Scene Labeling with Convolutional Neural Networks
Zeming Lin and Jack Lanchantin
April 28, 2015
Motivation

Many tasks require fine-grained labelling of pixels in an image. E.g., labelling the entire scene ahead for a self-driving car.
Project Objective

Input

Output
Model Architecture

- **3 Input planes:** full R,G,B planes
- **9 Output planes:** each is interpreted as a score for a given class
  - Construct labels based on max probability of all classes
- **3 Hidden layers**
  - 64,64,64 feature maps at each layer, respectively
  - Each hidden layer contains a convolution and max pooling operation
Downscaled Label Planes

Feeding individual patches in is slow!
Convolutions => batch processing images
Downscaled Label Planes

Convolution → Pooling → “sky”
Convolution → Pooling → “mountain”

Pooling reduces resolution!

Single depth slice:

<table>
<thead>
<tr>
<th>x</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

max pool with 2x2 filters and stride 2

6 8
3 4

Pooling reduces resolution!
Shift-And-Stitch to Handle Downscaling

Convolution → Pooling → “sky”
Convolution → Pooling → “lake”
Merging Label Planes

Algorithm:
1. Calculate patch size $s$, and let pad be $p = s / 2$.
2. Zero pad bottom and right by $p$.
3. for $x,y$ in $(0..p-1, 0..p-1)$ do
   a. Pad left and top by $(p-x, p-y)$ and call this the $(x,y)$ image plane.
   b. $(x,y) = \lfloor s \times (xs, ys) \rfloor + (xr, yr)$, where $xr$ and $yr$ are the remainders.
   c. The final pixel $(x, y)$ is just at the $(xr, yr)$ image plane at pixel position $(xs, ys)$
Accuracy and Efficiency

- Deeper Model (5 hidden layers) achieves up to ~70% accuracy
  - Take about 5 minutes to test a 240x320 image
- Shallower model (3 hidden layers) achieves ~67% accuracy
  - Takes about 1 minute to test a 240x320 image
Results
Our Model

Ground Truth
Improvements/Future Work

- Implement “fbcunn”: facebook’s deep learning modules for GPUs
  - speeds up convolutions, FFT based algorithm => $O(n \lg n)$
- Parallelize shifted inputs and then do merging once they have all completed
  - Train on every pixel of the training set
- Train on other datasets (e.g. medical images)
Website!

http://45.55.218.104:3000/

Please don’t overload our server with requests :) Each image takes about 10 seconds to run.
Code (written in Torch7)

https://github.com/jacklanchantin/SceneLabelingConvNet