DeepLabv3+: Encoder-Decoder with Atrous Separable Convolution
(2018 - ECCV)

Liang-Chieh Chen et al.
Resume

March 4, 2019

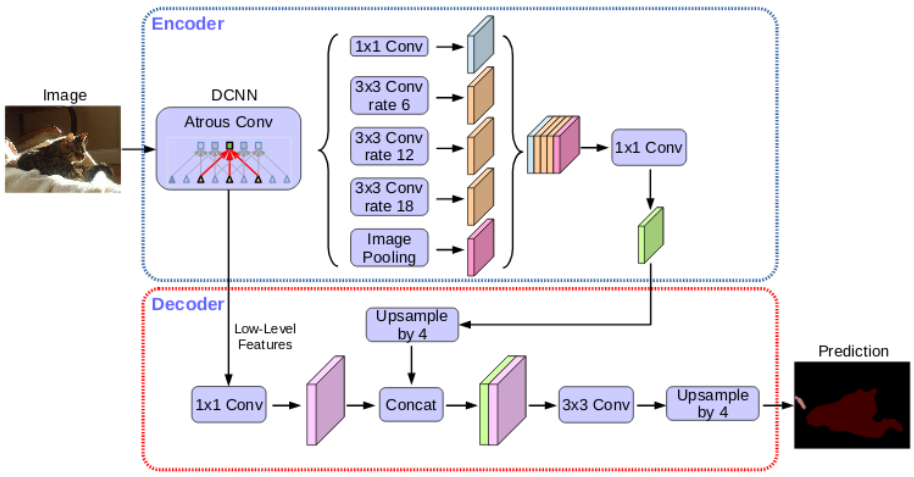
---

1 Introduction

In this new version of deeplab, the authors contributions are:

- A novel encoder-decoder structure which employs DeepLabv3 as a powerful encoder module and a simple yet effective decoder module.
- The possibility to arbitrarily control the resolution of extracted encoder features by atrous convolution to trade-off precision and runtime, which is not possible with existing encoder-decoder models.
- Using the Xception model as a backbone for the segmentation task and apply depthwise separable convolution to both ASPP module and decoder module, resulting in a faster and stronger encoder-decoder network.

2 The model



2.1 Atrous convolutions

With a variable size of rates (number of zeros between two non zero elements of the filter), and convolutions becom atrous convolutions, by using different rates of dilation, we can adjust the filter's

field of view and capture multiscale information without the need to use pooling operations and reduce the spatial dimensions if the feature maps.

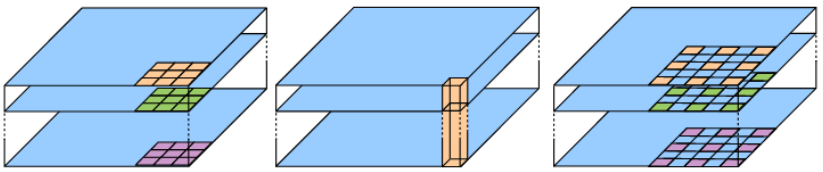


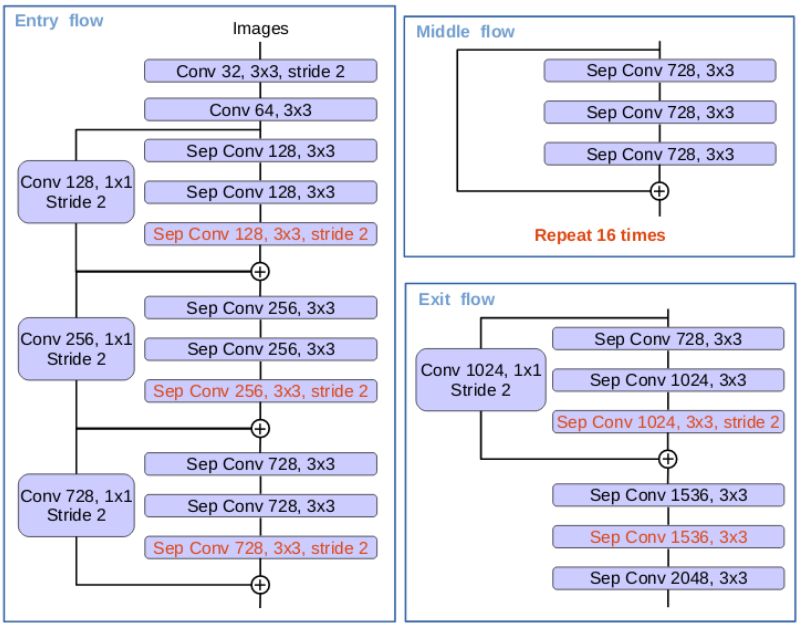
Fig. 3. 3×3 Depthwise separable convolution decomposes a standard convolution into (a) a depthwise convolution (applying a single filter for each input channel) and (b) a pointwise convolution (combining the outputs from depthwise convolution across channels). In this work, we explore *atrous separable convolution* where atrous convolution is adopted in the depthwise convolution, as shown in (c) with $rate = 2$.

2.2 Depthwise separable convolution

with depthwise convolutions, we represent a convolution with a depth wise convolutions (for CxHxW input feature maps, will use C (FxF) filters, each filter will be appield to only one channel of the input feature maps), followed by apoint wise convolution (1x1 convolution) to take into consideration the computed features in the chennels dimension.

2.3 Encoder

For the encoder we take the a backbone, either resnet of a modifed version of Xception (aligned exception), for resnet, without any modifications we have an ouput stride of 32, if we want an ouput stride of 16 we replace the convolution of the fourth resnet block with a stride of 2 with a convolution with a stride of one and a dilation of 2, and if we want an output stride of 8, we replace the last strided convolutions of the last two blocks with a 1 strided convolutions and a dilation of 2 and 4 respectively.



For Xception, if we want an output stride of 16, the model presented in the figure bellow remains unchanged, the only note is that the last three seperable convolution in the exit flow have a dilation of 2, now if we want an output stride of 8, we'll need to remove the striding in the first block in the exit flow, have a dilation of two for all the seperable convolutions in the middle flow, and then

have a dilation of 2 for block in the exit flow and a dilation rate of 4 for the last three convolutions in the exit flow.

After getting the output of the backbone architecture, we have our ASPP module taking as input the 256 x H/16 x W/16 (in case of resnet), this volume is passed through 1x1 convolution, three 3x3 convolutions of dilation rates : 6, 12, 18 (this is in case of an output stride of 16, it is 12, 24, 36 if the output stride equals 8), and we apply an average pooling over the upcomming volume and we upsample it to the H/16 x W/16, all the five outputs are then concatenated and passed through a 1x1 convolution to adjust the depth from 256x5 to 256.

2.4 Decoder

In the decoder, we obtain the output of the atrous spatial pyramid pooling with a depth of 256, we upsample it with a rate of 4, and we also have the low level features that have a depth of 256 and spatial dimension of H/4 x W/4, these low level feature are extracted from the first block of the resnet, and after the first block in Xception with a depth of 128, and these low level feature are passed through 1x1 convoltutions to reduce the depth from 256/128 ot 48, and are then concatenated with the upsampled outputs of the ASPP and passed through 3x3 convolution and are then upsamled to the size of the input image.

3 Experiments

Encoder train OS	Encoder eval OS	Decoder	MS	Flip	mIOU	Multiply-Adds
16	16				77.21%	81.02B
16	8				78.51%	276.18B
16	8		✓		79.45%	2435.37B
16	8		✓	✓	79.77%	4870.59B
16	16	✓			78.85%	101.28B
16	16	✓	✓		80.09%	898.69B
16	16	✓	✓	✓	80.22%	1797.23B
16	8	✓			79.35%	297.92B
16	8	✓	✓		80.43%	2623.61B
16	8	✓	✓	✓	80.57%	5247.07B
32	32				75.43%	52.43B
32	32	✓			77.37%	74.20B
32	16	✓			77.80%	101.28B
32	8	✓			77.92%	297.92B

Table 3. Inference strategy on the PASCAL VOC 2012 *val* set using *ResNet-101*. **train OS**: The *output stride* used during training. **eval OS**: The *output stride* used during evaluation. **Decoder**: Employing the proposed decoder structure. **MS**: Multi-scale inputs during evaluation. **Flip**: Adding left-right flipped inputs.

Encoder train OS	Encoder eval OS	Decoder	MS	Flip	SC	COCO	JFT	mIOU	Multiply-Adds
16	16							79.17%	68.00B
16	16		✓					80.57%	601.74B
16	16		✓	✓				80.79%	1203.34B
16	8							79.64%	240.85B
16	8		✓					81.15%	2149.91B
16	8		✓	✓				81.34%	4299.68B
16	16	✓						79.93%	89.76B
16	16	✓	✓					81.38%	790.12B
16	16	✓	✓	✓				81.44%	1580.10B
16	8	✓						80.22%	262.59B
16	8	✓	✓					81.60%	2338.15B
16	8	✓	✓	✓				81.63%	4676.16B
16	16	✓			✓			79.79%	54.17B
16	16	✓	✓	✓	✓			81.21%	928.81B
16	8	✓			✓			80.02%	177.10B
16	8	✓	✓	✓	✓			81.39%	3055.35B
16	16	✓			✓	✓		82.20%	54.17B
16	16	✓	✓	✓	✓	✓		83.34%	928.81B
16	8	✓			✓	✓		82.45%	177.10B
16	8	✓	✓	✓	✓	✓		83.58%	3055.35B
16	16	✓			✓	✓	✓	83.03%	54.17B
16	16	✓	✓	✓	✓	✓	✓	84.22%	928.81B
16	8	✓			✓	✓	✓	83.39%	177.10B
16	8	✓	✓	✓	✓	✓	✓	84.56%	3055.35B

Table 5. Inference strategy on the PASCAL VOC 2012 *val* set when using modified *Xception*. **train OS**: The *output stride* used during training. **eval OS**: The *output stride* used during evaluation. **Decoder**: Employing the proposed decoder structure. **MS**: Multi-scale inputs during evaluation. **Flip**: Adding left-right flipped inputs. **SC**: Adopting depthwise separable convolution for both ASPP and decoder modules. **COCO**: Models pretrained on MS-COCO. **JFT**: Models pretrained on JFT.

Method	**mIOU**
Deep Layer Cascade (LC) [82]	82.7
TuSimple [77]	83.1
Large_Kernel_Matters [60]	83.6
Multipath-RefineNet [58]	84.2
ResNet-38_MS_COCO [83]	84.9
PSPNet [24]	85.4
IDW-CNN [84]	86.3
CASIA_IVA_SDN [63]	86.6
DIS [85]	86.8
DeepLabv3 [23]	85.7
DeepLabv3-JFT [23]	86.9
DeepLabv3+ (Xception)	87.8
DeepLabv3+ (Xception-JFT)	89.0

Table 6. PASCAL VOC 2012 *test* set results with top-performing models.