. However, this approach is inefficient for real-time segmentation due to its extensive computational overhead. Our proposed design enables global context extraction with significantly reduced computational requirements. Thus, the usage of SA module achieves adequate global context extraction by only using convolutions. In Figs. 4 and 5, after the SA module is performed, the feature map is concatenated with the previous feature map and a classifier is performed. We confirmed through experiments that pixel blur occurs in the semantic mask when there is no concatenation with the previous feature map in convolutions using a dilated kernel. In Fig. 3, the phenomenon negatively affects the precision of segmentation mask prediction.

The proposed SA module consists of depth-wise convolution, which minimizes the computational overhead and reduces peak-SRAM. It is optimized for embedded and edge devices. Therefore, it is the most hardware-friendly segmentation decoding module.

3.3 Model Architecture

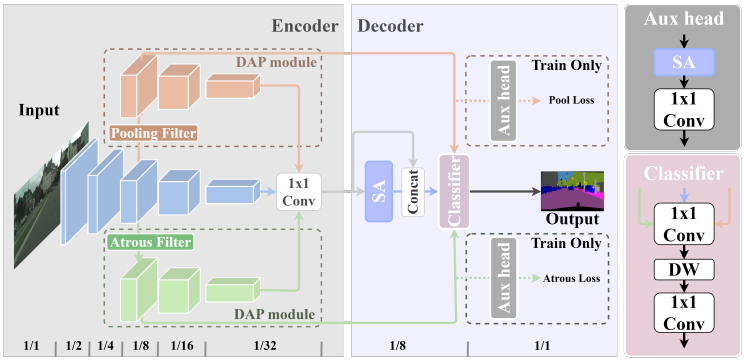

Fig. 4: The architecture of HARD-GPU. We apply two DAP modules in the encoder to extract global context information. In addition, the Pooling and Atrous filters are configured with an auxiliary head to provide auxiliary output for calculating extra semantic loss. Term DW means Depth-wise Convolution.

Model Architecture for High-Performance on GPU: For real-time semantic segmentation, it is important to sufficiently reduce the feature map size. Therefore, the HARD-GPU is a three-branch structure that uses encoders and decoders. The HARD-GPU encoder module combines standard convolutions with the inverted bottleneck proposed in MobileNetV2 [37]. To enhance performance, downsampling occurs in the standard convolution considering inter-channel correlations. The inverted bottleneck is useful for extracting spatial information with a small number of parameters. Fig. 4 includes five encoder modules, and the encoding process ends when downsampling reduces the input resolution to $\times\frac{1}{32}$. Two feature maps are extracted in the DAP module. The final output from the encoder performs concatenation. Then, contextual information is fused by 1×1 convolution to reduce it to $\times\frac{1}{3}$ of the channels. After the encoding process is completed, the output is interpolated to $\times\frac{1}{8}$ of the input resolution. The global context is extracted through SA module, then the feature map is used as the input to the classifier. The channels are reduced for the dataset class and interpolated to the input resolution for output.

Model Architecture for Embedded Computing Devices Compared with image classification and object detection, semantic segmentation models require more computational overhead due to the image interpolation in the decoder. To overcome the problem, more lightweight HARD-S, XS, and XXS architectures are designed. In Fig. 5, the lightweight HARD-S, XS, and XXS for embedded devices adopt an inverted bottleneck encoder. The inverted bottlenecks

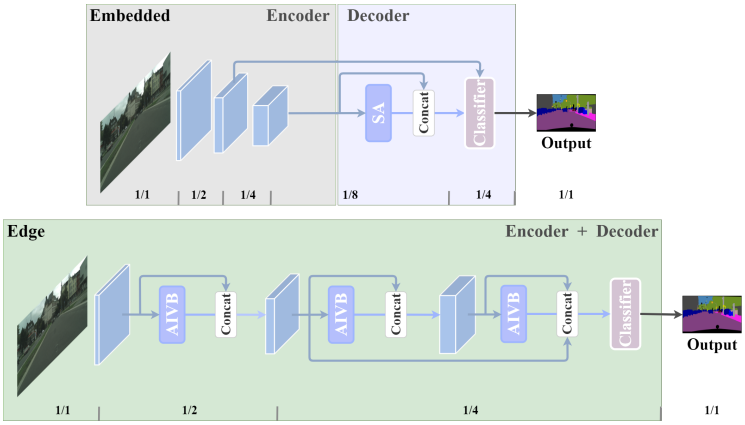

Fig. 5: On the top, the architecture of HARD-S, XS, and XXS is shown. On the bottom, the architecture of HARD-Edge is illustrated.

effectively reduce the number of parameters and computational overhead. While massive parallel processing is possible on GPUs, the lack of enough parallelism on low-power CPUs and embedded computing GPUs, such as the NVIDIA Jetson, makes fast inference difficult. Therefore, the HARD-S, XS, XXS models downsample the input to $\times\frac{1}{8}$ size using an inverted bottleneck for real-time inference, and the decoding process is performed in the SA module.

Model Architecture for Low-Cost Edge Devices We propose a novel tiny segmentation model called HARD-Edge for deployment on microcontroller units (MCUs). As far as we know, the proposed HARD-Edge is the first model for the semantic segmentation on MCUs. HARD-Edge is designed to ensure real-time frames per second (FPS) on MCUs with limited Flash, internal SRAM, and low-power CPUs such as ARM Cortex-M7. For example, ARM Cortex-M7 has a maximum of about 320 KB SRAM. These MCUs are limited in the resolution of the images that they can process. Consequently, HARD-Edge is optimized to operate on images with a size under 128×128 pixels. To minimize computational overhead while extracting adequate contextual information, HARD-Edge adopts a combined architecture with an encoder and decoder. This structure employs two downsampling followed by a single interpolation to produce a feature map with the original resolution. Furthermore, in order to reduce the model size and achieve fast inference, 8-bit quantization is applied to HARD-Edge, which shows negligible performance degradation. The quantization approach makes it suitable for low-power MCUs, enabling efficient operations.

3.4 Extra Semantic Loss

We utilize Atrous and Pooling filters as auxiliary classifiers to enhance the ability of HARD-GPU. Inception [38] used a secondary classifier to avoid the vanishing gradient problem with increasing model depth. In a similar way, PIDNet [47] attached classifiers to model branches to generate extra semantic loss, thereby optimizing the model. In many cases, auxiliary classifiers have been employed to improve segmentation performance in vision tasks.

$$Loss_{CE}(x, y) = -\frac{1}{N}\sum_{i} log\frac{e^{x_{y_i}}}{\sum_j e^{x_j}}. \tag{6}$$

$$Loss = Loss_{CE}(x, y) + 0.3 \times (Loss_{CE}(x_{Pool}, y) + Loss_{CE}(x_{Atrous}, y)). \tag{7}$$

Therefore, the proposed approach applied extra semantic loss during training to improve the performance of lightweight models. In Fig. 4, two filters of the DAP module configure their own decoder heads, respectively. In Eq. 7, we propose that the three outputs from the model are used to calculate cross-entropy loss denoted as $Loss_{CE}$ with the ground truth. These semantic losses are weighted and summed together. The detailed reason for setting the weight to 0.3 is in the ablation study.

4 Experimental Results and Analysis

4.1 Experimental Setup

We evaluated the performance of the proposed HARD on Cityscapes [6], CamVid [3], and COCO-Stuff-10k [4]. We pretrained the encoder of HARD on ImageNet-1k [7], then fine-tuned it on the semantic segmentation datasets. The training of proposed models was performed with a set of hyper-parameters as: we adopted AdamW [26] optimizer, having the weight decay set as 1e−4. Initial learning rate increased from 1e−5 to 1e−2 for the first 6550 iterations. Then, the learning rate was annealed to 1e−5 using a cosine scheduler [27]. The inference speed of all models was measured on a single NVIDIA RTX 4090. In order to show fair comparisons, all reported FPSs were estimated on the same input resolution. Furthermore, the proposed HARD and other counterparts were evaluated in terms of FPS on NVIDIA Jetson ORIN NX and ARM Cortex-M7. For the evaluation with ARM Cortex-M7, we adopted STM32F746NG from STMicroelectronics.

4.2 Comparison on Cityscapes

Cityscapes is a well-known urban scene segmentation dataset having 5,000 images collected from the perspective of a car. Table 1 shows the experimental results on the Cityscapes dataset. Our experiments were conducted for 30K iterations on two RTX 4090 GPUs. The proposed HARD-GPU had fewer number

Table 1: Comparisons with SOTA real-time methods on Cityscapes validation set.

Resolution	Model	Params	FLOPs	mIoU	FPS	Resolution	Model	Params	FLOPs	mIoU	FPS
512×1024	ENet [32]	0.38M	5.65G	58.3	112	512×1024	LiteHRNet [52]	1.09M	4.66G	70.6	40
512×1024	FSSNet [17]	0.29M	3.89G	58.8	240	512×1024	SeaFormer-S [42]	4.1M	1.8G	70.7	120
512×1024	ESPNet [29]	0.36M	4.1G	60.3	392	1024×2048	SGCPNet [14]	0.61M	4.5G	70.9	138
512×1024	MiniNet [1]	1.41M	6.71G	61.5	426	512×1024	FBSNet [12]	0.61M	22.06G	70.9	29
360×640	**HARD-XXS**	0.11M	0.93G	64.1	**533**	512×1024	EdgeNet [10]	-	-	71.0	31
360×640	CGNet [46]	0.50M	28.0G	64.8	177	512×1024	FDDWNet [22]	0.77M	12.38G	71.5	126
512×1024	NDNet [49]	0.50M	3.9G	65.1	251	512×1024	MiniNetV2 [2]	0.51M	9.26G	71.8	195
512×1024	ESPNetV2 [30]	1.25M	5.65G	66.2	167	512×1024	MSCFNet [13]	1.15M	17.1G	71.9	50
512×1024	EDANet [24]	0.69M	8.88G	67.3	240	512×1024	STDC1 [11]	12.5M	23.1G	72.2	82
512×1024	ADSCNet [43]	0.51M	12.68G	67.5	360	512×1024	SeaFormer-B [42]	8.7M	3.1G	72.2	98
512×1024	ERFNet [35]	2.06M	29.93G	68	222	512×1024	LETNet [40]	0.95M	13.59G	72.8	36
1024×2048	FastSCNN [34]	1.14M	6.72G	68.6	348	512×1024	SCTNet-S [48]	4.6M	7.1G	72.8	275
360×640	**HARD-XS**	0.39M	1.93G	69.6	532	360×640	**HARD-S**	0.76M	3.24G	72.8	427
1024×2048	CFPNet [28]	0.27M	21.07G	70.1	66	512×1024	PP-LiteSeg-T [33]	4.4M	4.3G	73.1	257
512×1024	FPENet [28]	0.4M	12.8G	70.1	180	512×1024	BiseNetv2 [50]	5.2M	35.5G	73.4	244
512×1024	SwiftNetRN [31]	12.1M	32.1G	70.2	186	512×1024	AFFormer-Base [8]	3.0M	8.6G	73.5	50
512×1024	LEDNet [45]	0.94M	11.31G	70.6	127	512×1024	**HARD-GPU**	3.9M	11.5G	**73.8**	315

of parameters, ranging from 10% to 60% than recent models such as SCTNet-S, PP-LiteSeg-T, BiseNetv2, Seaformer-B, and STDC1. Nevertheless, the proposed HARD-GPU achieved 73.8 mIoU, which is 1 mIoU higher than SCTNet. Additionally, it achieved 315 FPS, which is 40 FPS faster in terms of inference speed compared with the SCTNet. HARD-S and HARD-XS achieved 72.8 and 69.6 mIoU, respectively, which showed better performance than LETNet, MiniNetv2, SGCPNet, and LEDNet with comparable numbers of parameters. HARD-XXS had the fewest parameters among the models. However, it can achieve 5 mIoU higher performance than Enet and ESPNet. It also had the fastest inference speed among all segmentation models, reaching 533 FPS. In conclusion, the proposed HARD achieved excellent trade-offs between performance and inference when compared with other real-time counterparts.

4.3 Comparison on CamVid

Table 2: Comparisons with SOTA real-time methods on CamVid validation set.

Resolution	Model	Params	FLOPs	mIoU	FPS	Resolution	Model	Params	FLOPs	mIoU	FPS
360×480	ENet[32]	0.36M	1.86G	51.3	140	360×480	FDDWNet[22]	0.8M	4.08G	66.9	127
360×480	ESPNet[29]	0.36M	1.2G	55.6	195	720×960	LBN-AA[9]	6.2M	-	68.0	-
360×480	NDNet[49]	0.5M	0.56G	57.2	257	360×480	FBSNet[12]	0.62M	7.27G	68.9	30
360×480	FSSNet[17]	0.26M	1.28G	58.6	252	360×480	MSCFNet[13]	1.15M	-	69.3	-
360×480	CFPNet[28]	0.55M	1.7G	64.3	68	720×960	LETNet[40]	0.95M	-	70.5	37
720×960	DFANet[20]	7.8M	-	64.7	-	360×480	**HARD-XXS**	0.11M	0.69G	74.4	**532**
720×960	BiseNet[51]	13.0M	40.0G	65.6	249	360×480	**HARD-XS**	0.39M	1.43G	75.8	530
360×480	DABNet[19]	0.76M	3.37G	66.4	232	360×480	**HARD-S**	0.76M	2.41G	**76.3**	424

CamVid provides 701 images of driving scenes, having 960×720 image resolution, where our experiments adopted 11 classes. In Table 2, when a similar number of parameters is given, HARD-S achieved 76.3 mIoU, significantly outperforming LETNet, SGCPNet, and FBSNet. On the other hand, although

HARD-XXS has the fewest parameters in the models of Table 2, it achieved 74.4 mIoU and 532 FPS, outperforming other counterparts in Table 2.

4.4 Comparison on COCO-Stuff-10K

Table 3: Comparisons with SOTA real-time methods on COCO-Stuff-10K.

Resolution	Model	Params	FLOPs	mIoU	FPS
480×480	HARD-XXS	0.11M	1.49G	23.8	**527**
640×640	BiSeNetV2-L [50]	5.2M	27.8G	28.7	225
512×512	DeepLabV3+(MV2) [5]	15.4M	25.9G	29.9	-
640×640	DDRNet-23 [16]	20.1M	27.9G	32.1	200
640×640	PSPNet [54]	49.0M	288.2G	32.6	78
480×480	**HARD-XS**	0.39M	2.42G	33.1	527
640×640	SeaFormer-B [42]	8.6M	2.39G	34.1	-
512×512	AFFormer-B [8]	3.0M	4.6G	35.1	109
640×640	SCTNet-B [48]	17.4M	23.37G	35.9	216
480×480	**HARD-S**	0.76M	3.87G	37.0	358
640×640	**HARD-GPU**	3.9M	9.27G	**41.0**	271

The COCO-Stuff-10k dataset is a large-scale segmentation dataset having 171 classes. On COCO-Stuff-10k, it is known that real-time semantic segmentation is very challenging due to the large number of classes. In Table 3, HARD-GPU achieved 41.0 mIoU, demonstrating 5.1% higher performance compared with the state-of-the-art model SCTNet. Furthermore, HARD-GPU achieved state-of-the-art performance with a much faster inference speed. Notably, HARD-XS achieved 33.1 mIoU by having only 0.39M parameters. Although the above number of parameters was smaller than those of Seaformer-B and PSPNet, HARD-XS produced comparable performance. Besides, HARD-XS achieved 527 FPS, being faster than other models in Table 3.

4.5 Semantic Segmentation on Embedded Computing Device

Table 4: Comparisons on NVIDIA Jetson ORIN NX.

Resolution	Model	Params	mIoU	FPS	Resolution	Model	Params	mIoU	FPS
512×1024	FBSNet [12]	0.61M	70.9	3	512×1024	ESPNetV2 [30]	1.25M	66.2	18
1024×2048	CFPNet [28]	0.27M	70.1	4	512×1024	ENet [32]	0.38M	58.3	19
1024×2048	CGNet [46]	0.50M	64.8	4	512×1024	EDANet [24]	0.69M	67.3	19
512×1024	FDDWNet [22]	0.77M	71.5	7	512×1024	NDNet [49]	0.50M	65.1	20
512×1024	LETNet [40]	0.95M	72.8	8	1024×2048	FastSCNN [34]	1.14M	68.6	20
512×1024	LiteHRNet [52]	1.09M	70.6	9	512×1024	FSSNet [17]	0.29M	58.8	26
512×1024	ERFNet [35]	2.06M	68	11	512×1024	ESPNet [29]	0.36M	60.3	27
512×1024	LEDNet [45]	0.94M	70.6	12	360×640	**HARD-S**	0.76M	**72.8**	30
512×1024	ADSCNet [43]	0.51M	67.5	14	512×1024	MiniNet [1]	1.41M	61.5	36
512×1024	FPENet [23]	0.4M	70.1	15	360×640	**HARD-XS**	0.39M	69.6	40
512×1024	MiniNetV2 [2]	0.51M	71.8	15	360×640	**HARD-XXS**	0.11M	64.1	**64**
1024×2048	SGCPNet [14]	0.61M	70.9	17					

Most of all, real-time semantic segmentation studies have been conducted using high-performance GPUs. To prove the effectiveness of the proposed models in embedded computing environments, our experiments were performed on

NVIDIA Jetson Orin NX 16GB. In the above experiments, FPS was measured in each model based on the resolution from Cityscapes. The experimental results are presented in Table 4. Most real-time segmentation models could not achieve 30 FPS. However, the proposed HARD-XXS demonstrated 64 FPS, which was more than $\times 2$ faster than that of ESPNet. HARD-S achieved the highest performance at 72.8 mIoU while also demonstrating a high inference speed of over 30 FPS. Besides, the inference speed was $\times 4$ faster than FDDWNet, having slightly more advanced results. HARD-S achieved 12 mIoU higher performance than ESPNet and comparable inference speed. The above summary of Table 4 shows that the proposed HARD can outperform other real-time segmentation models on the embedded computing device.

4.6 Semantic Segmentation on MCU

Table 5: Comparisons on STM32F746NG. When the target model exceeds its memory constraints, it is marked as OOM denoting "Out Of Memory".

Model	Param	Flash (FP32)	SRAM (FP32)	Flash (Int8)	SRAM (Int8)	mIoU	Latency
ENet [32]	0.36M	1.41MB (OOM)	13.18MB (OOM)	735.46kB	3.35MB (OOM)	51.3	-
ESPNet [29]	0.36M	1.43MB (OOM)	14.5MB (OOM)	1.15MB (OOM)	3.55MB (OOM)	55.6	-
NDNet [49]	0.5M	1.87MB (OOM)	14.54MB (OOM)	-	-	57.2	-
CFPNet [28]	0.55M	1.15MB (OOM)	14.62MB (OOM)	1.2MB (OOM)	7.44MB (OOM)	64.3	-
FSSNet [17]	0.26M	719.77kB	5.54MB (OOM)	439.41kB	3.68MB (OOM)	58.6	-
DABNet [19]	0.76M	2.89MB (OOM)	14.54MB (OOM)	898.77kB	7.44MB (OOM)	66.4	-
HARD-Edge	0.11M	620.72kB	694.04kB (OOM)	**271.41kB**	**283.05kB**	**57.3**	**30.1ms**

In the evaluations of HARD-Edge on an MCU, STM32F746NG (Cortex-M7 processor, 320kB SRAM, and 1MB Flash) was adopted. Table 5 shows the comparisons of lightweight segmentation models trained on the CamVid dataset in terms of performance and latency. We performed experiments on FP32 real-valued and Int8 quantized models, respectively. In FP32, the activations and model outputs of all models were out of SRAM capacity in the target device. Besides, ENet, ESPNet, NDNet, CFPNet, and DABNet cannot be stored in Flash memory because the memory requirements for storing the above models were over 1 MB or more. Therefore, Int8 quantization was necessary for the deployment in MCUs.

The 8-bit quantized lightweight segmentation models were deployed in the target MCU for an apple-to-apple comparison. Although most models were designed for embedded computing devices, only the HARD-Edge could be deployed on an MCU. Although only FSSNet can meet the limitation with 719 kB Flash, it exceeded the memory capacity of 320kB SRAM. In Int8 quantization, ENet, FSSNet, and DABNet did not exceed Flash memory capacity. However, SRAM capacity was insufficient for the above models. On the other hand, HARD-Edge was designed to have minimal computational overhead so it can achieve a fast inference speed of 30.1ms (33.2 FPS) on STM32F746NG. As far as we know, HARD-Edge is the first real-time segmentation model that can be deployed on MCUs.

4.7 Ablation Study

Table 6: Effects on DAP module and Extra Semantic Loss (denoted as ES Loss) of HARD-GPU.

Pooling filter	Atrous filter	ES Loss	mIoU
			69.8
✓			69.9
	✓		70.5
✓	✓		73.0
✓	✓	✓	**73.8**

Effects on DAP module and Extra Semantic Loss We performed ablation studies to validate the effectiveness of the proposed DAP module and ES Loss. In Table 6, there was a degradation of 4 mIoU when Atrous and Pooling filters were removed. The performance was significantly degraded when either the Atrous filter or the Pooling filter was not deployed. Therefore, we concluded that the training with multi-contextual information is significantly important for the semantic segmentation task. The extra semantic loss was effective in improving the performance without increasing the number of parameters and computational overhead. In Fig. 4, it is worth noting that HARD-GPU incorporates an auxiliary classifier solely during the training process. In Table 6, the training HARD-GPU with an auxiliary classifier resulted in a 0.8 mIoU performance improvement. Moreover, after extensive experiments, a weight of 0.3 was found to be the most effective for the weighting ES Loss in Eq. 7. When the weight was set to 0.1, there was only a performance improvement of 0.07 mIoU, and when the weight was set as 0.5, the performance improvement was only 0.2 mIoU.

Table 7: Comparisons with lightweight segmentation models on low-resolution CamVid.

Model	Resolution	Param	mIoU	Flash	SRAM	FPS (GPU)	Latency (MCU)
FSSNet [17]	64×64	0.26M	26.7	553.2kB	**182.57kB**	256	44.7ms
ENet [32]	64×64	0.36M	37.1	821.52kB	211.15kB	149	40.5ms
ESPNet [29]	64×64	0.36M	39.5	1.14MB (OOM)	238.72kB	167	OOM
NDNet [49]	64×64	0.5M	39.6	1.2MB (OOM)	367.44kB (OOM)	261	OOM
DABNet [19]	64×64	0.76M	40.0	1.26MB (OOM)	308.11kB	232	OOM
CFPNet [28]	64×64	0.55M	46.6	1.2MB (OOM)	367.44kB (OOM)	66	OOM
HARD-Edge	64×64	**0.11M**	**57.3**	**271.42kB**	283.05kB	**535**	**30.1ms**

Low-Resolution Performance Table 7 shows the training results with the CamVid dataset when images were resized to 64×64. FPS was measured on a single NVIDIA RTX 4090 GPU, and latency was measured on an STM32F746NG. While FSSNet was faster than ENet on GPUs, it was shown that ENet was faster on the STM32F746NG. It was noted that FSSNet required more memory access time due to the residual connection between the encoder and decoder feature maps. Therefore, it showed a 4.2 ms slower latency on the MCU. However,

HARD-Edge achieved the highest performance at low resolution with the fewest number of parameters. It also showed 30.1 ms, which was 10.4 ms faster than ENet.

4.8 Comparison on Visualization Results

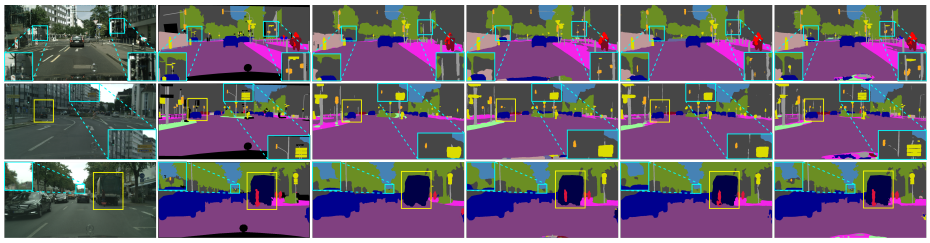

Fig. 6: Visualized comparisons on the Cityscapes validation set. From left to right are original input images, ground truths, and segmentation results from BiSeNetv2 [50], STDC [11], SCTNet [48], and the proposed HARD.

Fig. 6 shows the visualization results for the cityscapes validation set. Compared with SCTNet, STDC, and BiseNetv2, which have similar numbers of parameters, HARD-GPU achieved higher-quality results. It effectively produced precise segmentation masks for small and narrow objects like utility poles, signs, and traffic lights. Notably, HARD-GPU showed higher quality long-range context extraction and better preservation of object boundaries than SCTNet using Transformer.

5 Conclusion

In this paper, we propose HARD that can be deployed on GPUs, embedded computing devices, and MCUs. We demonstrate through intensive experiments on a variety of datasets and devices where HARD achieves new SOTA results. Also, the proposed DAP and SA modules can successfully extract long-range contextual information. Notably, HARD is the first method to deploy semantic segmentation deployable on MCUs. HARD can extend its applications of real-time semantic segmentation, considering computational resources.

Acknowledgements This work was supported by the IITP(Institute of Information & Coummunications Technology Planning & Evaluation)-ICAN(ICT Challenge and Advanced Network of HRD) grant funded by the Korea government(Ministry of Science and ICT)(IITP-2024-RS-2024-00437788) and K-CHIPS(Korea Collaborative & High-tech Initiative for Prospective Semiconductor Research)(1415188224, RS-2023-00301703, 23045-15TC) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea).

References

1. Alonso, I., Riazuelo, L., Murillo, A.C.: Enhancing v-slam keyframe selection with an efficient convnet for semantic analysis. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 4717–4723 (2019). https://doi.org/10.1109/ICRA.2019.8793923
2. Alonso, I., Riazuelo, L., Murillo, A.C.: Mininet: An efficient semantic segmentation convnet for real-time robotic applications. IEEE Transactions on Robotics **36**(4), 1340–1347 (2020). https://doi.org/10.1109/TRO.2020.2974099
3. Brostow, G.J., Fauqueur, J., Cipolla, R.: Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters **xx**(x), xx–xx (2008)
4. Caesar, H., Uijlings, J., Ferrari, V.: Coco-stuff: Thing and stuff classes in context. In: Computer vision and pattern recognition (CVPR), 2018 IEEE conference on. IEEE (2018)
5. 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). pp. 801–818 (2018)
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
8. Dong, B., Wang, P., Wang, F.: Head-free lightweight semantic segmentation with linear transformer. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 516–524 (2023)
9. Dong, G., Yan, Y., Shen, C., Wang, H.: Real-time high-performance semantic image segmentation of urban street scenes. IEEE Transactions on Intelligent Transportation Systems **22**(6), 3258–3274 (2020)
10. Dourado, A., De Campos, T.E., Kim, H., Hilton, A.: Edgenet: Semantic scene completion from a single rgb- d image. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 503–510 (2021). https://doi.org/10.1109/ICPR48806.2021.9413252
11. Fan, M., Lai, S., Huang, J., Wei, X., Chai, Z., Luo, J., Wei, X.: Rethinking bisenet for real-time semantic segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9716–9725 (2021)
12. Gao, G., Xu, G., Li, J., Yu, Y., Lu, H., Yang, J.: Fbsnet: A fast bilateral symmetrical network for real-time semantic segmentation. IEEE Transactions on Multimedia (2022)
13. Gao, G., Xu, G., Yu, Y., Xie, J., Yang, J., Yue, D.: Mscfnet: A lightweight network with multi-scale context fusion for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems **23**(12), 25489–25499 (2021)
14. Hao, S., Zhou, Y., Guo, Y., Hong, R., Cheng, J., Wang, M.: Real-time semantic segmentation via spatial-detail guided context propagation. IEEE Transactions on Neural Networks and Learning Systems (2022)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

16. Hong, Y., Pan, H., Sun, W., Jia, Y.: Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes. arXiv preprint arXiv:2101.06085 (2021)
17. Hu, X., Wang, H.: Efficient fast semantic segmentation using continuous shuffle dilated convolutions. IEEE Access **8**, 70913–70924 (2020)
18. Lee, J., Kim, D., Ponce, J., Ham, B.: Sfnet: Learning object-aware semantic correspondence. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2278–2287 (2019)
19. Li, G., Yun, I., Kim, J., Kim, J.: Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. arXiv preprint arXiv:1907.11357 (2019)
20. Li, H., Xiong, P., Fan, H., Sun, J.: Dfanet: Deep feature aggregation for real-time semantic segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9522–9531 (2019)
21. Lin, G., Milan, A., Shen, C., Reid, I.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1925–1934 (2017)
22. Liu, J., Zhou, Q., Qiang, Y., Kang, B., Wu, X., Zheng, B.: Fddwnet: a lightweight convolutional neural network for real-time semantic segmentation. In: ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 2373–2377. IEEE (2020)
23. Liu, M., Yin, H.: Feature pyramid encoding network for real-time semantic segmentation. arXiv preprint arXiv:1909.08599 (2019)
24. Lo, S.Y., Hang, H.M., Chan, S.W., Lin, J.J.: Efficient dense modules of asymmetric convolution for real-time semantic segmentation. In: Proceedings of the 1st ACM International Conference on Multimedia in Asia. pp. 1–6 (2019)
25. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
26. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
27. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts (2017)
28. Lou, A., Loew, M.: Cfpnet: Channel-wise feature pyramid for real-time semantic segmentation. In: 2021 IEEE International Conference on Image Processing (ICIP). pp. 1894–1898. IEEE (2021)
29. Mehta, S., Rastegari, M., Caspi, A., Shapiro, L., Hajishirzi, H.: Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In: Proceedings of the european conference on computer vision (ECCV). pp. 552–568 (2018)
30. Mehta, S., Rastegari, M., Shapiro, L., Hajishirzi, H.: Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9190–9200 (2019)
31. Orsic, M., Kreso, I., Bevandic, P., Segvic, S.: In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12607–12616 (2019)
32. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: A deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147 (2016)

33. Peng, J., Liu, Y., Tang, S., Hao, Y., Chu, L., Chen, G., Wu, Z., Chen, Z., Yu, Z., Du, Y., et al.: Pp-liteseg: A superior real-time semantic segmentation model. arXiv preprint arXiv:2204.02681 (2022)
34. Poudel, R.P., Liwicki, S., Cipolla, R.: Fast-scnn: Fast semantic segmentation network. arXiv preprint arXiv:1902.04502 (2019)
35. Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R.: Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems **19**(1), 263–272 (2017)
36. 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 18. pp. 234–241. Springer (2015)
37. 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. pp. 4510–4520 (2018)
38. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
39. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
40. Ta, N., Chen, H., Liu, X., Jin, N.: Let-net: locally enhanced transformer network for medical image segmentation. Multimedia Systems **29**(6), 3847–3861 (2023)
41. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
42. Wan, Q., Huang, Z., Lu, J., Yu, G., Zhang, L.: Seaformer: Squeeze-enhanced axial transformer for mobile semantic segmentation. arXiv preprint arXiv:2301.13156 (2023)
43. Wang, J., Xiong, H., Wang, H., Nian, X.: Adscnet: asymmetric depthwise separable convolution for semantic segmentation in real-time. Applied Intelligence **50**(4), 1045–1056 (2020)
44. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al.: Deep high-resolution representation learning for visual recognition. IEEE transactions on pattern analysis and machine intelligence **43**(10), 3349–3364 (2020)
45. Wang, Y., Zhou, Q., Liu, J., Xiong, J., Gao, G., Wu, X., Latecki, L.J.: Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In: 2019 IEEE international conference on image processing (ICIP). pp. 1860–1864. IEEE (2019)
46. Wu, T., Tang, S., Zhang, R., Cao, J., Zhang, Y.: Cgnet: A light-weight context guided network for semantic segmentation. IEEE Transactions on Image Processing **30**, 1169–1179 (2020)
47. Xu, J., Xiong, Z., Bhattacharyya, S.P.: Pidnet: A real-time semantic segmentation network inspired by pid controllers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 19529–19539 (2023)
48. Xu, Z., Wu, D., Yu, C., Chu, X., Sang, N., Gao, C.: Sctnet: Single-branch cnn with transformer semantic information for real-time segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 6378–6386 (2024)

49. Yang, Z., Yu, H., Fu, Q., Sun, W., Jia, W., Sun, M., Mao, Z.H.: Ndnet: Narrow while deep network for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems **22**(9), 5508–5519 (2020)
50. Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., Sang, N.: Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. International Journal of Computer Vision **129**, 3051–3068 (2021)
51. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European conference on computer vision (ECCV). pp. 325–341 (2018)
52. Yu, C., Xiao, B., Gao, C., Yuan, L., Zhang, L., Sang, N., Wang, J.: Lite-hrnet: A lightweight high-resolution network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10440–10450 (2021)
53. Zhao, H., Qi, X., Shen, X., Shi, J., Jia, J.: Icnet for real-time semantic segmentation on high-resolution images. In: Proceedings of the European conference on computer vision (ECCV). pp. 405–420 (2018)
54. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2881–2890 (2017)