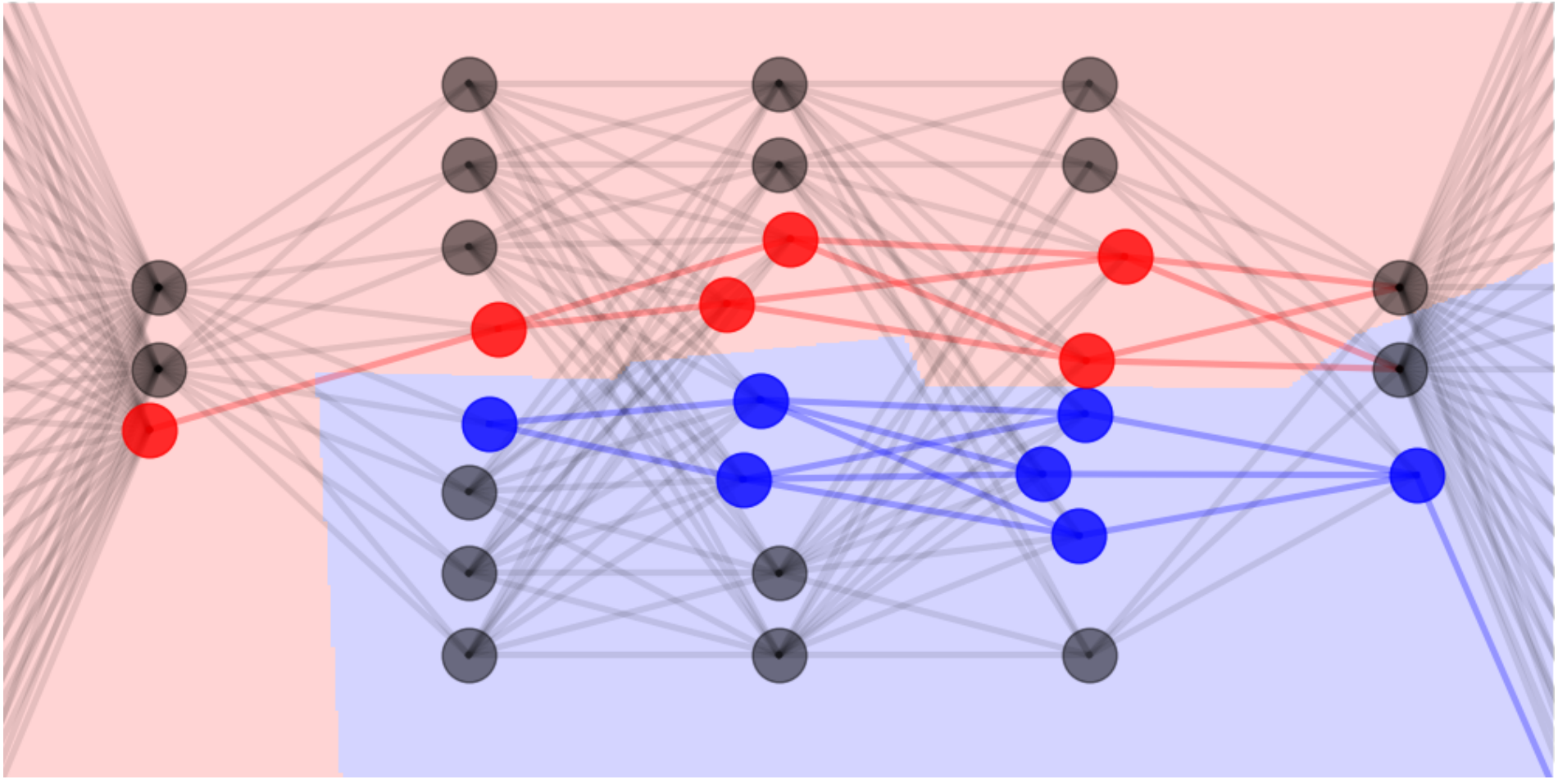
TU-Delft Machine and Deep Learning



CNNs

Lecture 5

Lecturer: Jan van Gemert

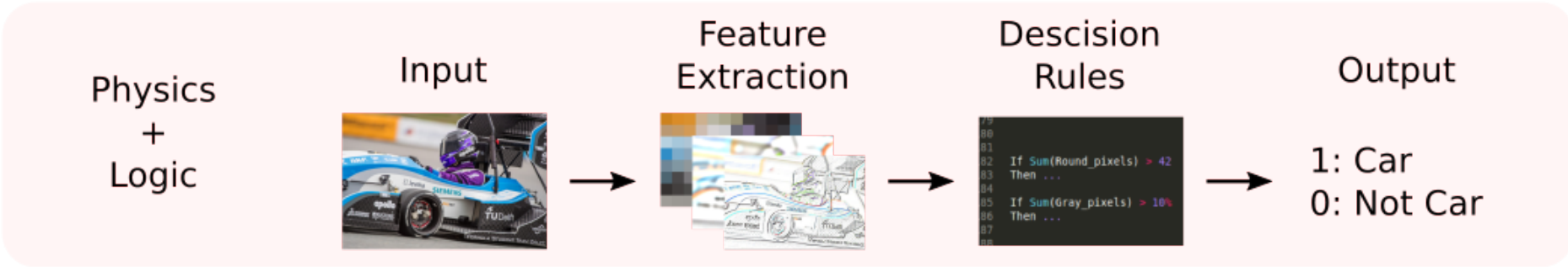
Questions?

Today: Convolutional Neural Networks

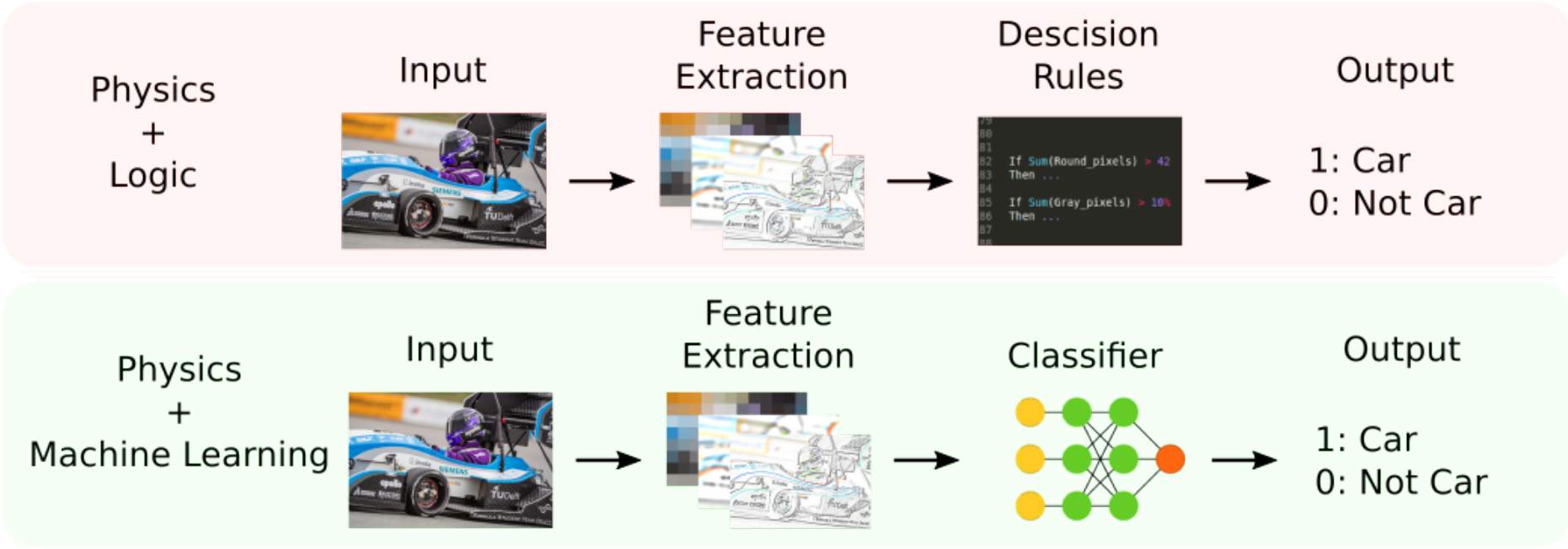
- Convolution
- Convolutional Network
- Relation to Fully Connected layers
- Properties of CNNs
- Going deeper

Chapters 'deep learning' book: 9.1, 9.2, 9.3, 9.5
'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/),

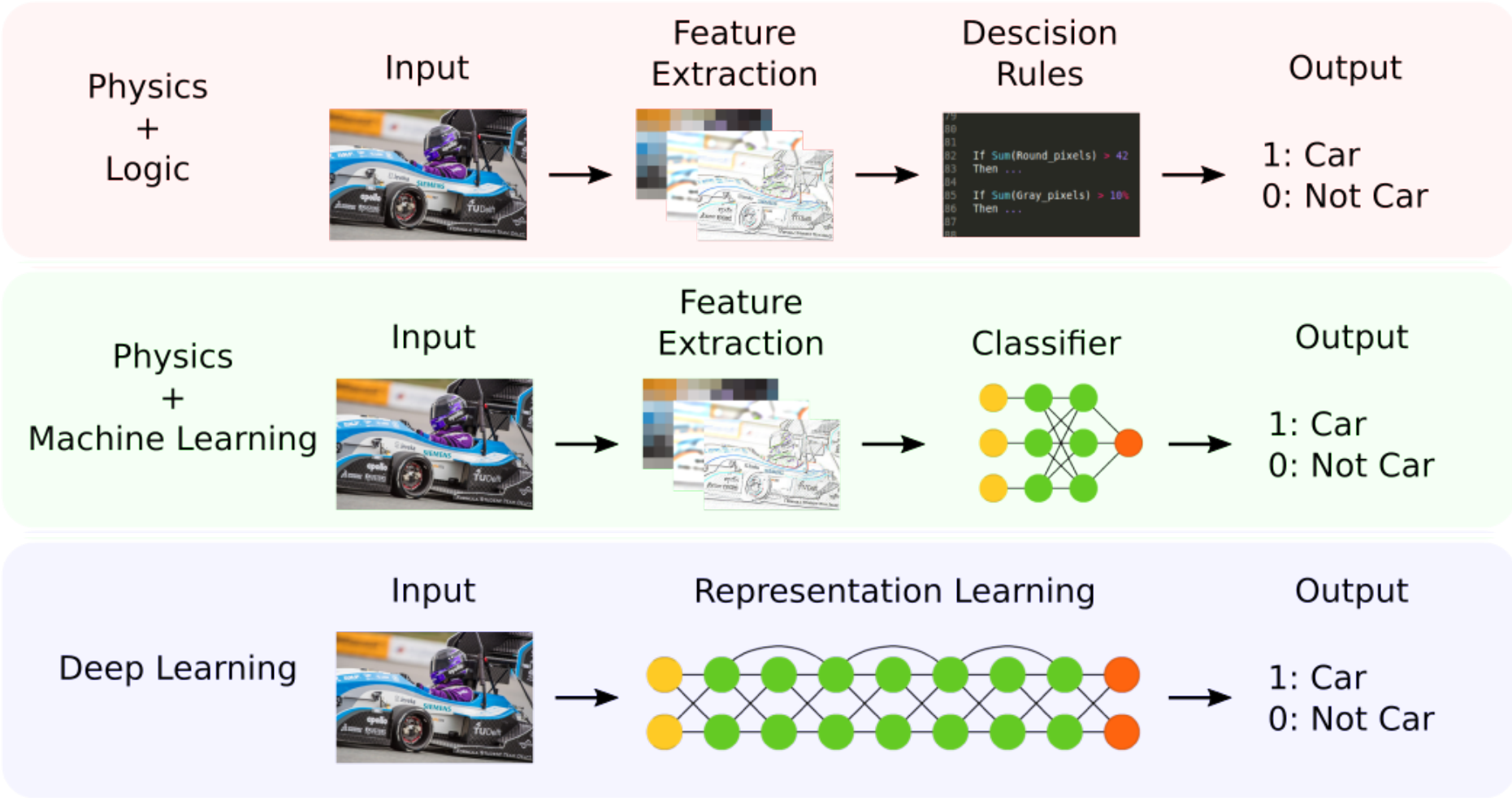
Feature extraction vs deep learning



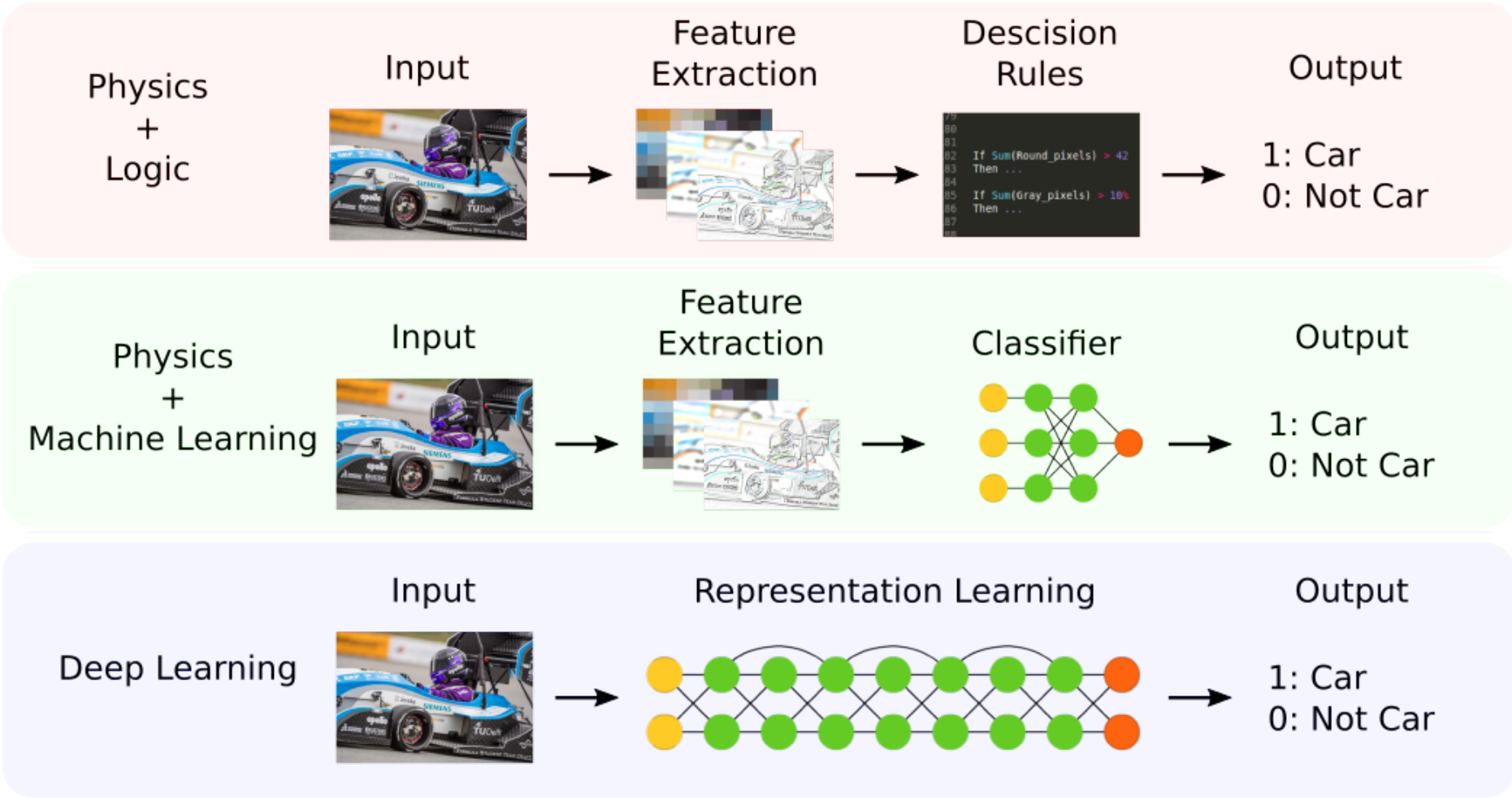
Feature extraction vs deep learning



Feature extraction vs deep learning



Feature extraction vs deep learning



End-to-End learning: End goal (output) used to learn feature extraction (input)

A bit of image processing

Q: How to get rid of noisy pixels?

A bit of image processing

Q: How to get rid of noisy pixels?

A: Simple solution: replace pixel by neighborhood average

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0								

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0								

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10							

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10							

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10	20						

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10	20						

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10	20	30					

Moving Neighborhood Average

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Moving Neighborhood Average

Q: What do you notice?

$F(x, y)$

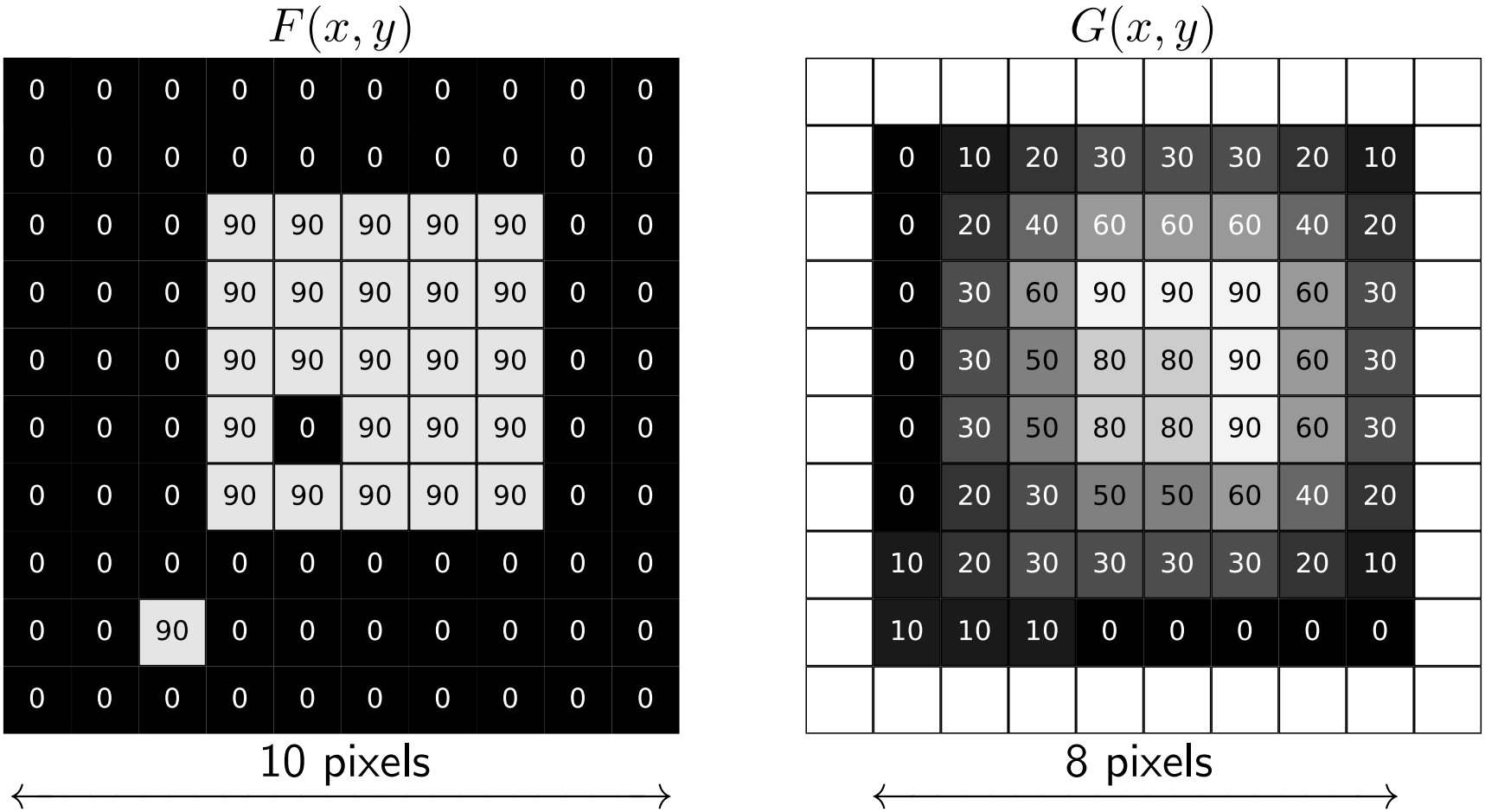
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Moving Neighborhood Average

Q: What do you notice?



A: 2 pixels lost to boundary (1 on each side)

Moving Neighborhood Average

Q: What do you notice?

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

10 pixels

$G(x, y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

8 pixels

A: 2 pixels lost to boundary (1 on each side)

Q: How to solve?

Moving Neighborhood Average

Q: What do you notice?

$F(x, y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

10 pixels

$G(x, y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

8 pixels

A: 2 pixels lost to boundary (1 on each side)

Q: How to solve? A: Add values outside the border (padding)

Convolution

Technically: not convolution but cross-correlation; look up the difference if you are interested.

Chapter 9.1

Generalize to a kernel: **Multiply weights variables per pixel and sum**

$F(x,y)$ $\star$ $H(x,y)$ $=$ $G(x,y)$

0	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	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

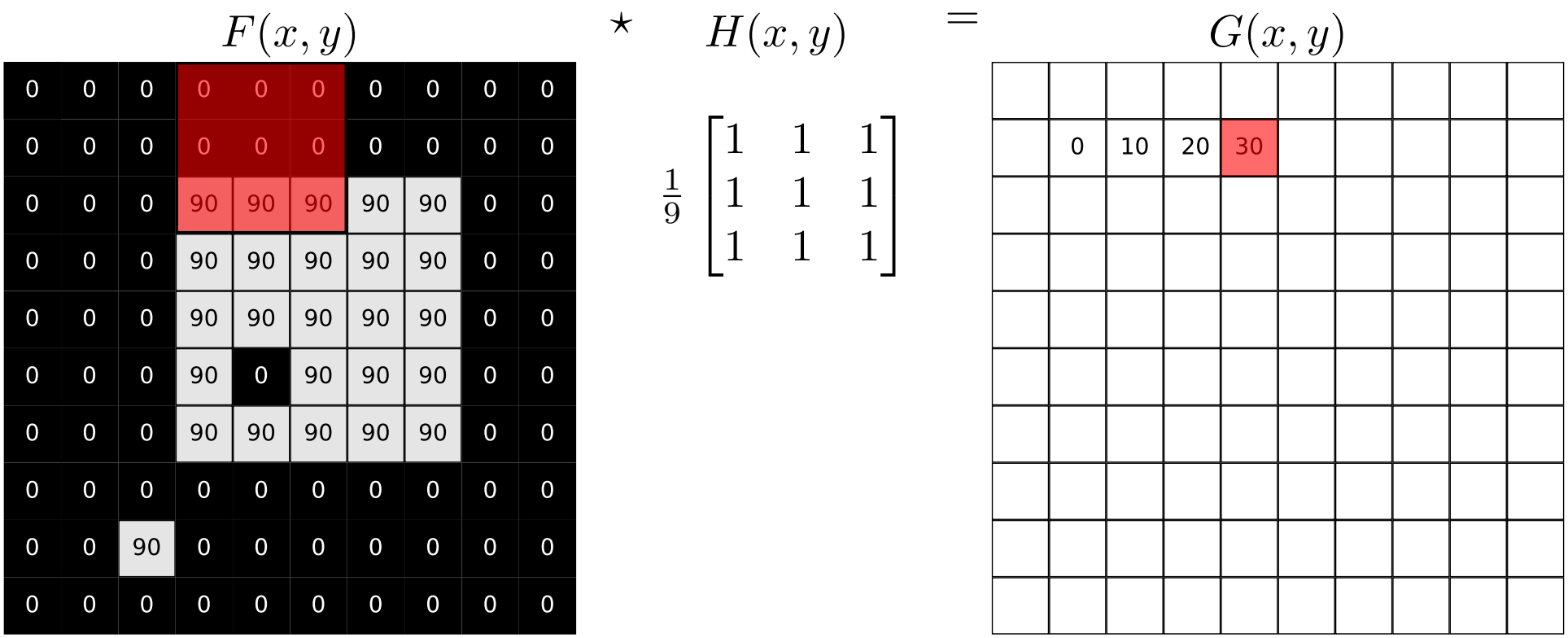
	0	10	20	30					

Q: What values to fill in kernel H to obtain a moving neighborhood average?

Convolution

Chapter 9.1

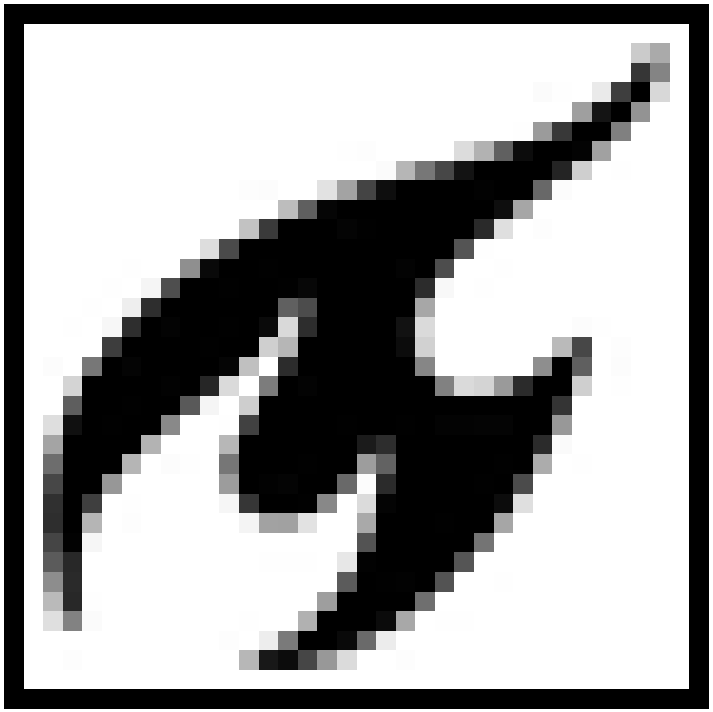
Generalize to a kernel: **Multiply weights variables per pixel and sum**



Q: What values to fill in kernel H to obtain a moving neighborhood average?

Practice with kernels

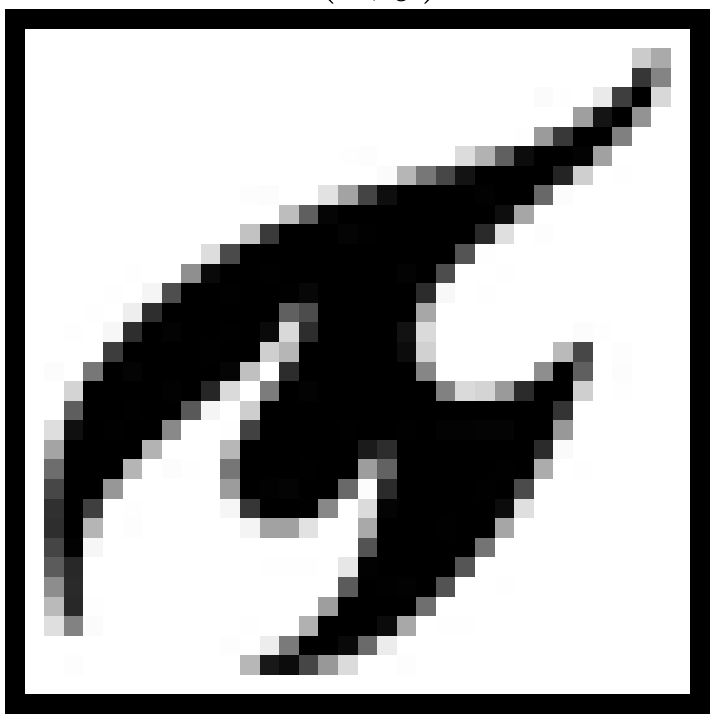
$F(x,y)$ $\star$ $H(x,y)$ $=$ $G(x,y)$



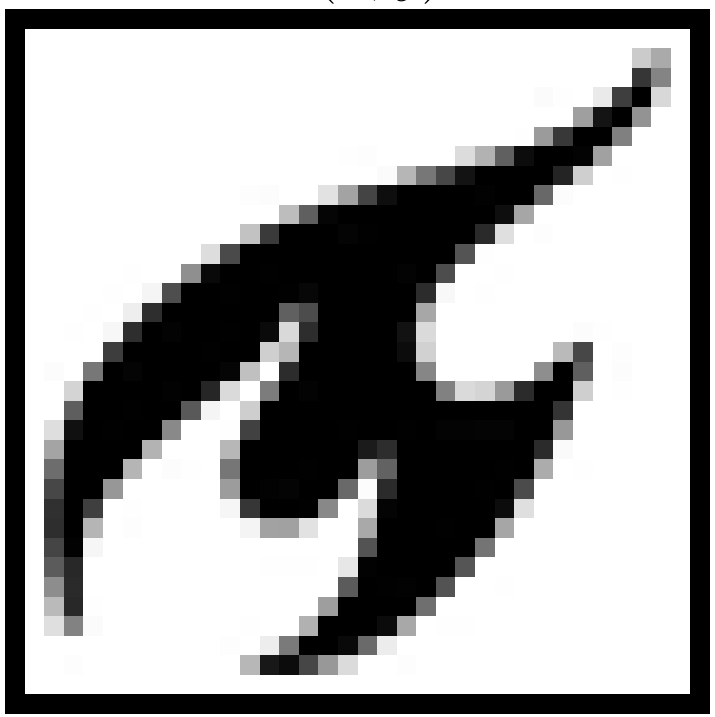
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Practice with kernels

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$

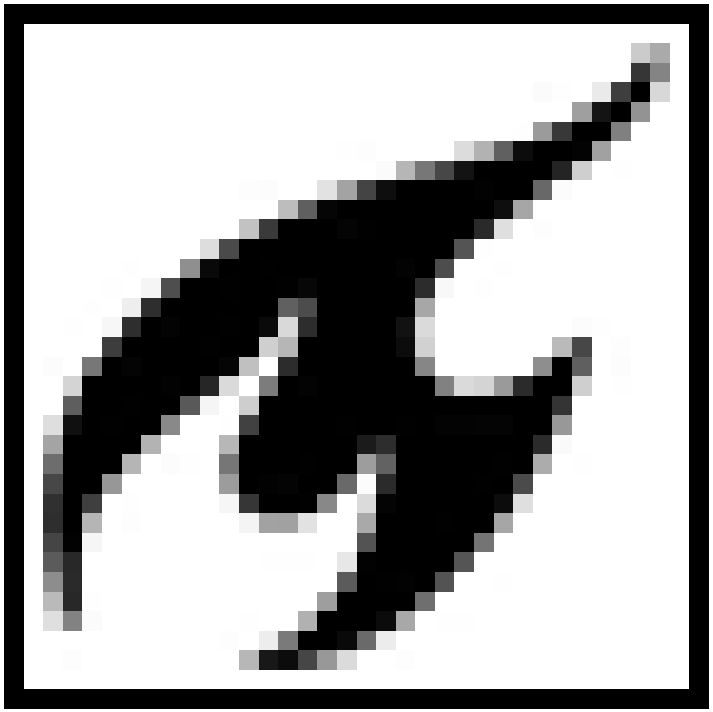


$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



Practice with kernels

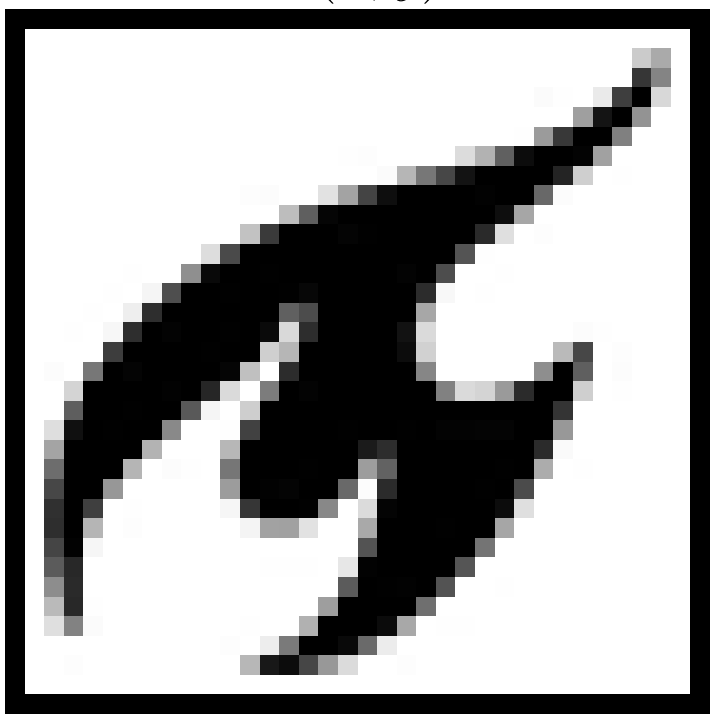
$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



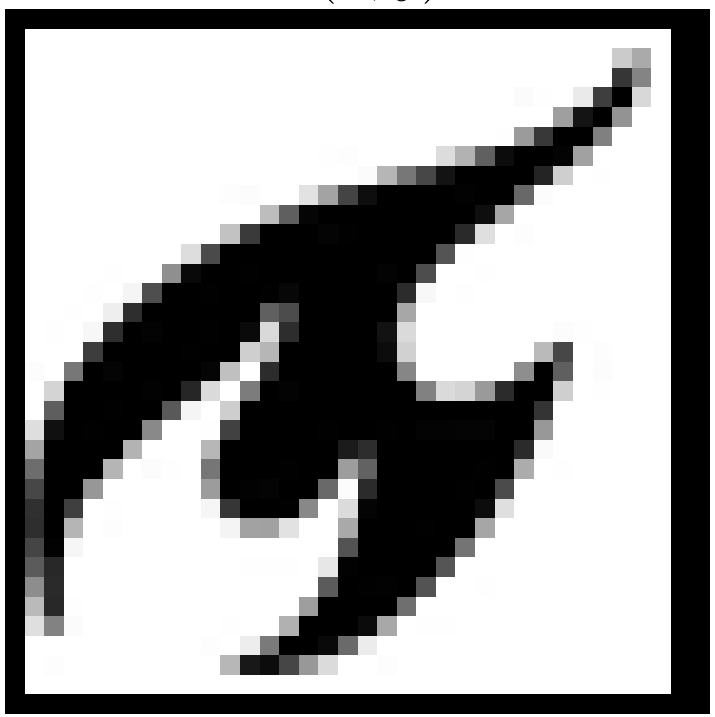
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Practice with kernels

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$

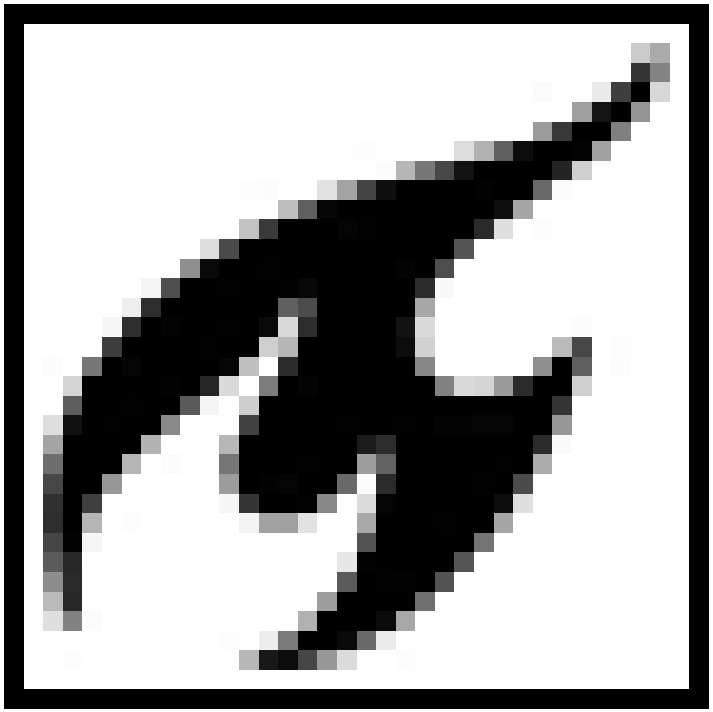


$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



Practice with kernels

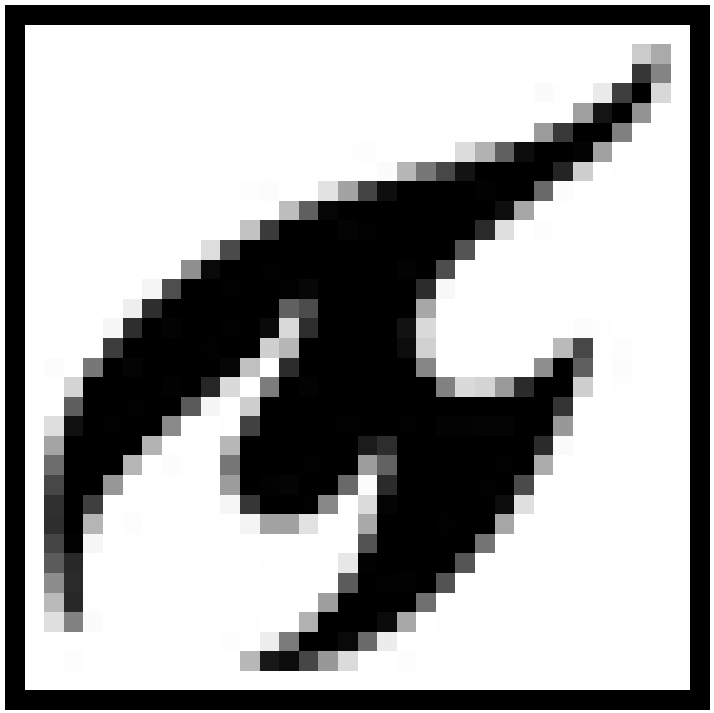
$F(x,y)$ $\star$ $H(x,y)$ $=$ $G(x,y)$



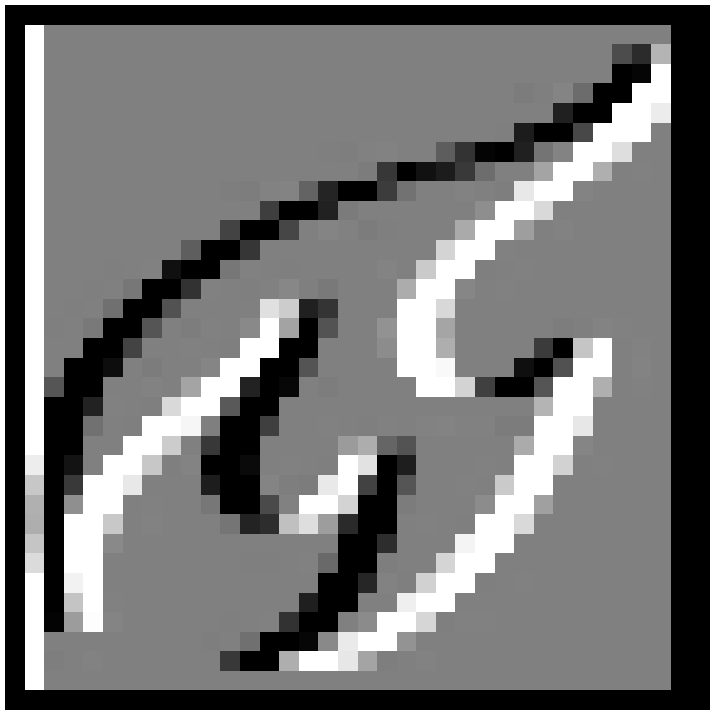
$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Practice with kernels

$F(x,y)$ $\star$ $H(x,y)$ $=$ $G(x,y)$

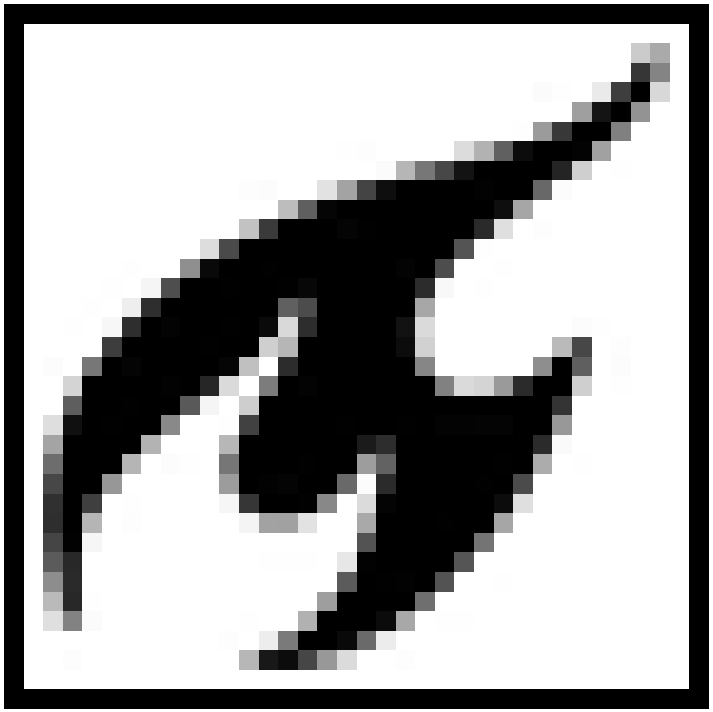


$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

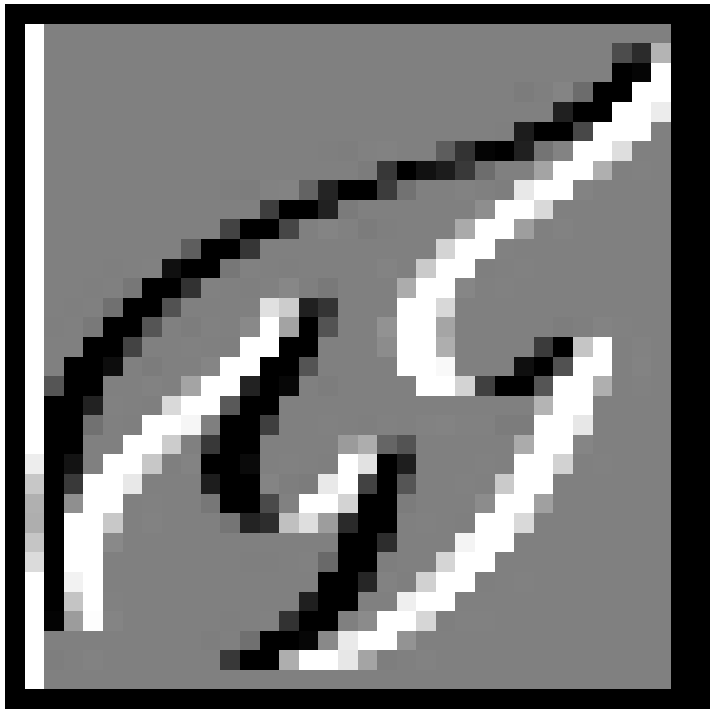


Practice with kernels

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



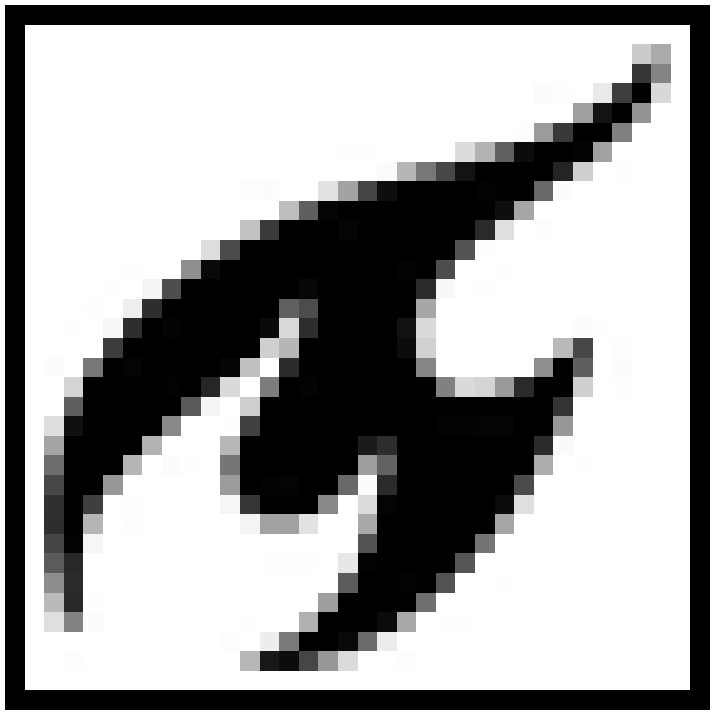
$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



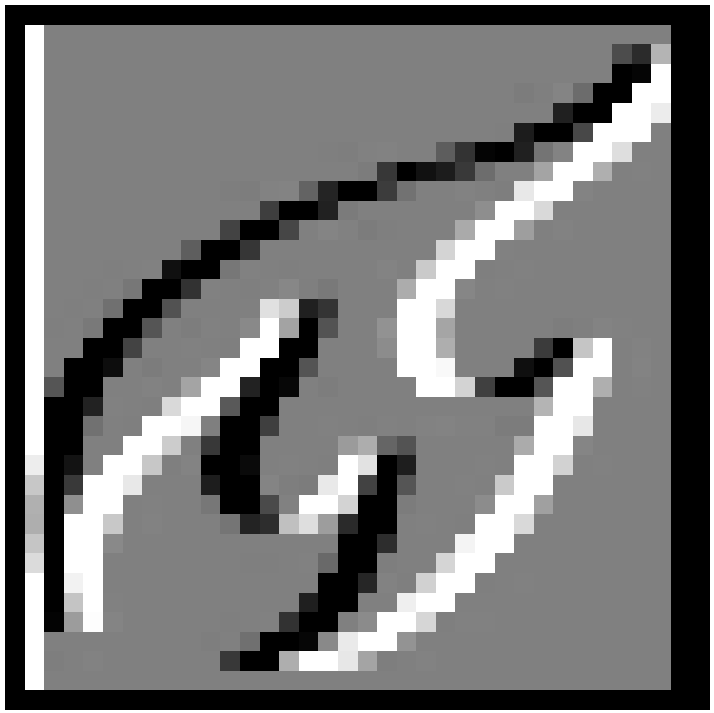
Q: For both images (left and right), what range are the values?

Practice with kernels

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



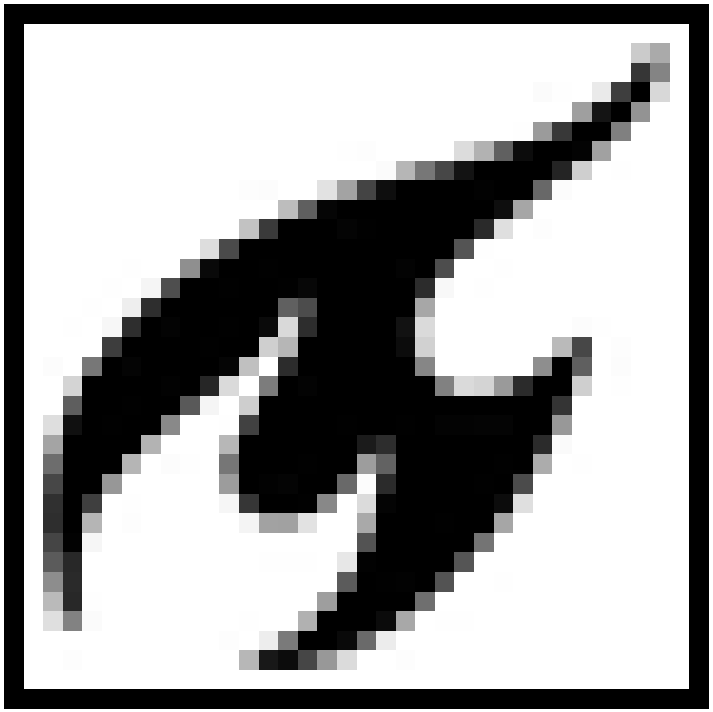
$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



Q: For both images (left and right), what range are the values?
A: Left image: 0-255; Right image: [-255, 255]

Practice with kernels

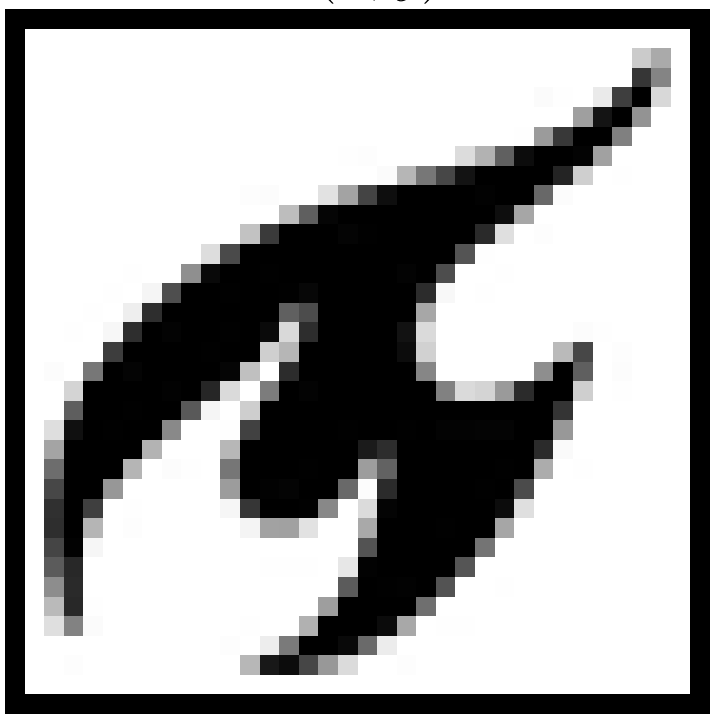
$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



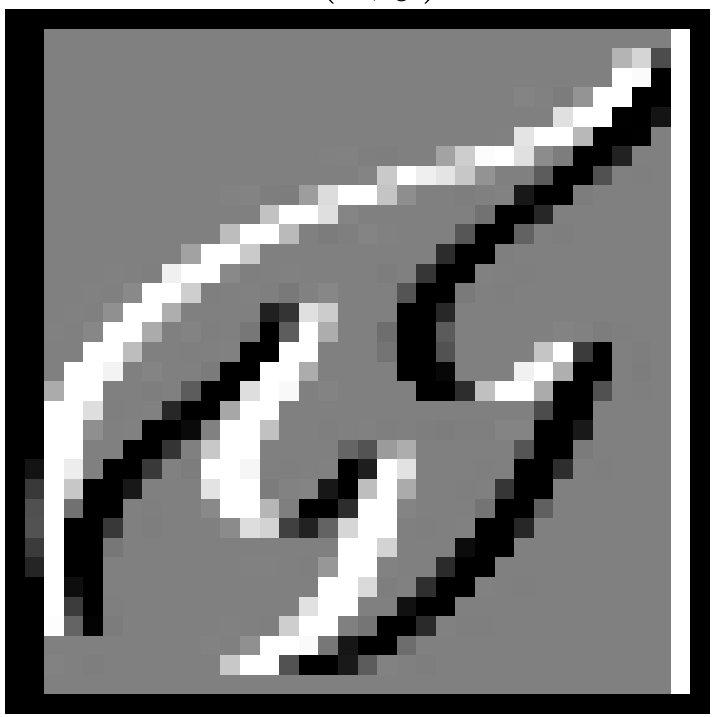
$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

Practice with kernels

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



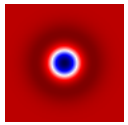
$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$



Practice with kernels

(Red = positive; blue is negative)

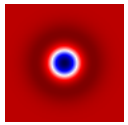
$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$

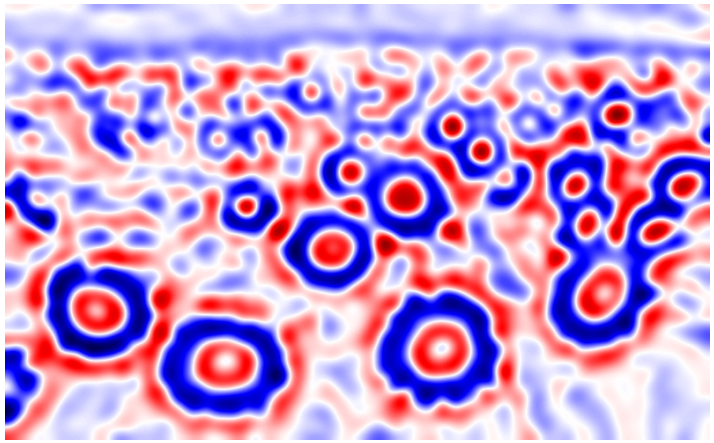


Practice with kernels

(Red = positive; blue is negative)

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$

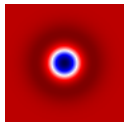


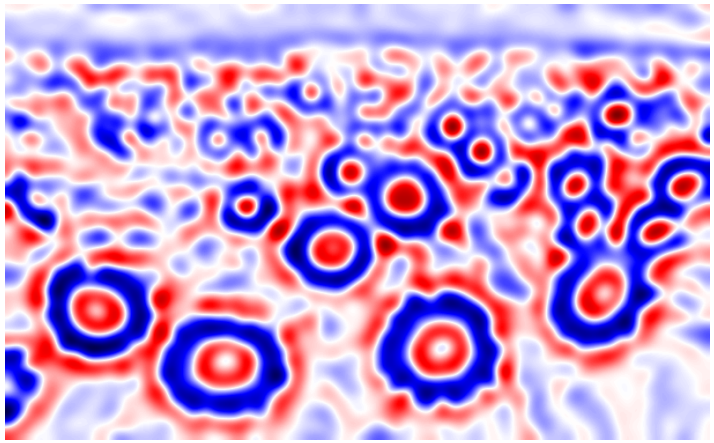


Practice with kernels

(Red = positive; blue is negative)

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$



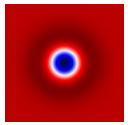


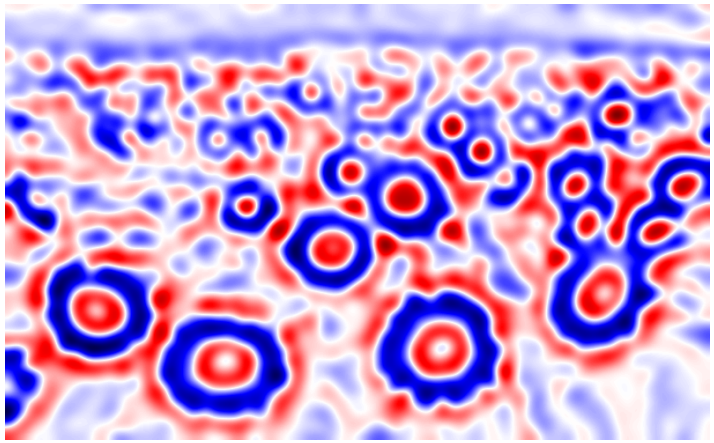
Q: How about a smaller kernel?

Practice with kernels

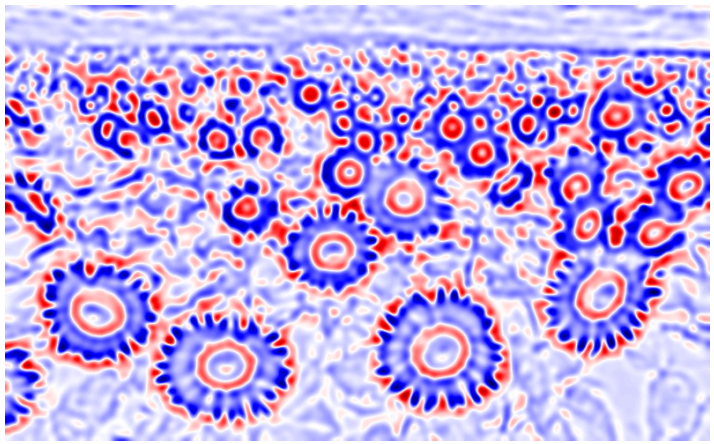
(Red = positive; blue is negative)

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$





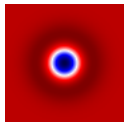
Q: How about a smaller kernel?

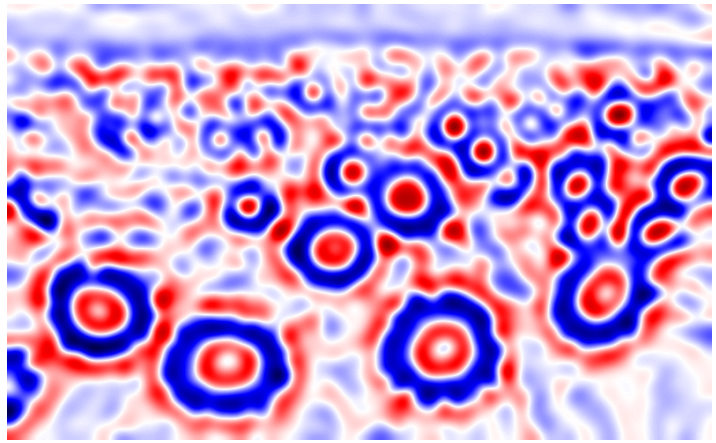


Practice with kernels

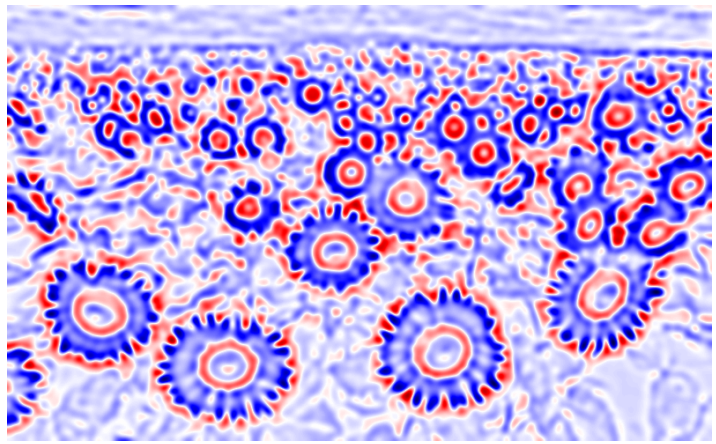
(Red = positive; blue is negative)

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$





Q: How about a smaller kernel?

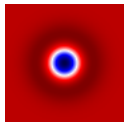


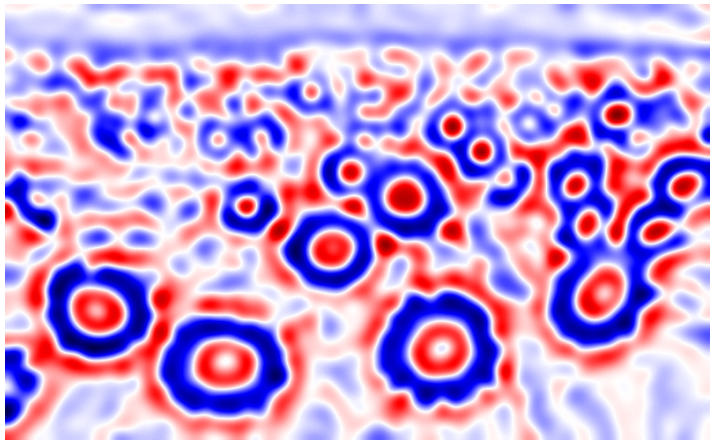
Q: How about a smaller input image?

Practice with kernels

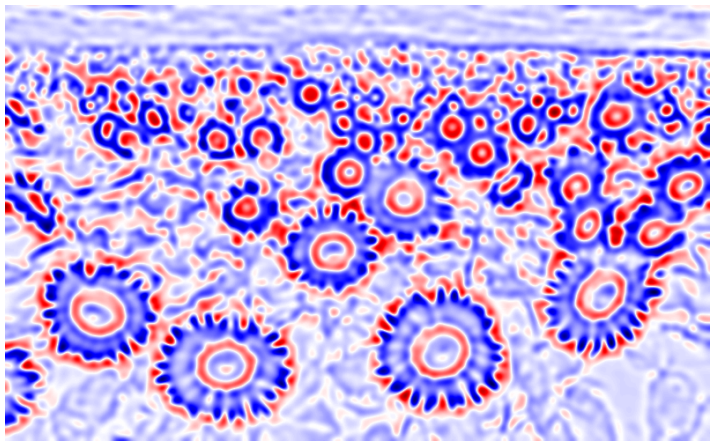
(Red = positive; blue is negative)

$F(x, y)$ $\star$ $H(x, y)$ $=$ $G(x, y)$





Q: How about a smaller kernel?



Q: How about a smaller input image?

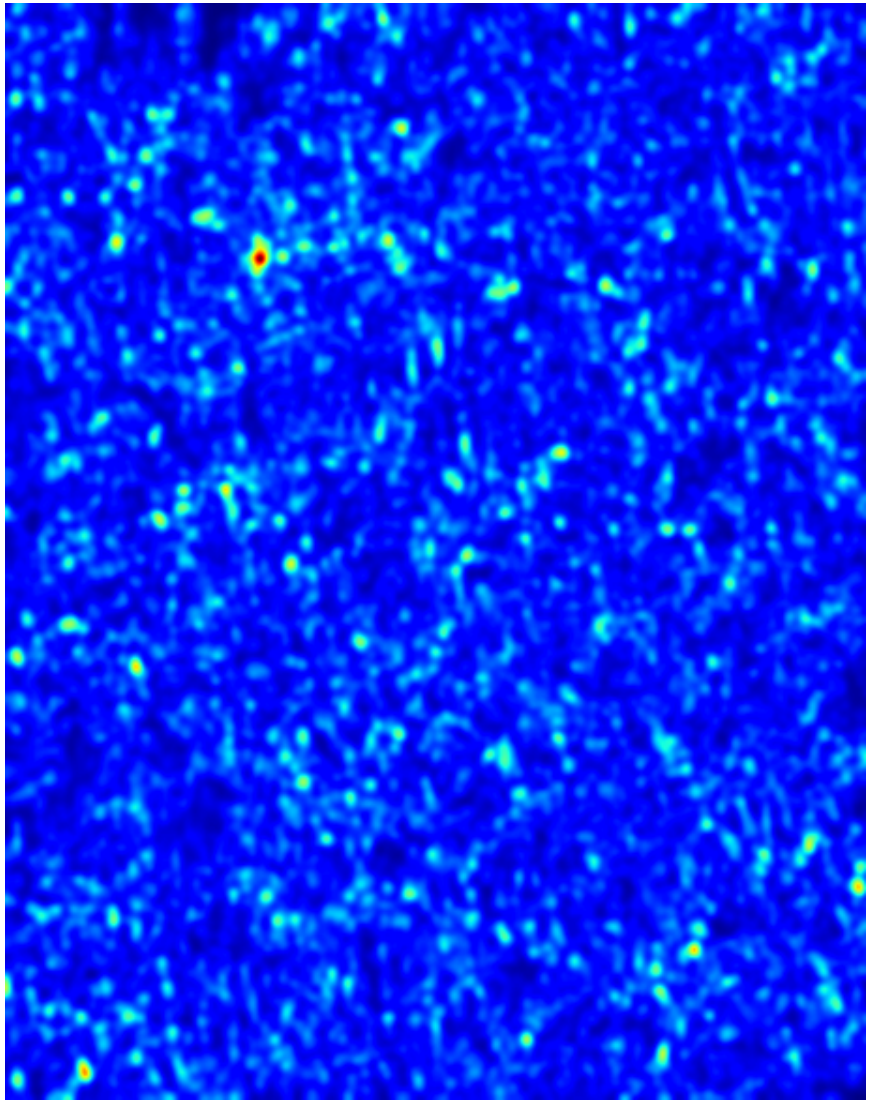
A: Smaller image = larger kernel.

Where is Waldo?

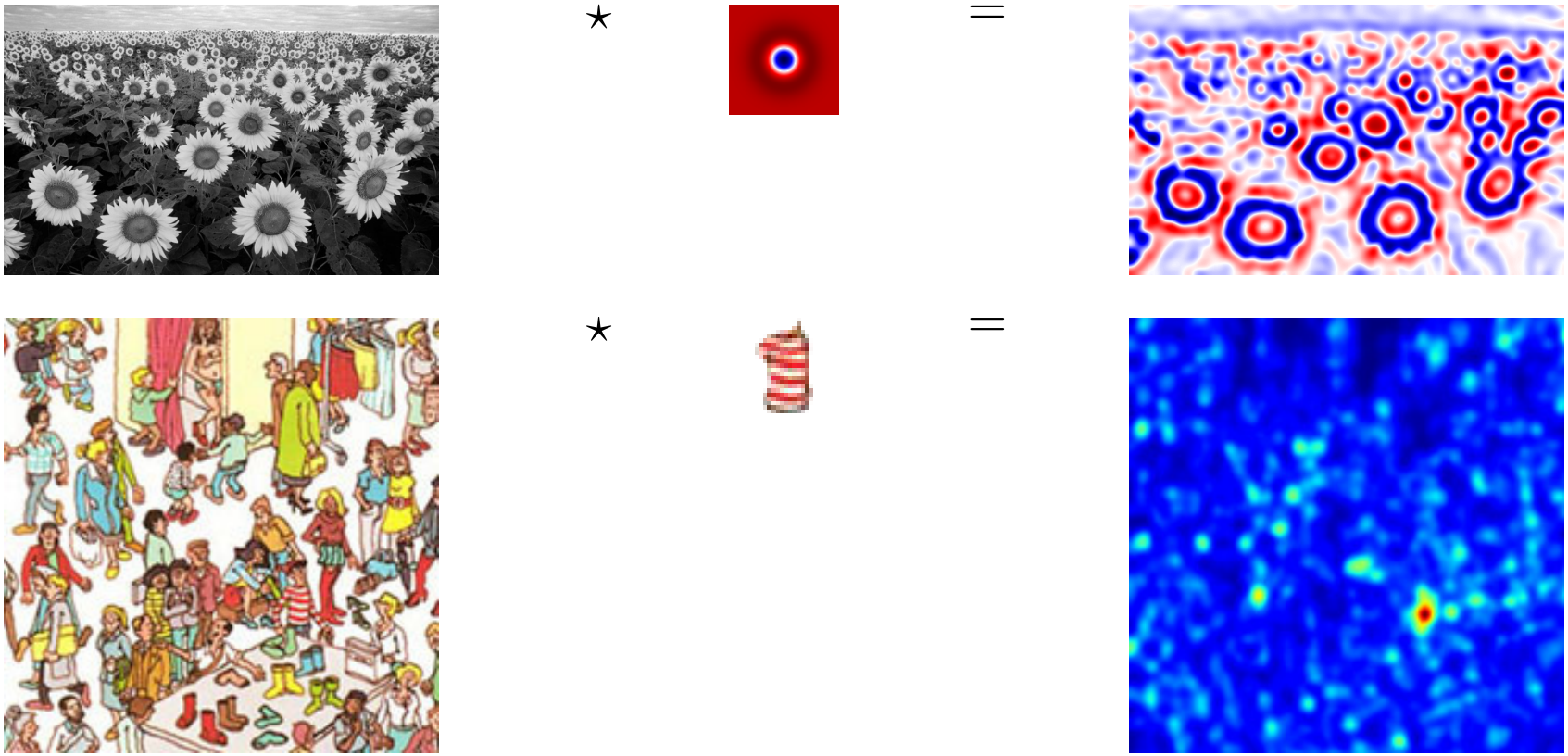
Where is Waldo?

Use this normalized kernel:

Where is Waldo?



Representation learning: Main conclusion

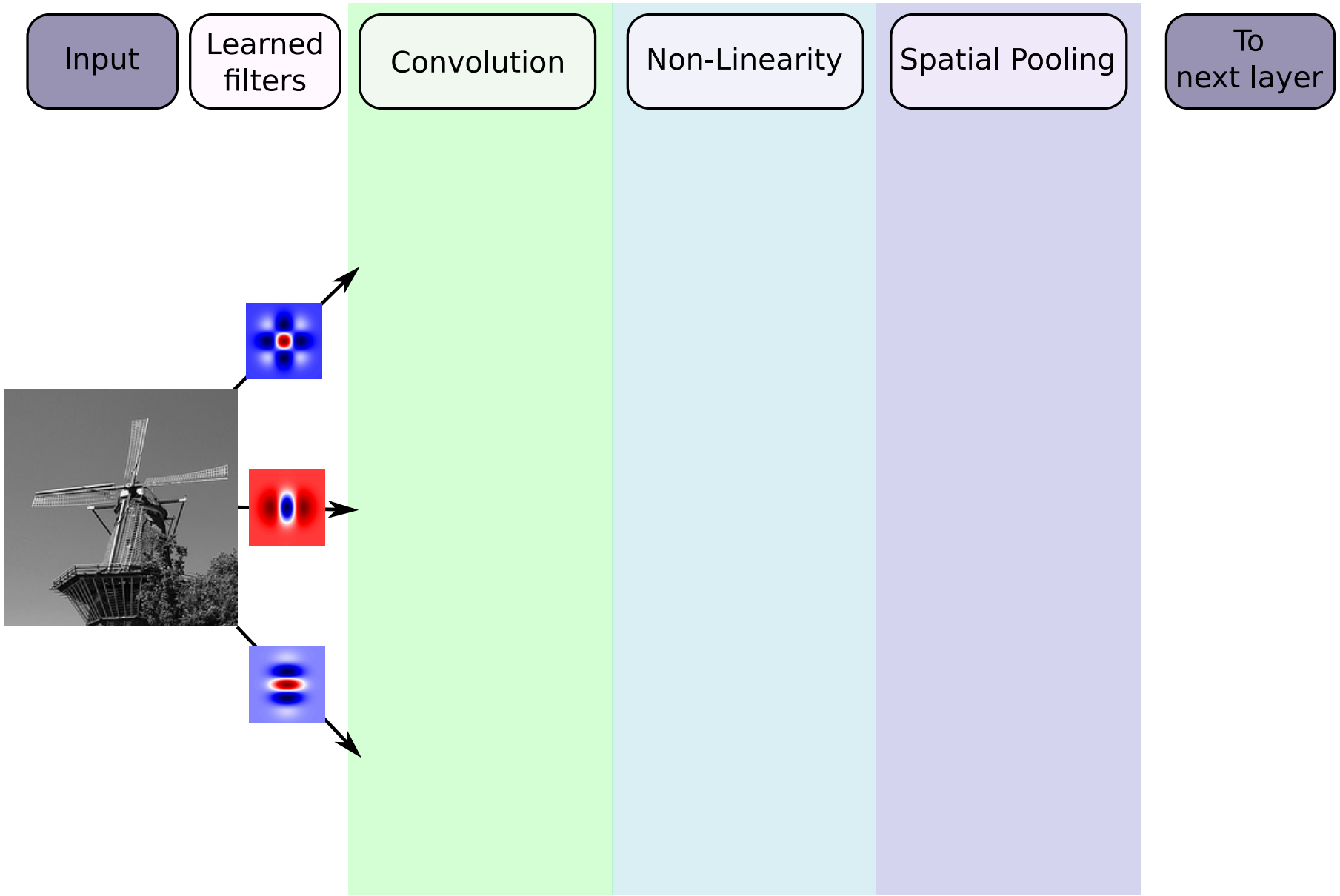


- Kernel weights are feature detectors
- Learning weights = Learning features
- Convnet learns the feature representation

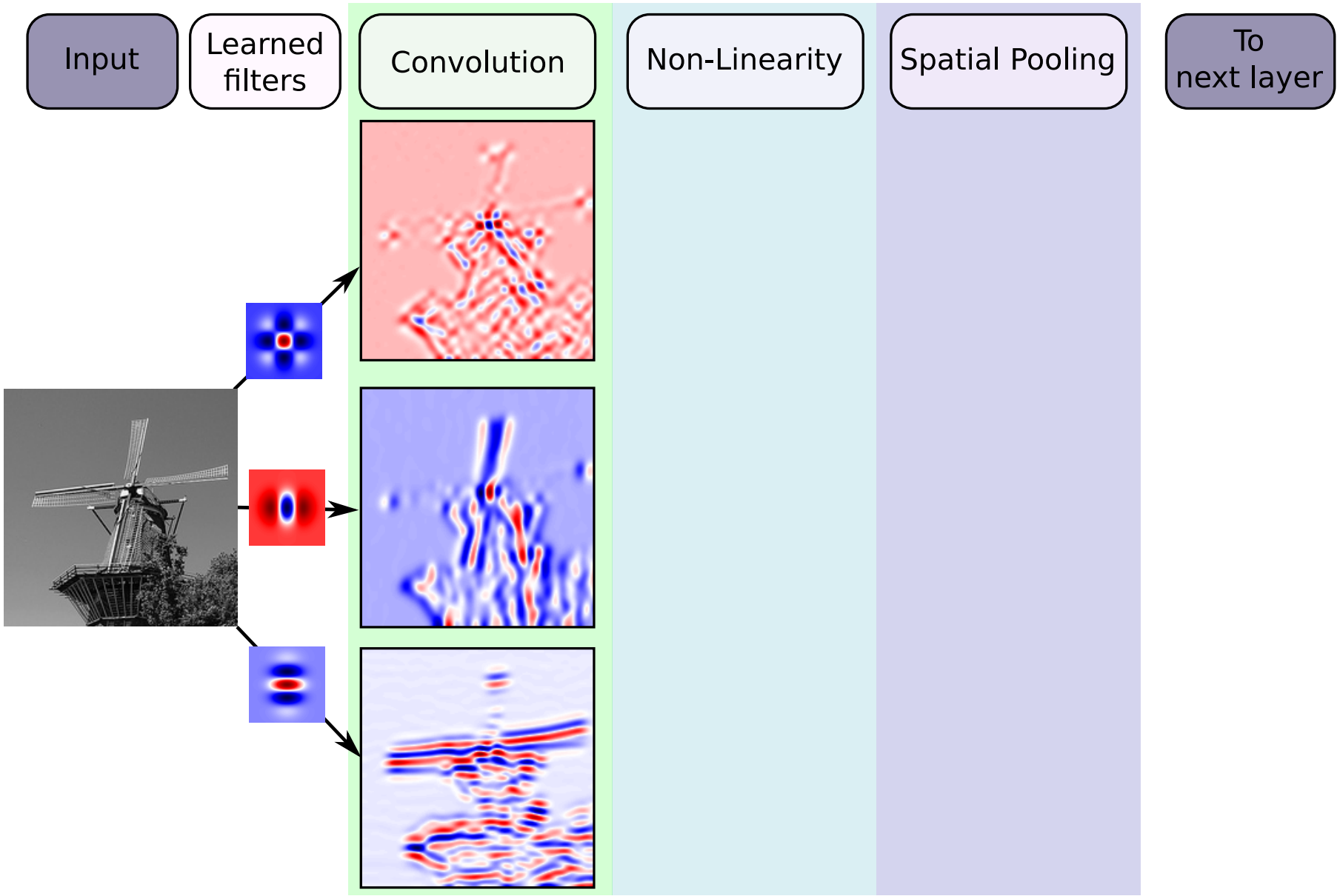
Questions?

Convolutional Network

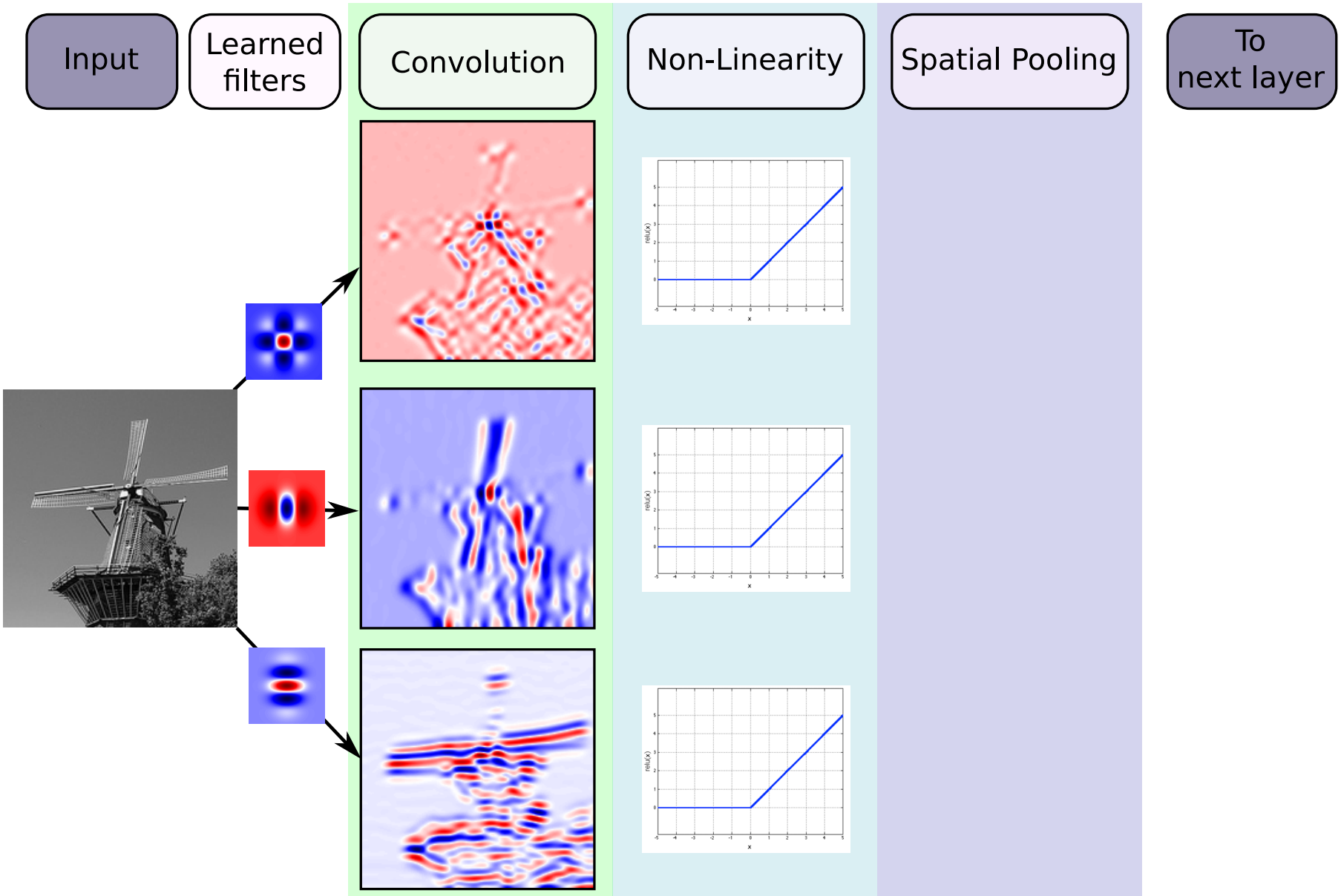
Convolutional Network: Filter (Red = positive; blue is negative)



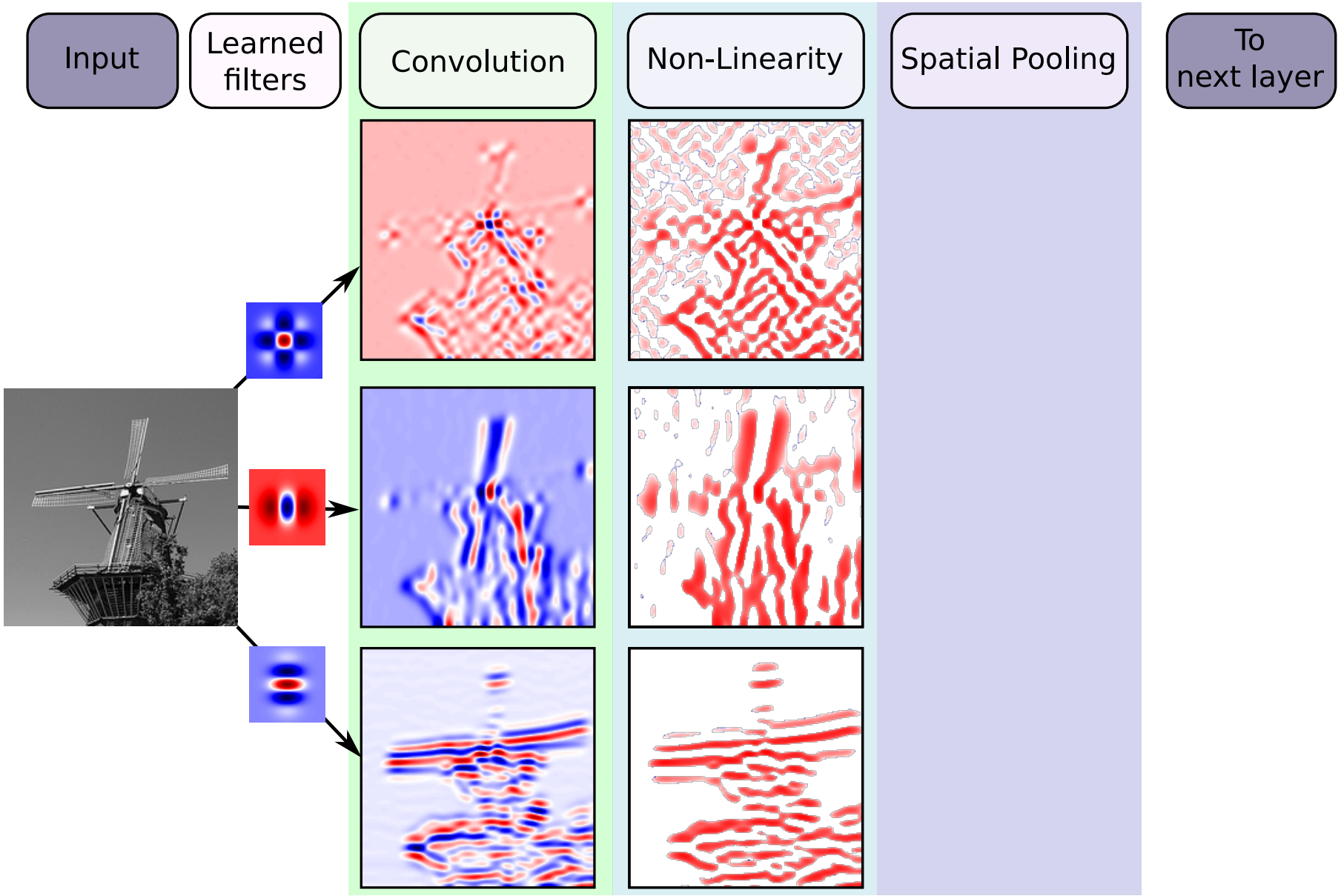
Convolutional Network: Featuremaps



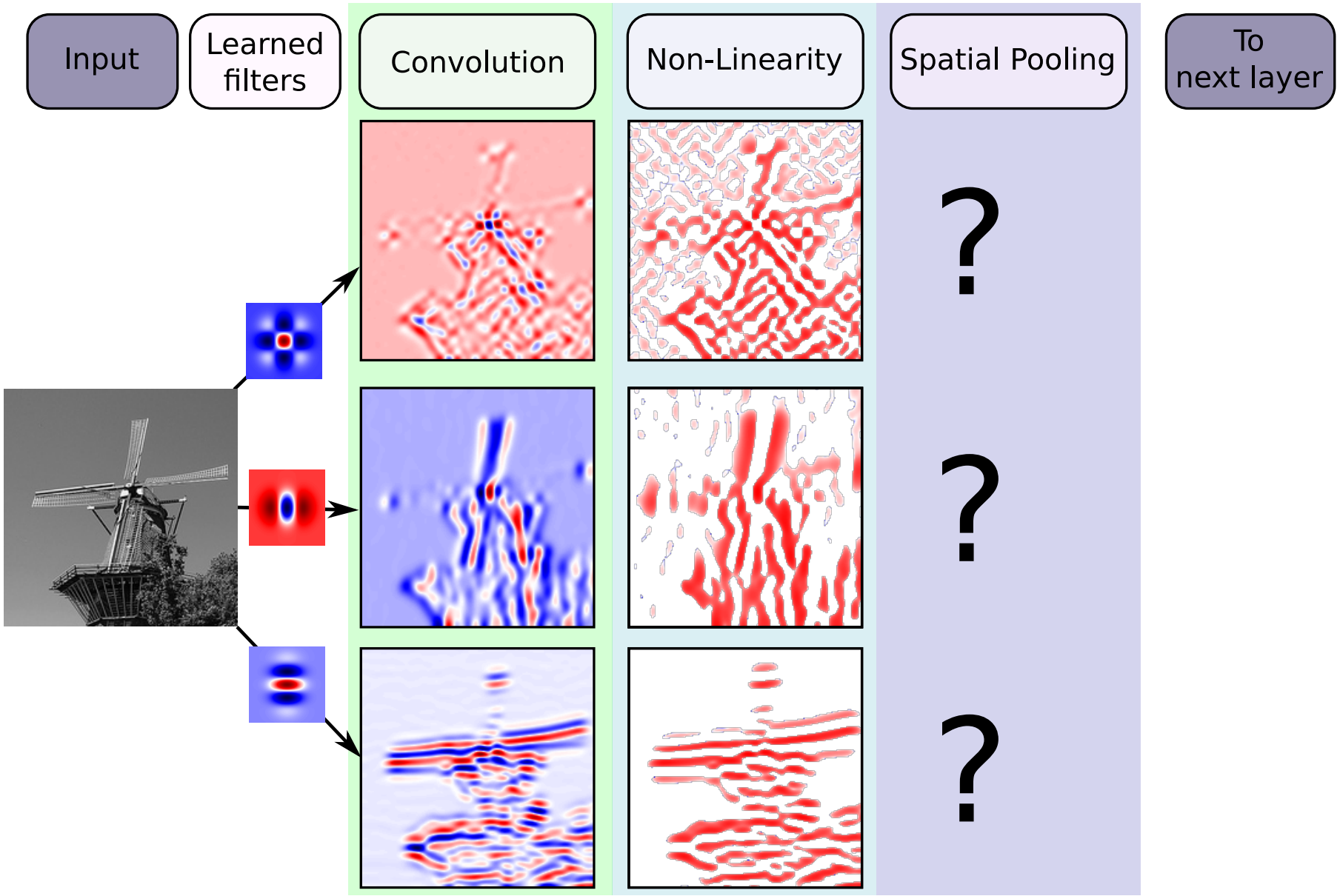
Convolutional Network: ReLu ($f(x) = \max(0, x)$)



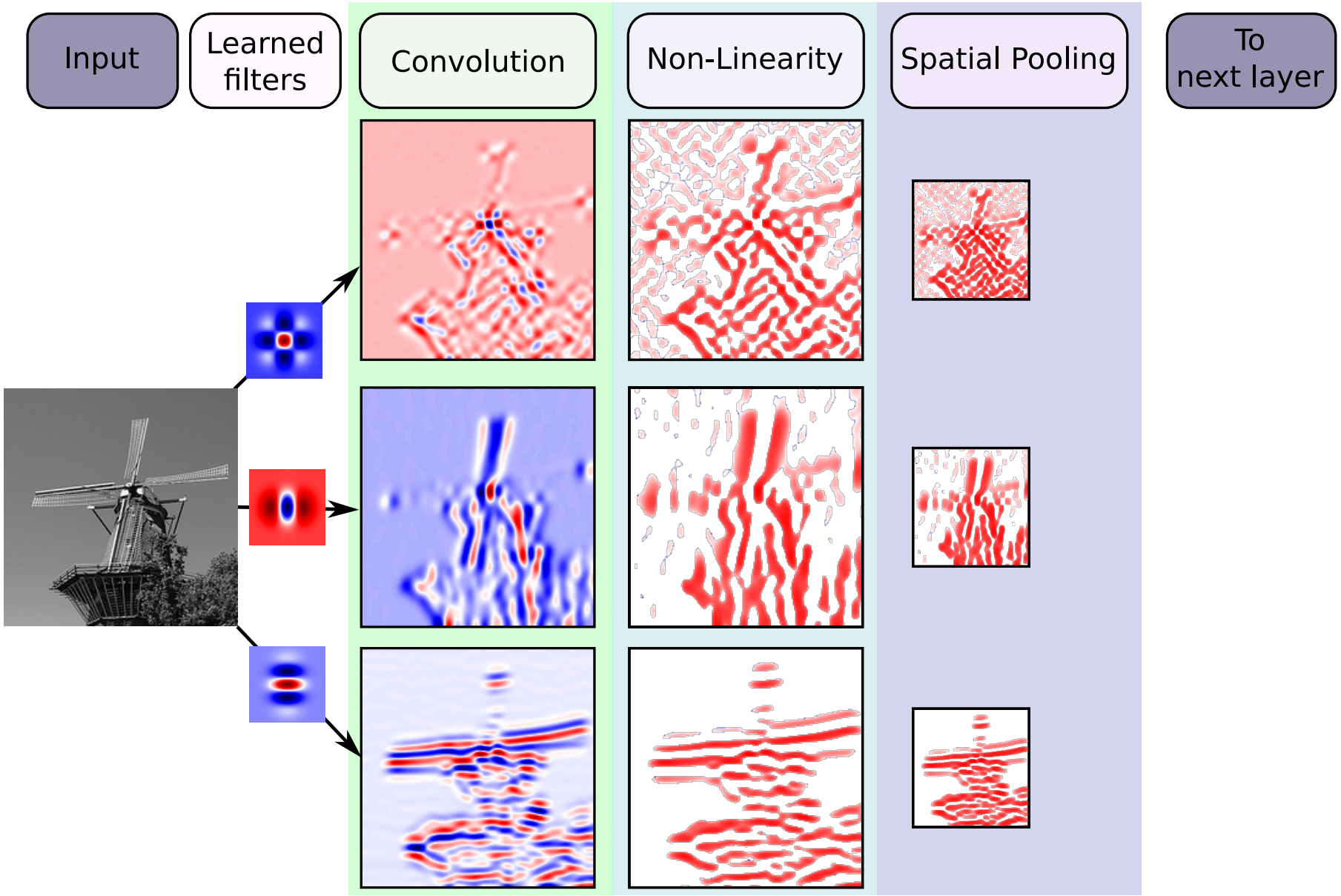
Convolutional Network: All negative values removed



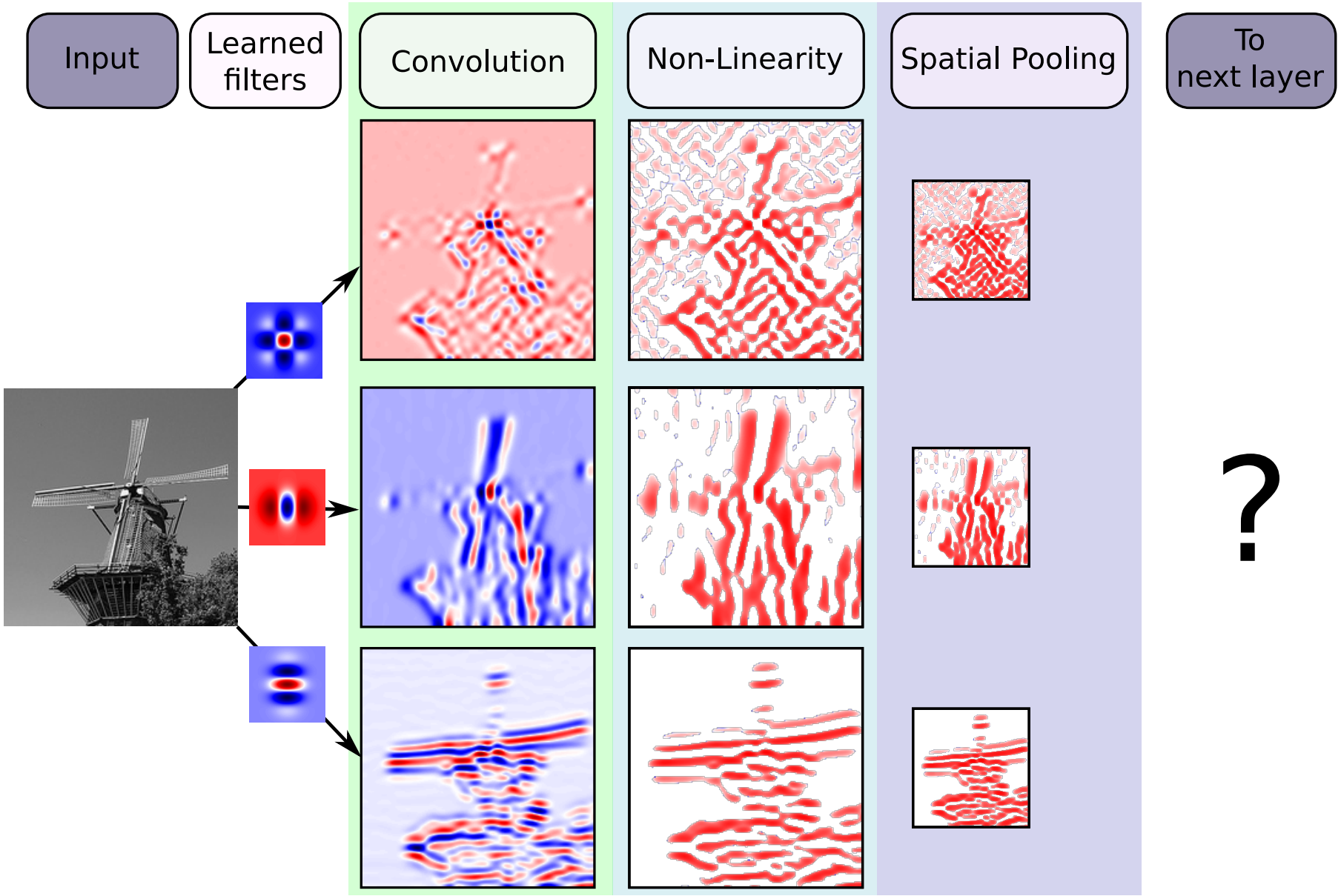
Convolutional Network: $2 \times 2\,\max$, 2×2 sub-sample



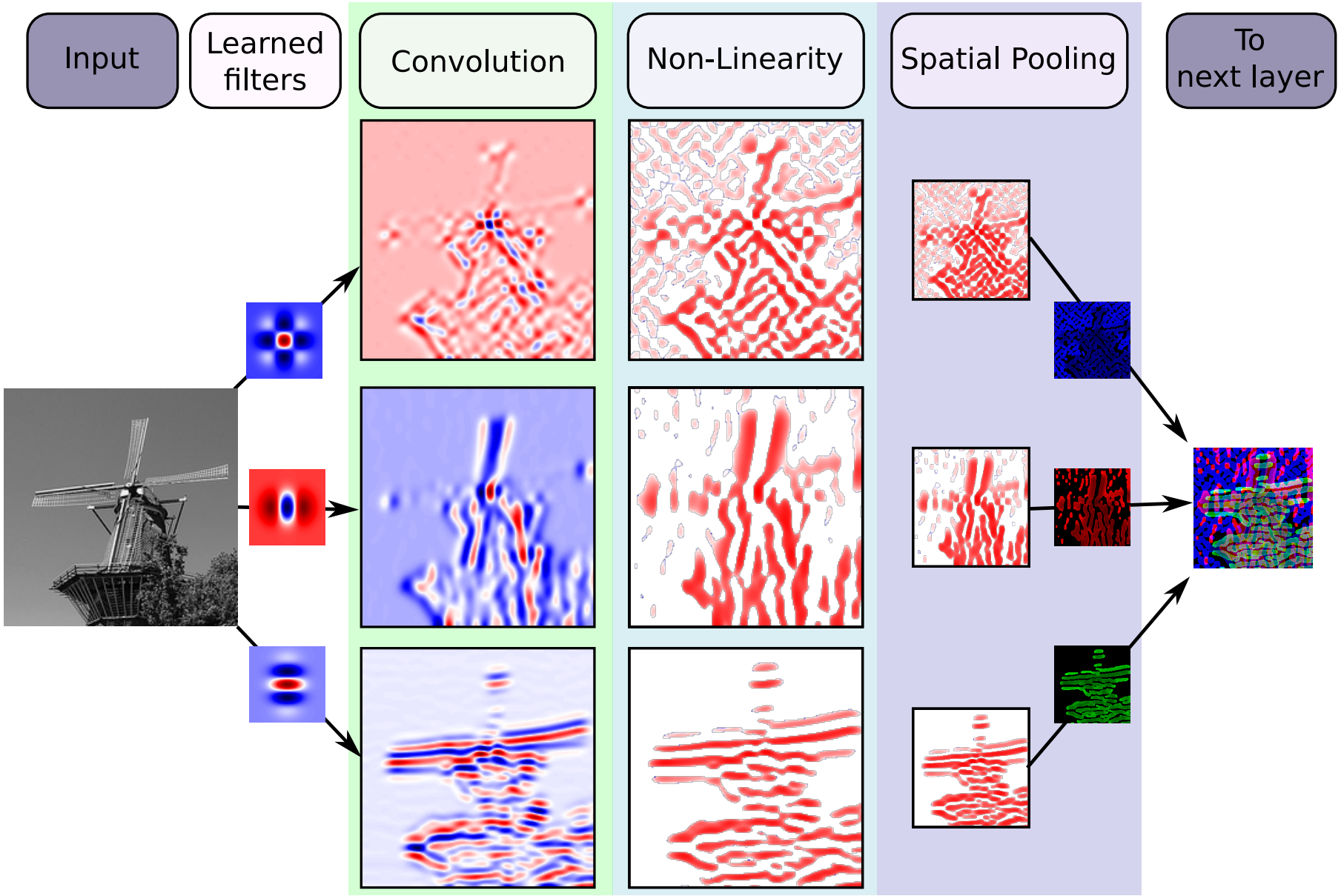
Convolutional Network: Smaller feature maps



Convolutional Network: To the next layer

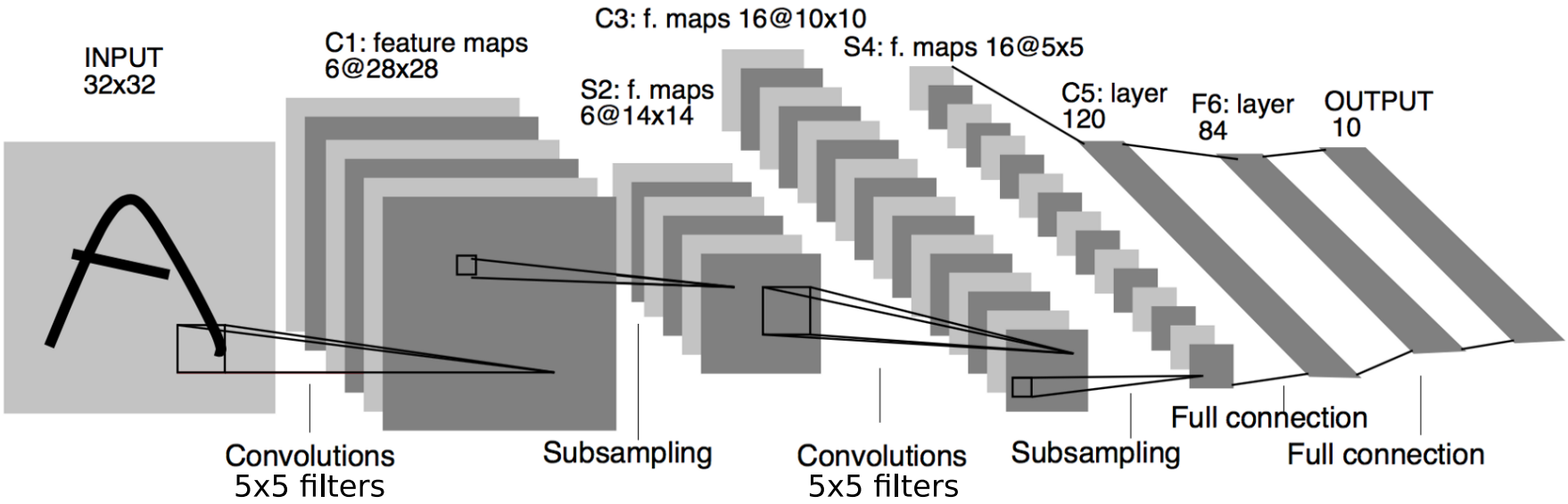


Convolutional Network: Featuremaps as new channels



Questions?

Questions?



Q: Can you explain this CNN?
(start with listing all hyper-parameters shown; explain each number shown)

Questions?

Convolutional versus fully connected

Q[1]: $[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ 0,\ +1] =$

[1] No padding

Convolutional versus fully connected

Q[1]: $[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ 0,\ +1] = [-1,\ 99,\ 100,\ 1]$

[1] No padding

Convolutional versus fully connected

Q[1]: $[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ 0,\ +1] = [-1,\ 99,\ 100,\ 1]$

- Q: Write as a (fully connected) matrix multiplication? (how many rows and cols?)

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} \cdot & \cdot & \cdots \\ \cdot & \cdot & \cdots \\ \vdots & \ddots & \cdots \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

[1] No padding

Convolutional versus fully connected

Q[1]: $[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ 0,\ +1] = [-1,\ 99,\ 100,\ 1]$

- Q: Write as a (fully connected) matrix multiplication? (how many rows and cols?)

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} \cdot & \cdot & \cdots \\ \cdot & \cdot & \cdots \\ \vdots & \ddots & \cdots \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- A: Toeplitz matrix (or diagonal-constant matrix):

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

[1] No padding

Convolutional versus fully connected

Q[1]: $[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ 0,\ +1] = [-1,\ 99,\ 100,\ 1]$

- Q: Write as a (fully connected) matrix multiplication? (how many rows and cols?)

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} \cdot & \cdot & \cdots \\ \cdot & \cdot & \cdots \\ \vdots & \ddots & \cdots \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- A: Toeplitz matrix (or diagonal-constant matrix):

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

Can be extended to 2D by doubly block-Toeplitz matrices

[1] No padding

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?
  A: CNN is –by design– a limited parameter version of fully-connected

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?
  A: CNN is –by design– a limited parameter version of fully-connected
- Q: How many parameters for the matrix vs convolution?

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?
  A: CNN is –by design– a limited parameter version of fully-connected
- Q: How many parameters for the matrix vs convolution? A: 24 vs 3

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?
  A: CNN is –by design– a limited parameter version of fully-connected
- Q: How many parameters for the matrix vs convolution? A: 24 vs 3
- Q: Wait.. Less parameters is less flexibility. Why is this a *good* thing?

Convolutional versus fully connected

Chapter 9.2

Toeplitz matrix (or diagonal-constant matrix) is the same as convolution.

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [-1,\ 99,\ 100,\ 1]$$

- Q: What are three things to notice about this matrix?
  - Sparse (many zeros)
  - Local (non-zero values occur next to each other)
  - Sharing parameters (same values repeat; the convolutional kernel values)
- Q: Difference between fully-connected and convolutional networks?
  A: CNN is –by design– a limited parameter version of fully-connected
- Q: How many parameters for the matrix vs convolution? A: 24 vs 3
- Q: Wait.. Less parameters is less flexibility. Why is this a *good* thing?
  A: Curse of dimensionality; each parameter has to be learned from data.

Questions?

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.
- Q: Let g be a translation (shift), and f be a convolution, what does this equivariance mean? (in 'plain English')

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.
- Q: Let g be a translation (shift), and f be a convolution, what does this equivariance mean? (in 'plain English')
- A: First translating, and then convolving, is the same as first convolving and then translating.

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.
- Q: Let g be a translation (shift), and f be a convolution, what does this equivariance mean? (in 'plain English')
- A: First translating, and then convolving, is the same as first convolving and then translating.
- Q: How does translation equivariance relate to images?

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.
- Q: Let g be a translation (shift), and f be a convolution, what does this equivariance mean? (in 'plain English')
- A: First translating, and then convolving, is the same as first convolving and then translating.
- Q: How does translation equivariance relate to images?
- A: Camera position is accidental, objects may appear anywhere

Equivariance

Chapter 9.2

- Q: What do you notice here?

$$[100,\ 101,\ 99,\ 200,\ 199,\ 201] \star [-1,\ +1] = [1,\ -2,\ 101,\ -1,\ 2]$$
$$[101,\ 99,\ 200,\ 199,\ 201,\ 201] \star [-1,\ +1] = [-2,\ 101,\ -1,\ 2,\ 0]$$

- A: If the input shifts to the left, the output shifts to the left (equivariance)
- Function $f()$ is equivariant to $g()$: $f(g(x)) = g(f(x))$.
- Q: Let g be a translation (shift), and f be a convolution, what does this equivariance mean? (in 'plain English')
- A: First translating, and then convolving, is the same as first convolving and then translating.
- Q: How does translation equivariance relate to images?
- A: Camera position is accidental, objects may appear anywhere

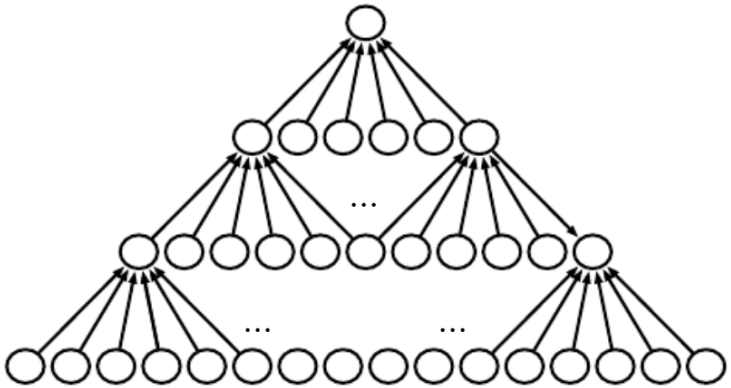
Add prior knowledge (convolution) to deep nets: huge gain in params and compute

Questions?

Padding

Chapter 9.5, fig 9.13

Lets do a convolution with size 6.
(Start at the bottom)

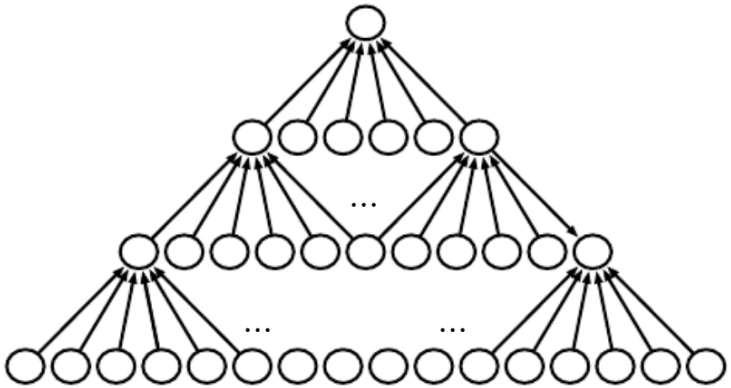


- Q: How to prevent shrinking?

Padding

Chapter 9.5, fig 9.13

Lets do a convolution with size 6.
(Start at the bottom)



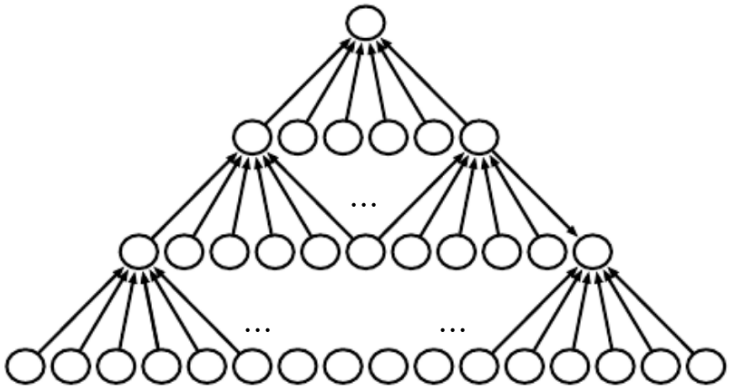
- Q: How to prevent shrinking?

- A: Padding

Padding

Chapter 9.5, fig 9.13

Lets do a convolution with size 6.
(Start at the bottom)



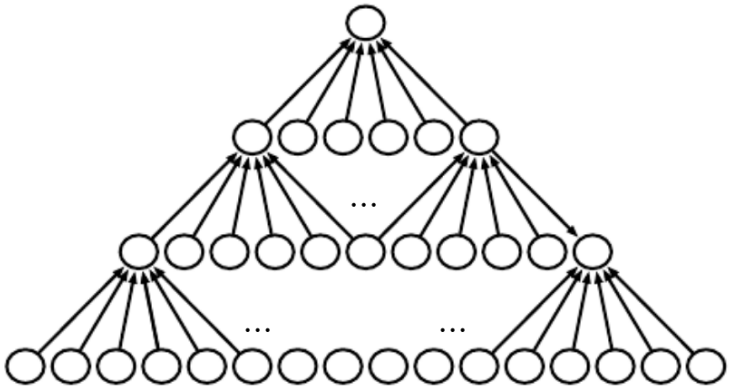
- Q: How to prevent shrinking?

- A: Padding
- Q: What do you notice?

Padding

Chapter 9.5, fig 9.13

Lets do a convolution with size 6.
(Start at the bottom)



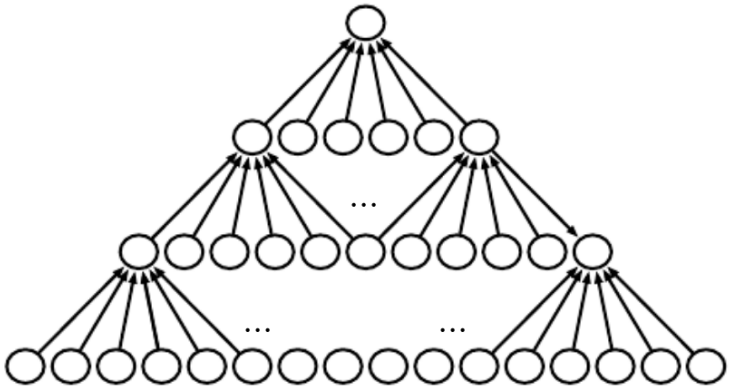
- Q: How to prevent shrinking?

- A: Padding
- Q: What do you notice?
- A: Unequal border: 3 left, 2 right.

Padding

Chapter 9.5, fig 9.13

Lets do a convolution with size 6.
(Start at the bottom)



- Q: How to prevent shrinking?

- A: Padding
- Q: What do you notice?
- A: Unequal border: 3 left, 2 right.
- A: Kernel is even, has no center.

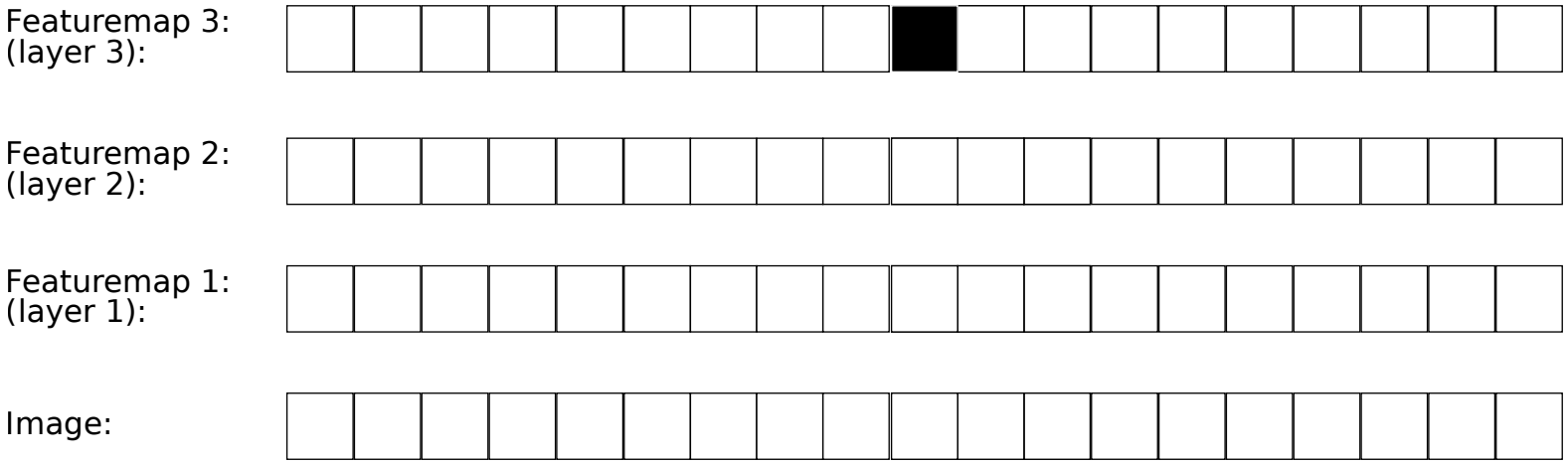
Questions?

Receptive field: Convolution

Chapter 9.5

Example: Convolution of size 3×1.

- Q: how much does the shaded activation in FM3 'see' of the input image?

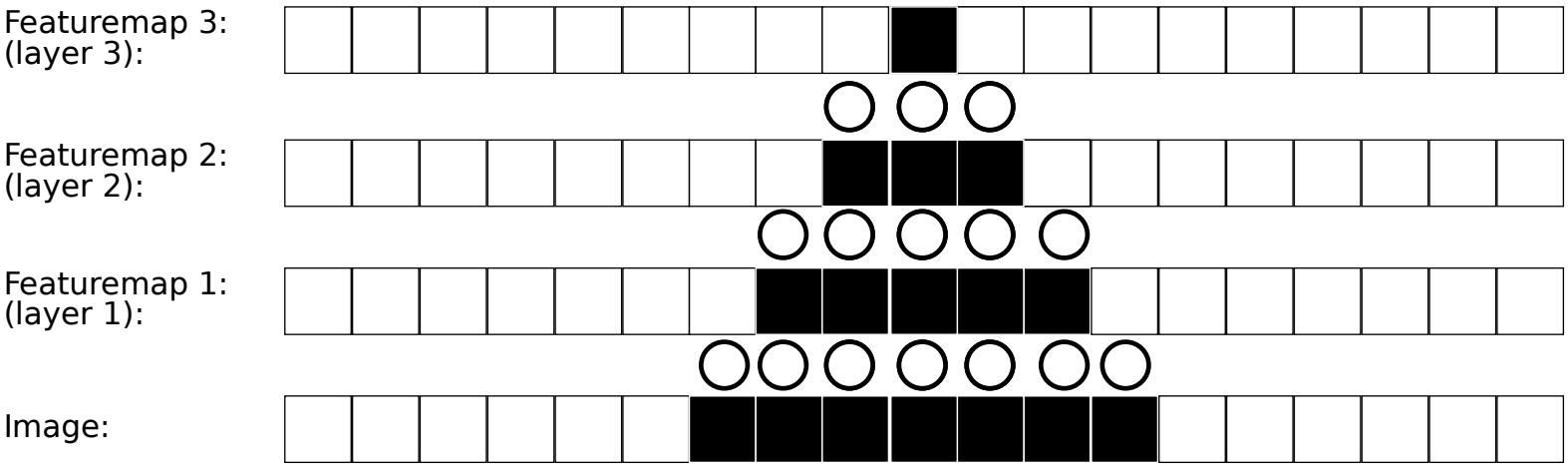


Receptive field: Convolution

Chapter 9.5

Example: Convolution of size 3×1.

- Q: how much does the shaded activation in FM3 'see' of the input image?

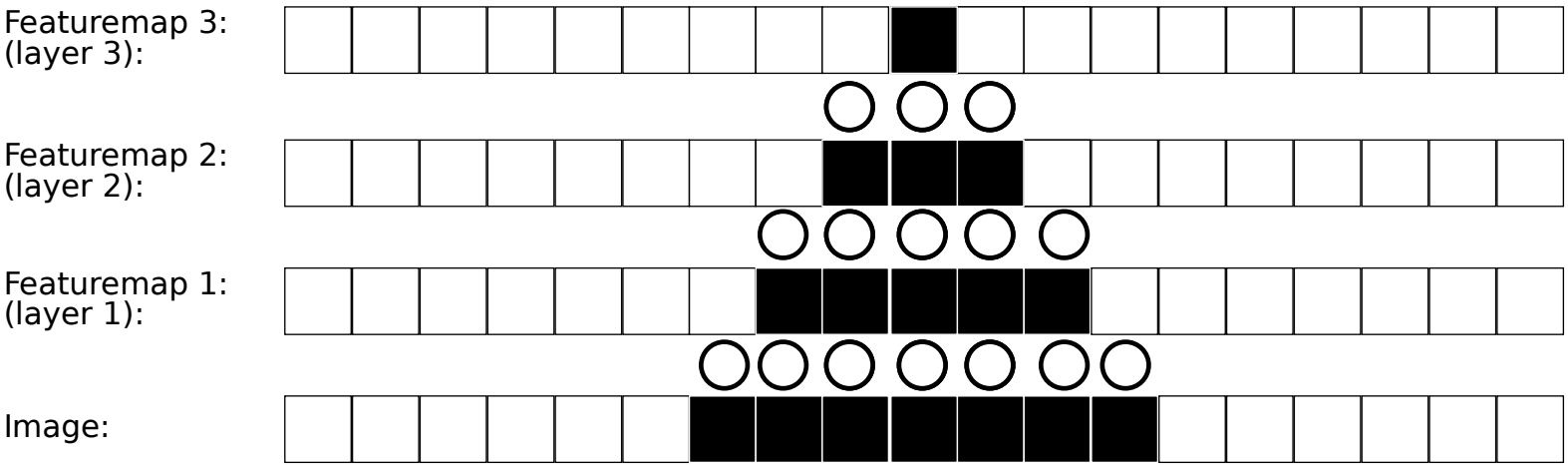


Receptive field: Convolution

Chapter 9.5

Example: Convolution of size 3×1.

- Q: how much does the shaded activation in FM3 'see' of the input image?



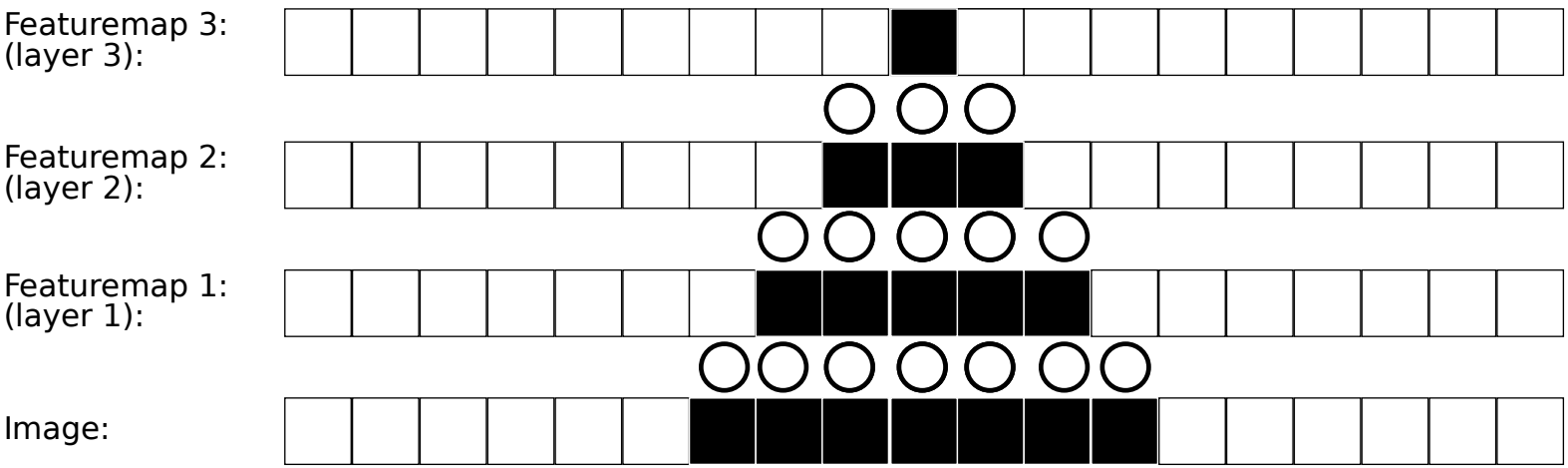
- A: 7 pixels

Receptive field: Convolution

Chapter 9.5

Example: Convolution of size 3×1.

- Q: how much does the shaded activation in FM3 'see' of the input image?



- A: 7 pixels

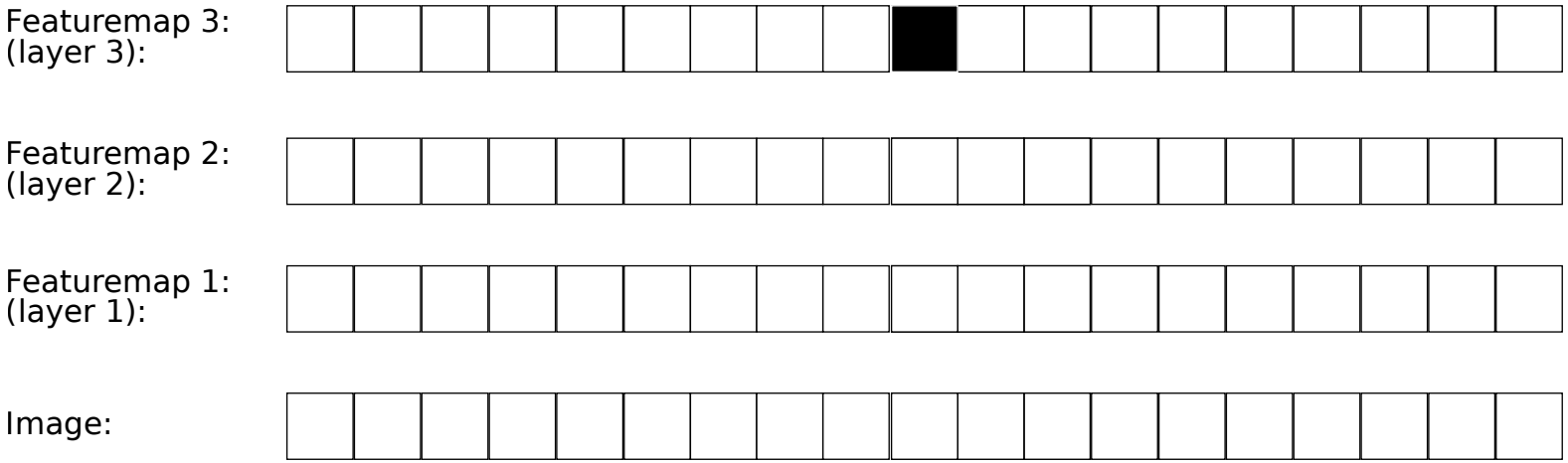
Activations in deeper layers see more of the input (additive in nr of layers).

Receptive field: Striding/sub-sampling

Chapter 9.5

Example: Convolution of size 3×1, every time sub-sampling by a factor of 2.

- Q: how much does the shaded activation in FM3 'see' of the image?

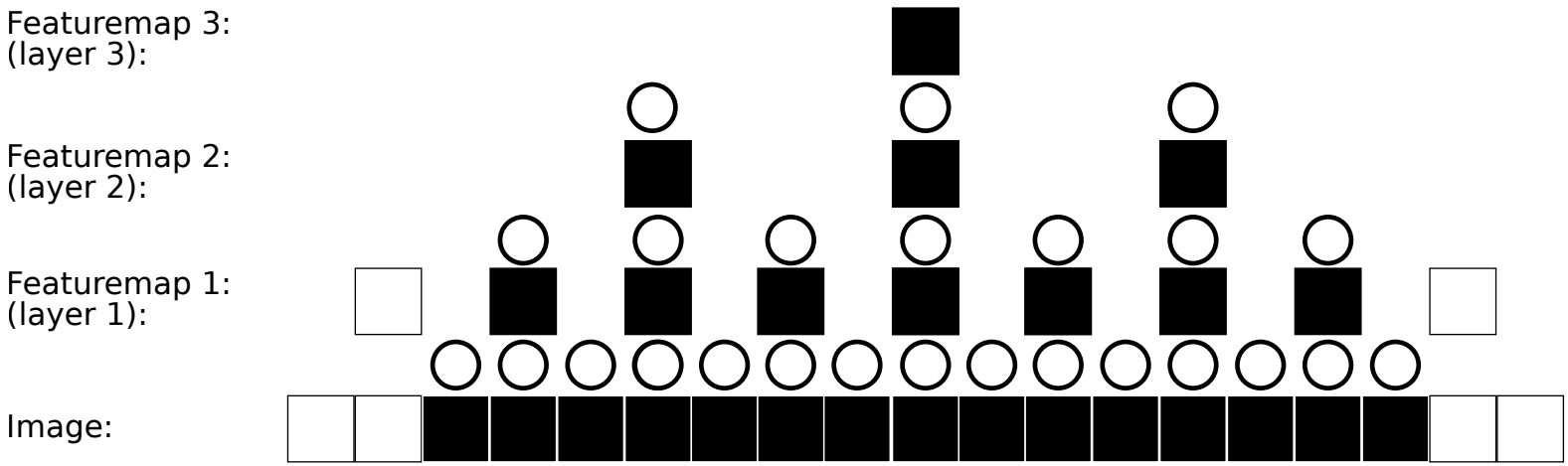


Receptive field: Striding/sub-sampling

Chapter 9.5

Example: Convolution of size 3×1, every time sub-sampling by a factor of 2.

- Q: how much does the shaded activation in FM3 'see' of the image?

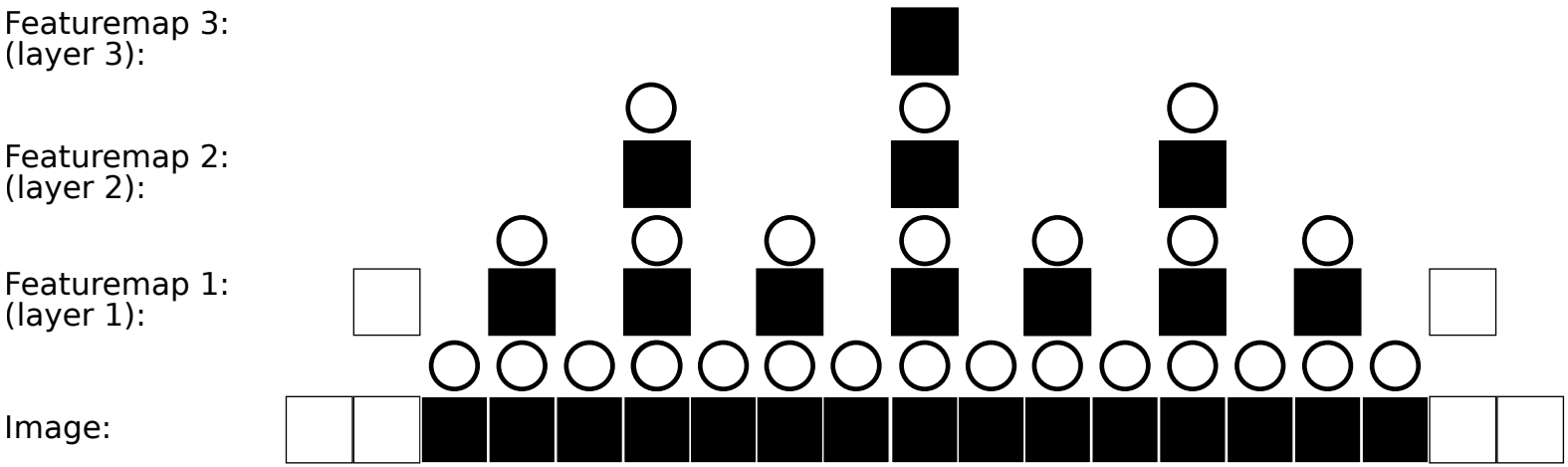


Receptive field: Striding/sub-sampling

Chapter 9.5

Example: Convolution of size 3×1, every time sub-sampling by a factor of 2.

- Q: how much does the shaded activation in FM3 'see' of the image?



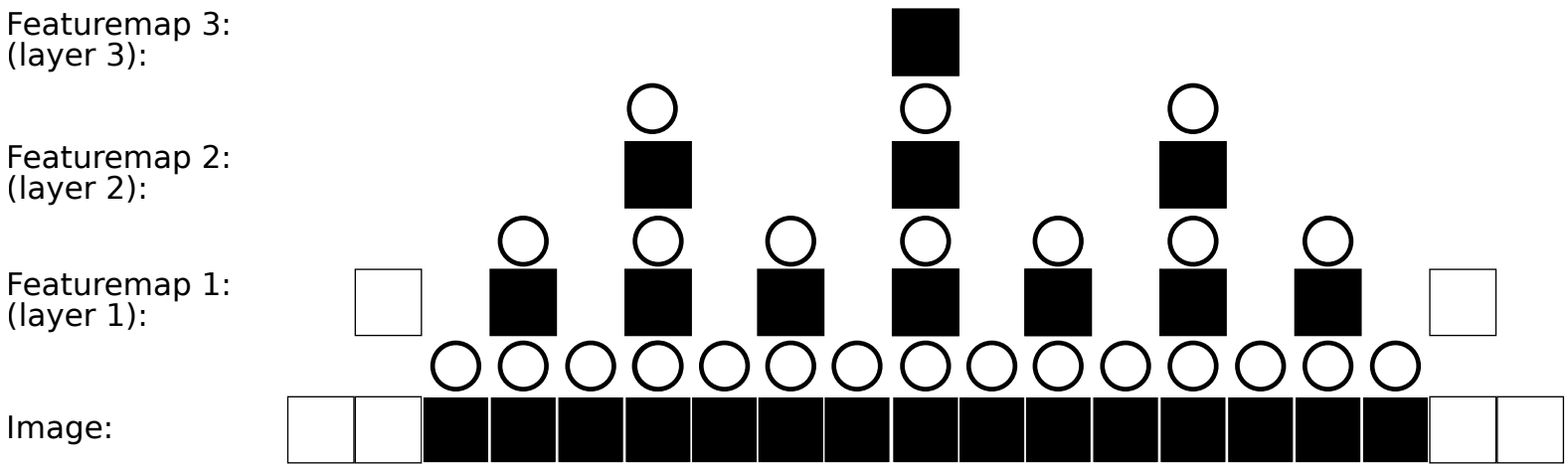
- A: 15 pixels

Receptive field: Striding/sub-sampling

Chapter 9.5

Example: Convolution of size 3×1, every time sub-sampling by a factor of 2.

- Q: how much does the shaded activation in FM3 'see' of the image?



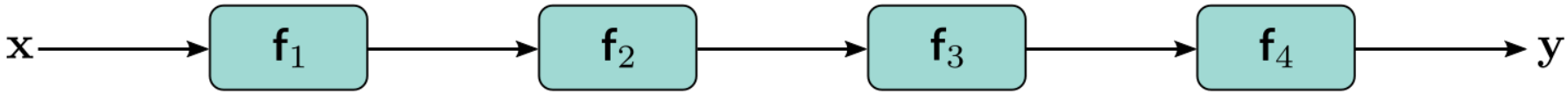
- A: 15 pixels

Sub-sampling allows to quickly 'see' more of the image (multiple of the stride).

Questions?

Going deeper (more layers): Residual/skip connections

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.1/11.4

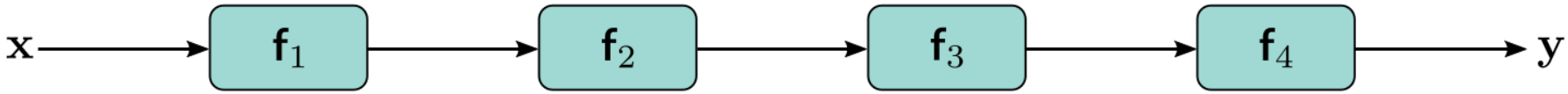


Deep Networks: a function chain $f(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

Q: What is the derivative? A: $\frac{df}{d\mathbf{x}} =$

Going deeper (more layers): Residual/skip connections

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.1/11.4



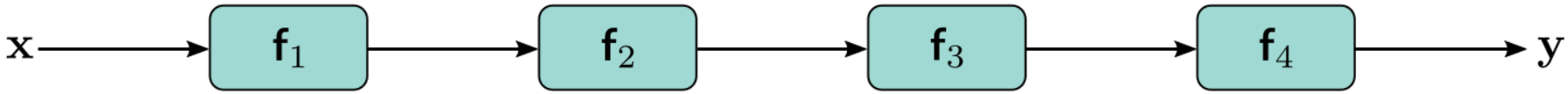
Deep Networks: a function chain $f(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

Q: What is the derivative? A: $\frac{df}{d\mathbf{x}} = \frac{df_4}{df_3}\frac{df_3}{df_2}\frac{df_2}{df_1}\frac{df_1}{d\mathbf{x}}$

Q: What if one of the derivatives is very close to 0?

Going deeper (more layers): Residual/skip connections

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.1/11.4



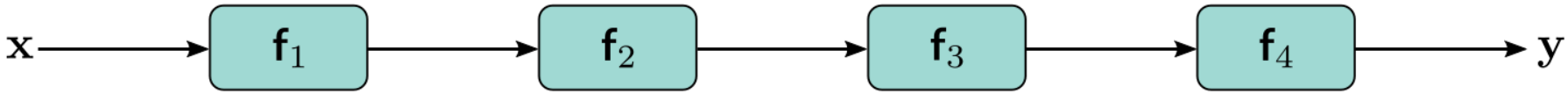
Deep Networks: a function chain $f(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

Q: What is the derivative? A: $\frac{df}{d\mathbf{x}} = \frac{df_4}{df_3}\frac{df_3}{df_2}\frac{df_2}{df_1}\frac{df_1}{d\mathbf{x}}$

Q: What if one of the derivatives is very close to 0? A: Whole chain becomes zero.

Going deeper (more layers): Residual/skip connections

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.1/11.4



Deep Networks: a function chain $f(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

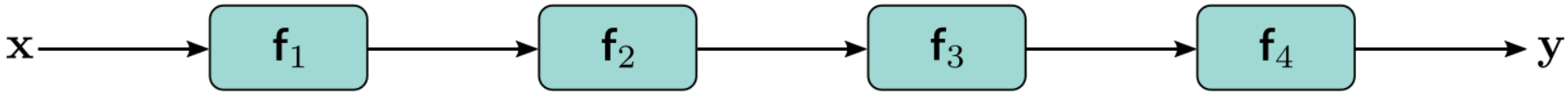
Q: What is the derivative? A: $\frac{df}{d\mathbf{x}} = \frac{df_4}{df_3}\frac{df_3}{df_2}\frac{df_2}{df_1}\frac{df_1}{d\mathbf{x}}$

Q: What if one of the derivatives is very close to 0? A: Whole chain becomes zero.

Instead, sum an ensemble of sub-networks.

Going deeper (more layers): Residual/skip connections

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.1/11.4



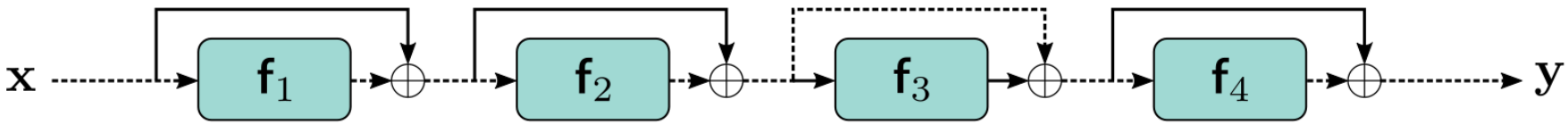
Deep Networks: a function chain $f(\mathbf{x}) = f_4(f_3(f_2(f_1(\mathbf{x}))))$

Q: What is the derivative? A: $\frac{df}{d\mathbf{x}} = \frac{df_4}{df_3}\frac{df_3}{df_2}\frac{df_2}{df_1}\frac{df_1}{d\mathbf{x}}$

Q: What if one of the derivatives is very close to 0? A: Whole chain becomes zero.

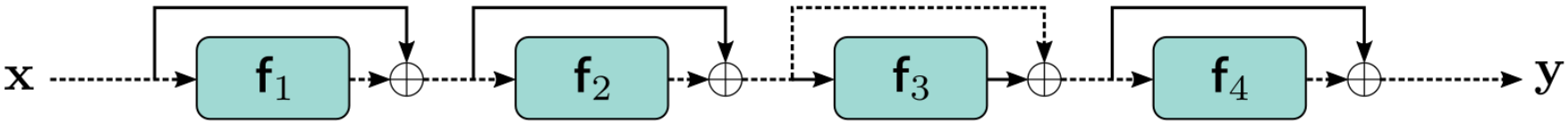
Instead, sum an ensemble of sub-networks. Residual/skip connections:

$$\begin{aligned}\text{Featuremap } \mathbf{h_1} &= \mathbf{x} + f_1(\mathbf{x}) \\ \text{Featuremap } \mathbf{h_2} &= \mathbf{h_1} + f_2(\mathbf{h_1}) \\ \text{Featuremap } \mathbf{h_3} &= \mathbf{h_2} + f_3(\mathbf{h_2}) \\ f(\mathbf{x}) &= \mathbf{h_3} + f_4(\mathbf{h_3})\end{aligned}$$



Going deeper by adding more layers

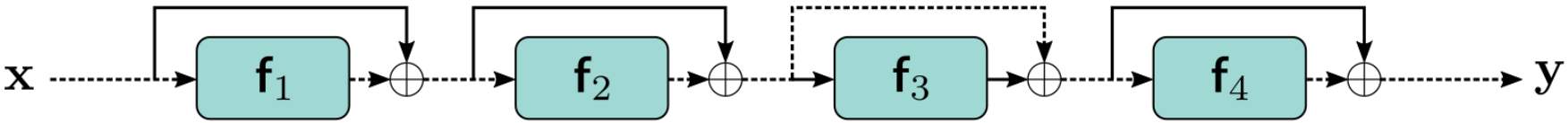
'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4



$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Going deeper by adding more layers

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4

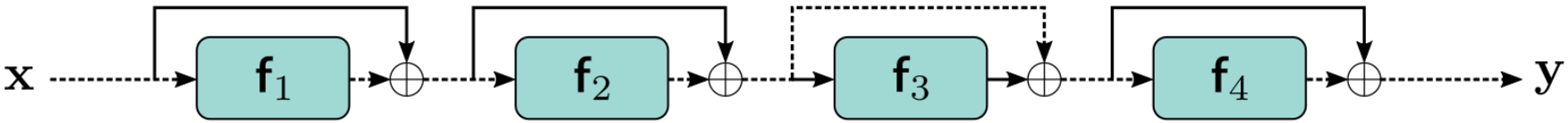


$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Q: If $\mathbf{h_1}$ has size d; what size is $f_2(\mathbf{h_1})$?

Going deeper by adding more layers

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4

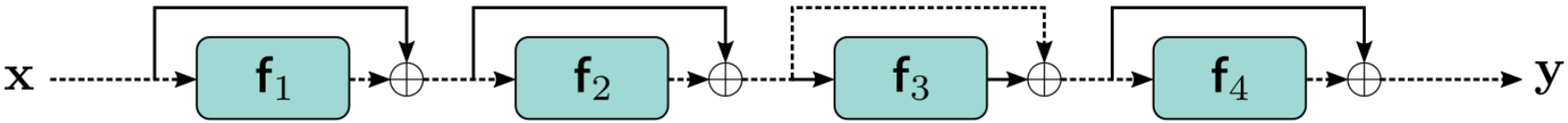


$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Q: If $\mathbf{h_1}$ has size d; what size is $f_2(\mathbf{h_1})$? A: Same (to allow $\mathbf{h_1} + f_2(\mathbf{h_1})$)

Going deeper by adding more layers

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4



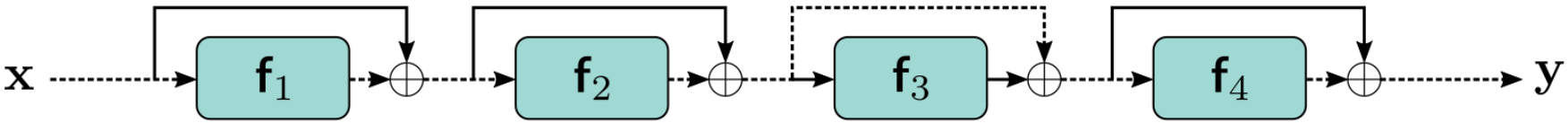
$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Q: If $\mathbf{h_1}$ has size d; what size is $f_2(\mathbf{h_1})$? A: Same (to allow $\mathbf{h_1} + f_2(\mathbf{h_1})$)

Q: Why is residual/skip connections a summed ensemble ? (write out the $\mathbf{h}$s in terms of f)

Going deeper by adding more layers

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4



$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Q: If $\mathbf{h_1}$ has size d; what size is $f_2(\mathbf{h_1})$? A: Same (to allow $\mathbf{h_1} + f_2(\mathbf{h_1})$)

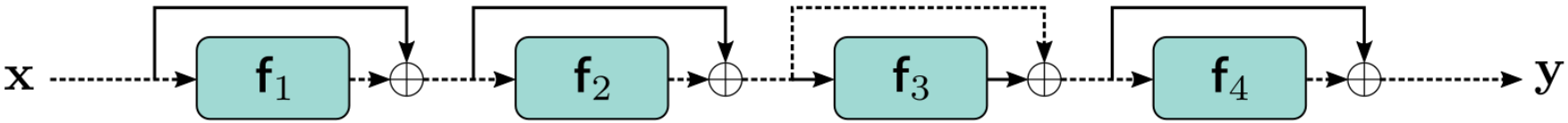
Q: Why is residual/skip connections a summed ensemble ? (write out the $\mathbf{h}$s in terms of f) A:

$$\begin{aligned} f(\mathbf{x}) = \mathbf{x} &+ f_1(\mathbf{x}) \\ &+ f_2(\mathbf{x} + f_1(\mathbf{x})) \\ &+ f_3(\mathbf{x} + f_1(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x}))) \\ &+ f_4\Big(\mathbf{x} + f^{(1)}(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x})) + f_3(\mathbf{x} + f_1(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x})))\Big) \end{aligned}$$

Q: How many summed sub-networks?

Going deeper by adding more layers

'Understanding Deep Learning' chapter 11 (https://udlbook.github.io/udlbook/), fig 11.4



$$\mathbf{h_1} = \mathbf{x} + f_1(\mathbf{x}); \quad \mathbf{h_2} = \mathbf{h_1} + f_2(\mathbf{h_1}); \quad \mathbf{h_3} = \mathbf{h_2} + f_3(\mathbf{h_2}); \quad f(\mathbf{x}) = \mathbf{h_3} + f_4(\mathbf{h_3}).$$

Q: If $\mathbf{h_1}$ has size d; what size is $f_2(\mathbf{h_1})$? A: Same (to allow $\mathbf{h_1} + f_2(\mathbf{h_1})$)

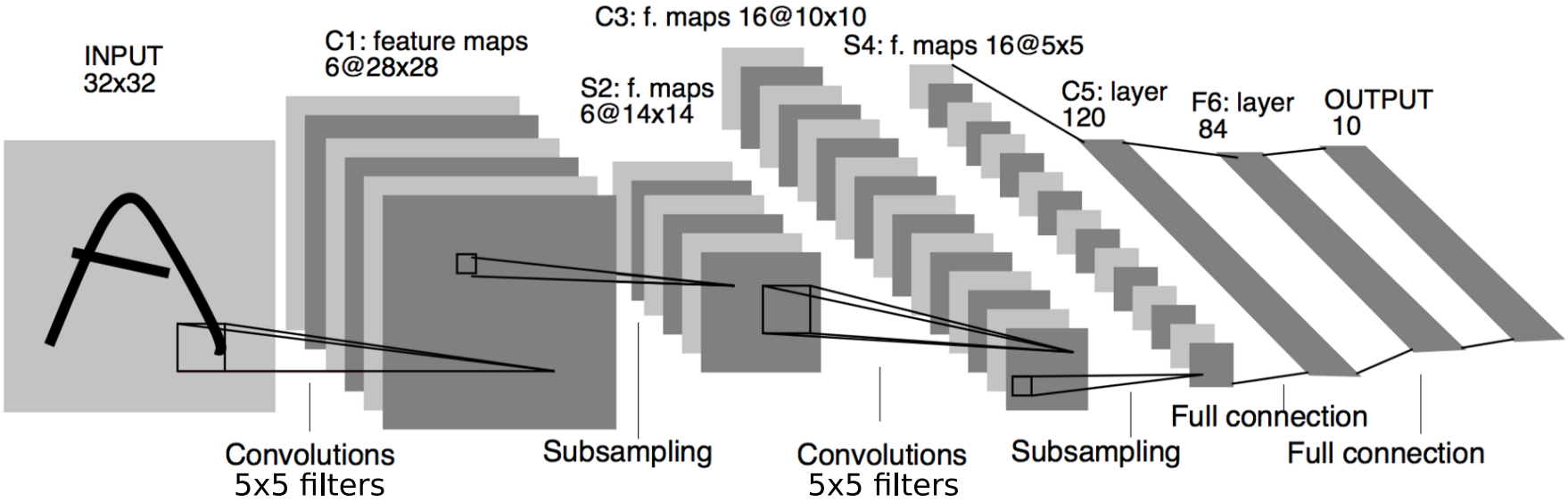
Q: Why is residual/skip connections a summed ensemble ? (write out the $\mathbf{h}$s in terms of f) A:

$$\begin{aligned} f(\mathbf{x}) = \mathbf{x} &+ f_1(\mathbf{x}) \\ &+ f_2(\mathbf{x} + f_1(\mathbf{x})) \\ &+ f_3(\mathbf{x} + f_1(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x}))) \\ &+ f_4\Big(\mathbf{x} + f^{(1)}(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x})) + f_3(\mathbf{x} + f_1(\mathbf{x}) + f_2(\mathbf{x} + f_1(\mathbf{x})))\Big) \end{aligned}$$

Q: How many summed sub-networks? A: 4

Questions?

Questions?



Q: How to compute the number of parameters of the full network?

Q: How to compute the receptive field of a pixel in S4?