Convolution

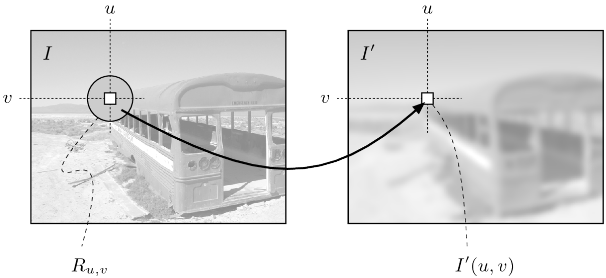
Foundations of Data Analysis

April 19, 2022

Spatial Filters

Definition

A **spatial filter** is an image operation where each pixel value $I(u, v)$ is changed by a function of the intensities of pixels in a neighborhood of (u, v).



What Spatial Filters Can Do

Blurring/Smoothing

$\rightarrow$

What Spatial Filters Can Do

Sharpening

$\rightarrow$

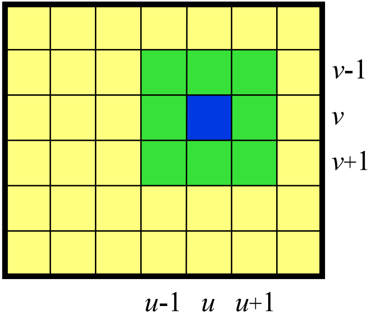
What Spatial Filters Can Do

Weird Stuff

→

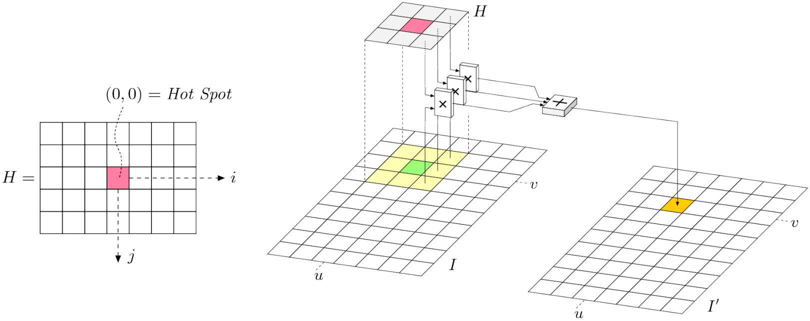
Example: The Mean of a Neighborhood

Consider taking the mean in a 3 × 3 neighborhood:



$$I'(u,v) = \frac{1}{9} \sum_{i=-1}^{1} \sum_{j=-1}^{1} I(u+i, v+j)$$

How a Linear Spatial Filter Works



H is the filter “kernel” or “matrix”

For the neighborhood mean: $$H(i,j) = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

General Filter Equation

Notice that the kernel H is just a small image!

Let $H : R_H \to [0, K-1]$

$$I'(u,v) = \sum_{(i,j) \in R_H} I(u+i, v+j) \cdot H(i,j)$$

This is known as a **correlation** of I and H

What Does This Filter Do?

0	0	0
0	1	0
0	0	0

Identity function (leaves image alone)

What Does This Filter Do?

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Mean (averages neighborhood)

What Does This Filter Do?

0	0	0
0	0	1
0	0	0

Shift left by one pixel

What Does This Filter Do?

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 17 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Sharpen (identity minus mean filter)

Filter Normalization

- Notice that all of our filter examples sum up to one
- Multiplying all entries in H by a constant will cause the image to be multiplied by that constant
- To keep the overall brightness constant, we need H to sum to one

$$
\begin{aligned}
I'(u,v) &= \sum_{i,j} I(u+i, v+j) \cdot (cH(i,j)) \\
&= c \sum_{i,j} I(u+i, v+j) \cdot H(i,j)
\end{aligned}
$$

Effect of Filter Size

Mean Filters:

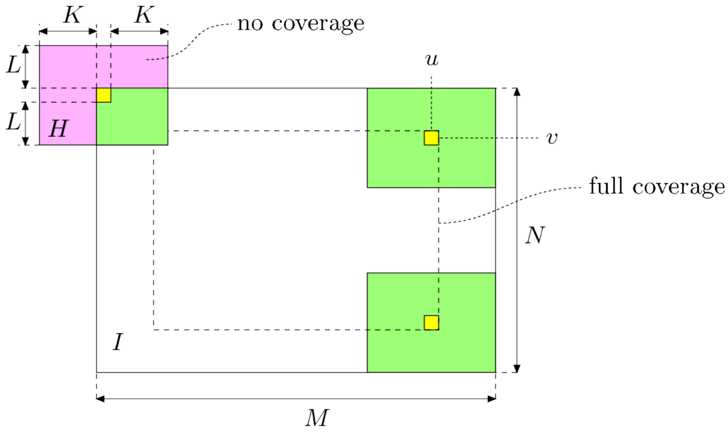
Original

7×7

15×15

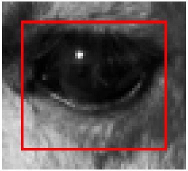
41×41

What To Do At The Boundary?



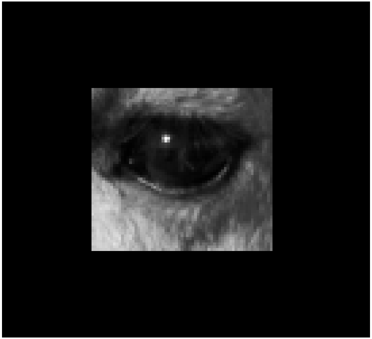
What To Do At The Boundary?

- Crop



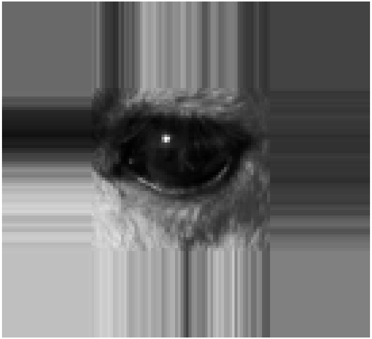
What To Do At The Boundary?

- Crop
- Pad



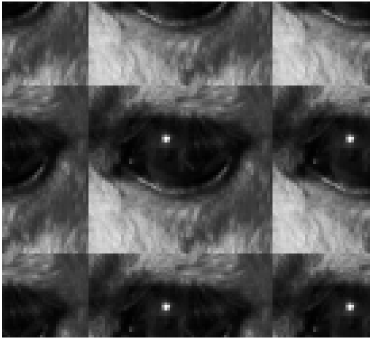
What To Do At The Boundary?

- Crop
- Pad
- Extend



What To Do At The Boundary?

- Crop
- Pad
- Extend
- Wrap



Convolution

Definition

Convolution of an image I by a kernel H is given by

$$I'(u,v) = \sum_{(i,j)\in R_H} I(u-i, v-j) \cdot H(i,j)$$

This is denoted: $I' = I * H$

- Notice this is the same as correlation with H, but with negative signs on the I indices
- Equivalent to vertical and horizontal flipping of H:

$$I'(u,v) = \sum_{(-i,-j)\in R_H} I(u+i, v+j) \cdot H(-i,-j)$$

Linear Operators

Definition

A **linear operator** F on an image is a mapping from one image to another, $I' = F(I)$, that satisfies:

1. $F(cI) = cF(I)$,
2. $F(I_1 + I_2) = F(I_1) + F(I_2)$,

where I, I_1, I_2 are images, and c is a constant.

Both correlation and convolution are linear operators

Infinite Image Domains

Let's define our image and kernel domains to be infinite:

$$\Omega = \mathbb{Z} \times \mathbb{Z}$$

Remember $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$

Now convolution is an infinite sum:

$$I'(u,v) = \sum_{i=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} I(u-i, v-j) \cdot H(i,j)$$

This is denoted $I' = I * H$.

Infinite Image Domains

The infinite image domain $\Omega = \mathbb{Z} \times \mathbb{Z}$ is just a trick to make the theory of convolution work out.

We can still imagine that the image is defined on a bounded (finite) domain, $[0, w] \times [0, h]$, and is set to zero outside of this.

Properties of Convolution

Commutativity:

$$I * H = H * I$$

This means that we can think of the image as the kernel and the kernel as the image and get the same result.

In other words, we can leave the image fixed and slide the kernel or leave the kernel fixed and slide the image.

Properties of Convolution

Associativity:

$$(I * H_1) * H_2 = I * (H_1 * H_2)$$

This means that we can apply H_1 to I followed by H_2, or we can convolve the kernels $H_2 * H_1$ and then apply the resulting kernel to I.

Properties of Convolution

Linearity:

$$(a \cdot I) * H = a \cdot (I * H)$$

$$(I_1 + I_2) * H = (I_1 * H) + (I_2 * H)$$

This means that we can multiply an image by a constant before or after convolution, and we can add two images before or after convolution and get the same results.

Properties of Convolution

Shift-Invariance:

Let S be the operator that shifts an image I:

$$S(I)(u, v) = I(u + a, v + b)$$

Then

$$S(I * H) = S(I) * H$$

This means that we can convolve I and H and then shift the result, or we can shift I and then convolve it with H.

Properties of Convolution

Theorem: The only shift-invariant, linear operators on images are convolutions.

Computational Complexity of Convolution

If my image I has size $M \times N$ and my kernel H has size $(2R+1) \times (2R+1)$, then what is the complexity of convolution?

$$I'(u,v) = \sum_{i=-R}^{R} \sum_{j=-R}^{R} I(u-i, v-j) \cdot H(i,j)$$

Answer: $O(MN(2R+1)(2R+1)) = O(MNR^2)$.
Or, if we consider the image size fixed, $O(R^2)$.

Which is More Expensive?

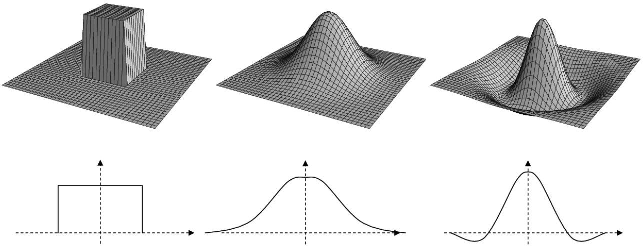
The following both shift the image 10 pixels to the left:

1. Convolve with a 21×21 shift operator (all zeros with a 1 on the right edge)
2. Repeatedly convolve with a 3 $\times$ 3 shift operator 10 times

The first method requires $21^2 \cdot wh = 441 \cdot wh$.
The second method requires $(9 \cdot wh) \cdot 10 = 90 \cdot wh$.

Some More Filters

Box | Gaussian | Laplace



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

(a)

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

(b)

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

(c)

Edge Detection

What is an Edge?

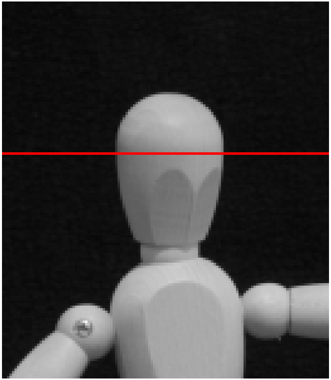
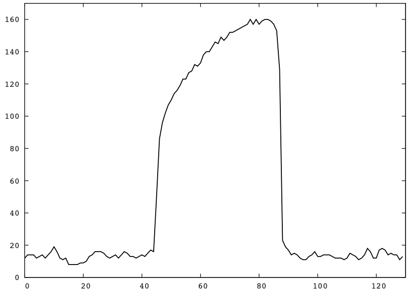


Image Value vs X-Position



An abrupt transition in intensity between two regions

What is an Edge?

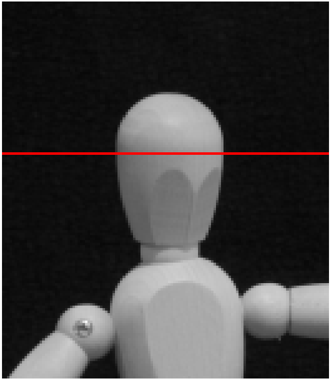


Image X-Derivative vs X-Position

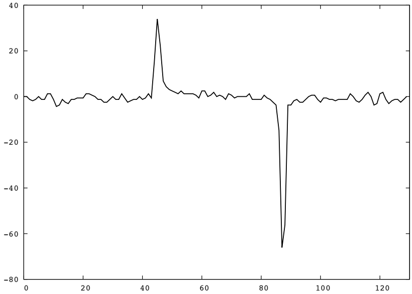
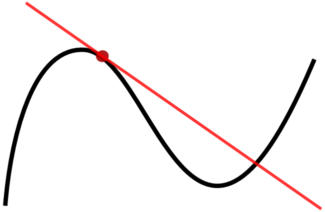


Image derivatives are high (or low) at edges

Review: Derivative of a Function

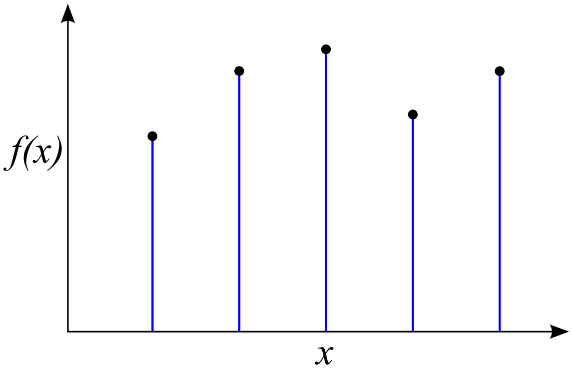
Given a function $f : \mathbb{R} \to \mathbb{R}$, its derivative is defined as

$$\frac{df}{dx}(x) = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$



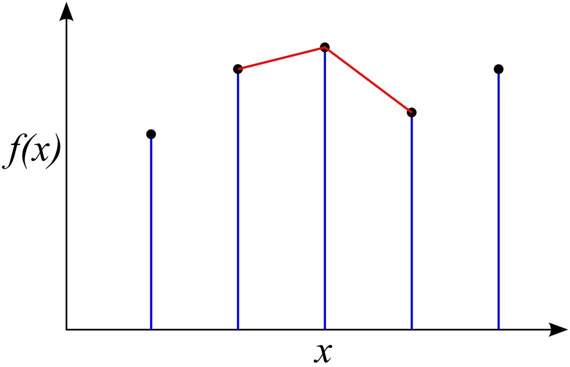
Derivative of f is the slope of the tangent to the graph of f

Derivatives of Discrete Functions



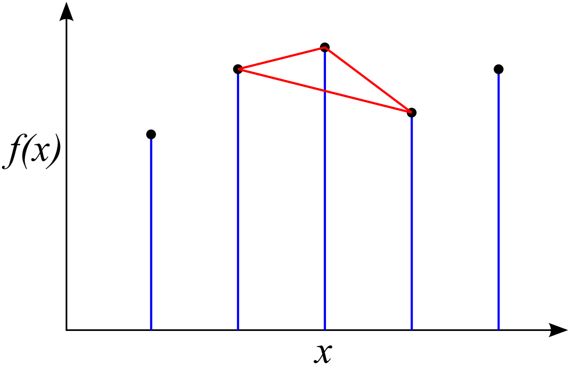
Discrete function defined on integer values of x

Derivatives of Discrete Functions



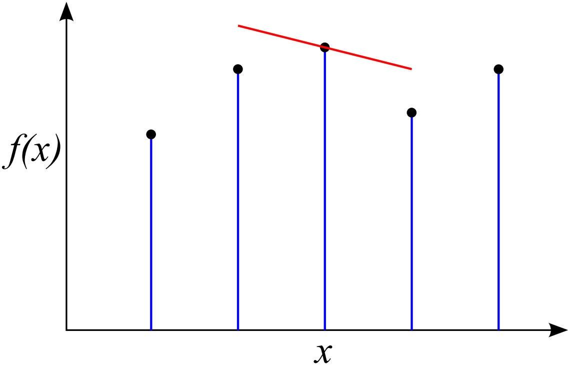
Slopes (derivatives) don't match on left and right

Derivatives of Discrete Functions



Instead take the average of the two (or secant)

Derivatives of Discrete Functions



Instead take the average of the two (or secant)

Finite Differences

Forward Difference

$$\Delta_+ f(x) = f(x+1) - f(x)$$ right slope

Backward Difference

$$\Delta_- f(x) = f(x) - f(x-1)$$ left slope

Central Difference

$$\Delta f(x) = \frac{1}{2}\left(f(x+1) - f(x-1)\right)$$ average slope

Finite Differences as Convolutions

Forward Difference

$$\Delta_+ f(x) = f(x+1) - f(x)$$

Take a convolution kernel: $H = [1 \;\; -1 \;\; 0]$

$$\Delta_+ f = f * H$$

(Remember that the kernel H is flipped in convolution)

Finite Differences as Convolutions

Central Difference

$$\Delta f(x) = \frac{1}{2}\left(f(x+1) - f(x-1)\right)$$

Convolution kernel here is: $H = \left[\begin{array}{ccc} \frac{1}{2} & 0 & -\frac{1}{2} \end{array}\right]$

$$\Delta f(x) = f * H$$

Notice: Derivative kernels sum to zero!

Derivatives of Images

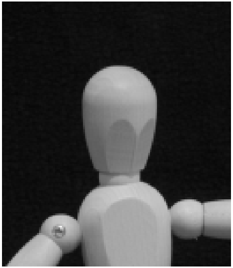
- Images have two parameters: $I(x, y)$
- We can take derivatives with respect to x or y
- Central differences:

$$\Delta_x I = I * H_x, \quad \text{and} \quad \Delta_y I = I * H_y,$$

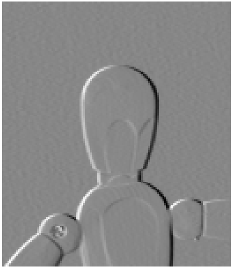
where $H_x = \begin{bmatrix} 0.5 & 0 & -0.5 \end{bmatrix}$ and $H_y = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix}$

Derivatives of Images

x-derivative using central difference:

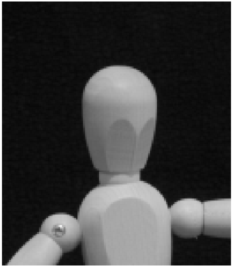


$$* \quad \left[\, \tfrac{1}{2} \;\; 0 \;\; -\tfrac{1}{2} \,\right] \quad =$$

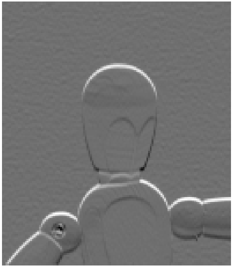


Derivatives of Images

y-derivative using central difference:



$$* \begin{bmatrix} 0.5 \\ 0 \\ -0.5 \end{bmatrix} =$$



Combining x and y Derivatives

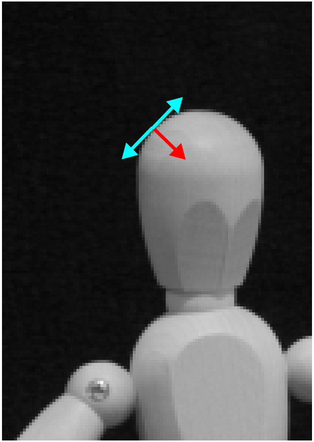
The **discrete gradient** of $I(x, y)$ is the 2D vector:

$$\nabla I(x,y) = \begin{bmatrix} \Delta_x I(x,y) \\ \Delta_y I(x,y) \end{bmatrix}$$

The **gradient magnitude** is

$$\|\nabla I(x,y)\| = \sqrt{(\Delta_x I(x,y))^2 + (\Delta_y I(x,y))^2}$$

Image Gradient



- Gradient points in direction of maximal increasing intensity
- Length (magnitude) of gradient equals amount of change in that direction
- Gradient is perpendicular (90 degrees) to edge contour

Convolutional Neural Networks (CNNs)

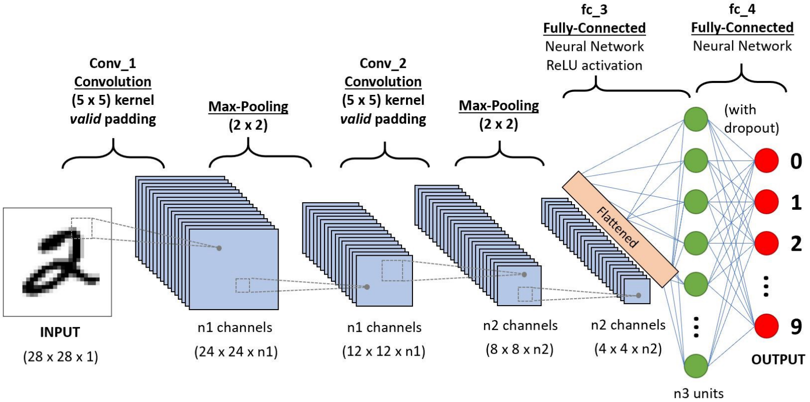
Learning a Filter

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

?

Filter consists of weights that need to be learned.

Convolutional Neural Networks



http://towardsdatascience.com/