CS6501: Deep Learning for Visual Recognition

Image Processing
Today’s Class

Image Representations: Matrices
Image Representations: RGB, HSV, etc
Image Processing: Brightness
Image Filtering: Mean Filter, Median Filter
Image Filtering: Convolutions
Blurring, Sharpening, Computing Gradients
About the Course

CS6501-004: Deep Learning for Visual Recognition

• Instructor: Vicente Ordonez
• Email: vicente@virginia.edu
• Website: http://vicenteordonez.com/deeplearning/
• Class Location: Thornton Hall E303
• Class Times: Monday-Wednesday 3:30pm and 4:45pm
• Piazza: https://piazza.com/virginia/spring2019/cs6501004/home
Teaching Assistants

Ziyan Yang
(tw8cb@virginia.edu)
Hours: 11am to 12:30pm
Tues and Thurs at Rice 436

Tianlu Wang
(tw8cb@virginia.edu)
Hours: 10am to noon on Weds at Rice 436
Grading

- Assignments: 50% (5 assignments)
  (10% + 10% + 10% + 10% + 10%)

- Course Project: 40%
  Groups of 3 students.

- Participation / Required Reading Summaries: 10%
  Required Reading Activities will be listed in the course page well in advance.
Additionally you will need for your Course Project:

Free credits for students! $50.

- g2.2xlarge: 4GB ($0.65 / hour)
- p2.xlarge: 12GB ($0.90 / hour)
Or maybe, even better:

GTX 1080 Ti (11GB): $700
GTX 1080 (8GB): $500
GTX 1060 (6GB): $150-$250
Also...

• Assignment 1 will be released on course website
• In the meantime complete, the pytorch/jupyter/Google Colaboratory tutorial
Reminder of what is an image for a computer.
Images as Functions

\[ z = f(x, y) \]
Images as Functions

\[ z = f(x, y) \]

- The domain of \( x \) and \( y \) is \([0, \text{img-width})\) and \([0, \text{img-height})\).
- \( x \) and \( y \) are discretized into integer values.
Light

• What determines the color of a pixel?

Figure from Szeliski
The Retina

**Cones**
- cone-shaped
- less sensitive
- operate in high light
- color vision

**Rods**
- rod-shaped
- highly sensitive
- operate at night
- gray-scale vision

[What the Frog's Eye Tells the Frog's Brain]
Electromagnetic Spectrum

Human Luminance Sensitivity Function

http://www.yorku.ca/eye/photopik.htm
Basic Image Processing

\[ I \quad \alpha I \]

\( \alpha > 1 \)
Basic Image Processing

\[ I \quad \alpha I \]

\[ 0 < \alpha < 1 \]
Color Images as Tensors

channel x height x width
Color Images as Tensors

Channels are usually RGB: Red, Green, and Blue

Other color spaces: HSV, HSL, LUV, XYZ, Lab, CMYK, etc
Color spaces: RGB

Some drawbacks
• Strongly correlated channels
• Non-perceptual

Default color space

Slide by James Hays

Color spaces: HSV

Intuitive color space

H
(S=1, V=1)

S
(H=1, V=1)

V
(H=1, S=0)

Slide by James Hays
Color spaces: L*a*b*

“Perceptually uniform”* color space

Slide by James Hays
Most information in intensity

Only color shown – constant intensity
Most information in intensity

Only intensity shown – constant color
Most information in intensity

Original image
Image filtering
Image filtering

Image filtering
Image filtering: e.g. Mean Filter
Image filtering: e.g. Mean Filter
Image filtering: e.g. Median Filter

Image filtering: Convolution operator

\[ g(x, y) = \sum_u \sum_v k(u, v) f(x - u, y - v) \]

Image filtering: Convolution operator

e.g. mean filter

\[ k(x, y) = \begin{bmatrix}
\frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\
\frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\
\frac{1}{9} & \frac{1}{9} & \frac{1}{9}
\end{bmatrix} \]

Image filtering: Convolution operator
e.g. mean filter

\[ k(x, y) = \begin{bmatrix}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{bmatrix} \]

Example: box filter

\[
g[\cdot,\cdot] = \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

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

Credit: S. Seitz
Image filtering

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]
Image filtering

\[ f[\cdot,\cdot] \]

\[ h[\cdot,\cdot] \]

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

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ g[\cdot, \cdot] \frac{1}{9} \]

\[ h[\ldots] \]

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

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

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

Credit: S. Seitz
Image filtering

\[
f[\ldots]\]

\[
h[\ldots]\]

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

Credit: S. Seitz
Image filtering

\[ f[\ldots] \quad g[\ldots] \]

\[
\begin{array}{ccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{cccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 20 & 40 & 60 & 60 & 60 & 60 & 40 & 20 & 10 \\
0 & 30 & 60 & 90 & 90 & 90 & 90 & 60 & 30 \\
0 & 30 & 50 & 80 & 80 & 80 & 90 & 60 & 30 \\
0 & 30 & 50 & 80 & 80 & 80 & 90 & 60 & 30 \\
0 & 0 & 20 & 30 & 50 & 50 & 60 & 40 & 20 \\
10 & 20 & 30 & 30 & 30 & 30 & 20 & 10 \\
10 & 10 & 10 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

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

Credit: S. Seitz
What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

Slide credit: David Lowe (UBC)
Image filtering: e.g. Mean Filter
Image filtering: Convolution operator

Important filter: gaussian filter (gaussian blur)

\[ k(x, y) = \begin{bmatrix}
1/16 & 1/8 & 1/16 \\
1/8 & 1/4 & 1/8 \\
1/16 & 1/8 & 1/16 
\end{bmatrix} \]

Important filter: Gaussian

- Weight contributions of neighboring pixels by nearness

\[ G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \]

5 x 5, \( \sigma = 1 \)

<table>
<thead>
<tr>
<th></th>
<th>0.003</th>
<th>0.013</th>
<th>0.022</th>
<th>0.013</th>
<th>0.003</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
<td></td>
</tr>
<tr>
<td>0.022</td>
<td>0.097</td>
<td>0.159</td>
<td>0.097</td>
<td>0.022</td>
<td></td>
</tr>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
<td></td>
</tr>
<tr>
<td>0.003</td>
<td>0.013</td>
<td>0.022</td>
<td>0.013</td>
<td>0.003</td>
<td></td>
</tr>
</tbody>
</table>

Slide credit: Christopher Rasmussen
Image filtering: Convolution operator
e.g. gaussian filter (gaussian blur)

Practical matters

• What about near the edge?
  • the filter window falls off the edge of the image
  • need to extrapolate
  • methods:
    • clip filter (black)
    • wrap around
    • copy edge
    • reflect across edge

Source: S. Marschner
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Filtered
(no change)

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

0 0 0
0 0 1
0 0 0

Shifted left
By 1 pixel

Source: D. Lowe
Practice with linear filters

Original

(Note that filter sums to 1)

Source: D. Lowe
Practice with linear filters

Original

**Sharpening filter**
- Accentuates differences with local average

Source: D. Lowe
Sharpening

before

after

Source: D. Lowe
Key properties of linear filters

**Linearity:**

\[ \text{imfilter}(I, f_1 + f_2) = \text{imfilter}(I, f_1) + \text{imfilter}(I, f_2) \]

**Shift invariance:** same behavior regardless of pixel location

\[ \text{imfilter}(I, \text{shift}(f)) = \text{shift}(\text{imfilter}(I, f)) \]

Any linear, shift-invariant operator can be represented as a convolution

Source: S. Lazebnik
Image filtering: Convolution operator

– Enhance images
  • Denoise, resize, increase contrast, etc.
– Extract information from images
  • Texture, edges, distinctive points, etc.
– Detect patterns
  • Template matching
– Deep Convolutional Networks
Image filtering: Convolution operator
Important Filter: Sobel operator

\[ k(x, y) = \begin{bmatrix}
1 & 0 & -1 \\
2 & 0 & -2 \\
1 & 0 & -1 \\
\end{bmatrix} \]

Other filters

Sobel

Vertical Edge
(absolute value)
Other filters

Sobel

Horizontal Edge (absolute value)

Slide by James Hays
Sobel operators are equivalent to 2D partial derivatives of the image

- Vertical sobel operator – Partial derivative in $X$ (width)
- Horizontal sobel operator – Partial derivative in $Y$ (height)

- Can compute magnitude and phase at each location.

- Useful for detecting edges
Sobel filters are (approximate) partial derivatives of the image

Let \( f(x, y) \) be your input image, then the partial derivative is:

\[
\frac{\partial f(x, y)}{\partial x} = \lim_{h \to 0} \frac{f(x + h, y) - f(x, y)}{h}
\]

Also:

\[
\frac{\partial f(x, y)}{\partial x} = \lim_{h \to 0} \frac{f(x + h, y) - f(x - h, y)}{2h}
\]
But digital images are not continuous, they are discrete

Let $f[x, y]$ be your input image, then the partial derivative is:

$$\Delta_x f[x, y] = f[x + 1, y] - f[x, y]$$

Also:

$$\Delta_x f[x, y] = f[x + 1, y] - f[x - 1, y]$$
But digital images are not continuous, they are discrete

Let \( f[x, y] \) be your input image, then the partial derivative is:

\[
\Delta_x f[x, y] = f[x + 1, y] - f[x, y]
\]

Also:

\[
\Delta_x f[x, y] = f[x + 1, y] - f[x - 1, y]
\]
Sobel Operators Smooth in Y and then Differentiate in X

\[ k(x, y) = \begin{array}{c}
1 \\
2 \\
1 \\
\end{array} \ast \begin{array}{ccc}
1 & 0 & -1 \\
2 & 0 & -2 \\
1 & 0 & -1 \\
\end{array} = \begin{array}{ccc}
1 & 0 & -1 \\
2 & 0 & -2 \\
1 & 0 & -1 \\
\end{array} \]

Similarly to differentiate in Y
Image Features
What are Image Features?

- Color histogram?
- Maximum color on sub-areas of the image?
- Any statistics on the input image?
- The output of some image processing on the input image?
Why are they useful?

As inputs to a machine learning model
Why are they useful?

To compare images (i.e. retrieve similar images)

Distance function (e.g. Euclidean distance)
Image Features: Color

Photo by: marielito

slide by Tamara L. Berg
Image Features: Color

80 million tiny images: a large dataset for non-parametric object and scene recognition

Antonio Torralba, Rob Fergus and William T. Freeman
Color often not a powerful feature

However, these are all images of people but the colors in each image are very different.
Suggested Reading for Next Class

• [What the Frog's Eye Tells the Frog's Brain]

[Csurka's Bag of Keypoints. ECCV 2004]
Questions?