CAP5415
Computer Vision

Yogesh S Rawat
yogesh@ucf.edu
HEC-241
Administrative

• GPU accounts
• Bonus programming assignment
• Homework performance
  • Make use of office hours
Introduction to Convolutional Neural Networks

Lecture 6
A quote from a famous scientist...

Deep Learning

What society thinks I do
A quote from a famous scientist...
A quote from a famous scientist...

Deep Learning

What society thinks I do
What my friends think I do
What other computer scientists think I do
A quote from a famous scientist...

Deep Learning

What society thinks I do
What my friends think I do
What other computer scientists think I do

What mathematicians think I do
A quote from a famous scientist...

Deep Learning

What society thinks I do

What my friends think I do

What other computer scientists think I do

What mathematicians think I do

What I think I do
A quote from a famous scientist...
CNN – example: depth estimation

CNN – example: depth estimation
CNN – example: depth estimation

Convolutional Neural Network (CNN)

- A class of Neural Networks
  - Takes image as input (mostly)
  - Make predictions about the input image
History

• The LeNet architecture (1990s)

*Gradient-based learning applied to document recognition*

First Strong Results

• AlexNet 2012
  • Winner of ImageNet Large-Scale Visual Recognition Challenge (ILSVRC 2012)
  • Error rate – 15.4% (the next best entry was at 26.2%)

*Imagenet classification with deep convolutional neural networks*
Today: CNNs are everywhere

Classification
Today: CNNs are everywhere

Object detection

Semantic Segmentation

Faster R-CNN: Ren, He, Girshick, Sun 2015

Today: CNNs are everywhere

**Image captioning**

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

A skateboarder does a trick on a ramp.

A dog is jumping to catch a frisbee.

A group of young people playing a game of frisbee.

Two hockey players are fighting over the puck.

A little girl in a pink hat is blowing bubbles.

A refrigerator filled with lots of food and drinks.

A herd of elephants walking across a dry grass field.

A close up of a cat laying on a couch.

A red motorcycle parked on the side of the road.

A yellow school bus parked in a parking lot.

"Show and tell: A neural image caption generator."


**Style transfer**

A Neural Algorithm of Artistic Style

L. Gatys et al. 2015.)
CNN – Not just images

• Natural Language Processing (NLP)
  • Text classification
  • Word to vector

• Audio Research
  • Speech recognition
  • Can be represented as spectrograms

• Converting data to a matrix (2-D) format
  • 1D convolution – Audio, EEG, etc.
  • 3D convolution - Videos
Background
What we already know!
General CNN architecture
General CNN architecture
What is a (digital) Image? - recap

• Definition: A digital image is defined by integrating and sampling continuous (analog) data in a spatial domain [Klette, 2014].
General CNN architecture
Filtering - recap

- Image filtering: compute function of local neighborhood at each position

\[ h[m, n] = \sum_{k, l} f[k, l] I[m + k, n + l] \]

- Space domain: 2d coords = k, l
- Image domain: 2d coords = m, n
Filtering - recap

• Output is linear combination of the neighborhood pixels

\[
\begin{array}{ccc}
1 & 3 & 0 \\
2 & 10 & 2 \\
4 & 1 & 1 \\
\end{array} \quad \otimes \quad \begin{array}{ccc}
1 & 0 & -1 \\
1 & 0.1 & -1 \\
1 & 0 & -1 \\
\end{array} = \begin{array}{ccc}
& & \\
& & \\
& & 5 \\
\end{array}
\]
Correlation (linear relationship) - recap

\[ f \otimes h = \sum_k \sum_l f(k, l)h(k, l) \]

\[ f = \text{Image} \]

\[ h = \text{Kernel} \]

\[
\begin{array}{ccc}
  f_1 & f_2 & f_3 \\
  f_4 & f_5 & f_6 \\
  f_7 & f_8 & f_9
\end{array}
\]

\[
\begin{array}{ccc}
  h_1 & h_2 & h_3 \\
  h_4 & h_5 & h_6 \\
  h_7 & h_8 & h_9
\end{array}
\]

\[ f \otimes h = f_1h_1 + f_2h_2 + f_3h_3 + f_4h_4 + f_5h_5 + f_6h_6 + f_7h_7 + f_8h_8 + f_9h_9 \]
Convolution – recap

\[ f * h = \sum_k \sum_l f(k, l)h(-k, -l) \]

- \( f \) = Image
- \( h \) = Kernel

\[ f \]
\[ \begin{array}{ccc}
    f_1 & f_2 & f_3 \\
    f_4 & f_5 & f_6 \\
    f_7 & f_8 & f_9 \\
\end{array} \]

\[ h \]
\[ \begin{array}{ccc}
    h_1 & h_2 & h_3 \\
    h_4 & h_5 & h_6 \\
    h_7 & h_8 & h_9 \\
\end{array} \]

\[ X - \text{flip} \]
\[ Y - \text{flip} \]

\[ f * h = f_1h_9 + f_2h_8 + f_3h_7 \]
\[ + f_4h_6 + f_5h_5 + f_6h_4 \]
\[ + f_7h_3 + f_8h_2 + f_9h_1 \]
Sobel Edge Detector

Image $I$ 

\[
\begin{bmatrix}
1 & 0 & -1 \\
2 & 0 & -2 \\
1 & 0 & -1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & -2 & -1 \\
\end{bmatrix}
\]

$\frac{d}{dx} I$ $\frac{d}{dy} I$ 

\[\sqrt{\left(\frac{d}{dx} I\right)^2 + \left(\frac{d}{dy} I\right)^2}\]

Threshold 

Edges
General CNN architecture
Multi-layer perceptron (MLP) – recap

• ...is a ‘fully connected’ neural network with non-linear activation functions.

• ‘Feed-forward’ neural network
General CNN architecture
Learning phases

Training

Images

Labels

Image Features

Training

Trained classifier

Testing

Image not in training set

Image Features

Apply classifier

Prediction

Slide credit: D. Hoiem and L. Lazebnik
General CNN architecture

End to end learning!
Neural Network vs CNN

• Image as input in neural network
  • Size of feature vector = HxWxC
  • For 256x256 RGB image
    • 196608 dimensions

• CNN - Special type of neural network
  • Operate with volume of data
  • Weight sharing in form of kernels

Source: http://cs231n.github.io
Fundamental operation
Convolution

- Core building block of a CNN
  - Spatial structure of image is preserved

A filter/kernel is convolved with the image
Convolution

- Convolution at one spatial location

32x32x3 image

3x3x3 filter

Result of convolution
Convolution

- Convolution over whole image

32x32x3 image

3x3x3 filter

Convolve over all spatial locations

Activation map (feature map)
Convolution

• Multiple filters

32x32x3 image

2 3x3x3 filter

Convolve over all spatial locations

Activation maps (feature maps)

32x32x3 image

32

32

32

30

30

1

3
Convolution layer

- One convolution layer
  - 6 3x3x3 kernels

32x32x3 image → Convolution layer → Activation maps

32x32x3 image

Convolution layer

Activation maps

32 32 32

6 30 30
Convolutional Network

• Convolution network is a sequence of these layers
Convolutional Network

- Convolution network is a sequence of these layers
Parameters

Convolve over all spatial locations

32x32x3 image

3x3x3 filter

Activation map (feature map)

32

32

3

30

30

1
Parameters

32x32x3 image → Convolution layer → Activation maps

6 3x3x3 kernels – 6x3x3x3 parameters = 162
Convolution Operation

- Convolution of two functions \( f \) and \( g \)

\[
(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)g(t - \tau)d\tau
\]

In CNN we use 2D convolutions (mostly)
Sobel Edge Detector – recap

Image $I$:

- $\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$

- $\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

- $\frac{d}{dx} I$

- $\frac{d}{dy} I$

- $\sqrt{\left(\frac{d}{dx} I\right)^2 + \left(\frac{d}{dy} I\right)^2}$

Threshold → Edges

[Diagram of Sobel Edge Detector process]
Demo

Input image

filter

output

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 0 0 0

1 0 1
0 1 0
1 0 1

4
Demo

Input image

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

filter

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

output

4 3
Demo

Input image

Output

Filter

1 0 1
0 1 0
1 0 1

4 3 4
Demo

Input image

filter

output
Demo

Input image

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

filter

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

output

<table>
<thead>
<tr>
<th>4</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Convolution - Intuition

0 0 0 0 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
0 0 0 0 0
Convolution - Intuition
Convolution - Intuition

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\times
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\Rightarrow
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[1 \times 1 + 1 \times 1 + \ldots + 1 \times 1 = 5\]
Convolution - Intuition
Convolution - Intuition

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\begin{array}{c}
\times \\
\end{array}
\begin{array}{c}
1x1 = 1 \\
\end{array}
\]
Convolution

- Multiple filters

32x32x3 image

2 3x3x3 filter

Activation maps (feature maps)

Convolve over all spatial locations
Convolution - Intuition

Source: https://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/
2D Convolution - dimensions

7x7 map

3x3 filter
2D Convolution - dimensions

7x7 map

3x3 filter
2D Convolution - dimensions

7x7 map

3x3 filter
2D Convolution - dimensions

7x7 map  

3x3 filter
2D Convolution - dimensions

- 7x7 map
- 3x3 filter
- Output activation map 5x5
- Output size
- \( N - F + 1 \)
- \( (7 - 3 + 1) = 5 \)
- \( N \) – input size
- \( F \) – filter size

Output size formula:
\[
N - F + 1 = 5
\]

Input size:
\[
N = \text{original size}
\]

Filter size:
\[
F = \text{filter size}
\]

Image size:
\[
7 = \text{image size}
\]
Stride

7x7 map

3x3 filter

Filter applied with stride 2
Stride

7x7 map

3x3 filter

Filter applied with stride 2
Stride

7x7 map

3x3 filter

Filter applied with stride 2

Activation map size 3x3

Output size

\[
\frac{(7-3)}{2} + 1 = 3
\]

\[
\frac{(N-F)}{S} + 1
\]
Pop Quiz?
Stride

7x7 map

5x5 filter

Filter applied with stride 1

Activation map size?

30 seconds!
Stride

7x7 map

5x5 filter

Filter applied with stride 1

Activation map size 3x3
Stride

7x7 map

3x3 filter

Filter applied with stride 3
Stride

7x7 map

3x3 filter

Filter applied with stride 3

Cannot cover perfectly

Not all parameters will fit
Stride

7x7 map

3x3 filter
Output size \((N-F)/S + 1\)
\(N = 7, F = 3\)

Stride 1
\((7-3)/1 + 1 \Rightarrow 5\)

Stride 2
\((7-3)/2 + 1 \Rightarrow 3\)

Stride 3
\((7-3)/3 + 1 \Rightarrow 2.33\)
Padding

• Zero padding in the input

For 7x7 input and 3x3 filter

If we have padding of one pixel

Output

7x7

Size (recall (N-F)/S+1)

(N-F+2P)/S + 1
## Padding

- Zero padding in the input

![Padding Example](image)

Common to see, (F-1)/2 padding with stride 1 to preserve the map size

\[
N = \frac{(N-F+2P)}{S} + 1
\]

\[
\Rightarrow (N-1)S = N-F+2P
\]

\[
\Rightarrow P = \frac{(F-1)}{2}
\]
Pooling

• Invariance to small translations of the input
Pooling

• Makes the representations smaller
• Operates over each activation map independently
Pooling

- Kernel size
- Stride

Single depth slice

max pool with 2x2 filters and stride 2
Pop Quiz?

Input feature map: 112x112x16
Max-pooling kernel size: 3x3
Stride: 2
How many parameters required?
30 seconds!
Visualizing CNN

Source: http://cs231n.github.io
AlexNet: Network Size

- Input 227x227x3
- 5 convolution layers
- 3 dense layers
- Output 1000-D vector
AlexNet : Network Size

• Input: 227x227x3 images
• First layer (CONV1): 96 11x11 filters applied at stride 4
• What is the output volume size? \((227-11)/4+1 = 55\)
• What is the number of parameters? \(11x11x3x96 = 35K\)
AlexNet: Network Size

- After CONV1: 55x55x96
- Second layer (POOL1): 3x3 filters applied at stride 2
- What is the output volume size? \((55-3)/2+1 = 27\)
- What is the number of parameters in this layer? 0
AlexNet : Network Size

- After POOL1: 27x27x96
- Third layer (NORM1): Normalization
- What is the output volume size? 27x27x96
AlexNet: Network Size

1. [227x227x3] INPUT
2. [55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
3. [27x27x96] MAX POOL1: 3x3 filters at stride 2
4. [27x27x96] NORM1: Normalization layer
5. [27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
6. [13x13x256] MAX POOL2: 3x3 filters at stride 2
7. [13x13x256] NORM2: Normalization layer
8. [13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
9. [13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
10. [13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
11. [6x6x256] MAX POOL3: 3x3 filters at stride 2
12. [4096] FC6: 4096 neurons
13. [4096] FC7: 4096 neurons
14. [1000] FC8: 1000 neurons (class scores)
Visualizing Convolution

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]
Why not correlation neural network?

• It could be
  • Deep learning libraries actually implement correlation

• Correlation relates to convolution via a 180deg rotation
  • When we learn kernels, we could easily learn them flipped
Questions?

Sources for this lecture include materials from works by Abhijit Mahalanobis, Andrej Karpathy, and Fei Fei Li