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], Yunhai Tong[1]

[1] Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
[2] Huazhong University of Science and Technology, [3] DeepMotion
{lxtpku,youansheng,yhtong}@pku.edu.cn, zzhu.cs@hust.edu.cn, kuiyuanyang@deepmotion.ai

Abstract

In this paper, we focus on effective methods 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—astrous convolutions and feature pyramid fusion, are either computation intensive or ineffective. Inspired by 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 very light-weight backbone networks, such as ResNet-18. Extensive experiments are conducted on several challenging datasets, including Cityscapes, PASCAL Context, ADE20K and CamVid. Particularly, our network is the first to achieve 80.4% mIoU on Cityscapes with a frame rate of 26 FPS. The code will be available at `https://github.com/donnyyou/torchcv`.

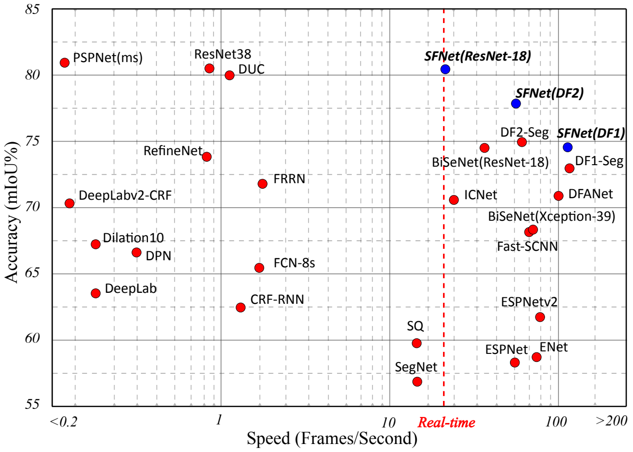


Figure 1. Inference speed versus mIoU performance on test set of Cityscapes. Previous models are marked as red points, and our models are shown in blue points which achieve the best speed/accuracy trade-off. Note that our method with light-weight ResNet-18 as backbone even achieves comparable accuracy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

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 have prominent impacts on the performance are: detailed resolution information [42] and strong semantics representation [5, 60]. The seminal work of Long *et. al.* [29] 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 built-in down-sampling pooling and convolutional layers. To alleviate this problem, many state-of-the-art methods [13, 60, 61, 65] apply atrous convolutions [52] 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.

Nevertheless, doing so inevitably requires huge 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 regular FCN using ResNet-18 [17] as the backbone network has a frame rate of 57.2 FPS for a 1024 × 2048 image, after applying atrous convolutions [52] to the network as done in [60, 61], the modified network only has a frame rate of 8.7 FPS. Moreover, under single GTX 1080Ti GPU with no ongoing programs, the state-of-art model PSPNet [60] has a frame rate of only 1.6 FPS for 1024 × 2048 input images. As a consequence, this is excessively problematic to many advanced real-world applications, such as self-driving cars and robots navigation, which desperately

*equal contribution

demand real-time online data processing.

On the other hand, recent fast models still have a large accuracy gap to accurate models, e.g., DFANet [23] only achieves 71.2% mIoU though running at 112 FPS shown in Figure 1. In summary, fast and accurate models for scene paring are demanding for real-time applications.

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 [21, 28] models which leverage the lateral path to fuse feature maps in a top-down manner. In this way, the deep features of 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, such methods [1, 42] still undergo unsatisfying accuracy issues when compared to those networks holding thick and big feature maps in the last several stages.

We believe the main reason lies in the ineffective semantics delivery from deep layers to shallow layers of these methods. To mitigate this issue, we propose to learn the **Semantic Flow** between layers with different resolutions. Semantic Flow is inspired from the optical flow method [11] used to align pixels between adjacent frames in the video processing task [64]. Based on Semantic Flow, we construct a novel network module called Flow Alignment Module(FAM) to the area of scene parsing. It takes features from adjacent levels as inputs, produces the offset field, and then warps the coarse feature to the fine feature with higher resolution according to the offset field. Because FAM effectively transmits the semantic information from deep layers to shallow layers through very simple operations, it shows superior efficacy in both improving the accuracy and keeping super efficiency. Moreover, the proposed module is end-to-end trainable, and can be plugged into any backbone networks to form new networks called **SFNet**. As depicted in Figure 1, our method with different backbones 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 Cityscapes test server with a frame rate of 26 FPS. When adopting DF2 [48] as backbone, our method achieves 77.8% mIoU with 61 FPS and 74.5% mIoU with 121 FPS when equipping the DF1 backbone. Moreover, when equipped with deeper backbone, such as ResNet-101, our method achieves comparable results with the state-of-the-art model DANet [13] and only requires 33% computation of DANet. Besides, experiments also clearly illustrate the generality of our SFNet across various datasets: Cityscapes [8], Pascal Context [34], ADE20K [62] and CamVid [3].

To conclude, our main contributions are three-fold:

- we propose a novel flow-based align module (FAM) to learn Semantic Flow between feature maps of adjacent levels and broadcast high-level features to high resolution features effectively and efficiently.
- We insert FAM into the feature pyramid framework and build a feature pyramid aligned network named 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, ADE20K and Camvid datasets. Specifically, our network achieves 80.4% mIoU on Cityscapes test server while attaining a real-time speed of 26 FPS on single GTX 1080Ti GPU.

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 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 is mostly based on astrous convolution [52], which keeps high-resolution feature maps in the latter network stages. Current state-of-the-art accurate methods [13, 60, 65] follow this paradigm and keep improving performance by designing sophisticated head networks to capture contextual information. PSPNet [60] proposes pyramid pooling module (PPM) to model multi-scale contexts, whilst DeepLab series [4–6] uses astrous spatial pyramid pooling (ASPP). In [13, 15, 16, 18, 25, 53, 66], non-local operator [46] and self-attention mechanism [45] are adopt to harvest pixel-wise context from whole image. Meanwhile, graph convolution networks [20, 26] are used to propagate information over the whole image by projecting features into an interaction space.

The second paradigm contains state-of-the-art fast methods, where high-level semantics are represented by low-resolution feature maps. A common strategy follows the same backbone networks for image classification without using astrous convolution, and fuses multi-level feature maps for both spatiality and semantics [1,29,38,42,47]. ICNet [59] uses multi-scale images as input and a cascade network to raise efficiency. DFANet [24] utilizes a light-weight backbone to speed up and is equipped with a cross-level feature aggregation to boost accuracy, while SwiftNet [38] uses lateral connections as the cost-effective solution to restore the prediction resolution while maintaining the speed. To further speed up, low-resolution image is used as input for high-level semantics [31, 59]. All these methods reduce feature maps into quite low resolution and upsample them back by a large factor, which causes inferior results especially

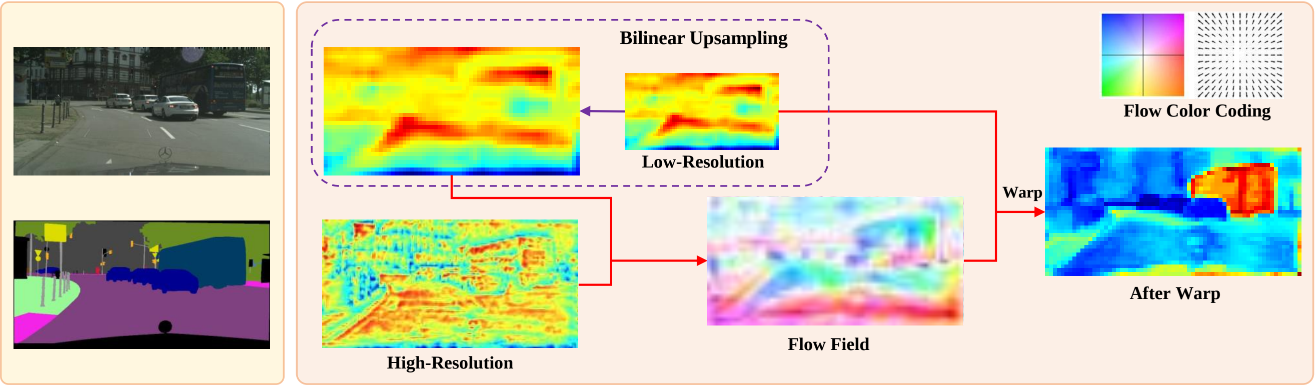


Figure 2. Visualization of feature maps and semantic flow field in FAM. Feature maps are visualized by averaging along the channel dimension, where large values are denoted by hot colors and vice versa. For visualizing semantic flow field, color code proposed by [2] and showed in top-right corner is adopted, where orientation and magnitude of flow vectors are represented by hue and saturation respectively.

for small objects and object boundaries. Guided upsampling [31] is closely related to our method, where semantic map is upsampled back to the input image size guided by the feature map from an early layer. However, the guidance is insufficient due to both the semantic and resolution gap, which make the model still incomparable to accurate models. In contrast, our method aligns feature maps from adjacent levels and enhances a feature pyramid towards both high resolution and strong semantics, and results in state-of-the-art models for both high accuracy and speed.

There are also a set of works focusing on designing light-weight networks for real-time scene parsing. ESPNets [32, 33] save computation by decomposing standard convolution into point-wise convolution and spatial pyramid of dilated convolutions. BiSeNet [50] introduces spatial path and semantic path to reduce computation. Recently, several methods [35,48,58] use auto-ML algorithms to search efficient architectures for scene parsing. Our method is complementary to these works, which will further boost the segmentation speed as demonstrated in our experiments.

Our proposed semantic flow is inspired by optical flow [11], which is widely used in video semantic segmentation for both high accuracy and speed. For accurate results, temporal information is exceedingly exploited by using optical flow. Gadde *et. al.* [14] warps internal feature maps and Nilsson *et. al.* [37] warps final semantic maps. 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 [27, 64]. Our work is different from them by propagating information hierarchically in another dimension, which is orthogonal to the temporal propagation for videos.

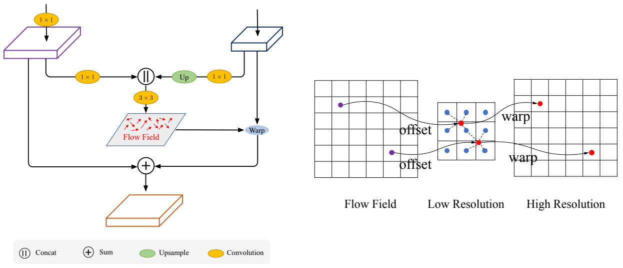
3. Method

In this section, we will first give some preliminary knowledge about scene parsing and introduce the misalignment problem therein. Then, Flow Alignment Module (FAM) is proposed 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 FPN framework [28] for fast and accurate scene parsing.

3.1. Preliminary

The task of scene parsing is to map a 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 [28], the image $\mathbf{X}$ is firstly mapped to a set of feature maps $\{\mathbf{F}_l\}_{l=2}^{5}$ by taking output of each residual block, 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 $H_l \times W_l, H_l = \frac{H}{2^l}, W_l = \frac{W}{2^l}$, and they have downsampling rates of $\{4, 8, 16, 32\}$ with respect to the input image. The coarsest feature map $\mathbf{F}_5$ comes from the deepest layer with strong semantics, FCN-32s directly do prediction on it and causes oversmoothed results without fine details, and improvements are achieved by fusing predictions from lower levels [29]. FPN takes a step further to gradually fuse high-level feature maps into low-level feature maps in a top-down pathway through 2x bilinear upsampling, it was originally proposed for object detection [28] and recently used for scene parsing [21,47].

While the whole design looks like symmetry with both downsampling encoder and upsampling decoder, there is an important issue lies in the common and simple operator, bilinear upsampling, which breaks the symmetry. Bilinear upsampling recovers the resolution of downsampled



(a) Flow Alignment Module

(b) Warp Procedure

Figure 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 (*i.e.*, offsets).

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 is far more complex. Therefore, correspondence between feature maps needs to be explicitly established to resolve their true misalignment.

3.2. Flow Alignment Module

The task is formally similar to aligning two video frames via optical flow [11], which motivates us to design a flow-based alignment module, and align feature maps of two adjacent levels by predicting a flow field. We define such flow field **Semantic Flow** which are generated between different levels in a feature pyramid. Specifically, we follow the design of FlowNet-S [11] for its efficiency. Given two adjacent feature maps $\mathbf{F}_l$ and $\mathbf{F}_{l-1}$, we first upsample $\mathbf{F}_l$ to the same size as $\mathbf{F}_{l-1}$ via bilinear interpolation, then concatenate them together for a convolutional layer using two kernels with spatial size of 3×3, and predict the semantic flow field $\Delta_{l-1} \in \mathbb{R}^{H_{l-1} \times W_{l-1} \times 2}$. Each position p_{l-1} on spatial grid Ω_{l-1} is mapped to a point p_l on the upper level l via $\frac{p_{l-1}+\Delta_{l-1}(p_{l-1})}{2}$, we then use the differentiable bilinear sampling mechanism proposed in the spatial transformer networks [19], which linearly interpolates the values of the 4-neighbors (top-left, top-right,bottom-left, and bottom-right) of p_l to approximate $\widetilde{\mathbf{F}}_l(p_{l-1})$, i.e., $\widetilde{\mathbf{F}}_l(p_{l-1}) = \mathbf{F}_l(p_l) = \sum_{p \in \mathcal{N}(p_l)} w_p \mathbf{F}_l(p)$. This process is shown in Figure 3(b). A similar strategy is used in [63] for self-supervised monodepth learning via view synthesis. The proposed module is light-weight and end-to-end trainable and 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 structured than normal bilinear upsampled feature and leads to more consistent representation inner the objects like bus and car.

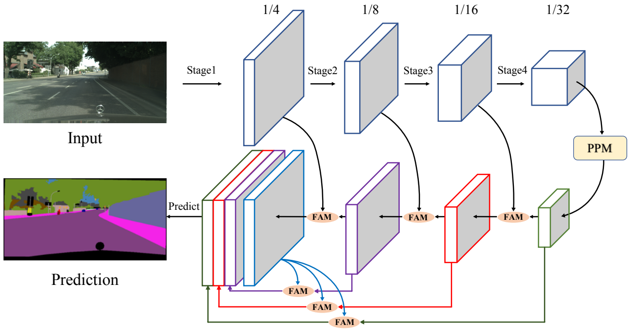


Figure 4. Overview of our proposed SFNet. ResNet-18 backbone with four stages is used for exemplar illustration. FAM: Flow Alignment Module. PPM: Pyramid Pooling Module [60].

3.3. Network Architectures

Figure 4 illustrates the whole network architecture, which contains a bottom-up pathway as encoder and a top-down pathway as decoder, the encoder has a backbone same as image classification by replacing fully connected layers with contextual modeling module, and the decoder is a FPN equipped with FAM. Details of each part are described as follows.

Backbone We choose standard networks pretrained from ImageNet [43] for image classification as our backbone network by removing the last fully connected layer. Specifically, ResNet series [17], ShuffleNet v2 [30] and DF series [48] are used in our experiments. All backbones have four stages with residual blocks, and each stage has stride 2 in the first convolution layer to downsample the feature map for both computational efficiency and larger receptive fields.

Contextual Module plays an important role in scene parsing to capture long-range contextual information [54, 60], and we adopt Pyramid Pooling Module (PPM) [60] in this work. Since PPM outputs the same resolution feature map as last residual module, we treat PPM and last residual module together as last stage for FPN. Other modules like ASPP [5] can be readily plugged into our architecture in a similar manner, which are also verified in the Experiment section.

Aligned FPN Decoder takes feature maps from the encoder and uses the aligned feature pyramid for final scene parsing. By replacing normal bilinear upsampling with FAM in the top-down pathway of FPN [28], $\{\mathbf{F}_l\}_{l=2}^4$ 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. For scene parsing, $\{\widetilde{\mathbf{F}}_l\}_{l=2}^4 \cup \{\mathbf{F}_5\}$ are upsampled 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 upsampling operations with the proposed FAM.

Cascaded Deeply Supervised Learning We use deeply supervised loss [60] to supervise intermediate outputs of the decoder for easier optimization. In addition, online hard example mining [44, 50] is used by only training on the 10% hardest pixels sorted by cross-entropy loss.

4. Experiment

We first carry out experiments on the Cityscapes [8] dataset, which is comprised of a large, diverse set of high-resolution (2048×1024) images recorded 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, 20,000 coarsely labeled images provided by this dataset are not used in this work. Besides, more experiments on Pascal Context [12], ADE20K [62] and CamVid [3] are summarised to further prove the effectiveness of our method.

4.1. Experiments on Cityscapes

Implementation details: We use PyTorch [40] 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" 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 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. Most ablation studies are conducted on the validation set, and we also compare our method with other state-of-the-art methods on the test set.

Comparison with baseline methods: Table 1 reports the comparison results against baselines on the validation set of Cityscapes [8], where ResNet-18 [17] 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) [60] 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%.

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

Table 1. Ablation study on baseline methods, where ResNet-18 serves as the backbone.

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

Table 2. Ablation study on different positions to insert FAM, where $\mathbf{F}_l$ denote the upsampling position between level l and level $l-1$, ResNet-18 with FPN decoder and PPM head serves as baseline.

Positions to insert FAM: We insert FAM to different stage positions in the FPN decoder and report the results as Table 2. 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 phenomenon 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 performance is achieved by adding FAM to all stages as listed in the last row.

Ablation study on different contextual heads: Considering current state-of-the-art contextual modules are used as heads on dilated backbone networks [5,13,49,56,60,61], we further try different contextual heads in our methods where coarse feature map is used for contextual modeling. Table 3 reports the comparison results, where PPM [60] delivers the best result, while more recently proposed methods such as Non-Local based heads [18, 46, 53] perform worse. Therefore, we choose PPM as our contextual head considering its better performance with lower computational cost.

Ablation study on different backbones: We further carry

Method	mIoU(%)	Δa	#GFLOPs
FAM	76.4	-	-
+PPM [60]	78.3	1.9↑	246.5
+NL [46]	76.8	0.4↑	295.5
+ASPP [5]	77.6	1.2↑	276.6
+DenseASPP [49]	77.5	1.1↑	282.3

Table 3. Ablation with different contextual modeling heads, where ResNet-18 with FAM serves as the baseline.

Backbone	mIoU(%)	Δa	#GFLOPs
ResNet-50 [17]	76.8	-	664.7
w/ FAM	79.2	2.4 ↑	673.1
ResNet-101 [17]	77.6	-	824.9
w/ FAM	79.8	2.2↑	833.5
ShuffleNetv2 [30]	69.8	-	35.3
w/ FAM	72.1	2.3 ↑	35.9
DF1 [48]	72.1	-	36.8
w/ FAM	74.3	2.2 ↑	36.9
DF2 [48]	73.2	-	96.0
w/ FAM	75.8	2.6 ↑	96.5

Table 4. Ablation study on different backbones, where FPN decoder with PPM head is used as baseline. The top part compare deep networks with sliding window testing, and the bottom part compares light-weight networks using single view testing.

out a set of experiments with different backbone networks including both deep and light-weight networks, where FPN decoder with PPM head is used as a strong baseline. For heavy networks, we choose ResNet-50 and ResNet-101 [17] as representation. For light-weight networks, ShuffleNetv2 [30] and DF1/DF2 [48] are experimented. All these backbones are pretrained on ImageNet [43]. Table 4 reports the results, where FAM significantly achieves better mIoU on all backbones with only slightly extra computational cost.

Visualization of Semantic Flow: Figure 5 visualizes semantic flow from FAM in different stages. Similar with traditional optical flow, semantic flow is visualized by color coding and is bilinearly interpolated to image size for quick overview. Besides, vector field is also visualized for detailed inspection. From the visualization, we observe that semantic flow tend to converge to 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.

Improvement analysis: Table 5 compares the detailed results of each category on the validation set, where ResNet-101 is used as backbone, and FPN decoder with PPM head serves as the baseline. Our method improves almost all categories, especially for 'truck' with more than 19% mIoU improvement. Figure 6 visualizes the prediction errors by both methods, where FAM considerably resolves ambiguities inside large objects (e.g., truck) and produces more precise boundaries for small and thin objects (e.g., poles, edges of wall).

Comparison with PSPNet: We compare our segmentation results with previous state-of-the-art model PSPNet [60] using ResNet-101 as backbone. We re-implement PSPNet using open-source code provided by the author and achieve 78.8% mIoU on validation set. Based on the same backbone ResNet-101 without using astrous convolution, our method achieves 79.8% mIoU while being about **3 times faster** than PSPNet. Figure 7 shows the comparison results, where our model gets more consistent results for large objects and keeps more detailed information benefited from the well fused multi-level feature pyramid in our decoder.

Comparison with state-of-the-art 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 6, our method based on DF1 achieves 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 [36] dataset for pretraining, our ResNet-18 based model achieves 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 about Mapillary pretraining and TensorRT acceleration can be referred in supplementary file.

Comparison with state-of-the-art accurate models: State-of-the-art accurate models [13, 49, 60, 65] perform multi-scale and horizontal flip inference to achieve better results on the Cityscapes test server. Although our model can run fast in real-time scenario with single-scale inference, for fair comparison, we also report multi-scale with flip testing results, which is common settings following previous methods [13,60]. Number of model parameters and computation FLOPs are also listed for comparison. Table 7 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 [60] while only requiring 11% of its computation. Our ResNet-101 based model achieves comparable results with DAnet [13] and only requires 32% of its computation.

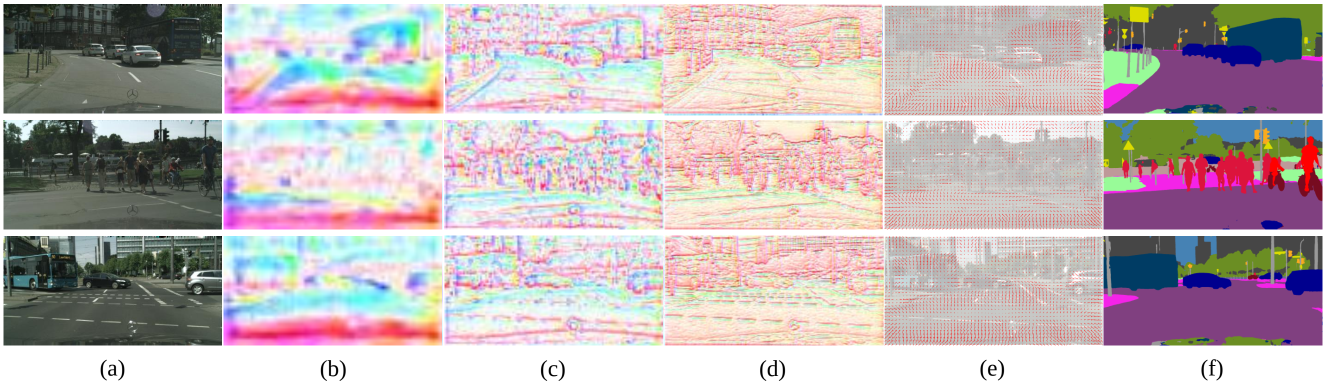


Figure 5. Visualization of the learned semantic flow fields. Column (a) lists three exemplar images. Column (b)-(d) show the semantic flow of the three FAMs in an ascending order of resolution during the decoding process, following the same color coding of Figure 2. Column (e) is the arrowhead visualization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

Method	road	swalk	build	wall	fence	pole	tlight	sign	veg.	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
BaseLine	98.1	84.9	92.6	54.8	62.2	66.0	72.8	80.8	92.4	60.6	94.8	83.1	66.0	94.9	65.9	83.9	70.5	66.0	78.9	77.6
w/ FAM	98.3	85.9	93.2	62.2	67.2	67.3	73.2	81.1	92.8	60.5	95.6	83.2	65.0	95.7	84.1	89.6	75.1	67.7	78.8	79.8

Table 5. Quantitative per-category comparison results on Cityscapes validation set, where ResNet-101 backbone with the FPN decoder and PPM head serves as the strong baseline. Sliding window crop with horizontal flip is used for testing. Obviously, FAM boosts the performance of almost all the categories.

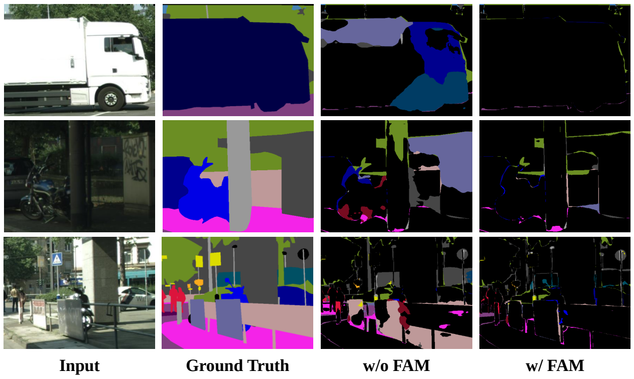


Figure 6. Qualitative comparison in terms of errors in predictions, where correctly predicted pixels are shown as black background while wrongly predicted pixels are colored with their groundtruth label color codes.

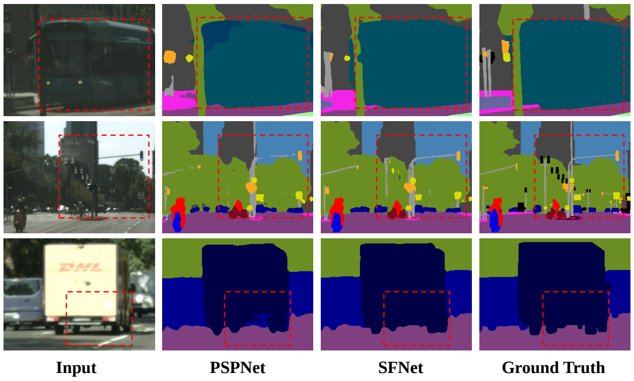


Figure 7. Scene parsing results comparison against PSPNet [60], where significantly improved regions are marked with red dashed boxes. Our method performs better on both small scale and large scale objects.

4.2. Experiment on More Datasets

To further prove the effectiveness of our method, we perform more experiments on other three data-sets including Pascal Context [34], ADE20K [62] and CamVid [3]. Standard settings of each benchmark are used, which are summarized in supplementary file.

PASCAL Context: provides pixel-wise segmentation annotation for 59 classes, and contains 4,998 training images and 5,105 testing images. The results are illustrated as Table 8, 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 significant lower computational cost.

ADE20K: is a challenging scene parsing dataset annotated with 150 classes, and it contains 20K/2K images for training and validation. Images in this dataset are from different scenes with more scale variations. Table 9 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 [60, 61] with much less computation.

Method	InputSize	mIoU (%)	#FPS	#Params
ENet [39]	640 × 360	58.3	60	0.4M
ESPNet [32]	512 × 1024	60.3	132	0.4M
ESPNetv2 [33]	512 × 1024	62.1	80	0.8M
ERFNet [41]	512 × 1024	69.7	41.9	-
BiSeNet(ResNet-18) [50]	768 × 1536	74.6	43	12.9M
BiSeNet(Xception-39) [50]	768 × 1536	68.4	72	5.8M
ICNet [59]	1024 × 2048	69.5	34	26.5M
DF1-Seg [48]	1024 × 2048	73.0	80	8.55M
DF2-Seg [48]	1024 × 2048	74.8	55	8.55M
SwiftNet [38]	1024 × 2048	75.5	39.9	11.80M
SwiftNet-ens [38]	1024 × 2048	76.5	18.4	24.7M
DFANet [23]	1024 × 1024	71.3	100	7.8M
CellNet [58]	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	13.47M
SFNet(ResNet-18)†	1024 × 2048	**80.4**	18/26	13.47M

†Mapillary dataset used for pretraining.

Table 6. 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.

Method	Backbone	mIoU (%)	#Params	#GFLOPs†
SAC [57]	ResNet-101	78.1	-	-
DepthSeg [22]	ResNet-101	78.2	-	-
PSPNet [60]	ResNet-101	78.4	65.7M	2128.2
BiSeNet [50]	ResNe-18	77.7	12.3M	
BiSeNet [50]	ResNet-101	78.9	51.0M	437.4
DFN [51]	ResNet-101	79.3	90.7M	2241.3
PSANet [61]	ResNet-101	80.1	85.6M	2364.4
DenseASPP [49]	DenseNet-161	80.6	35.7M	1265.2
SPGNet [7]	2×ResNet-50	81.1	-	-
ANNet [66]	ResNet-101	81.3	63.0M	2178.7
CCNet [18]	ResNet-101	81.4	66.5M	2307.0
DANet [13]	ResNet-101	81.5	66.6M	2596.8
SFNet	ResNet-18	**79.5**	13.47M	246.5
SFNet	ResNet-101	**81.3**	50.32M	833.5

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

Table 7. Comparison on Cityscapes *test* set with state-of-the-art accurate models. For better accuracy, all models use multi-scale inference.

CamVid: is another road scene dataset for autonomous driving. This dataset involves 367 training images, 101 validation images and 233 testing images with resolution of 480×360. We apply our method with different light-weight backbones on this dataset and report comparison results in Table 10. With DF2 as backbone, FAM improves its baseline by 3.4% mIoU. Our method based on ResNet-18 performs best with 72.4% mIoU while running at 45.2 FPS.

Method	Backbone	mIoU (%)	#GFLOPs†
PSPNet [60]	ResNet-101	49.8	334.7
Ding *et al.* [10]	ResNet-101	51.6	-
EncNet [55]	ResNet-50	49.2	-
EncNet [55]	ResNet-101	51.7	-
DANet [13]	ResNet-50	50.1	372.5
DANet [13]	ResNet-101	52.6	513.7
ANNet [66]	ResNet-101	52.8	487.0
BAFPNet [9]	ResNet-101	53.6	-
CFNet [56]	ResNet-101	54.1	-
EMANet [25]	ResNet-101	53.1	418.1
w/o FAM	ResNet-50	49.8	148.6
SFNet	ResNet-50	**51.5**(1.7 ↑)	150.4
w/o FAM	ResNet-101	52.0	185.0
SFNet	ResNet-101	**54.6**(2.6 ↑)	186.8

† #GFLOPs calculation adopts 480 × 480 image as input. To be noted, we only compute the Flops for methods with open-sourced codes.

Table 8. Comparison with the state-of-art methods on Pascal Context testing set [34]. All the models use multi-scale inference with horizontal flip.

Method	Backbone	mIoU (%)	#GFLOPs†
PSPNet [60]	ResNet-50	42.78	335.0
PSPNet [60]	ResNet-101	43.29	476.3
PSANet [61]	ResNet-101	43.77	529.3
EncNet [55]	ResNet-101	44.65	-
CFNet [56]	ResNet101	44.82	-
w/o FAM	ResNet-50	41.12	149.3
SFNet	ResNet-50	42.81(1.69 ↑)	151.1
w/o FAM	ResNet-101	43.08	185.7
SFNet	ResNet-101	44.67(1.59 ↑)	187.5

† #GFLOPs calculation adopts 480 × 480 image as input.

Table 9. Results on ADE20K dataset, where our models achieve the best trade-off on speed and accuracy, all models are evaluated using multi-scale inference with horizontal flip.

Method	Backbone	mIoU (%)	FPS
SegNet [1]	-	55.6	29.4
ENet [39]	-	51.3	61.2
ICNet [59]	ResNet-50	67.1	34.5
BiSegNet [50]	Xception-39	65.6	-
BiSegNet [50]	ResNet-18	68.7	-
DFANet A [24]	-	64.7	120
DFANet B [24]	-	59.3	160
w/o FAM	DF2	64.5	155.1
SFNet	DF2	67.9(3.4 ↑)	153.8
SFNet	ResNet-18	**72.4**	45.2

Table 10. Accuracy and efficiency comparison with previous state-of-the-art real-time models on CamVid [3] test set, where the input size is 360 × 480 and single scale inference is used.

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 flowed to 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. Since our network is super efficient and shares the same spirit as optical flow for aligning different maps (*i.e.*, feature maps of different video frames), it can be naturally extended to video semantic segmentation to align feature maps hierarchically and temporally. Besides, we're also interested in extending the idea of semantic flow to other related areas like panoptic segmentation, *etc*.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *PAMI*, 2017.

[2] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, Mar 2011.

[3] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, xx(x):xx–xx, 2008.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *PAMI*, 2018.

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[7] Bowen Cheng, Liang-Chieh Chen, Yunchao Wei, Yukun Zhu, Zilong Huang, Jinjun Xiong, Thomas S. Huang, Wen-Mei Hwu, and Honghui Shi. Spgnet: Semantic prediction guidance for scene parsing. In *ICCV*, October 2019.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[9] Henghui Ding, Xudong Jiang, Ai Qun Liu, Nadia Magnenat-Thalmann, and Gang Wang. Boundary-aware feature propagation for scene segmentation. 2019.

[10] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *CVPR*, 2018.

[11] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *CVPR*, 2015.

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[13] Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. *arXiv preprint arXiv:1809.02983*, 2018.

[14] Raghudeep Gadde, Varun Jampani, and Peter V. Gehler. Semantic video cnns through representation warping. In *ICCV*, Oct 2017.

[15] Junjun He, Zhongying Deng, and Yu Qiao. Dynamic multi-scale filters for semantic segmentation. In *ICCV*, October 2019.

[16] Junjun He, Zhongying Deng, Lei Zhou, Yali Wang, and Yu Qiao. Adaptive pyramid context network for semantic segmentation. In *CVPR*, June 2019.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[18] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. 2019.

[19] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *ArXiv*, abs/1506.02025, 2015.

[20] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ArXiv*, abs/1609.02907, 2016.

[21] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollar. Panoptic feature pyramid networks. In *CVPR*, June 2019.

[22] Shu Kong and Charless C. Fowlkes. Recurrent scene parsing with perspective understanding in the loop. In *CVPR*, 2018.

[23] Hanchao Li, Pengfei Xiong, Haoqiang Fan, and Jian Sun. Dfanet: Deep feature aggregation for real-time semantic segmentation. In *CVPR*, June 2019.

[24] Hanchao Li, Pengfei Xiong, Haoqiang Fan, and Jian Sun. Dfanet: Deep feature aggregation for real-time semantic segmentation. In *CVPR*, June 2019.

[25] Xia Li, Zhisheng Zhong, Jianlong Wu, Yibo Yang, Zhouchen Lin, and Hong Liu. Expectation-maximization attention networks for semantic segmentation. In *ICCV*, 2019.

[26] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. In *NIPS*. 2018.

[27] Yule Li, Jianping Shi, and Dahua Lin. Low-latency video semantic segmentation. In *CVPR*, June 2018.

[28] Tsung-Yi Lin, Piotr Dollr, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[29] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[30] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, September 2018.

[31] Davide Mazzini. Guided upsampling network for real-time semantic segmentation. In *BMVC*, 2018.

[32] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *ECCV*, September 2018.

[33] Sachin Mehta, Mohammad Rastegari, Linda Shapiro, and Hannaneh Hajishirzi. Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In *CVPR*, June 2019.

[34] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[35] Vladimir Nekrasov, Hao Chen, Chunhua Shen, and Ian Reid. Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In *CVPR*, June 2019.

[36] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

[37] David Nilsson and Cristian Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. In *CVPR*, June 2018.

[38] Marin Orsic, Ivan Kreso, Petra Bevandic, and Sinisa Segvic. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In *CVPR*, June 2019.

[39] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation.

[40] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[41] Eduardo Romera, Jose M. Alvarez, Luis Miguel Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Trans. Intelligent Transportation Systems*, pages 263–272, 2018.

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015.

[43] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[44] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[46] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, June 2018.

[47] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

[48] Zheng Pan Jiashi Feng Xin Li, Yiming Zhou. Partial order pruning: for best speed/accuracy trade-off in neural architecture search. In *CVPR*, 2019.

[49] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *CVPR*, 2018.

[50] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018.

[51] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. In *CVPR*, 2018.

[52] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.

[53] Yuhui Yuan and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018.

[54] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, June 2018.

[55] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.

[56] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. In *CVPR*, June 2019.

[57] Rui Zhang, Sheng Tang, Yongdong Zhang, Jintao Li, and Shuicheng Yan. Scale-adaptive convolutions for scene parsing. In *ICCV*, 2017.

[58] Yiheng Zhang, Zhaofan Qiu, Jingen Liu, Ting Yao, Dong Liu, and Tao Mei. Customizable architecture search for semantic segmentation. In *CVPR*, June 2019.

[59] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, September 2018.

[60] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[61] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018.

[62] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *arXiv preprint arXiv:1608.05442*, 2016.

[63] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

[64] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *CVPR*, July 2017.

[65] Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *CVPR*, June 2019.

[66] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *ICCV*, 2019.