Fast-SCNN: Fast Semantic Segmentation Network

Rudra PK Poudel  
Stephan Liwicki  
Roberto Cipolla

Cambridge Research Laboratory  
Toshiba Research Europe, UK

BMVC 2019
Real-time Semantic Image Segmentation

- *What* am I seeing and *where* is it?
- Real-time perception is critical for autonomous systems

Decision Support System in ADAS
Real-time Semantic Image Segmentation

- *What* am I seeing and *where* is it?
- Real-time perception is critical for autonomous systems
Motivation

**Problem:** SOTA models are accurate but resource hungry
- Compute: floating point ops
- Power consumption
- Memory

**Observations:**
1. First few layers of DCNN extract low-level features (Zeiler et al., 2014)
2. Larger receptive field (context) is important for accuracy (Poudel et al., 2018)
3. Spatial details is necessary to preserve boundary (Shelhamer et al. 2016)
4. SOTA efficient models adapt multi-resolution and multi-branch architecture
Motivation: First Few Layers Learn Low-level Features

Zeiler et al., ECCV 2014
Motivation: Importance of Larger Receptive Field

Deep Network for Context

Shallow Network for Spatial Detail

Convolution Block

Bottleneck Residual Block

Depth-wise Separable Convolution Block

Feature Fusion Unit

ContextNet (Poudel et al., BMVC 2018)
Motivation: Importance of Spatial Details

U-Net (Ronneberger et al., MICCAI 2015)
Motivation: Efficient Multi-resolution Architectures

ICNet (Zhao et al., ECCV 2018).
Motivation

**Problem:** SOTA models are accurate but resource hungry
- Compute: floating point ops
- Power consumption
- Memory

**Observations:**
1. First few layers of DCNN extract low-level features (Zeiler et al., 2014)
2. Larger receptive field (context) is important for accuracy (Poudel et al., 2018)
3. Spatial details is necessary to preserve boundary (Shelhamer et al. 2016)
4. SOTA efficient models adapt multi-resolution and multi-branch architecture
Proposed Model: Overview

- **Hypothesis**: jointly learn the low level features of multi-branch networks to increase the model efficiency.

- **Learning to Down-sample** jointly learns the low level features
Proposed Model: Learning to Down-sample

- **Learning to Down-sample** sharing computation of multi-resolution branches improves efficiency

- No need for multiple resizes and memory copies of the original input
**Proposed Model: Larger Receptive Field**

- **Going deeper with convnet** Fast-SCNN can be reduced to convnet

- Early sub-sampling/max-pooling layers increase receptive field and efficiency
Proposed Model: Skip-Connection

- **Spatial details** skip-connection helps to recover boundary information

- We preferred simple feature fusion module i.e. addition only
Proposed Model: Fast-SCNN

- **Deeper path** at low resolution captures global context information
- **Shallow path** focuses on high resolution segmentation details

No need to learn low-level features separately

Quantization, network pruning and other techniques are also applicable
Proposed Model: Qualitative Validation

Input image

Skip-Connection: No

Skip-Connection: Yes
Proposed Model: Qualitative Validation

Input image

Skip-Connection: No

Skip-Connection: Yes
Fast-SCNN: Quantitative Evaluation

![Graph showing accuracy vs. runtime for different methods]

- **Our Fast-SCNN**
- **Fast-SCNN**
- **BiSeNet**
- **ContextNet**
- **ICNet**
- **ERFNet**
- **ENet**
- **Real-time**
- **ENet**
- **Other Methods**

* Nvidia Titan Xp (Pascal); Others Nvidia Titan X (Maxwell)
Fast-SCNN: Quantitative Evaluation

- Fast-SCNN balances accuracy and speed

<table>
<thead>
<tr>
<th>Model</th>
<th>Class mIoU%</th>
<th>Category mIoU%</th>
<th>Params in Millions</th>
<th>FPS on 1024x2048</th>
</tr>
</thead>
<tbody>
<tr>
<td>SegNet</td>
<td>56.1</td>
<td>79.8</td>
<td>29.46</td>
<td>1.6</td>
</tr>
<tr>
<td>ENet</td>
<td>58.3</td>
<td>80.4</td>
<td>0.37</td>
<td>20.4</td>
</tr>
<tr>
<td>ICNet</td>
<td>69.5</td>
<td>-</td>
<td>6.68</td>
<td>30.3</td>
</tr>
<tr>
<td>ERFNet</td>
<td>68.0</td>
<td>86.5</td>
<td>2.1</td>
<td>11.2</td>
</tr>
<tr>
<td>ContextNet</td>
<td>66.1</td>
<td>82.7</td>
<td>0.85</td>
<td>41.9</td>
</tr>
<tr>
<td>BiSeNet*</td>
<td>71.4</td>
<td>-</td>
<td>5.8</td>
<td>57.3</td>
</tr>
<tr>
<td>GUN*</td>
<td>70.4</td>
<td>-</td>
<td>-</td>
<td>33.3</td>
</tr>
<tr>
<td><strong>Fast-SCNN</strong></td>
<td><strong>68.0</strong></td>
<td><strong>84.7</strong></td>
<td><strong>1.11</strong></td>
<td><strong>123.5</strong></td>
</tr>
</tbody>
</table>

* Nvidia Titan Xp (Pascal); Others Nvidia Titan X (Maxwell)
Fast-SCNN: Input Size Variation

- Fast-SCNN is efficient on smaller as well as larger scale input sizes

<table>
<thead>
<tr>
<th>Input Size</th>
<th>Class mIoU%</th>
<th>Frame-Per-Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024 × 2048</td>
<td>68.0</td>
<td>123.5</td>
</tr>
<tr>
<td>512 × 1024</td>
<td>62.8</td>
<td>285.8</td>
</tr>
<tr>
<td>256 × 512</td>
<td>51.9</td>
<td>485.4</td>
</tr>
</tbody>
</table>
Is ImageNet Pre-Training is Necessary?

- Total number of gradient updates is important
- At least in validation and test sets ImageNet pre-training is not important!
- Similar finding on Rethinking ImageNet Pre-training by He et al. (ICCV 2019)

<table>
<thead>
<tr>
<th>Model</th>
<th>Class mIoU%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast-SCNN</td>
<td>68.62</td>
</tr>
<tr>
<td>Fast-SCNN + ImageNet</td>
<td>69.15</td>
</tr>
<tr>
<td>Fast-SCNN + Coarse</td>
<td>69.22</td>
</tr>
<tr>
<td>Fast-SCNN + Coarse + ImageNet</td>
<td>69.19</td>
</tr>
</tbody>
</table>

R. Poudel et al. (CRL) Fast-SCNN: Fast Semantic Segmentation Network
Conclusion

- **Fast-SCNN** is
  - memory, computation and power efficient
  - twice as fast as other state-of-the-art models
  - above real-time i.e. 123.5 fps on 1024×2048 images
  - efficient and competitive on smaller as well as larger scale input sizes

- We have shown accuracy without ImageNet pre-training is comparable

- Limitations: accuracy gap with bigger off-line models

- Future work: apply to depth estimation and instance segmentation


Public implementations on PyTorch and TensorFlow are available on Github!

Thank you!