Fully Convolutional Networks for Semantic Segmentation

Jonathan Long, Evan Shelhamer and Trevor Darrell (2014)

CSE 291 Presentation by Nicolas Jourdan
Outline

I. Semantic Segmentation
II. Approaches
III. FCN
   I. Classifiers for dense Prediction
   II. Concepts for Upsampling
   III. Model Architecture(s)
IV. Results and Analysis
   I. Strengths and Weaknesses
V. Follow Ups
Previously

• Image classification

• Object detection

Figure adapted from Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
Semantic Segmentation

Example

• Label each pixel with class

• Example applications in self-driving cars, medicine, SELFIES

Figures adapted from Stanford CS231n
Recent (2 Days old) Use: Youtube

Figure from CityScapes Dataset
Semantic Segmentation Challenge

Global Reasoning

Locally accurate Boundaries
Semantic Segmentation Approaches

Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014
Semantic Segmentation Approaches

What could be a Problem?

Figure adapted from Stanford cs231n
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014
Semantic Segmentation Approaches

What could be a Problem?
- Very inefficient
- No shared computation between overlapping patches

Figure adapted from Stanford cs231n
FCN Architecture

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Adapting classifiers for dense prediction

Used Classifiers „Base Nets“: VGG, GoogLeNet, AlexNet

Class Prediction: „Tabby Cat“

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Adapting classifiers for dense prediction

convolution

227 × 227  55 × 55  27 × 27  13 × 13  1 × 1

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Adapting classifiers for dense prediction

convolution

H x W  H/4 x W/4  H/8 x W/8  H/16 x W/16  H/32 x W/32

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture

Adapting classifiers for dense prediction

convolution

H × W
H/4 × W/4
H/8 × W/8
H/16 × W/16
H/32 × W/32
H × W

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Adapting classifiers for dense prediction

convolution

H × W  H/4 × W/4  H/8 × W/8  H/16 × W/16  H/32 × W/32  H × W

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture

Problem: Coarse Predictions

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Problem: Coarse Predictions

Slide adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture

How to get from coarse to dense predictions

• „Shift and Stitch“
• Decrease Subsampling
• Dilated Convolution / „à trous“ Trick (next Paper)
• Interpolation
• Deconvolution/ Transpose Convolution
Shift and Stitch
proposed in OverFeat (2013)

Max. Pooling Example

Shift and Stitch
proposed in OverFeat (2013)

Max. Pooling
Example

Shift the input and interlace the output
Shift and Stitch
proposed in OverFeat (2013)

Max. Pooling Example
Shift the input and interlace the output

Potential Downside?
Shift and Stitch
proposed in OverFeat (2013)

Max. Pooling
Example

Shift the input and
interlace the output

Potential Downside?
- Computation cost
  is increased by $f^2$
  ($f =$ Factor of Downsampling)

FCN Model Architecture
Upsampling/Deconvolution

Recall: Normal 3 x 3 convolution, stride 2 pad 1

Dot product between filter and input

Filter moves 2 pixels in the input for every one pixel in the output

Stride gives ratio between movement in input and output

Input: 4 x 4
Output: 2 x 2

FCN Model Architecture

Upsampling/Deconvolution

3 x 3 transpose convolution, stride 2 pad 1

Input gives weight for filter

Input: 2 x 2

Output: 4 x 4

FCN Model Architecture
Upsampling/Deconvolution

3 x 3 transpose convolution, stride 2 pad 1

Other names:
- Deconvolution (bad)
- Upconvolution
- Fractionally strided convolution
- Backward strided convolution

Input gives weight for filter

Sum where output overlaps

Filter moves 2 pixels in the output for every one pixel in the input

Stride gives ratio between movement in output and input

Input: 2 x 2
Output: 4 x 4

FCN Model Architecture
How to get from coarse to dense predictions

• „Shift and Stitch“
• Decrease Subsampling
• Dilated Convolution / „à trous“ Trick (next Paper)
• Interpolation

• Deconvolution/ Transpose Convolution
  • Initialized to bilinear interpolation
FCN Model Architecture

Performance after upsampling

• Performance is ok but results are still coarse
FCN Model Architecture

Combining what and where

• Fuse coarse semantic information with local appearance

→ Skip connections

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/ and Stanford cs231n
FCN Model Architecture
Combining what and where

• Fuse coarse semantic information with local appearance
  → Skip connections

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
FCN Model Architecture
Combining what and where

• Fuse coarse semantic information with local appearance
→ Skip connections

Process:
1. 1x1 Convolution on pool4 output why?
2. Crop to align with upsampled prediction
3. Elementwise Sum
4. Upsample to target size
FCN Model Architecture
Combining what and where

• Fuse coarse semantic information with local appearance

→ Skip connections
FCN Model Architecture
Combining what and where

• Fuse coarse semantic information with local appearance

→ Skip connections

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
Training the FCN model(s)

- Classifiers (VGG, AlexNet, GoogLeNet) pretrained on ImageNet
- Zero-Initialization of class-scoring layer
- Bilinear-Interpolation initialization for upsampling
  - Final Layer fixed, intermediates fine tuned
- Stochastic Gradient Descent (SGD) with Momentum
- No Class-Balancing or data augmentation
- Multinomial Logistic Loss (Cross Entropy)
Results
Metrics

- **pixel accuracy**: $\frac{\sum_i n_{ii}}{\sum_i t_i}$
- **mean accuracy**: $(1/n_{cl}) \sum_i n_{ii}/t_i$
- **mean IU**: $(1/n_{cl}) \sum_i n_{ii}/\left(t_i + \sum_j n_{ji} - n_{ii}\right)$
- **frequency weighted IU**: $\frac{\sum_k t_k}{\sum_i t_i n_{ii}}/\left(t_i + \sum_j n_{ji} - n_{ii}\right)$

$n_{ij}$
Number of pixels of class $i$ with predicted class $j$

$n_{cl}$
Number of classes

$t_i$
Total number of pixels of class $i$

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
Results

Architecture Comparison

**FCN-32 with different Classifiers**
(No Skip Connections) on PASCAL VOC 2011 VAL

<table>
<thead>
<tr>
<th></th>
<th>FCN-AlexNet</th>
<th>FCN-VGG16</th>
<th>FCN-GoogLeNet$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean IU</td>
<td>39.8</td>
<td><strong>56.0</strong></td>
<td>42.5</td>
</tr>
<tr>
<td>forward time</td>
<td>50 ms</td>
<td>210 ms</td>
<td>59 ms</td>
</tr>
<tr>
<td>conv. layers</td>
<td>8</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>parameters</td>
<td>57M</td>
<td>134M</td>
<td>6M</td>
</tr>
<tr>
<td>rf size</td>
<td>355</td>
<td>404</td>
<td>907</td>
</tr>
<tr>
<td>max stride</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
</tbody>
</table>

**FCN w/o Skip connections**
on subset of PASCAL VOC 2011 VAL

<table>
<thead>
<tr>
<th></th>
<th>pixel acc.</th>
<th>mean acc.</th>
<th>mean IU</th>
<th>f.w. IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCN-32s-fixed</td>
<td>83.0</td>
<td>59.7</td>
<td>45.4</td>
<td>72.0</td>
</tr>
<tr>
<td>FCN-32s</td>
<td>89.1</td>
<td>73.3</td>
<td>59.4</td>
<td>81.4</td>
</tr>
<tr>
<td>FCN-16s</td>
<td>90.0</td>
<td>75.7</td>
<td>62.4</td>
<td>83.0</td>
</tr>
<tr>
<td>FCN-8s</td>
<td><strong>90.3</strong></td>
<td><strong>75.9</strong></td>
<td><strong>62.7</strong></td>
<td><strong>83.2</strong></td>
</tr>
</tbody>
</table>

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
Results
PASCAL VOC 2011/2012 Testset

<table>
<thead>
<tr>
<th>Method</th>
<th>mean IU VOC2011 test</th>
<th>mean IU VOC2012 test</th>
<th>inference time</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN [12]</td>
<td>47.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SDS [16]</td>
<td>52.6</td>
<td>51.6</td>
<td>~ 50 s</td>
</tr>
<tr>
<td>FCN-8s</td>
<td>62.7</td>
<td>62.2</td>
<td>~ 175 ms</td>
</tr>
</tbody>
</table>

1112/8498 Training Images
20+1 Classes

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
Results

NYUDv2 → Multi-Modal Input

• Improvement by the use of depth information from Kinect Camera

• What is HHA?

<table>
<thead>
<tr>
<th></th>
<th>pixel acc.</th>
<th>mean acc.</th>
<th>mean IU</th>
<th>f.w. IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gupta et al. [14]</td>
<td>60.3</td>
<td>-</td>
<td>28.6</td>
<td>47.0</td>
</tr>
<tr>
<td>FCN-32s RGB</td>
<td>60.0</td>
<td>42.2</td>
<td>29.2</td>
<td>43.9</td>
</tr>
<tr>
<td>FCN-32s RGBD</td>
<td>61.5</td>
<td>42.4</td>
<td>30.5</td>
<td>45.5</td>
</tr>
<tr>
<td>FCN-32s HHA</td>
<td>57.1</td>
<td>35.2</td>
<td>24.2</td>
<td>40.4</td>
</tr>
<tr>
<td>FCN-32s RGB-HHA</td>
<td>64.3</td>
<td>44.9</td>
<td>32.8</td>
<td>48.0</td>
</tr>
<tr>
<td>FCN-16s RGB-HHA</td>
<td><strong>65.4</strong></td>
<td><strong>46.1</strong></td>
<td><strong>34.0</strong></td>
<td><strong>49.5</strong></td>
</tr>
</tbody>
</table>

1449 Images (795 Train/ 654 Test) 40 Classes

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/

Results

NYUDv2 → Multi-Modal Input

• Improvement by the use of depth information from Kinect Camera

• What is HHA?

• RGB-HHA Fused in Upsampling layer:

\[
\text{n.score\_fused} = \text{L.Eltwise(n.score\_fr\_color, n.score\_fr\_hha,}
\]
\[
\text{operation=P.Eltwise.SUM, coeff=[0.5, 0.5])}
\]
\[
\text{n.upscore} = \text{L.Deconvolution(n.score\_fused,}
\]
\[
\text{convolution\_param=\text{dict(num\_output=40, kernel\_size=64, stride=32, bias\_term=False)},}
\]
\[
\text{param=[\text{dict(lr\_mult=0})]}}
\]

<table>
<thead>
<tr>
<th></th>
<th>pixel acc.</th>
<th>mean acc.</th>
<th>mean IU</th>
<th>f.w. IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gupta et al. [14]</td>
<td>60.3</td>
<td>-</td>
<td>28.6</td>
<td>47.0</td>
</tr>
<tr>
<td>FCN-32s RGB</td>
<td>60.0</td>
<td>42.2</td>
<td>29.2</td>
<td>43.9</td>
</tr>
<tr>
<td>FCN-32s RGBD</td>
<td>61.5</td>
<td>42.4</td>
<td>30.5</td>
<td>45.5</td>
</tr>
<tr>
<td>FCN-32s HHA</td>
<td>57.1</td>
<td>35.2</td>
<td>24.2</td>
<td>40.4</td>
</tr>
<tr>
<td>FCN-32s RGB-HHA</td>
<td>64.3</td>
<td>44.9</td>
<td>32.8</td>
<td>48.0</td>
</tr>
<tr>
<td>FCN-16s RGB-HHA</td>
<td>65.4</td>
<td>46.1</td>
<td>34.0</td>
<td>49.5</td>
</tr>
</tbody>
</table>

1449 Images (795 Train/654 Test)
40 Classes

Results
SIFT Flow

• 33 Semantic Categories
• 3 Geometric Categories
• Two-Headed FCN-16s
  • Separate Prediction and Loss

2688 Images (2488 Train/ 200 Test)
36 Classes

<table>
<thead>
<tr>
<th></th>
<th>pixel acc.</th>
<th>mean acc.</th>
<th>mean IU</th>
<th>f.w. IU</th>
<th>geom. acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liu et al. [23]</td>
<td>76.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Tighe et al. [33]</td>
<td>-</td>
<td>75.6</td>
<td>41.1</td>
<td>-</td>
<td>90.8</td>
</tr>
<tr>
<td>Tighe et al. [34]</td>
<td>78.6</td>
<td>77.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Tighe et al. [34]</td>
<td>50.8</td>
<td>72.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Farabet et al. [8]</td>
<td>79.5</td>
<td>78.5</td>
<td>29.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Farabet et al. [8]</td>
<td>29.8</td>
<td>77.7</td>
<td>29.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pinheiro et al. [28]</td>
<td></td>
<td>85.2</td>
<td>51.7</td>
<td>39.5</td>
<td>76.1</td>
</tr>
<tr>
<td>FCN-16s</td>
<td>94.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figures adapted from paper author: http://people.eecs.berkeley.edu/~jonlong/
Strengths

• Milestone work

• Efficient use of classifier CNNs for dense prediction

• Inputs of arbitrary size

• Beats previous state-of-the-art

• Relatively simple architecture
Weaknesses

• Training
  • Staged Training Process (FCN-16, FCN-8)
  • End-to-End Training Difficult

• (Dense) Predictions
  • High Resolution Feature Maps are not used for Skip Connections
  • Small Decoder Network (0.5m Parameters vs 134m in encoder)

• Computational Cost
  • Does not run in real time
Improvements / Follow-Ups

• Later approaches usually keep general idea: Feature Extraction + Upsampling but focus on Decoder
  • Encoder – Decoder Style

E.g. SegNet (2015):

Improvements / Follow-Ups

• Post-Processing using Conditional Random Fields (CRFs)

• Also: Dilated Convolution but that is the next paper

Questions?
Additional Sources (not cited on slides)

- [http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review](http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review)
Backup Slides
Full Image Training vs Patch Training
Upper Limit on IU

<table>
<thead>
<tr>
<th>factor</th>
<th>mean IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>50.9</td>
</tr>
<tr>
<td>64</td>
<td>73.3</td>
</tr>
<tr>
<td>32</td>
<td>86.1</td>
</tr>
<tr>
<td>16</td>
<td>92.8</td>
</tr>
<tr>
<td>8</td>
<td>96.4</td>
</tr>
<tr>
<td>4</td>
<td>98.5</td>
</tr>
</tbody>
</table>