Semantic Flow for Fast and Accurate Scene Parsing

Xiangtai Li[1] ⋆, Ansheng You[1] *, Zhen Zhu[2], Houlong Zhao[3], Maoke Yang[3],
Kuiyuan Yang[3], Shaohua Tan[1], and Yunhai Tong[1]

[1] Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
[2] Huazhong University of Science and Technology
[3] DeepMotion

Abstract. In this paper, we focus on designing effective method for fast and accurate scene parsing. A common practice to improve the performance is to attain high resolution feature maps with strong semantic representation. Two strategies are widely used—atrous convolutions and feature pyramid fusion, are either computation intensive or ineffective. Inspired by the Optical Flow for motion alignment between adjacent video frames, we propose a Flow Alignment Module (FAM) to learn Semantic Flow between feature maps of adjacent levels, and broadcast high-level features to high resolution features effectively and efficiently. Furthermore, integrating our module to a common feature pyramid structure exhibits superior performance over other real-time methods even on light-weight backbone networks, such as ResNet-18. Extensive experiments are conducted on several challenging datasets, including Cityscapes, PASCAL Context, ADE20K and CamVid. Especially, our network is the first to achieve 80.4% mIoU on Cityscapes with a frame rate of 26 FPS. The code is available at https://github.com/donnyyou/torchcv.

Keywords: Scene Parsing, Semantic Flow, Flow Alignment Module

1 Introduction

Scene parsing or semantic segmentation is a fundamental vision task which aims to classify each pixel in the images correctly. Two important factors that are highly influential to the performance are: detailed information [46] and strong semantics representation [6, 64]. The seminal work of Long *et. al.* [33] built a deep Fully Convolutional Network (FCN), which is mainly composed from convolutional layers, in order to carve strong semantic representation. However, detailed object boundary information, which is also crucial to the performance, is usually missing due to the use of the down-sampling layers. To alleviate this problem, state-of-the-art methods [15, 64, 65, 68] apply atrous convolutions [55] at the last several stages of their networks to yield feature maps with strong semantic representation while at the same time maintaining the high resolution.

⋆ The first two authors have equal contribution.

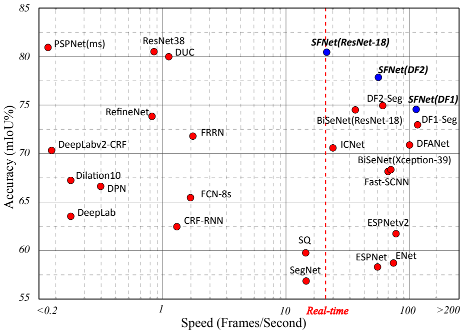


Fig. 1. Inference speed versus mIoU performance on test set of Cityscapes. Previous models are marked as red points, and our models are shown in blue points which achieve the best speed/accuracy trade-off. Note that our method with ResNet-18 as backbone even achieves comparable accuracy with all accurate models at much faster speed.

Nevertheless, doing so inevitably requires intensive extra computation since the feature maps in the last several layers can reach up to 64 times bigger than those in FCNs. Given that the FCN using ResNet-18 [19] as the backbone network has a frame rate of 57.2 FPS for a 1024 × 2048 image, after applying atrous convolutions [55] to the network as done in [64,65], the modified network only has a frame rate of 8.7 FPS. Moreover, under a single GTX 1080Ti GPU with no other ongoing programs, the previous state-of-the-art model PSPNet [64] has a frame rate of only 1.6 FPS for 1024 × 2048 input images. As a consequence, this is very problematic to many advanced real-world applications, such as self-driving cars and robots navigation, which desperately demand real-time online data processing.

In order to not only maintain detailed resolution information but also get features that exhibit strong semantic representation, another direction is to build FPN-like [23,32,46] models which leverage the lateral path to fuse feature maps in a top-down manner. In this way, the deep features of the last several layers strengthen the shallow features with high resolution and therefore, the refined features are possible to satisfy the above two factors and beneficial to the accuracy improvement. However, the accuracy of these methods [1,46] is still unsatisfactory when compared to those networks who hold large feature maps in the last several stages. We suspect the low accuracy problem arises from the ineffective propagation of semantics from deep layers to shallow layers.

To mitigate this issue, we propose to learn the **Semantic Flow** between two network layers of different resolutions. The concept of Semantic Flow is inspired from optical flow, which is widely used in video processing task [67] to represent the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by relative motion. In a flash of inspiration, we feel the relationship between two feature maps of arbitrary resolutions from the same image can also be represented with the "motion" of every pixel from one feature map to the other one. In this case, once precise Semantic Flow is obtained, the network is able to propagate semantic features with minimal information loss. It should be noted that Semantic Flow is apparently different from optical flow, since Semantic Flow takes feature maps from different levels as input and assesses the discrepancy within them to find a suitable flow field that will give dynamic indication about how to align these two feature maps effectively.

Based on the concept of Semantic Flow, we design a novel network module called Flow Alignment Module (FAM) to utilize Semantic Flow in the scene parsing task. Feature maps after FAM are embodied with both rich semantics and abundant spatial information. Because FAM can effectively transmit the semantic information from deep layers to shallow layers through very simple operations, it shows superior efficacy in both improving the accuracy and keeping superior efficiency. Moreover, FAM is end-to-end trainable, and can be plugged into any backbone networks to improve the results with a minor computational overhead. For simplicity, we call the networks that all incorporate FAM but have different backbones as **SFNet(backbone)**. As depicted in Figure 1, SFNet with different backbone networks outperforms other competitors by a large margin under the same speed. In particular, our method adopting ResNet-18 as backbone achieves **80.4%** mIoU on the Cityscapes test server with a frame rate of **26 FPS**. When adopting DF2 [29] as backbone, our method achieves 77.8% mIoU with 61 FPS and 74.5% mIoU with 121 FPS when equipped with the DF1 backbone. Moreover, when using deeper backbone networks, such as ResNet-101, SFNet achieves better results(81.8 %mIoU) than the previous state-of-the-art model DANet [15](81.5 %mIoU), and only requires **33%** computation of DANet during the inference. Besides, the consistent superior efficacy of SFNet across various datasets also clearly demonstrates its broad applicability.

To conclude, our main contributions are three-fold:

- We introduce the concept of Semantic Flow in the field of scene parsing and propose a novel flow-based align module (FAM) to learn the Semantic Flow between feature maps of adjacent levels and broadcast high-level features to high resolution features more effectively and efficiently.
- We insert FAMs into the feature pyramid framework and build a feature pyramid aligned network called SFNet for fast and accurate scene parsing.
- Detailed experiments and analysis indicate the efficacy of our proposed module in both improving the accuracy and keeping light-weight. We achieve state-of-the-art results on Cityscapes, Pascal Context, Camvid datasets and a considerable gain on ADE20K.

2 Related Work

For scene parsing, there are mainly two paradigms for high-resolution semantic map prediction. One paradigm tries to keep both spatial and semantic information along the main network pathway, while the other paradigm distributes spatial and semantic information to different parts in a network, then merges them back via different strategies.

The first paradigm mostly relies on some network operations to retain high-resolution feature maps in the latter network stages. Many state-of-the-art accurate methods [15, 64, 68] follow this paradigm to design sophisticated head networks to capture contextual information. PSPNet [64] proposes to leverage pyramid pooling module (PPM) to model multi-scale contexts, whilst DeepLab series [5–7, 52] uses astrous spatial pyramid pooling (ASPP). In [15, 17, 18, 20, 27, 56, 69], non-local operator [50] and self-attention mechanism [49] are adopted to harvest pixel-wise context from the whole image. Meanwhile, several works [22, 26, 30, 59, 60] use graph convolutional neural networks to propagate information over the image by projecting features into an interaction space.

The second paradigm contains several state-of-the-art fast methods, where high-level semantics are represented by low-resolution feature maps. A common strategy is to fuse multi-level feature maps for high-resolution spatiality and strong semantics [1,28,33,46,51]. ICNet [63] uses multi-scale images as input and a cascade network to be more efficient. DFANet [25] utilizes a light-weight backbone to speed up its network and proposes a cross-level feature aggregation to boost accuracy, while SwiftNet [42] uses lateral connections as the cost-effective solution to restore the prediction resolution while maintaining the speed. To further speed up, low-resolution images are used as input for high-level semantics [35, 63] which reduce features into low resolution and then upsample them back by a large factor. The direct consequence of using a large upsample factor is performance degradation, especially for small objects and object boundaries. Guided upsampling [35] is related to our method, where the semantic map is upsampled back to the input image size guided by the feature map from an early layer. However, this guidance is still insufficient for some cases due to the information gap between the semantics and resolution. In contrast, our method aligns feature maps from adjacent levels and further enhances the feature maps using a feature pyramid framework towards both high resolution and strong semantics, consequently resulting in the state-of-the-art performance considering the trade-off between high accuracy and fast speed.

There is another set of works focusing on designing light-weight backbone networks to achieve real-time performances. ESPNets [36, 37] save computation by decomposing standard convolution into point-wise convolution and spatial pyramid of dilated convolutions. BiSeNet [53] introduces spatial path and semantic path to reduce computation. Recently, several methods [29, 39, 62] use AutoML techniques to search efficient architectures for scene parsing. Our method is complementary to some of these works, which further boosts their accuracy. Since our proposed semantic flow is inspired by optical flow [13], which is used in video semantic segmentation, we also discuss several works in video semantic segmentation. For accurate results, temporal information is exceedingly exploited by using optical flow. Gadde *et. al.* [16] warps internal feature maps and Nilsson *et. al.* [41] warps final semantic maps from nearby frame predictions to the current map. To pursue faster speed, optical flow is used to bypass the low-level feature computation of some frames by warping features from their preceding frames [31, 67]. Our work is different from theirs by propagating information hierarchically in another dimension, which is orthogonal to the temporal propagation for videos.

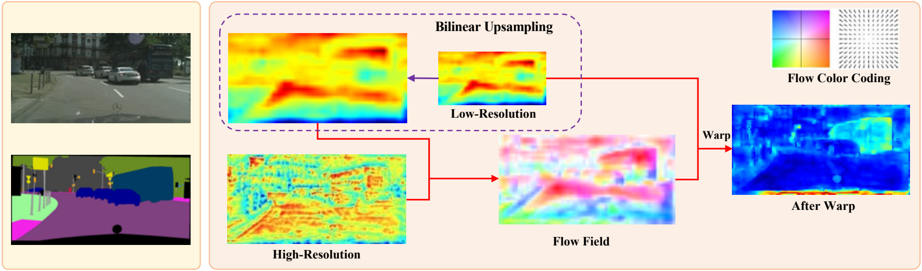


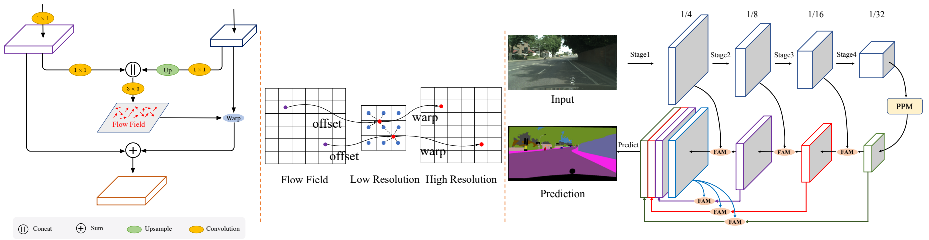
Fig. 2. Visualization of feature maps and semantic flow field in FAM. Feature maps are visualized by averaging along the channel dimension. Larger values are denoted by hot colors and vice versa. We use the color code proposed in [2] to visualize the Semantic Flow field. The orientation and magnitude of flow vectors are represented by hue and saturation respectively.

3 Method

In this section, we will first give some preliminary knowledge about scene parsing and introduce the misalignment problem therein. Then, we propose the Flow Alignment Module (FAM) to resolve the misalignment issue by learning Semantic Flow and warping top-layer feature maps accordingly. Finally, we present the whole network architecture equipped with FAMs based on the FPN framework [32] for fast and accurate scene parsing.

3.1 Preliminary

The task of scene parsing is to map an RGB image $\mathbf{X} \in \mathbb{R}^{H \times W \times 3}$ to a semantic map $\mathbf{Y} \in \mathbb{R}^{H \times W \times C}$ with the same spatial resolution $H \times W$, where C is the number of predefined semantic categories. Following the setting of FPN [32], the input image $\mathbf{X}$ is firstly mapped to a set of feature maps $\{\mathbf{F}_l\}_{l=2,...,5}$ from each network stage, where $\mathbf{F}_l \in \mathbb{R}^{H_l \times W_l \times C_l}$ is a C_l-dimensional feature map defined on a spatial grid Ω_l with size of $H_l \times W_l, H_l = \frac{H}{2^l}, W_l = \frac{W}{2^l}$. The coarsest feature map $\mathbf{F}_5$ comes from the deepest layer with strongest semantics. FCN-32s directly predicts upon $\mathbf{F}_5$ and achieves over-smoothed results without fine details. However, some improvements can be achieved by fusing predictions from lower levels [33]. FPN takes a step further to gradually fuse high-level feature maps with low-level feature maps in a top-down pathway through 2× bi-linear upsampling, which was originally proposed for object detection [32] and recently introduced for scene parsing [23,51]. The whole FPN framework highly relies on upsampling operator to upsample the spatially smaller but semantically stronger feature map to be larger in spatial size. However, the bilinear upsampling recovers the resolution of downsampled feature maps by interpolating a set of uniformly sampled positions (i.e., it can only handle one kind of fixed and predefined misalignment), while the misalignment between feature maps caused by a residual connection, repeated downsampling and upsampling, is far more complex. Therefore, position correspondence between feature maps needs to be explicitly and dynamically established to resolve their actual misalignment.



(a) Flow Alignment Module **(b) Warp Procedure** **(c) Overview of SFNet**

Fig. 3. (a) The details of Flow Alignment Module. We combine the transformed high-resolution feature map and low-resolution feature map to generate the semantic flow field, which is utilized to warp the low-resolution feature map to high-resolution feature map. (b) Warp procedure of Flow Alignment Module. The value of the high-resolution feature map is the bilinear interpolation of the neighboring pixels in low-resolution feature map, where the neighborhoods are defined according learned semantic flow field. (c) Overview of our proposed SFNet. ResNet-18 backbone with four stages is used for exemplar illustration. FAM: Flow Alignment Module. PPM: Pyramid Pooling Module [64]. Best view it in color and zoom in.

3.2 Flow Alignment Module

Design Motivation. For more flexible and dynamic alignment, we thoroughly investigate the idea of optical flow, which is very effective and flexible to align two adjacent video frame features in the video processing task [4,67]. The idea of optical flow motivates us to design a *flow-based alignment module* (**FAM**) to align feature maps of two adjacent levels by predicting a flow field inside the network. We define such flow field as *Semantic Flow*, which is generated between different levels in a feature pyramid. For efficiency, while designing our network, we adopt an efficient backbone network—FlowNet-S [13].

Module Details. FAM is built within the FPN framework, where feature map of each level is compressed into the same channel depth through two 1×1 convolution layers before entering the next level. Given two adjacent feature maps $\mathbf{F}_l$ and $\mathbf{F}_{l-1}$ with the same channel number, we up-sample $\mathbf{F}_l$ to the same size as $\mathbf{F}_{l-1}$ via a bi-linear interpolation layer. Then, we concatenate them together and take the concatenated feature map as input for a sub-network that contains two convolutional layers with the kernel size of 3 × 3. The output of the sub-network is the prediction of the semantic flow field $\Delta_{l-1} \in \mathbb{R}^{H_{l-1} \times W_{l-1} \times 2}$. Mathematically, the aforementioned steps can be written as:

$$\Delta_{l-1} = \mathrm{conv}_l(\mathrm{cat}(\mathbf{F}_l, \mathbf{F}_{l-1})), \tag{1}$$

where $\mathrm{cat}(\cdot)$ represents the concatenation operation and $\mathrm{conv}_l(\cdot)$ is the 3 × 3 convolutional layer. Since our network adopts strided convolutions, which could lead to very low resolution, for most cases, the respective field of the 3×3 convolution conv_l is sufficient to cover most large objects of that feature map. Note that, we discard the correlation layer proposed in FlowNet-C [13], where positional correspondence is calculated explicitly. Because there exists a huge semantic gap between higher-level layer and lower-level layer, explicit correspondence calculation on such features is difficult and tends to fail for offset prediction. Moreover, adopting such a correlation layer introduces heavy computation cost, which violates our goal for the network to be fast and accurate.

After having computed $\varDelta_{l-1}$, each position p_{l-1} on the spatial grid $\varOmega_{l-1}$ is then mapped to a point p_l on the upper level l via a simple addition operation. Since there exists a resolution gap between features and flow field shown in Fig 3(b), the warped grid and its offset should be halved as Eq 2,

$$p_l = \frac{p_{l-1} + \varDelta_{l-1}(p_{l-1})}{2}. \tag{2}$$

We then use the differentiable bi-linear sampling mechanism proposed in the spatial transformer networks [21], which linearly interpolates the values of the 4-neighbors (top-left, top-right, bottom-left, and bottom-right) of p_l to approximate the final output of the FAM, denoted by $\widetilde{\mathbf{F}}_l(p_{l-1})$. Mathematically,

$$\widetilde{\mathbf{F}}_l(p_{l-1}) = \mathbf{F}_l(p_l) = \sum_{p \in \mathcal{N}(p_l)} w_p \mathbf{F}_l(p), \tag{3}$$

where $\mathcal{N}(p_l)$ represents neighbors of the warped points p_l in $\mathbf{F}_l$ and w_p denotes the bi-linear kernel weights estimated by the distance of warped grid. This warping procedure may look similar to the convolution operation of the deformable kernels in deformable convolution network (DCN) [10]. However, our method has a lot of noticeable difference from DCN. First, our predicted offset field incorporates both higher-level and lower-level features to *align the positions* between high-level and low-level feature maps, while the offset field of DCN moves the positions of the kernels according to the predicted location offsets in order to *possess larger and more adaptive respective fields*. Second, our module focuses on aligning features while DCN works more like an attention mechanism that attends to the salient parts of the objects. More detailed comparison can be found in the experiment part.

On the whole, the proposed FAM module is light-weight and end-to-end trainable because it only contains one 3×3 convolution layer and one parameter-free warping operation in total. Besides these merits, it can be plugged into networks multiple times with only a minor extra computation cost overhead. Figure 3(a) gives the detailed settings of the proposed module while Figure 3(b) shows the warping process. Figure 2 visualizes feature maps of two adjacent levels, their learned semantic flow and the finally warped feature map. As shown in Figure 2, the warped feature is more structurally neat than normal bi-linear upsampled feature and leads to more consistent representation of objects, such as the bus and car.

3.3 Network Architectures

Figure 3(c) illustrates the whole network architecture, which contains a bottom-up pathway as the encoder and a top-down pathway as the decoder. While the encoder has a backbone network offering feature representations of different levels, the decoder can be seen as a FPN equipped with several FAMs.

Encoder Part. We choose standard networks pre-trained on ImageNet [47] for image classification as our backbone network by removing the last fully connected layer. Specifically, ResNet series [19], ShuffleNet v2 [34] and DF series [29] are used and compared in our experiments. All backbones have 4 stages with residual blocks, and each stage has a convolutional layer with stride 2 in the first place to downsample the feature map chasing for both computational efficiency and larger receptive fields. We additionally adopt the Pyramid Pooling Module (PPM) [64] for its superior power to capture contextual information. In our setting, the output of PPM shares the same resolution as that of the last residual module. In this situation, we treat PPM and the last residual module together as the last stage for the upcoming FPN. Other modules like ASPP [6] can also be plugged into our network, which are also experimentally ablated in Sec. 4.1.

Aligned FPN Decoder takes feature maps from the encoder and uses the aligned feature pyramid for final scene parsing. By replacing normal bi-linear up-sampling with FAM in the top-down pathway of FPN [32], $\{\mathbf{F}_l\}_{l=2}^{4}$ is refined to $\{\widetilde{\mathbf{F}}_l\}_{l=2}^{4}$, where top-level feature maps are aligned and fused into their bottom levels via element-wise addition and l represents the range of feature pyramid level. For scene parsing, $\{\widetilde{\mathbf{F}}_l\}_{l=2}^{4} \cup \{\mathbf{F}_5\}$ are up-sampled to the same resolution (*i.e.*, 1/4 of input image) and concatenated together for prediction. Considering there are still misalignments during the previous step, we also replace these up-sampling operations with the proposed FAM.

Cascaded Deeply Supervised Learning. We use deeply supervised loss [64] to supervise intermediate outputs of the decoder for easier optimization. In addition, following [53], online hard example mining [48] is also used by only training on the 10% hardest pixels sorted by cross-entropy loss.

4 Experiments

We first carry out experiments on the Cityscapes [9] dataset, which is comprised of a large set of high-resolution (2048×1024) images in street scenes. This dataset has 5,000 images with high quality pixel-wise annotations for 19 classes, which is further divided into 2975, 500, and 1525 images for training, validation and testing. To be noted, coarse data are not used in this work. Besides, more experiments on Pascal Context [14], ADE20K [66] and CamVid [3] are summarised to further prove the generality of our method.

4.1 Experiments on Cityscapes

Implementation details: We use PyTorch [44] framework to carry out following experiments. All networks are trained with the same setting, where stochastic gradient descent (SGD) with batch size of 16 is used as optimizer, with momentum of 0.9 and weight decay of 5e-4. All models are trained for 50K iterations with an initial learning rate of 0.01. As a common practice, the "poly"

Method	Stride	mIoU (%)	Δa(%)
FCN	32	71.5	-
Dilated FCN	8	72.6	1.1 ↑
+FPN	32	74.8	3.3 ↑
+FAM	32	77.2	5.7 ↑
+FPN + PPM	32	76.6	5.1 ↑
+FAM + PPM	32	**78.7**	7.2 ↑

(a) Ablation study on baseline model.

Method	$\mathbf{F}_3$	$\mathbf{F}_4$	$\mathbf{F}_5$	mIoU(%)	Δa(%)
FPN+PPM				76.6	-
	✓			76.9	0.3 ↑
		✓		77.0	0.4 ↑
			✓	77.5	0.9 ↑
		✓	✓	77.8	1.2 ↑
	✓	✓	✓	78.3	1.7 ↑

(b) Ablation study on insertion position.

Method	mIoU(%)	Δa(%)	#GFLOPs
FAM	76.4	-	-
+PPM [64]	78.3	1.9↑	123.5
+NL [50]	76.8	0.4↑	148.0
+ASPP [6]	77.6	1.2↑	138.6
+DenseASPP [52]	77.5	1.1↑	141.5

(c) Ablation study on context module.

Backbone	mIoU(%)	Δa(%)	#GFLOPs
ResNet-50 [19]	76.8	-	332.6
w/ FAM	79.2	2.4 ↑	337.1
ResNet-101 [19]	77.6	-	412.7
w/ FAM	79.8	2.2↑	417.5
ShuffleNetv2 [34]	69.8	-	17.8
w/ FAM	72.1	2.3 ↑	18.1
DF1 [29]	72.1	-	18.6
w/ FAM	74.3	2.2 ↑	18.7
DF2 [29]	73.2	-	48.2
w/ FAM	75.8	2.6 ↑	48.5

(d) Ablation on study on various backbones.

Table 1. Experiments results on network design using Cityscapes validation set.

learning rate policy is adopted to decay the initial learning rate by multiplying $(1-\frac{\text{iter}}{\text{total\_iter}})^{0.9}$ during training. Data augmentation contains random horizontal flip, random resizing with scale range of [0.75, 2.0], and random cropping with crop size of 1024 × 1024. During inference, we use the whole picture as input to report performance unless explicitly mentioned. For quantitative evaluation, mean of class-wise intersection-over-union (mIoU) is used for accurate comparison, and number of float-point operations (FLOPs) and frames per second (FPS) are adopted for speed comparison.

Comparison with baseline methods: Table 1(a) reports the comparison results against baselines on the validation set of Cityscapes [9], where ResNet-18 [19] serves as the backbone. Comparing with the naive FCN, dilated FCN improves mIoU by 1.1%. By appending the FPN decoder to the naive FCN, we get 74.8% mIoU by an improvement of 3.2%. By replacing bilinear upsampling with the proposed FAM, mIoU is boosted to 77.2%, which improves the naive FCN and FPN decoder by 5.7% and 2.4% respectively. Finally, we append PPM (Pyramid Pooling Module) [64] to capture global contextual information, which achieves the best mIoU of 78.7 % together with FAM. Meanwhile, FAM is complementary to PPM by observing FAM improves PPM from 76.6% to 78.7%.

Positions to insert FAM: We insert FAM to different stage positions in the FPN decoder and report the results as Table 1(b). From the first three rows, FAM improves all stages and gets the greatest improvement at the last stage, which demonstrate that misalignment exists in all stages on FPN and is more severe in coarse layers. This is consistent with the fact that coarse layers containing stronger semantics but with lower resolution, and can greatly boost segmentation performance when they are appropriately upsampled to high resolution. The best result is achieved by adding FAM to all stages in the last row.

Method	mIoU (%)
bilinear upsampling	78.3
deconvolution	77.9
nearest neighbor	78.2

(a) Ablation study on Upsampling operation in FAM.

Method	mIoU (%)	Gflops
$k = 1$	77.8	120.4
$k = 3$	78.3	123.5
$k = 5$	78.1	131.6
$k = 7$	78.0	140.5

(b) Ablation study on kernel size k in FAM where 3 FAMs are involved.

Method	mIoU (%)	Δa(%)
FPN +PPM	76.6	-
correlation [13]	77.2	0.6 ↑
Ours	77.5	0.9 ↑

(c) Ablation with FlowNet-C [13] in FAM.

Method	$\mathbf{F}_3$	$\mathbf{F}_4$	$\mathbf{F}_5$	mIoU(%)	Δa(%)
FPN +PPM	-	-	-	76.6	-
DCN			✓	76.9	0.3 ↑
Ours			✓	77.5	0.9 ↑
DCN	✓	✓	✓	77.2	0.6 ↑
Ours	✓	✓	✓	78.3	1.7 ↑

(d) Comparison with DCN [10].

Table 2. Experiments results on FAM design using Cityscapes validation set.

Ablation study on network architecture design: Considering current state-of-the-art contextual modules are used as heads on dilated backbone networks [6,15,52,58,64,65], we further try different contextual heads in our methods where coarse feature map is used for contextual modeling. Table 1(c) reports the comparison results, where PPM [64] delivers the best result, while more recently proposed methods such as Non-Local based heads [50] perform worse. Therefore, we choose PPM as our contextual head considering its better performance with lower computational cost. We further carry out experiments with different backbone networks including both deep and light-weight networks, where FPN decoder with PPM head is used as a strong baseline in Table 1(d). For heavy networks, we choose ResNet-50 and ResNet-101 [19] as representation. For light-weight networks, ShuffleNetv2 [34] and DF1/DF2 [29] are employed. FAM significantly achieves better mIoU on all backbones with slightly extra computational cost.

Ablation study on FAM design: We first explore the effect of upsampling in FAM in Table 2(a). Replacing the bilinear upsampling with deconvolution and nearest neighbor upsampling achieves 77.9 mIoU and 78.2 mIoU, respectively, which are similar to the 78.3 mIoU achieved by bilinear upsampling. We also try the various kernel size in Table 2(b). Larger kernel size of 5×5 is also tried which results in a similar (78.2) but introduces more computation cost. In Table 2(c), replacing FlowNet-S with correlation in FlowNet-C also leads to slightly worse results (77.2) but increases the inference time. The results show that it is enough to use lightweight FlowNet-S for aligning feature maps in FPN. In Table 2(d), we compare our results with DCN [10]. We apply DCN on the concatenated feature map of bilinear upsampled feature map and the feature map of next level. We first insert one DCN in higher layers $\mathbf{F}_5$ where our FAM is better than it. After applying DCN to all layers, the performance gap is much larger. This denotes our method can also align low level edges for better boundaries and edges in lower layers, which will be shown in visualization part.

Aligned feature representation: In this part, we give more visualization on aligned feature representation as shown in Figure 4. We visualize the upsampled

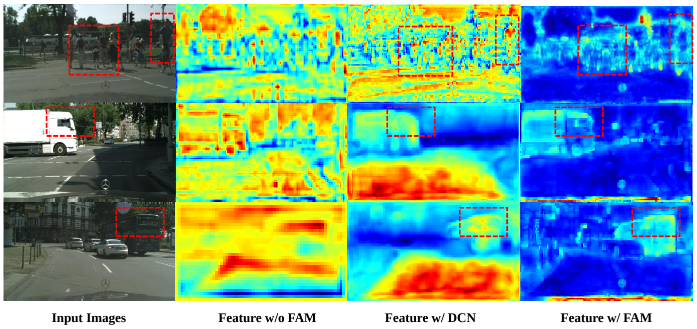


Fig. 4. Visualization of the aligned feature. Compared with DCN, our module outputs more structural feature representation.

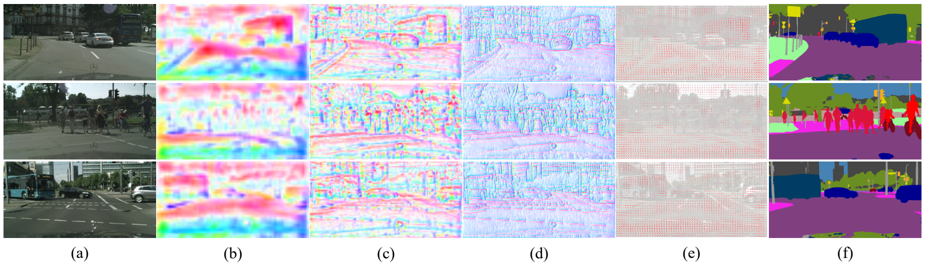


Fig. 5. Visualization of the learned semantic flow fields. Column (a) lists three exemplary images. Column (b)-(d) show the semantic flow of the three FAMs in an ascending order of resolution during the decoding process, following the same color coding of Figure 2. Column (e) is the arrowhead visualization of flow fields in column (d). Column (f) contains the segmentation results.

feature in the final stage of ResNet-18. It shows that compared with DCN [10], our FAM feature is more structural and has much more precise objects boundaries which is consistent with the results in Table 2(d). That indicates FAM is **not** an attention effect on feature similar to DCN, but actually aligns feature towards more precise shape as compared in red boxes.

Visualization of Semantic Flow: Figure 5 visualizes semantic flow from FAM in different stages. Similar with optical flow, semantic flow is visualized by color coding and is bilinearly interpolated to image size for quick overview. Besides, vector fields are also visualized for detailed inspection. From the visualization, we observe that semantic flow tends to diffuse out from some positions inside objects, where these positions are generally near object centers and have better receptive fields to activate top-level features with pure, strong semantics. Top-level features at these positions are then propagated to appropriate high-resolution positions following the guidance of semantic flow. In addition, semantic flows also have coarse-to-fine trends from top level to bottom level, which phenomenon is consistent with the fact that semantic flows gradually describe offsets between gradually smaller patterns.

Visual Improvement analysis: Figure 6(a) visualizes the prediction errors by both methods, where FAM considerably resolves ambiguities inside large objects

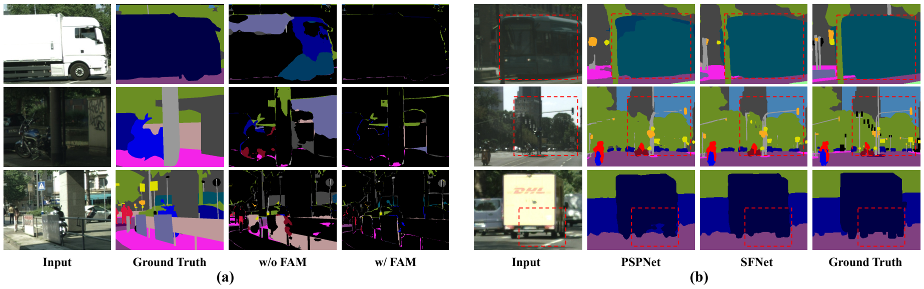


Fig. 6. (a), Qualitative comparison in terms of errors in predictions, where correctly predicted pixels are shown as black background while wrongly predicted pixels are colored with their groundtruth label color codes. (b), Scene parsing results comparison against PSPNet [64], where significantly improved regions are marked with red dashed boxes. Our method performs better on both small scale and large scale objects.

Method	InputSize	mIoU (%)	#FPS	#Params
ENet [43]	640 × 360	58.3	60	0.4M
ESPNet [36]	512 × 1024	60.3	132	0.4M
ESPNetv2 [37]	512 × 1024	62.1	80	0.8M
ERFNet [45]	512 × 1024	69.7	41.9	-
BiSeNet(ResNet-18) [53]	768 × 1536	74.6	43	12.9M
BiSeNet(Xception-39) [53]	768 × 1536	68.4	72	5.8M
ICNet [63]	1024 × 2048	69.5	34	26.5M
DF1-Seg [29]	1024 × 2048	73.0	80	8.55M
DF2-Seg [29]	1024 × 2048	74.8	55	8.55M
SwiftNet [42]	1024 × 2048	75.5	39.9	11.80M
SwiftNet-ens [42]	1024 × 2048	76.5	18.4	24.7M
DFANet [25]	1024 × 1024	71.3	100	7.8M
CellNet [62]	768 × 1536	70.5	108	-
SFNet(DF1)	1024 × 2048	**74.5**	74/121	9.03M
SFNet(DF2)	1024 × 2048	**77.8**	53/61	10.53M
SFNet(ResNet-18)	1024 × 2048	**78.9**	18/26	12.87M
SFNet(ResNet-18)†	1024 × 2048	**80.4**	18/26	12.87M

†Mapillary dataset used for pretraining.

Table 3. Comparison on Cityscapes *test* set with state-of-the-art real-time models. For fair comparison, input size is also considered, and all models use single scale inference.

(e.g., truck) and produces more precise boundaries for small and thin objects (e.g., poles, edges of wall). Figure6 (b) shows our model can better handle the small objects with shaper boundaries than dilated PSPNet due to the alignment on lower layers.

Comparison with real-time models: All compared methods are evaluated by single-scale inference and input sizes are also listed for fair comparison. Our speed is tested on one GTX 1080Ti GPU with full image resolution 1024 × 2048 as input, and we report speed of two versions, i.e., without and with TensorRT acceleration. As shown in Table 3, our method based on DF1 achieves a more accurate result(74.5%) than all methods faster than it. With DF2, our method outperforms all previous methods while running at 60 FPS. With ResNet-18 as backbone, our method achieves 78.9% mIoU and even reaches performance of accurate models which will be discussed in the next experiment. By additionally using Mapillary [40] dataset for pretraining, our ResNet-18 based model achieves

Method	Backbone	mIoU (%)	#Params	#GFLOPs[†]
SAC [61]	ResNet-101	78.1	-	-
DepthSeg [24]	ResNet-101	78.2	-	-
PSPNet [64]	ResNet-101	78.4	65.7M	1065.4
BiSeNet [53]	ResNe-18	77.7	12.3M	82.2
BiSeNet [53]	ResNet-101	78.9	51.0M	219.1
DFN [54]	ResNet-101	79.3	90.7M	1121.0
PSANet [65]	ResNet-101	80.1	85.6M	1182.6
DenseASPP [52]	DenseNet-161	80.6	35.7M	632.9
SPGNet [8]	2×ResNet-50	81.1	-	-
ANNet [69]	ResNet-101	81.3	63.0M	1089.8
CCNet [20]	ResNet-101	81.4	66.5M	1153.9
DANet [15]	ResNet-101	81.5	66.6M	1298.8
SFNet	ResNet-18	**79.5**	**12.87M**	**123.5**
SFNet	ResNet-101	**81.8**	**50.32M**	**417.5**

[†] #GFLOPs calculation adopts 1024 × 1024 image as input.

Table 4. Comparison on Cityscapes *test* set with state-of-the-art accurate models. For better accuracy, all models use multi-scale inference.

26 FPS with 80.4% mIoU, which sets the new state-of-the-art record on accuracy and speed trade-off on Cityscapes benchmark. More detailed information are in the supplementary file.

Comparison with accurate models: State-of-the-art accurate models [15,52, 64,68] perform multi-scale and horizontal flip inference to achieve better results on the Cityscapes test server. For fair comparison, we also report multi-scale with flip testing results following previous methods [15, 64]. Model parameters and computation FLOPs are also listed for comparison. Table 4 summarizes the results, where our models achieve state-of-the-art accuracy while costs much less computation. In particular, our method based on ResNet-18 is 1.1% mIoU higher than PSPNet [64] while only requiring **11%** of its computation. Our ResNet-101 based model achieves better results than DAnet [15] by 0.3% mIoU and only requires **30%** of its computation.

4.2 Experiment on More Datasets

We also perform more experiments on other three data-sets including Pascal Context [38], ADE20K [66] and CamVid [3] to further prove the effectiveness of our method. More detailed setting can be found in the supplemental file.

PASCAL Context: The results are illustrated as Table 5(a), our method outperforms corresponding baselines by 1.7% mIoU and 2.6% mIoU with ResNet-50 and ResNet-101 as backbones respectively. In addition, our method on both ResNet-50 and ResNet-101 outperforms their existing counterparts by large margins with significantly lower computational cost.

ADE20K: is a challenging scene parsing dataset. Images in this dataset are from different scenes with more scale variations. Table 5(b) reports the performance comparisons, our method improves the baselines by 1.69% mIoU and 1.59% mIoU respectively, and outperforms previous state-of-the-art methods [64, 65] with much less computation.

Method	Backbone	mIoU (%)	#GFLOPs
Ding *et al.* [12]	ResNet-101	51.6	-
EncNet [57]	ResNet-50	49.2	-
EncNet [57]	ResNet-101	51.7	-
DANet [15]	ResNet-50	50.1	186.4
DANet [15]	ResNet-101	52.6	257.1
ANNet [69]	ResNet-101	52.8	243.8
BAFPNet [11]	ResNet-101	53.6	-
EMANet [27]	ResNet-101	53.1	209.3
w/o FAM	ResNet-50	49.0	74.5
SFNet	ResNet-50	**50.7**(1.7 ↑)	75.4
w/o FAM	ResNet-101	51.1	92.7
SFNet	ResNet-101	**53.8**(2.7 ↑)	93.6

(a) Results on Pascal Context. Evaluated on 60 classes.

Method	Backbone	mIoU (%)	#GFLOPs
PSPNet [64]	ResNet-50	42.78	167.6
PSPNet [64]	ResNet-101	43.29	238.4
PSANet [65]	ResNet-101	43.77	264.9
EncNet [57]	ResNet-101	44.65	-
CFNet [58]	ResNet101	44.82	-
w/o FAM	ResNet-50	41.12	74.8
SFNet	ResNet-50	42.81(1.69 ↑)	75.7
w/o FAM	ResNet-101	43.08	93.1
SFNet	ResNet-101	44.67(1.59 ↑)	94.0

(b) Results on ADE20K.

Table 5. Experiments results on Pascal Context and ADE20k(Multi scale inference). #GFLOPs calculation adopts 480 × 480 image as input.

Method	Backbone	mIoU (%)	FPS
ICNet [63]	ResNet-50	67.1	34.5
BiSegNet [53]	Xception-39	65.6	-
BiSegNet [53]	ResNet-18	68.7	-
DFANet A [25]	-	64.7	120
DFANet B [25]	-	59.3	160
w/o FAM	DF2	67.2	139.8
SFNet	DF2	**70.4** (3.2 ↑)	134.1
SFNet	ResNet-18	**73.8**	35.5

Table 6. Accuracy and Speed comparison with previous state-of-the-art real-time models on CamVid [3] test set where the input size is 960 × 720 with single scale inference.

CamVid: is another road scene dataset. This dataset involves 367 training images, 101 validation images and 233 testing images with resolution of 960×720. We apply our method with different light-weight backbones on this dataset and report comparison results in Table 6. With DF2 as backbone, FAM improves its baseline by 3.2% mIoU. Our method based on ResNet-18 performs best with 73.8% mIoU while running at 35.5 FPS.

5 Conclusion

In this paper, we devise to use the learned **Semantic Flow** to align multi-level feature maps generated by a feature pyramid to the task of scene parsing. With the proposed flow alignment module, high-level features are well fused into low-level feature maps with high resolution. By discarding atrous convolutions to reduce computation overhead and employing the flow alignment module to enrich the semantic representation of low-level features, our network achieves the best trade-off between semantic segmentation accuracy and running time efficiency. Experiments on multiple challenging datasets illustrate the efficacy of our method.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. PAMI (2017)
2. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision **92**(1), 1–31 (Mar 2011). https://doi.org/10.1007/s11263-010-0390-2, https://doi.org/10.1007/s11263-010-0390-2
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. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. ECCV (2004)
5. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. PAMI (2018)
6. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
7. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
8. Cheng, B., Chen, L.C., Wei, Y., Zhu, Y., Huang, Z., Xiong, J., Huang, T.S., Hwu, W.M., Shi, H.: Spgnet: Semantic prediction guidance for scene parsing. In: ICCV (October 2019)
9. 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: CVPR (2016)
10. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV (2017)
11. Ding, H., Jiang, X., Liu, A.Q., Magnenat-Thalmann, N., Wang, G.: Boundary-aware feature propagation for scene segmentation (2019)
12. Ding, H., Jiang, X., Shuai, B., Qun Liu, A., Wang, G.: Context contrasted feature and gated multi-scale aggregation for scene segmentation. In: CVPR (2018)
13. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: CVPR (2015)
14. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV (2010)
15. Fu, J., Liu, J., Tian, H., Fang, Z., Lu, H.: Dual attention network for scene segmentation. arXiv preprint arXiv:1809.02983 (2018)
16. Gadde, R., Jampani, V., Gehler, P.V.: Semantic video cnns through representation warping. In: ICCV (Oct 2017)
17. He, J., Deng, Z., Qiao, Y.: Dynamic multi-scale filters for semantic segmentation. In: ICCV (October 2019)
18. He, J., Deng, Z., Zhou, L., Wang, Y., Qiao, Y.: Adaptive pyramid context network for semantic segmentation. In: CVPR (June 2019)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
20. Huang, Z., Wang, X., Huang, L., Huang, C., Wei, Y., Liu, W.: Ccnet: Criss-cross attention for semantic segmentation (2019)
21. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. ArXiv **abs/1506.02025** (2015)

22. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. ArXiv **abs/1609.02907** (2016)
23. Kirillov, A., Girshick, R., He, K., Dollar, P.: Panoptic feature pyramid networks. In: CVPR (June 2019)
24. Kong, S., Fowlkes, C.C.: Recurrent scene parsing with perspective understanding in the loop. In: CVPR (2018)
25. Li, H., Xiong, P., Fan, H., Sun, J.: Dfanet: Deep feature aggregation for real-time semantic segmentation. In: CVPR (June 2019)
26. Li, X., Yang, Y., Zhao, Q., Shen, T., Lin, Z., Liu, H.: Spatial pyramid based graph reasoning for semantic segmentation. In: CVPR (2020)
27. Li, X., Zhong, Z., Wu, J., Yang, Y., Lin, Z., Liu, H.: Expectation-maximization attention networks for semantic segmentation. In: ICCV (2019)
28. Li, X., Houlong, Z., Lei, H., Yunhai, T., Kuiyuan, Y.: Gff: Gated fully fusion for semantic segmentation. In: AAAI (2020)
29. Li, X., Zhou, Y., Pan, Z., Feng, J.: Partial order pruning: for best speed/accuracy trade-off in neural architecture search. In: CVPR (2019)
30. Li, Y., Gupta, A.: Beyond grids: Learning graph representations for visual recognition. In: NIPS (2018)
31. Li, Y., Shi, J., Lin, D.: Low-latency video semantic segmentation. In: CVPR (June 2018)
32. Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: CVPR (2017)
33. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
34. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: ECCV (September 2018)
35. Mazzini, D.: Guided upsampling network for real-time semantic segmentation. In: BMVC (2018)
36. Mehta, S., Rastegari, M., Caspi, A., Shapiro, L., Hajishirzi, H.: Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In: ECCV (September 2018)
37. Mehta, S., Rastegari, M., Shapiro, L., Hajishirzi, H.: Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In: CVPR (June 2019)
38. Mottaghi, R., Chen, X., Liu, X., Cho, N.G., Lee, S.W., Fidler, S., Urtasun, R., Yuille, A.: The role of context for object detection and semantic segmentation in the wild. In: CVPR (2014)
39. Nekrasov, V., Chen, H., Shen, C., Reid, I.: Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In: CVPR (June 2019)
40. Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P.: The mapillary vistas dataset for semantic understanding of street scenes. In: ICCV (2017)
41. Nilsson, D., Sminchisescu, C.: Semantic video segmentation by gated recurrent flow propagation. In: CVPR (June 2018)
42. 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: CVPR (June 2019)
43. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: A deep neural network architecture for real-time semantic segmentation http://arxiv.org/abs/1606.02147
44. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)

45. Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R.: Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Trans. Intelligent Transportation Systems pp. 263–272 (2018)
46. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. MICCAI (2015)
47. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV (2015)
48. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR (2016)
49. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
50. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (June 2018)
51. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV (2018)
52. Yang, M., Yu, K., Zhang, C., Li, Z., Yang, K.: Denseaspp for semantic segmentation in street scenes. In: CVPR (2018)
53. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: ECCV (2018)
54. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Learning a discriminative feature network for semantic segmentation. In: CVPR (2018)
55. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. ICLR (2016)
56. Yuan, Y., Wang, J.: Ocnet: Object context network for scene parsing. arXiv preprint arXiv:1809.00916 (2018)
57. Zhang, H., Dana, K., Shi, J., Zhang, Z., Wang, X., Tyagi, A., Agrawal, A.: Context encoding for semantic segmentation. In: CVPR (2018)
58. Zhang, H., Zhang, H., Wang, C., Xie, J.: Co-occurrent features in semantic segmentation. In: CVPR (June 2019)
59. Zhang, L., Li, X., Arnab, A., Yang, K., Tong, Y., Torr, P.H.: Dual graph convolutional network for semantic segmentation. In: BMVC (2019)
60. Zhang, L., Xu, D., Arnab, A., Torr, P.H.: Dynamic graph message passing networks. In: CVPR (2020)
61. Zhang, R., Tang, S., Zhang, Y., Li, J., Yan, S.: Scale-adaptive convolutions for scene parsing. In: ICCV (2017)
62. Zhang, Y., Qiu, Z., Liu, J., Yao, T., Liu, D., Mei, T.: Customizable architecture search for semantic segmentation. In: CVPR (June 2019)
63. Zhao, H., Qi, X., Shen, X., Shi, J., Jia, J.: Icnet for real-time semantic segmentation on high-resolution images. In: ECCV (September 2018)
64. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017)
65. Zhao, H., Zhang, Y., Liu, S., Shi, J., Change Loy, C., Lin, D., Jia, J.: Psanet: Point-wise spatial attention network for scene parsing. In: ECCV (2018)
66. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ADE20K dataset. arXiv preprint arXiv:1608.05442 (2016)
67. Zhu, X., Xiong, Y., Dai, J., Yuan, L., Wei, Y.: Deep feature flow for video recognition. In: CVPR (July 2017)

68. Zhu, Y., Sapra, K., Reda, F.A., Shih, K.J., Newsam, S., Tao, A., Catanzaro, B.: Improving semantic segmentation via video propagation and label relaxation. In: CVPR (June 2019)
69. Zhu, Z., Xu, M., Bai, S., Huang, T., Bai, X.: Asymmetric non-local neural networks for semantic segmentation. In: ICCV (2019)