Approaches to Solving Semantic Segmentation

Sergey Rusakov

Lehigh University

September 25, 2019
Problem description
Problem description
Problem description
Problem description
Problem description
Problem description
Problem description
Problem description

(a) Image classification

(b) Object localization

(c) Semantic segmentation

(d) Instance segmentation
Optimisation opportunities

\[ \mathcal{X} \xrightarrow{f} \mathcal{Y} \]

- \( \mathcal{X} \) is a set of three dimensional matrices, \( \mathcal{X} \subseteq [0,1]^{n \times m \times 3} \)
- \( \mathcal{Y} \) is a set of three dimensional matrices, \( \mathcal{Y} \subseteq \mathcal{C}^{n \times m} \), where \( \mathcal{C} \) is the set, that can be tailored to the specific needs, but generally somehow represents a collection of classes
  - \( \mathcal{C} = \{1, 2, \ldots, C\} \), where \( C \) is the number of classes
  - \( \mathcal{C} = \{e_1, e_2, \ldots, e_C\} \), where \( C \) is the number of classes and \( e_i \) is a one-hot vector with 1 at position \( i \)
  - \( \mathcal{C} = \{e_1, e_2, \ldots, e_C\} \cup [0,1]^3 \)
Optimisation usually consists of variants of minimizing sum of pixel-wise cross entropy with your favorite first order method

$$\frac{1}{N} \min_w \sum_{i=1}^N \sum_p CE_p(\hat{f}_w(x_i), y_i)$$

where $p$ goes through all pixels
Obvious optimisation opportunities

Instead of blindly playing with architectures, one can play with regularization

- Border smoothness
- Number of connected regions
- Any possible intuition about the desired result
### Data from 2017 survey [2]

<table>
<thead>
<tr>
<th>Name and Reference</th>
<th>Purpose</th>
<th>Year</th>
<th>Classes</th>
<th>Data</th>
<th>Resolution</th>
<th>Sequence</th>
<th>Synthetic/Real</th>
<th>Samples (training)</th>
<th>Samples (validation)</th>
<th>Samples (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASCAL VOC 2012 Segmentation [27]</td>
<td>Generic</td>
<td>2012</td>
<td>21</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>1461</td>
<td>1439</td>
<td>Private</td>
</tr>
<tr>
<td>PASCAL Context [28]</td>
<td>Generic</td>
<td>2014</td>
<td>540 (59)</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>10103</td>
<td>N/A</td>
<td>9637</td>
</tr>
<tr>
<td>PASCAL Part [29]</td>
<td>Generic</td>
<td>2014</td>
<td>20</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>10103</td>
<td>N/A</td>
<td>9637</td>
</tr>
<tr>
<td>SBD [30]</td>
<td>Generic</td>
<td>2011</td>
<td>21</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>8498</td>
<td>2857</td>
<td>N/A</td>
</tr>
<tr>
<td>Microsoft COCO [31]</td>
<td>Generic</td>
<td>2014</td>
<td>140</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>82793</td>
<td>40504</td>
<td>81434</td>
</tr>
<tr>
<td>SYNTHIA [32]</td>
<td>Urban</td>
<td>2016</td>
<td>11</td>
<td>2D</td>
<td>960 x 720</td>
<td>x</td>
<td>S</td>
<td>13497</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Cityscapes (fine) [33]</td>
<td>Urban</td>
<td>2015</td>
<td>30 (8)</td>
<td>2D</td>
<td>2048 x 1024</td>
<td>x</td>
<td>R</td>
<td>2975</td>
<td>500</td>
<td>1525</td>
</tr>
<tr>
<td>CamVid [34]</td>
<td>Urban</td>
<td>2015</td>
<td>30 (8)</td>
<td>2D</td>
<td>2048 x 1024</td>
<td>x</td>
<td>R</td>
<td>22973</td>
<td>500</td>
<td>N/A</td>
</tr>
<tr>
<td>CamVid-Sturgio [35]</td>
<td>Urban</td>
<td>2012</td>
<td>3</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>923</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>KITTI-Layout [36]</td>
<td>Urban</td>
<td>2015</td>
<td>11</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>170</td>
<td>N/A</td>
<td>46</td>
</tr>
<tr>
<td>KITTI-Ros [38]</td>
<td>Urban</td>
<td>2015</td>
<td>11</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>140</td>
<td>N/A</td>
<td>112</td>
</tr>
<tr>
<td>KITTI-Zhong [39]</td>
<td>Urban</td>
<td>2015</td>
<td>10</td>
<td>2D/3D</td>
<td>1226 x 570</td>
<td>x</td>
<td>R</td>
<td>701</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Stanford background [40]</td>
<td>Outdoor</td>
<td>2009</td>
<td>8</td>
<td>2D</td>
<td>320 x 240</td>
<td>x</td>
<td>R</td>
<td>725</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>SiftFlow [41]</td>
<td>Outdoor</td>
<td>2011</td>
<td>33</td>
<td>2D</td>
<td>256 x 256</td>
<td>x</td>
<td>R</td>
<td>2685</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Youtube-Objects-Jain [42]</td>
<td>Objects</td>
<td>2014</td>
<td>10</td>
<td>2D</td>
<td>480 x 360</td>
<td>x</td>
<td>R</td>
<td>10107</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Adobe’s Portrait Segmentation [26]</td>
<td>Portrait</td>
<td>2016</td>
<td>2</td>
<td>2D</td>
<td>600 x 800</td>
<td>x</td>
<td>R</td>
<td>1500</td>
<td>300</td>
<td>N/A</td>
</tr>
<tr>
<td>MISC [43]</td>
<td>Materials</td>
<td>2015</td>
<td>23</td>
<td>2D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>7061</td>
<td>2500</td>
<td>5000</td>
</tr>
<tr>
<td>DAVIS [44]</td>
<td>Indoor</td>
<td>2016</td>
<td>4</td>
<td>2D</td>
<td>480p</td>
<td>x</td>
<td>R</td>
<td>4219</td>
<td>2023</td>
<td>2180</td>
</tr>
<tr>
<td>NYUv2 [46]</td>
<td>Indoor</td>
<td>2012</td>
<td>40</td>
<td>2.5D</td>
<td>480 x 640</td>
<td>x</td>
<td>R</td>
<td>791</td>
<td>654</td>
<td>N/A</td>
</tr>
<tr>
<td>SUN3D [47]</td>
<td>Indoor</td>
<td>2013</td>
<td>37</td>
<td>2.5D</td>
<td>610 x 480</td>
<td>x</td>
<td>R</td>
<td>196410</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>SUNRGBD [48]</td>
<td>Indoor</td>
<td>2015</td>
<td>37</td>
<td>2.5D</td>
<td>Variable</td>
<td>x</td>
<td>R</td>
<td>2406</td>
<td>260</td>
<td>5050</td>
</tr>
<tr>
<td>RGB-D Object Dataset [49]</td>
<td>Household objects</td>
<td>2011</td>
<td>51</td>
<td>2.5D</td>
<td>640 x 480</td>
<td>x</td>
<td>R</td>
<td>207620</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>ShapeNet Part [50]</td>
<td>Object/Part</td>
<td>2016</td>
<td>16/50</td>
<td>3D</td>
<td>N/A</td>
<td>x</td>
<td>S</td>
<td>31963</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Stanford 2D-3D-S [51]</td>
<td>Indoor</td>
<td>2017</td>
<td>13</td>
<td>2D/2.5D/3D</td>
<td>1080 x 1080</td>
<td>x</td>
<td>R</td>
<td>71469</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3D Mesh [52]</td>
<td>Object/Part</td>
<td>2009</td>
<td>19</td>
<td>3D</td>
<td>N/A</td>
<td>x</td>
<td>S</td>
<td>380</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Sydney Urban Objects Dataset [53]</td>
<td>Urban (Objects)</td>
<td>2013</td>
<td>26</td>
<td>3D</td>
<td>N/A</td>
<td>x</td>
<td>R</td>
<td>41</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Large-Scale Point Cloud Classification Benchmark [54]</td>
<td>Urban/Nature</td>
<td>2016</td>
<td>8</td>
<td>3D</td>
<td>N/A</td>
<td>x</td>
<td>R</td>
<td>15</td>
<td>N/A</td>
<td>15</td>
</tr>
</tbody>
</table>
### Research horizons

#### Approaches to Solving Semantic Segmentation

<table>
<thead>
<tr>
<th>Name and Reference</th>
<th>Architecture</th>
<th>Targets</th>
<th>Accuracy</th>
<th>Efficiency</th>
<th>Training</th>
<th>Instance</th>
<th>Sequences</th>
<th>Multi-modal</th>
<th>3D</th>
<th>Source Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Convolutional Network [68]</td>
<td>VGG-16(FCN)</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
</tr>
<tr>
<td>SegNet [66]</td>
<td>VGG-16 + Decoder</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
</tr>
<tr>
<td>Bayesian SegNet [67]</td>
<td>SegNet</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
</tr>
<tr>
<td>DeepLab [68]</td>
<td>VGG-16/RooNet-101</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
<td>✔️ ✔️ ✔️</td>
</tr>
<tr>
<td>MINC-CNN [43]</td>
<td>GoogleLeNet(FCN)</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>CRNet [70]</td>
<td>VGG-16</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Dilatation [71]</td>
<td>VGG-16</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>ENet [72]</td>
<td>ENet bottleneck</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Multi-scale-CNN-Ra [73]</td>
<td>VGG-16(FCN)</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Multi-scale-CNN-Eigen [74]</td>
<td>Custom</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Multi-scale-CNN-RoY [75]</td>
<td>Multi-scale-CNN-Eigen</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Multi-scale-CNN-Bin [76]</td>
<td>FCN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>ParseNet [77]</td>
<td>VGG-16</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>ReSeg [78]</td>
<td>VGG-16 + ReNet</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>LSTM-CF [79]</td>
<td>Fast R-CNN + DeepMask</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>2D-LSTM [80]</td>
<td>MDRNN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>c-CNN [81]</td>
<td>MDRNN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>DAG-RNN [82]</td>
<td>Elman network</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>SDS [10]</td>
<td>R-CNN + Box CNN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>DeepMask [83]</td>
<td>VGG-A</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>SharpMask [84]</td>
<td>DeepMask</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>MultiPathNet [85]</td>
<td>Fast R-CNN + DeepMask</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Huang-3DCNN [86]</td>
<td>Own 3DCNN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>PointNet [87]</td>
<td>Own MLP-3D</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>Clockwork ConvNet [88]</td>
<td>FCN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>3DCNN-Zhang</td>
<td>Own 3DCNN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
<tr>
<td>End2End Vox2Vox [89]</td>
<td>CSN</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
<td>✗ ✗ ✗</td>
</tr>
</tbody>
</table>

**Contribution(s):**
- Encoder-decoder
- Uncertainty modeling
- Standalone CRF, atrous convolutions
- Patchwise CNN, Standalone CRF
- CRF reformulated as RNN
- Dilated convolutions
- Bottleneck module for efficiency
- Multi-scale architecture
- Multi-scale sequential refinement
- Multi-scale coarse-to-fine refinement
- Independently trained multi-scale FCNs
- Global context feature fusion
- Extension of RaNet to semantic segmentation
- Fusion of contextual information from multiple sources
- Image context modelling
- Different input sizes, image context
- Graph image structure for context modelling
- Simultaneous detection and segmentation
- Proposals generation for segmentation
- Top-down refinement module
- Multi path information flow through network
- 3DCNN for voxelized point clouds
- Segmentation of unordered point sets
- Clockwork scheduling for sequences
- 3D convolutions and graph cut for sequences
- 3D convolutions/deconvolutions for sequences
Notable detail - architectures are often modular, significant parts are just borrowed from classification context

- Is it worth to consider architecture search over vast blocks instead of individual weights/layers?
- Focus computational resources on connecting structures
- Even simple automatisation of exhaustive search over large architecture blocks can be beneficial, considering the plethora of existing results (more of a commercial opportunity; TensorFlow might already include the functionality)
Specifically for pixel-wise segmentation, initial intended use for road/roadside segmentation

- Novel encoder-decoder architecture (at the time)
- Very simple method overall
SegNet, 2016

Input
RGB Image

Convolutional Encoder-Decoder

Pooling Indices

Output
Segmentation

Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax
Encoder is VGG16 without dense layers

Decoder uses saved indices from max-pooling with subsequent convolutions to restore the original size

Removal of dense layers allows to literally use the CE pixel sum objective without resorting to training on separate regions

Modular structure of the network allows for a more detailed analysis of decoder structure
SegNet, 2016

Convolution with trainable decoder filters

Max-pooling for upsampling

Deconvolution for upsampling

Dimensionality reduction

Encoder feature map

SegNet

FCN

$x_1$, $x_2$, $x_3$, $x_4$, $x_5$, $x_6$, $x_7$, $x_8$, $x_9$, $x_{10}$, $x_{11}$, $x_{12}$, $x_{13}$, $x_{14}$, $x_{15}$, $x_{16}$, $y_1$, $y_2$, $y_3$, $y_4$, $y_5$, $y_6$, $y_7$, $y_8$, $y_9$, $y_{10}$, $y_{11}$, $y_{12}$, $y_{13}$, $y_{14}$, $y_{15}$, $y_{16}$
SegNet, although attempting to restore image information during decoding, still loses some of it.

As mentioned, one way to try and improve the quality of the segmentation regions is maybe an addition of informed regularisation.

Another way is to make an informed choice of the architecture, trying to save low/mid/high level feature information along all levels of processing.
RefineNet, 2017

(a) 1/4 → 1/8 → 1/16 → 1/32 → ResNet

(b) 1/4 → 1/8 → 1/8 → 1/8 → 1/8 → Dilated convolutions
ResNet convolutional architecture is motivated by lowering computational resources requirements.

- Dilated convolution saves the resolution of an image, but still requires storage of large amounts of filter application results.
- We want the benefit of both without disadvantages of any.
RefineNet, 2017

Multi-Path Refinement

Prediction

Sergey Rusakov
Approaches to Solving Semantic Segmentation
Having architecture, that is empirically proven to be good is still not enough.

Sometimes the problem comes from class under-representation in data.

This can be argued to be an even more serious problem, than searching for architectures.
We will modify the initial formulation through change in $\mathcal{X}$

Now $\mathcal{X}$ is a set of **triplets** of three dimensional matrices, $(x_T, x_S, B_S)$, where we want to obtain a segmentation of image $x_T$ and $x_S$ serves as "support" image, with $B_S$ being its binary segmentation mask.

Value of the support image comes from the fact, that it can be taken from underrepresented class and, combined with input, still achieve good segmentation result on barely seen classes.

For $\mathcal{Y}$, we are only concerned with segmentation of a single object, so just a binary mask.
Approaches to Solving Semantic Segmentation

Sergey Rusakov

CANet, 2018
Aside from the change in the problem formulation, significant parts of the architecture are borrowed. ASPP is a module from DeepLabV3, which serves the same purpose as structure of RefineNet. Only further supports the idea about "large scale" architecture search.
Paper, again, deals with the issue of insufficient data for training

Considered case is video with sparsely annotated frames

Proposed solution is to use video prediction tools to simultaneously predict frames and labels
Automatic Data Augmentation, 2019

- Obvious idea - use existing frame prediction methods to predict future frames and apply the result on labels
- Particular implementation predicts \((u, v)\) translation of the pixel in the frame and then applies this translation to corresponding label pixel
- Since we have access to all frames, we then pair known frames and label prediction
- Approach encounters some problems
Solution - pair **predicted** labels with **predicted** frames

Predicted frames might be incorrect, but labeling will be more in line with them, which is our goal, when augmenting a data set

We can even *condition our predictive model on future frames*, since the only information, that we don’t have is label assignment; turns prediction into reconstruction
Still, if want to construct labels even for several frames into the future, we need to deal with severe artifacts of the prediction model.

Proposed solution - instead of maximising a probability of one class for pixels, which are placed on the border between objects, we will maximise the joint probability of labels, corresponding to these classes.

Surprisingly, paper shows, that this helps, which allows authors to use up to 5 frames into past and future, effectively multiplying the size of the data set by 10.
Automatic Data Augmentation, 2019

Approaches to Solving Semantic Segmentation
Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. 
*CoRR*, abs/1511.00561, 2015.

Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and José García Rodríguez. A review on deep learning techniques applied to semantic segmentation. 

*CoRR*, abs/1611.06612, 2016.
Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen.  
Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning.  

Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn D. Newsam, Andrew Tao, and Bryan Catanzaro.  
Improving semantic segmentation via video propagation and label relaxation.  