2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Fast and Accurate Real-Time Semantic Segmentation with Dilated Asymmetric Convolutions

Leonel Rosas-Arias*, Gibran Benitez-Garcia †, José Portillo-Portillo*, Gabriel Sánchez-Pérez * and Keiji Yanai†
*Instituto Politecnico Nacional, ESIME Culhuacan, Mexico City, Mexico
Email: lrosasa1800@alumno.ipn.mx, jportillop@ipn.mx, gasanchezp@ipn.mx
†Department of Informatics, The University of Electro-Communications, Tokyo, Japan
Email: gibran@ieee.org, yanai@cs.uec.ac.jp

***Abstract*—Recent works have shown promising results applied to real-time semantic segmentation tasks. To maintain fast inference speed, most of the existing networks make use of light decoders, or they simply do not use them at all. This strategy helps to maintain a fast inference speed; however, their accuracy performance is significantly lower in comparison to non-real-time semantic segmentation networks. In this paper, we introduce two key modules aimed to design a high-performance decoder for real-time semantic segmentation for reducing the accuracy gap between real-time and non-real-time segmentation networks. Our first module, Dilated Asymmetric Pyramidal Fusion (DAPF), is designed to substantially increase the receptive field on the top of the last stage of the encoder, obtaining richer contextual features. Our second module, Multi-resolution Dilated Asymmetric (MDA) module, fuses and refines detail and contextual information from multi-scale feature maps coming from early and deeper stages of the network. Both modules exploit contextual information without excessively increasing the computational complexity by using asymmetric convolutions. Our proposed network entitled "FASSD-Net" reaches 78.8% of mIoU accuracy on the Cityscapes validation dataset at 41.1 FPS on full resolution images (1024×2048). Besides, with a light version of our network, we reach 74.1% of mIoU at 133.1 FPS (full resolution) on a single NVIDIA GTX 1080Ti card with no additional acceleration techniques. The source code and pre-trained models are available at github.com/GibranBenitez/FASSD-Net.**

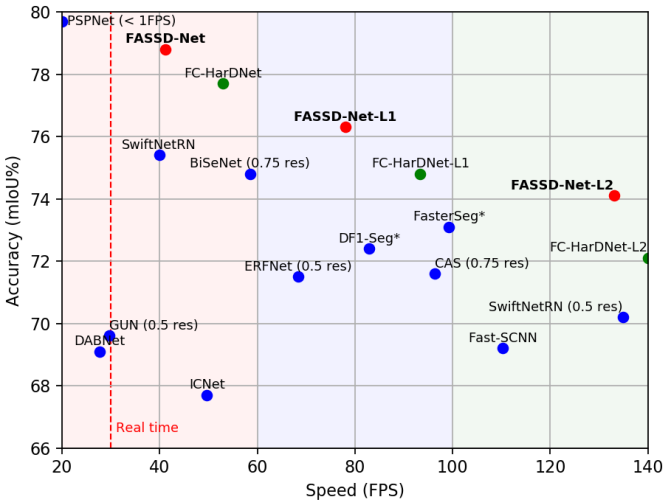

Fig. 1. Speed and Accuracy comparison between state-of-the-art methods for real-time semantic segmentation on the Cityscapes validation set. For a fair comparison, the speed of methods marked by (*) are approximated without TensorRT acceleration. PSPNet and FC-HarDNet-L2 speeds are placed on the x-axis edges for the sake of better visualization.

I. Introduction

The research of semantic segmentation is considered as a fundamental task in computer vision [1]–[3]. It aims to assign semantic class labels to each pixel in a given input image. In recent years, due to the development of new deep learning techniques, semantic segmentation has been widely applied to several challenging fields, including: autonomous driving, robot sensing, medical imaging, augmented reality, and video surveillance, to name a few [4].

Some applications require the inference speed to be as fast as possible with the maximum possible accuracy. Furthermore, according to the available hardware or budget, factors such as low energy consumption and memory usage also become crucial [5]. In particular, applications such as autonomous driving cars require to keep a balance between high accuracy prediction and low inference time to be able to take action rapidly [3], [6]. However, that speed and accuracy are two factors that seemingly contradict each other, making real-time semantic segmentation a challenging task [2].

In order to increase the accuracy performance, some state-of-the-art networks for real-time semantic segmentation [6]–[8] use *U-shape*-like architectures [9] to recover hierarchical features from previous stages of the network. Nevertheless, their accuracy performance is still significantly lower in comparison to non-real-time semantic segmentation networks.

In either encoder or decoder stages of the network, one common way to achieve a further increment in accuracy is leveraging the use of dilated (atrous) convolutions, which enlarge the receptive field of the convolution kernel [10]. The problem is, that methods designed to exploit this property require a considerable number of floating-point operations, such as the Atrous Spatial Pyramid Pooling module (ASPP) [11]. On top of that, the ASPP module heavily relies on dilated convolutions, which by themselves are slow to compute due

978-1-7281-8808-9/20/$31.00 ©2020 IEEE

to framework optimization constrains [8].

The slow-down caused by the use of dilated convolutions can be alleviated by implementing convolution factorization, as demonstrated in [5]. Therefore, in this paper, we introduce two modules designed to increase the prediction accuracy, exploiting contextual information in multiple stages of the network. We name these two modules: Dilated Asymmetric Pyramidal Fusion (DAPF) and Multi-resolution Dilated Asymmetric (MDA) module, respectively.

In order to increase the kernel receptive field and keep low computational complexity, we carefully employ 3×3 dilated convolutions factorized into two consecutive 1D dilated convolutions. From hereafter, we refer to this type of convolutions as *dilated asymmetric convolutions*, due to the asymmetric nature of the convolution kernel and the dilation implementation.

Our DAPF module is designed to significantly increase the receptive field of the last stage of the encoder network, obtaining richer contextual features. We follow a pyramidal scheme similar to the DeepLabV3+'s ASPP module [11], but all 3×3 dilated convolutions are replaced with dilated asymmetric convolutions. Our design, allows the number of pyramidal feature maps of DAPF to vary according to the number of input feature maps, which further reduces the computational complexity of the module.

Similarly, our MDA module, fuses multi-resolution feature maps coming from previous encoder and decoder levels of the network. In this module, feature maps are processed simultaneously by two parallel branches: the asymmetric branch and the non-asymmetric branch. The asymmetric convolutional branch exploits the contextual information of the input feature maps, whereas the non-asymmetric branch focuses on recovering details. This design allows a simultaneous refinement of detail and contextual information in multiple stages of the decoder.

Our proposed network, entitled FASSD-Net, combines the DAPF and MDA modules and effectively increases the semantic segmentation accuracy with a relatively low computational cost. Additionally, we present two light variations that provide a balanced trade-off between accuracy and inference speed, namely, FASSD-Net-L1 and FASSD-Net-L2.

Overall, our proposed networks bridge the accuracy gap between existing real-time (around 70% mIoU) [2], [5], [6], [8], [12]–[14] and non-real-time networks (about 80% mIoU) for semantic segmentation [1], [11], [15]–[20]. This new gap is illustrated in Figure 1.

Our main contributions are summarized as follows:

- We introduce DAPF, an efficient plug-in-and-play spatial pyramidal fusion module inspired by ASPP [11]. DAPF demands far less computational complexity, which enables its use for real-time applications.
- We introduce the MDA module, which allows better learning from two different stages of the network, simultaneously refining spatial and contextual information.
- As shown in Figure 1, our proposed networks obtain state-of-the-art mIoU results on the Cityscapes validation set for the task of real-time semantic segmentation. Furthermore, FASSD-Net is comparable to non-real-time methods such as DeepLabV3+ [11] and PSPNet [15] in terms of accuracy, while being about 40× faster.

II. Related work

Models such as PSPNet [15] and DeepLabV3+ [11] exploit contextual information by processing the same set of feature maps. PSPNet [15] does it by downsampling the feature maps at four different rates, perform a series of convolutions on them, and finally perform a fusion process. Likewise, DeepLabV3+ [11] processes the feature maps by applying atrous convolutions at different rates. These models have achieved top results on several segmentation benchmarks by leveraging the use of multi-scale information in a pyramidal fashion. However, even in modern GPUs, the required computational resources for these methods are prohibited [13]. Thus, making them unfeasible for real-time applications. In contrast, our proposed FASSD-Net handles the same pyramidal strategy used in DeepLabV3+ [11] but using dilated asymmetric convolutions, allowing its use in real-time applications.

In addition, fusion strategies for multi-resolution feature maps have been used in recent works such as HarDNet [7], SwiftNet [8] and FasterSeg [12]. These networks either concatenate or add two sets of feature maps and further process them with a single convolution. This straightforward fusion strategy requires a small amount of computation, but do not exploit contextual information. In contrast, our MDA module concatenates two sets of feature maps, and it processes them by using two parallel branches, simultaneously refining features rich in detail information and features rich in context.

On the other hand, techniques for reducing the computational complexity of the networks such as depth-wise separable convolutions [21]–[25], zoomed convolutions [12], or convolution factorization [5], [13], [14] have been proposed and applied to the task of real-time semantic segmentation. Networks that employ these techniques such as Fast-SCNN [25], ERFNet [13], and FasterSeg [12] achieve real-time performance, usually at the cost of significantly lower accuracy compared to non-real-time methods. Similarly, our three network proposals rely on factorized (asymmetric) convolutions used in the DAPF and MDA modules. However, as shown in Figure 1, our proposed networks outperform Fast-SCNN [25], ERFNet [13] and FasterSeg [12] in terms of accuracy, keeping comparable speed performance.

Additional methods for accelerating neural networks include filtering or channel pruning, network distillation, Network Architecture Search (NAS), and Neural Network Quantization [12], [26]–[29]. Such methods, mainly reduce the number of parameters and weight of the model, boosting the inference speed. However, most of them, either utilize sophisticated methodologies, require a considerable amount of memory, or cannot be directly applied to more elaborated network architectures [30], [31]. Specifically, FasterSeg [12] and CAS [28] obtain state-of-the-art inference speed by utilizing NAS techniques. By comparison, our proposed models are designed manually and still, outperform these NAS networks in accuracy and speed performance.

Networks such as ESPNetv2 [32], ESNet [32], and LEDNet [33] employ lightweight pyramidal multi-resolution strategies similar to our DAPF module. More specifically, LEDNet [33] and ESNet [14] incorporate asymmetric convolutions in their core modules. However, they heavily rely on the use of this technique, which hurts the inference speed performance in comparison to highly optimized standard convolutions [8]. Contrary to these methods, our DAPF and MDA modules utilize asymmetric convolutions only if dilation is applied concurrently, alleviating the reduction of inference speed performance caused by the atrous convolutions [5]. When compared to our proposals, LEDNet [33] and ESNet [14] are slower and much less accurate.

III. Methodology

Most of the existing state-of-the-art methods for semantic segmentation are built on top of high-performance baselines for image classification such as ResNet, WiderResNet, or Xception [1], [11], [16]–[18]. Following this trend, we adopt and extend the work of *Chao et al.* [7] by incorporating the DAPF and the MDA modules. The proposed network structure is shown in Figure 2. In the figure, the stem convolution block consists of four consecutive convolution layers. The core element of all encoder and decoder blocks is the HarDBlock (Harmonic Dense Block), proposed in HarDNet [7]. Our proposed DAPF is placed at the end of the encoder, while MDA modules connect each decoder block with its corresponding encoder, in a *U-shape* fashion. Finally, the last block of the network consists of a single 1×1 convolution for making the final prediction. Bilinear upsampling is used to reestablish the original input size (1024×2048).

A. Network overview

HarDNet (Harmonic DenseNet) [7], is a recent state-of-the-art network inspired by DenseNet (Densely Connected Network) [34]. Compared to ResNet [35] and DenseNet [34], HarDNet achieves comparable accuracy with significantly lower GPU runtime for classification tasks. Its core component, the *HarDBlock* (Harmonic Dense Block), is specifically designed to address the problem of the GPU memory traffic. The HarDBlock follows a concatenation scheme aimed to improve the throughput of the feature maps in the network, avoiding unnecessary DRAM (dynamic random-access memory) accesses. Additionally, the HarDBlock is optimized to increase the density of computations of the layers, defined by the number of Multiply-Accumulate operations (MACs) over the Convolutional Input/Output (CIO). These key improvements are based on the observation that when the density of computation is low, DRAM traffic can influence inference time more substantially than the model size and the number of operations.

Our baseline model, FC-HarDNet-70, is the implementation of HarDNet for the task of semantic segmentation. FC-HarDNet-70 is a *U-shape*-like architecture [9] with five encoder blocks and four decoder blocks (all of them HarDBlocks). The convolution layers in the last encoder stage of FC-HarDNet process a high number of feature maps with $1/64$ input size resolution, which is essential for classification tasks. However, we believe that this is not always the case for semantic segmentation tasks, where such small size feature maps lose track of small objects present in the scene (e.g., for a 1024×2048 image and a downsampling rate of 64, the size becomes 16×32). In our FASSD-Net implementation, the last encoder block and the first decoder block of FC-HarDNet-70 are replaced with our DAPF module, so that the smallest feature maps processed by our network are $1/32$ of the input size resolution. Similarly, the FC-HarDNet-70 multi-resolution fusion scheme is substituted by our MDA module.

B. Dilated Asymmetric Pyramidal Fusion module

As shown in Figure 3, the ASPP module heavily relies on standard atrous convolutions and produces a fixed number of feature maps $\mathbf{Q}$ in each of its five pyramidal branches. Pyramidal branches consist of: $1\times$ Conv 1×1, $1\times$ Pooling + Conv 1×1, and $3\times$ atrous Conv 3×3 with dilation rates r = 12, 24, and 36, respectively.

Our proposed DAPF module draws inspiration from the ASPP module, but with some key differences aimed to reduce its computational burden: Firstly, all 3×3 atrous convolutions are factorized into two consecutive 1D atrous convolutions, specifically, a 3×1 convolution followed by a 1×3 convolution. Secondly, the image pooling branch is removed since it computes feature maps that might as well be learned through the 1×1 convolutional branch. Lastly, the number of feature maps generated by each pyramidal branch, is no longer fixed. Instead, it is defined by the number of input feature maps of the module $\mathbf{K} \times \frac{1}{\alpha}$, where α serves as the compression factor. In our implementation, we set $\alpha = 2$. However, α can be adjusted to any other value according to the available computational budget. Figure 3 illustrates the differences between DAPF and the ASPP modules.

Undoubtedly, the major contributing factor for reducing the computational burden in our DAPF module is the use of asymmetric convolutions. For instance, in the pyramidal branches, for each standard 2D convolution, we would have to perform $K \times d \times d \times F$ operations, where K is the number of input channels (feature maps), d is the kernel size, and F is the number of output channels. On the other hand, following our asymmetric strategy with $\alpha = 2$, we perform $(K \times d \times \frac{1}{2}K) + (\frac{1}{2}K \times d \times \frac{1}{2}K)$ operations. For a 3×3 kernel, this factorization strategy only requires $\frac{1}{2}$ of the original number of operations, thus saving 50% of the needed computations and parameters in comparison to its non-asymmetric convolution equivalent. Moreover, factorization can also improve the learning capacity of the module as a result of the intermediate activation layers used between the two 1D convolutions [13].

C. Muti-resolution Dilated Asymmetric module

We design our MDA module to simultaneously exploit contextual information and recover spatial information. As shown in Figure 4, the two multi-resolution feature maps

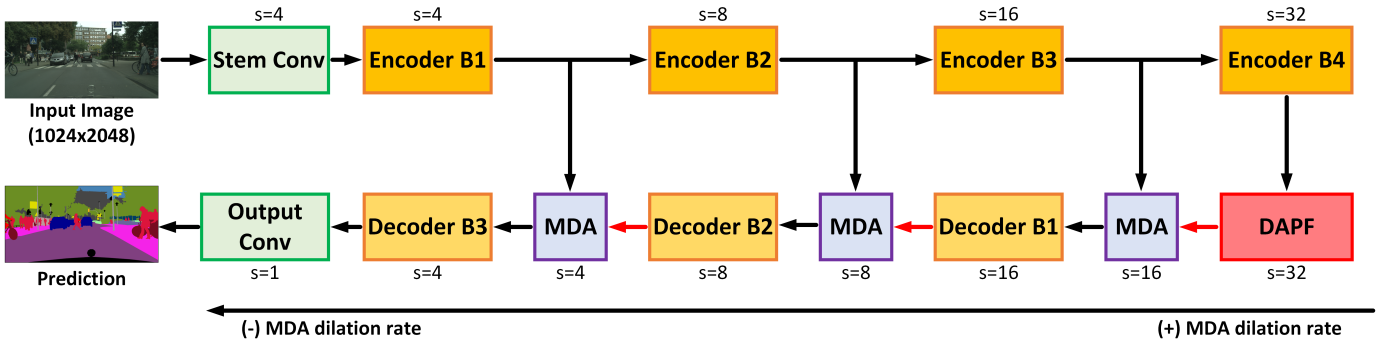

Fig. 2. Block diagram of the proposed network. "s" indicates the downsampling rate of the feature maps with respect to the original input image (e.g., *s=32* indicates output feature maps of size 32×64).

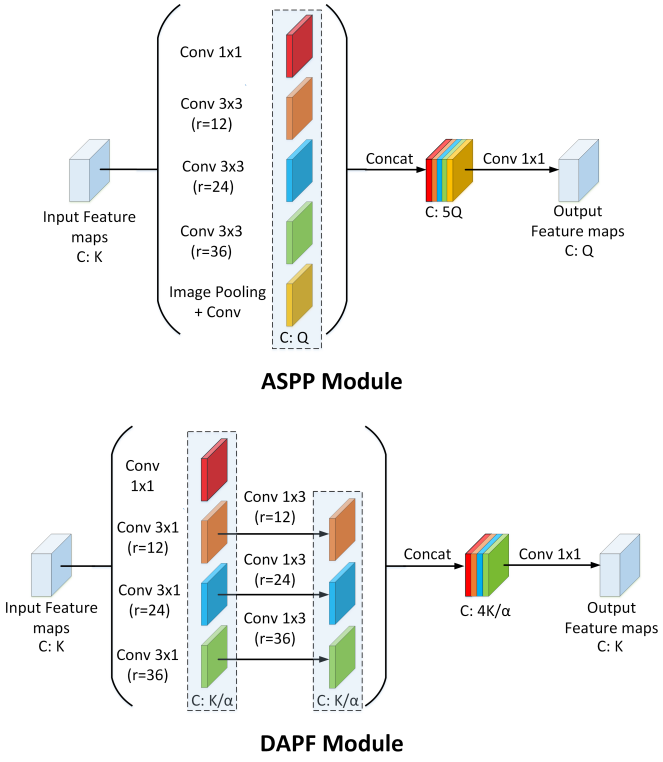

Fig. 3. ASPP and DAPF modules comparison. Each convolution is followed by its respective batch normalization and activation layers. "C" denotes the number of feature maps.

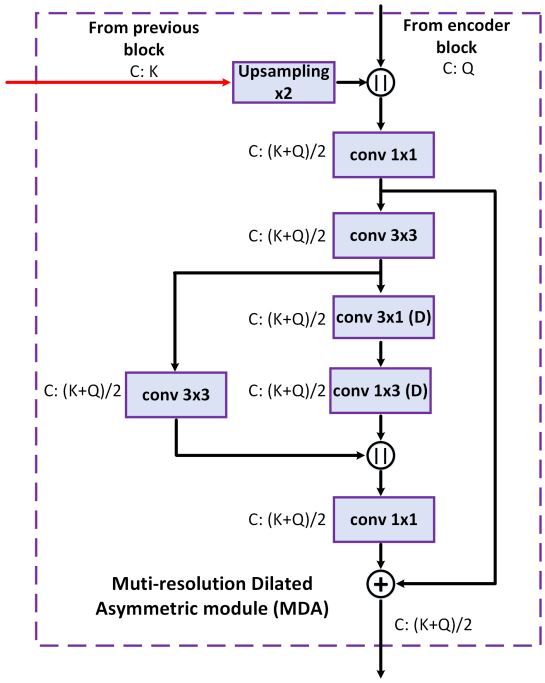

Fig. 4. Muti-resolution Dilated Asymmetric module. "C" denotes the number of output feature maps, "||" indicates concatenation and "D" indicates dilated convolution.

K and **Q** are concatenated and fused together with a 1×1 convolution. This convolution reduces the number of feature maps by half to reduce the number of computations in the module and subsequent stages.

After an additional 3×3 convolution that refines the initially fused feature maps, two parallel branches process the output feature maps. The asymmetric convolutional branch aims to exploit the contextual information present in the feature maps by leveraging the use of dilated convolutions. In contrast, the non-asymmetric branch focuses on refining the details. The resulting feature maps are concatenated and processed by a 1×1 convolution to match the number of feature maps of the first 1×1 convolution. Finally, feature maps of both 1×1 convolutions are summed up through a residual connection, helping to improve the gradient flow.

The dilation rate r of the asymmetric branch gradually decreases in every MDA block, from the *deepest* to the *shallowest* stage of the decoder (see Figure 2). Specifically, the dilation rates are $r = (8, 4, 2)$ from Decoder B1 to Decoder B3, respectively. The intuition behind this idea is that inner feature maps of the network are richer in contextual information and can be leveraged by atrous convolutions with larger dilation rates.

TABLE I
ABLATION STUDY OF OUR PROPOSED MODULES ON THE CITYSCAPES VALIDATION SET.

Method	GFLOPs	No. Parameters	Δp	FPS	mIoU
FC-HarDNet-70 [7]	35.4	4.10M	-	52.3	76.4
Baseline	**32.9**	**1.90M**	0M	**56.3**	75.2
+ ASPP	36.8	3.85M	1.95M	50.2	75.8
+ DAPF	33.9	2.36M	**0.46M**	53.9	77.7
+ MDA	44.2	2.38M	0.48M	42.2	77.4
+ ASPP + MDA	48.0	4.33M	2.43M	39.1	76.8
+ DAPF + MDA	45.1	2.85M	0.95M	41.1	**78.2**

TABLE II
FASSD-NET ARCHITECTURE. L DENOTES THE NUMBER OF CONVOLUTION LAYERS IN THE HARDBLOCK.

Stage	Name	Type	Output size
Input	-	-	1024×2048×3
Encoder	Stem Conv	Conv 3×3 (s=2)*	512×1024×16
		Conv 3×3**	512×1024×24
		Conv 3×3 (s=2)	256×512×32
		Conv 3×3	256×512×48
	Encoder B1	HarDBlock (L=4)	256×512×64
	Encoder B2	2D Average Pooling HarDBlock (L=4)	128×256×96
	Encoder B3	2D Average Pooling HarDBlock (L=8)	64×128×160
	Encoder B4	2D Average Pooling HarDBlock (L=8)	32×64×224
	DAPF	-	32×64×224
Decoder	MDA	-	64×128×192
	Decoder B1	HarDBlock (L=8)	64×128×160
	MDA	-	128×256×119
	Decoder B2	HarDBlock (L=4)	128×256×78
	MDA	-	256×512×63
	Decoder B3	HarDBlock (L=4)	256×512×48
	Output Conv	Conv 1×1	256×512×19
		Upsampling ×4	1024×2048×19

*Changes to (s=3) for FASSD-Net-L1
**Changes to (s=2) for FASSD-Net-L2

IV. EXPERIMENTS

We evaluate our proposed network architectures on the Cityscapes benchmark [36]. Performance is mainly measured in mean Intersection over Union accuracy (mIoU) and Frames per Second (FPS). Besides, we report the number of parameters and computational complexity in GFLOPs. All experiments are conducted using the publicly available Cityscapes dataset [36]. This dataset consists of 5,000 finely annotated 1024×2048 images: 2,975 for training, 1,525 for testing, and another 500 images for validation. Additionally, 19,998 images with coarse annotations are also provided. For a fair comparison, we only use the fine annotated images, and do not employ any augmentation technique for testing, such as multi-scale or multi-crops, which increase the accuracy at the cost of inference time.

A. Implementation Details

We use PyTorch 1.0 with CUDA 10.2 for all experiments. The same training setting is used for all models, where Stochastic Gradient Descent (SGD) with weight-decay 5×10^{-4} and momentum 0.9 is used as the optimizer. We employ the "poly" learning rate strategy $lr = initial\_lr \times (\frac{iter}{total\_iter})^{0.9}$, and an initial learning rate of 0.02. Cross-entropy loss is computed following the online bootstrapping strategy [37]. Data augmentation consists of random horizontal flip, random scale in the range $[0.5, 2]$, and random cropping with 1024×1024 crop size. We trained all models for 90k iterations with batch size 16. For the final models, we follow the same training protocol for 30K more iterations, setting the batch size to 24 and the initial learning rate to 0.001.

All networks are pre-trained on the ImageNet dataset [38], and the inference speed (in FPS) is measured on an Intel Core i7-9700K desktop with one NVIDIA GTX 1080ti card unless specified otherwise. For all experiments, the speed is calculated from the average FPS rate of $10,000$ iterations measured on images of size $1024 \times 2048 \times 3$.

B. Ablation Study

We show the performance comparison between our proposed DAPF and the DeepLabV3+'s ASPP module [11]. We evaluate the effectiveness of our MDA module and its combination with DAPF and ASPP. Table I summarizes the corresponding results. *Baseline* denotes the modified FC-HarDNet-70, where the last encoder and the first decoder blocks are removed, as previously described in Section III. Note that, for a fair comparison, all methods shown in Table I are trained without further fine-tuning.

Our DAPF module outperforms DeepLabV3+'s ASPP in all four metrics (GFLOPS, Parameters, FPS and mIoU). In addition, the increase of parameters (Δp) by DAPF from the *Baseline* is significantly fewer than the ASPP module (0.46M vs 1.95M). In resume, our module is more than four times lighter than ASPP, and presents an increase of 1.9% of mIoU. It can be observed that the addition of our MDA strategy outperforms the mIoU of the baseline network to almost the same degree as DAPF. Likewise, our MDA strategy consumes roughly the same number of parameters. However, since MDA is used in three different levels of the decoder, the FPS drop becomes evident compared to DAPF.

The combination of our two proposals achieves the best mIoU accuracy. Specifically, it outperforms the baseline and the state-of-the-art results of FC-HarDNet-70 [7] by 3% and 1.8%, respectively. Additionally, the increase of parameters (Δp) of our proposal is less than 50% over the case when ASSP is used. On top of that, the total number of parameters is significantly lower than those needed by FC-HArDNet-70 (2.85M vs 4.10M). Note that *Baseline + DAPF + MDA* corresponds to our final model called FASSD-Net, as shown in Figure 2.

C. Light variations of FASSD-Net

In addition to our network FASSD-Net, we introduce two light versions designed to maintain a better tradeoff between

TABLE III
PER-CLASS MIOU SCORE COMPARISON OF OUR PROPOSALS ON THE CITYSCAPES VALIDATION SET.

Method	road	s.walk	build.	wall	fence	pole	t.light	t.sign	veg.	terr.	sky	person	rider	car	truck	bus	train	mbik	bike	mIoU
FC-HarDNet-70 [7]	98.1	84.6	92.6	**60.0**	63.5	64.9	69.7	78.8	92.2	62.8	95.0	81.4	60.7	95.0	73.9	82.4	77.4	**61.7**	76.2	77.4
FASSD-Net	**98.3**	**86.0**	**92.9**	59.6	**63.9**	**67.1**	**71.7**	**79.6**	**92.5**	**63.7**	**95.0**	**82.1**	**63.2**	**95.4**	**80.8**	**87.8**	**79.6**	61.3	**76.8**	**78.8**
FASSD-Net-L1	98.2	84.6	92.4	54.7	61.3	63.3	68.2	77.1	92.1	61.8	94.9	80.0	59.8	94.9	76.0	82.7	74.7	59.1	74.6	76.3
FASSD-Net-L2	97.9	83.1	91.6	55.6	57.0	58.0	62.5	71.8	91.7	63.0	94.4	77.3	57.0	93.8	75.5	81.9	71.1	53.1	70.8	74.1

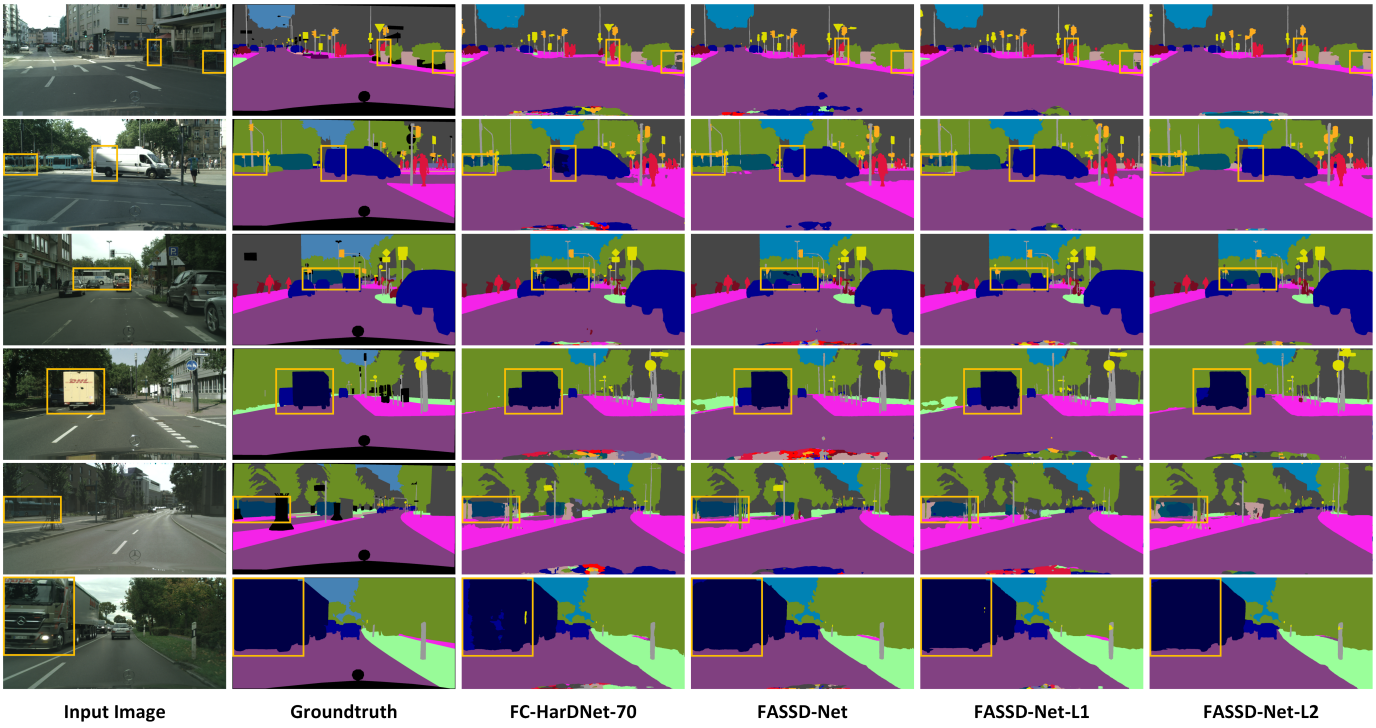

Fig. 5. Qualitative results of the proposed networks. Regions of improvement are highlighted with yellow squares.

speed and accuracy. We call these networks: FASSD-Net-L1 and FASSD-Net-L2.

Table II shows the detailed architecture of FASSD-Net, including the output and number of channels for each element. From Table II, FASSD-Net-L1 differs only in the first convolution layer, where the convolution stride is increased from 2 to 3. Such modification, preserves the same number of parameters of the network and leads to a faster inference speed at the cost of a small drop in accuracy. Specifically, it is 1.9× faster and 2.5% less accurate. FASSD-Net-L2, on the other hand, is designed to be the fastest among our three proposals. It adopts an additional convolution stride of 2 in the second convolution layer of the stem block. In addition, all the HarDBlocks in the decoder are replaced by conventional 3×3 convolution layers of 64 channels. Thus, FASSD-Net-L2 is 3.2× faster than FASSD-Net with only a 4.7% drop in accuracy performance.

We report the results of per-class mIoU accuracy in Table III. FASSD-Net obtains the highest score in 17 out of 19 categories, outperforming the current state-of-the-art model, FC-HarDNet-70. Most significative improvements occur in the truck and bus classes with 6.9% and 5.4%, respectively. Similarly, FASSD-Net-L1 and FASSD-Net-L2 also obtain better results overall in these two classes compared to FC-HarDNet-70, despite being less accurate models. Qualitative results of our FASSD-Net variations are shown in Figure 5. As the quantitative results suggest, the most significant improvements occur on pixels belonging to large objects, such as trucks and buses. Compared to FC-HarDNet-70, all three FASSD-Net variations better differentiate between car, bus, and truck classes. For a fair comparison, we have conducted the evaluation of our final models against FC-HarDNet-70 with its official weights from its open-source implementation[1].

D. Comparison with state-of-the-art real-time methods

Table IV shows the overall comparison of our network proposals versus other state-of-the-art methods for real-time semantic segmentation. The table is divided into three categories, based on the inference speed directly comparable to each of our three proposed networks. For a more complete and

[1] github.com/PingoLH/FCHarDNet

TABLE IV
COMPARISON BETWEEN STATE-OF-THE-ART NETWORKS FOR REAL-TIME SEMANTIC SEGMENTATION.

Method	Input Size	GPU	GFLOPs	No. Parameters	FPS	FPS (norm.)	mIoU
DeepLabV3+ [11]	512×1024	Titan X (P)	-	-	≈ 1	≈ 1	79.6
PSPNet [15]	713×713	Titan X (P)	-	-	< 1	< 1	79.7
Fast-SCNN [25]	1024×2048	Titan XP	-	1.11M	123.5	110.3	69.2
SwiftNetRN-18 [8]	512×1024	GTX 1080ti	26.0	11.8M	134.9	134.9	70.2
FC-HarDNet-70 (L2)	1024×2048	GTX 1080ti	6.6	2.86M	152.8	152.8	72.1
FASSD-Net-L2 (ours)	1024×2048	GTX 1080ti	**8.7**	2.3M	133.1	133.1	74.1
ERFNet [13]	512×1024	Titan X (M)	-	2.10M	41.7	68.4	71.5
CAS [28]	768×1536	Titan XP	-	-	108.0	96.4	71.6
DF1-Seg-d8 [26]	1024×2048	GTX 1080ti	-	-	136.9*	82.9	72.4
FasterSeg [12]	1024×2048	GTX 1080ti	28.2	4.4M	**163.9***	99.3	73.1
FC-HarDNet-70 (L1)	1024×2048	GTX 1080ti	15.7	4.1M	93.4	93.4	74.8
FASSD-Net-L1 (ours)	1024×2048	GTX 1080ti	20.0	2.85M	78.0	78.0	76.3
ICNet [2]	1024×2048	Titan X (M)	28.3	26.5M	30.3	49.7	67.7
DABNet [5]	1024×2048	GTX 1080ti	41.8	**0.76M**	27.7	27.7	69.1
GUN [39]	512×1024	Titan XP	-	-	33.3	29.7	69.6
BiSeNet [3]	768×1536	Titan XP	-	49M	65.5	58.5	74.8
SwiftNetRN-18 [8]	1024×2048	GTX 1080ti	104.0	11.8M	39.9	39.9	75.4
FC-HarDNet-70 [7]	1024×2048	GTX 1080ti	35.4	4.1M	53.0	53.0	77.4
FASSD-Net (ours)	1024×2048	GTX 1080ti	45.1	2.85M	41.1	41.1	**78.8**

* Speed measured with TensorRT acceleration

fair comparison with respect to our baseline, FC-HarDNet-70 (L1) and FC-HarDNet-70 (L2) are our modified implementations of FC-HarDNet-70 that closely resemble to our two light networks, following the same changes and our same training protocol. FC-HarDNet-70 (L1) and FC-HarDNet-70 (L2) are directly modified from the official source code of FC-HarDNet-70.

For fair comparison under different GPU architectures, we follow the same protocol as *Orsic et al.* [8] and let the column *FPS (norm.)* in Table IV to provide a speed estimation of the model running on a GTX 1080ti GPU. Scaling factors are: 1.0 for GTX 1080ti, 0.61 for Titan X (Maxwell), 1.03 for Titan X (Pascal), and 1.12 for Titan XP.

Our main network, FASSD-Net, surpasses by a considerable margin the mIoU score of all other methods for real-time semantic segmentation, requiring $1.44\times$ fewer parameters and being 1.4% more accurate than the closest competitor FC-HarDNet-70.

Our second network, FASSD-Net-L1, resembles BiSeNet in mIoU accuracy and FPS. However, the speed of BiSeNet has been originally measured on 768×1536 images with an NVIDIA Titan XP card. For a fair comparison, and according to *Zhuang et al.* [40], we let its speed to be 37 FPS evaluated on 1024×2048 images on an NVIDIA GTX 1080ti card. Therefore, resulting in our network being about $2.1\times$ faster, 1.5% more accurate, and requiring $17.2\times$ fewer parameters.

Similarly, FASSD-Net-L2 can be compared to FasterSeg and DF1-Seg-d8, which were designed and optimized by NAS methodologies. Both methods utilize TensorRT acceleration [41] to increase their speed performance. For a fair comparison, we let $1.65\times$ be the acceleration factor of TensorRT. This value is approximated from works that present results with and without TensorRT, such as FasterSeg [12]. Under these assumptions, our FASSD-Net-L2 is faster than FasterSeg and DF1-Seg-d8, while being 1% and 1.7% more accurate, respectively.

V. CONCLUSION

In this paper, we focus on reducing the accuracy gap between real-time and non-real-time semantic segmentation networks. For this purpose, we have proposed the DAPF and MDA modules. These modules exploit the contextual information in several stages of the decoder and boost the accuracy performance of the baseline network, keeping relatively low computational cost. Using our two modules jointly, we have designed three network variations that can be chosen depending on the computational budget. Our main network, FASSD-Net, sets the new state-of-the-art mIoU accuracy for the task of real-time semantic segmentation on the Cityscapes validation set. In addition, our proposed FASSD-Net-L2 ranks as the fastest network when evaluated on 1024×2048 images without using additional network acceleration techniques. As future work, we would like to evaluate our proposals on different scenarios, such as indoor understanding, or medical images. We also would like to apply acceleration techniques, such as network quantization or network distillation, to increase the speed of our models further.

ACKNOWLEDGMENT

The authors would like to thank to the University of Electro-Communications, the National Polytechnic Institute ESIME-Culhuacan and CONACyT (Mexico) for providing financial support for the development of this work. This work was also supported by JSPS KAKENHI Grant Number 15H05915, 17H01745, 17H06100 and 19H04929.

REFERENCES

[1] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-Cross Attention for Semantic Segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[2] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for Real-Time Semantic Segmentation on High-Resolution Images," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[3] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[4] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41 – 65, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494618302813

[5] G. Li and J. Kim, "DABNet: Depth-wise Asymmetric Bottleneck for Real-time Semantic Segmentation," in *British Machine Vision Conference*, 2019.

[6] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep Feature Aggregation for Real-Time Semantic Segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[7] P. Chao, C.-Y. Kao, Y.-S. Ruan, C.-H. Huang, and Y.-L. Lin, "HarDNet: A Low Memory Traffic Network," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[8] M. Orsic, I. Kreso, P. Bevandic, and S. Segvic, "In Defense of Pre-Trained ImageNet Architectures for Real-Time Semantic Segmentation of Road-Driving Images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[10] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1511.07122

[11] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," *arXiv e-prints*, p. arXiv:1802.02611, Feb 2018.

[12] W. Chen, X. Gong, X. Liu, Q. Zhang, Y. Li, and Z. Wang, "FasterSeg: Searching for Faster Real-time Semantic Segmentation," in *International Conference on Learning Representations*, 2020.

[13] E. Romera, J. M. Álvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, pp. 263–272, 2018.

[14] Y. Wang, Q. Zhou, and X. Wu, "ESNet: An Efficient Symmetric Network for Real-time Semantic Segmentation," *arXiv e-prints*, p. arXiv:1906.09826, Jun 2019.

[15] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[16] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-SCNN: Gated Shape CNNs for Semantic Segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[17] Z. Tian, T. He, C. Shen, and Y. Yan, "Decoders Matter for Semantic Segmentation: Data-Dependent Decoding Enables Flexible Feature Aggregation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[18] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, and H. Liu, "Expectation-Maximization Attention Networks for Semantic Segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[19] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," *ArXiv*, vol. abs/1909.11065, 2019.

[20] R. P. K. Poudel, U. Bonde, S. Liwicki, and C. Zach, "Contextnet: Exploring context and detail for semantic segmentation in real-time," in *BMVC*, 2018.

[21] F. Chollet, "Xception: Deep Learning With Depthwise Separable Convolutions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[22] 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," *arXiv e-prints*, p. arXiv:1704.04861, Apr 2017.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[24] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for MobileNetV3," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[25] R. P. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: fast semantic segmentation network," *arXiv preprint arXiv:1902.04502*, 2019.

[26] X. Li, Y. Zhou, Z. Pan, and J. Feng, "Partial Order Pruning: For Best Speed/Accuracy Trade-Off in Neural Architecture Search," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[27] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] Y. Zhang, Z. Qiu, J. Liu, T. Yao, D. Liu, and T. Mei, "Customizable Architecture Search for Semantic Segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[29] R. Zhao, Y. Hu, J. Dotzel, C. De Sa, and Z. Zhang, "Improving neural network quantization without retraining using outlier channel splitting," in *International Conference on Machine Learning*, 2019, pp. 7543–7552.

[30] X. Zheng, R. Ji, L. Tang, Y. Wan, B. Zhang, Y. Wu, Y. Wu, and L. Shao, "Dynamic distribution pruning for efficient network architecture search," *arXiv preprint arXiv:1905.13543*, 2019.

[31] H. Cai, L. Zhu, and S. Han, "ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://arxiv.org/pdf/1812.00332.pdf

[32] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNetv2: A Light-Weight, Power Efficient, and General Purpose Convolutional Neural Network," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[33] 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 *2019 IEEE International Conference on Image Processing (ICIP)*, Sep. 2019, pp. 1860–1864.

[34] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[36] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[37] Z. Wu, C. Shen, and A. van den Hengel, "High-performance Semantic Segmentation Using Very Deep Fully Convolutional Networks," *CoRR*, vol. abs/1604.04339, 2016. [Online]. Available: http://arxiv.org/abs/1604.04339

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[39] D. Mazzini, "Guided Upsampling Network for Real-Time Semantic Segmentation," *arXiv e-prints*, p. arXiv:1807.07466, Jul 2018.

[40] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, "ShelfNet for Fast Semantic Segmentation," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[41] NVIDIA, "TensorRT," September 2010, [Online; accessed February 14, 2020]. [Online]. Available: https://developer.nvidia.com/tensorrt