Lecture 14b: Beyond classification – segmentation and autoencoders

Machine Learning and Imaging

BME 590L
Roarke Horstmeyer
Class project option – work with a new image dataset from Kenya!

- Collaboration with Dr. Wendy Prudhomme-O’Meara at the Duke Global Health Institute
- Certain species of mosquitoes carry the malaria parasite
- Classification of different mosquitoes into different species at different locations/villages can offer some really useful information!

**Dataset:**
- **4 species imaged:** 1710 images identified as gambiæ, 402 as unidentified, 107 images as funestus but **only 17** as demeilloni

- Each species imaged 4 times from 4 directions
- In one of 4 states: unfed, blood-fed, gravid, half gravid

**Task:** Prep dataset, develop a classification (or other) network to establish some preliminary findings
Semantic Segmentation

Label each pixel in the image with a category label

Don’t differentiate instances, only care about pixels

Other possible examples:

M. Drozdzal et al., Learning Normalized Inputs for Iterative Estimation in Medical Image Segmentation (2017)

Z. Zhang et al., Recent Advances in the Applications of Convolutional Neural Networks to Medical Image Contour Detection (2017)
Semantic Segmentation Idea: Fully Convolutional?

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!

Input: $3 \times H \times W$

Convolutions: $D \times H \times W$

Scores: $C \times H \times W$

Predictions: $H \times W$

Problem: convolutions at original image resolution will be very expensive...
Instead, *compress x-y dimensions of input image*

- Compress spatial features into learned filters
- Then, decompress learned filters back into same spatial dimensions

---

**U-Net Architecture**

**U-Net: Convolutional Networks for Biomedical Image Segmentation**

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/
Semantic Segmentation Idea: Fully Convolutional

Downsampling: Pooling, strided convolution

Design network as a bunch of convolutional layers, with downsampling and upsampling inside the network!

Upsampling: ???

Input: 3 x H x W

High-res: \( D_1 \times \frac{H}{2} \times \frac{W}{2} \)

Low-res: \( D_3 \times \frac{H}{4} \times \frac{W}{4} \)

Med-res: \( D_2 \times \frac{H}{4} \times \frac{W}{4} \)

Med-res: \( D_2 \times \frac{H}{4} \times \frac{W}{4} \)

High-res: \( D_1 \times \frac{H}{2} \times \frac{W}{2} \)

Predictions: \( H \times W \)

In-Network upsampling: “Unpooling”

Nearest Neighbor

```
Input: 2 x 2

1 2
3 4

Output: 4 x 4

1 1 2 2
1 1 2 2
3 3 4 4
3 3 4 4
```

“Bed of Nails”

```
Input: 2 x 2

1 2
3 4

Output: 4 x 4

1 0 2 0
0 0 0 0
3 0 4 0
0 0 0 0
```
In-Network upsampling: “Max Unpooling”

Max Pooling
Remember which element was max!

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

Input: 4 x 4

Output: 2 x 2

Max Unpooling
Use positions from pooling layer

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>2</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Input: 2 x 2

Output: 4 x 4

Rest of the network

Corresponding pairs of downsampling and upsampling layers

Stanford CS231n - http://cs231n.stanford.edu
Learnable Upsampling: Transpose Convolution

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

Input: 4 x 4

Dot product between filter and input

Output: 2 x 2

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

Stride gives ratio between movement in input and output
Learnable Upsampling: Transpose Convolution

3 x 3 transpose convolution, stride 2 pad 1

Input: 2 x 2

Output: 4 x 4

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
Learnable Upsampling: 1D Example

Output contains copies of the filter weighted by the input, summing at where at overlaps in the output.

Need to crop one pixel from output to make output exactly 2x input.

Stanford CS231n - http://cs231n.stanford.edu
Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

\[
x \ast a = X \tilde{a}
\]

\[
\begin{bmatrix}
x & y & x & 0 & 0 & 0 \\
0 & x & y & x & 0 & 0 \\
0 & 0 & x & y & x & 0 \\
0 & 0 & 0 & x & y & x \\
0 & 0 & 0 & 0 & x & y \\
0 & 0 & 0 & 0 & 0 & x \\
\end{bmatrix}
\begin{bmatrix}
0 \\
a \\
b \\
c \\
d \\
0 \\
\end{bmatrix}
= 
\begin{bmatrix}
ay + bz \\
ax + by + cz \\
bx + cy + dz \\
cx + dy \\
0 \\
0 \\
\end{bmatrix}

Example: 1D conv, kernel size=3, stride=1, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

\[
x \ast^T \tilde{a} = X^T \tilde{a}
\]

\[
\begin{bmatrix}
x & 0 & 0 & 0 \\
y & x & 0 & 0 \\
z & y & x & 0 \\
0 & z & y & x \\
0 & 0 & z & y \\
0 & 0 & 0 & z \\
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
c \\
d \\
0 \\
0 \\
0 \\
\end{bmatrix}
= 
\begin{bmatrix}
ax \\
ay + bx \\
ax + by + cx \\
bd + cy + dx \\
cz + dy \\
dz \\
\end{bmatrix}

When stride=1, convolution transpose is just a regular convolution (with different padding rules)
Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

$$\vec{x} \ast \vec{a} = X \vec{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & a & b & c & d & 0 \\ 0 & 0 & x & y & z & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ay + bz \\ bx + cy + dz \end{bmatrix}$$

Example: 1D conv, kernel size=3, stride=2, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} \ast^T \vec{a} = X^T \vec{a}$$

$$\begin{bmatrix} x & 0 \\ y & 0 \\ z & x \\ 0 & y \\ 0 & z \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} ax \\ ay \\ az + bx \\ bz \\ by \end{bmatrix}$$

When stride>1, convolution transpose is no longer a normal convolution!
Learned sensing for improved image segmentation

Variably illuminated images

\[ w_1 I_1 + w_2 I_2 + w_3 I_3 + \ldots \]

U-Net CNN

Segmentation Map

Optimized illumination for nuclei segmentation

Mean Red channel | Mean Green channel | Mean Blue channel

Standard illumination | Learned illumination

+5-10% accuracy
Image segmentation – current workflow

**Capture**: BF images

![Optimally illuminated image](image)

**Learned illumination**

**Capture**: Fluorescence

![Fluorescence image](image)

(e.g., DAPI-stained nuclei)

**Threshold**

Segmentation Mask

**Inference via a trained U-Net**
in silico labeling: fluorescence image inference from bright-field data

In Silico Labeling: Predicting Fluorescent Labels in Unlabeled Images

Eric M. Christiansen,1,5,6,7 Samuel J. Yang,1 D. Michael Ando,1 Ashkan Jahayerian,1,2,8 Gaia Skibinski,2,9 Scott Lipnick,5,4,6,7 Elliot Mount,5,6,7 Alison O’Neil,5,6,7 Kevan Shah,5,6,7 Alicia K. Lee,5,6,7 Piyush Goyal,5,10 William Fedus,5,6,7,8,9 Ryans Poplin,1,5,7,8 Andre Esteva,2,7,10 Marc Berndt,2 Lee L. Rubin,7 Philip Nelson,7 and Steven Pinkinheimer5,6

1Google, Inc., Mountain View, CA 94043, USA
2Howard Hughes Medical Institute, Cambridge, MA 02139, USA
3Department of Biomedical Informatics, Harvard Medical School, Boston, MA 02115, USA
4Department of Stem Cell and Regenerative Biology, Harvard University, Cambridge, MA 02139, USA
5Department of Biomolecular Engineering, Stanford University, Stanford, CA 94305, USA
6Department of Neurology, University of California, San Francisco, CA 94158, USA
7Department of Learning Algorithms, University of Montreal, Montreal, QC, Canada
8Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA
9Center for Assessment Technology and Continuous Health, Massachusetts General Hospital, Boston, MA 02114, USA

BF Focal stack (26+ images)

Task: bright-field to fluorescence image Inference
Image segmentation versus *in silico* labeling (fluorescence inference)

**Capture:** Bright field

**Capture:** Fluorescence

Learned illumination for fluorescent image inference?

We can just Inference the fluorescence image itself...

Optimally illuminated
Instead, compress $x$-$y$ dimensions of input image

- Compress spatial features into learned filters
- Then, decompress learned filters back into same spatial dimensions
- Can be an autoencoder
- Analogous to image compression
- A very powerful idea…
Another example: Denoising Autoencoder

U-Net Architecture

- Input noisy image
- Compress
- Decompress
- "Label" is noiseless image!

Diagram details:
- Conv 3x3, ReLU
- Copy and crop
- Max pool 2x2
- Up-conv 2x2
- Conv 1x1
Example: Variational Autoencoder (VAE)

- Force this vector to follow a Gaussian PDF

- With Gaussian PDF, can start to add/subtract latent vector in a normalized vector space

Minimize (KL) distance between latent vector and Gaussian normal

Adding new features to samples
Example: Variational Autoencoder (VAE)

- Force this vector to follow a Gaussian PDF
- With Gaussian PDF, can start to add/subtract latent vector in a normalized vector space

Minimize (KL) distance between latent vector and Gaussian normal

Generative Example (once trained):
- Encode image with glasses, obtain latent vector PDF $P_g$
- Encode image without glasses, obtain PDF $P_{ng}$
- Compute $\text{diff} = P_g - P_{ng}$
- Encode new image to obtain $P_{\text{new}}$, add in $\text{diff}$
- Decode $P_{\text{new}} + \text{diff}$ to get guy with glasses!
Code review: See the following:

Jupyter Notebook: A simple Autoencoder in Tensorflow/Keras

https://deepimaging.github.io/lectures/