

# Shallow Intro to Deep Learning'19

CNN, Chapters: 9.1, 9.2, 9.3, 9.5

5 June 2019



Delft University of Technology

Lecturer: Jan van Gemert

# Shallow Intro to Deep Learning'19

CNN, Chapters: 9.1, 9.2, 9.3, 9.5



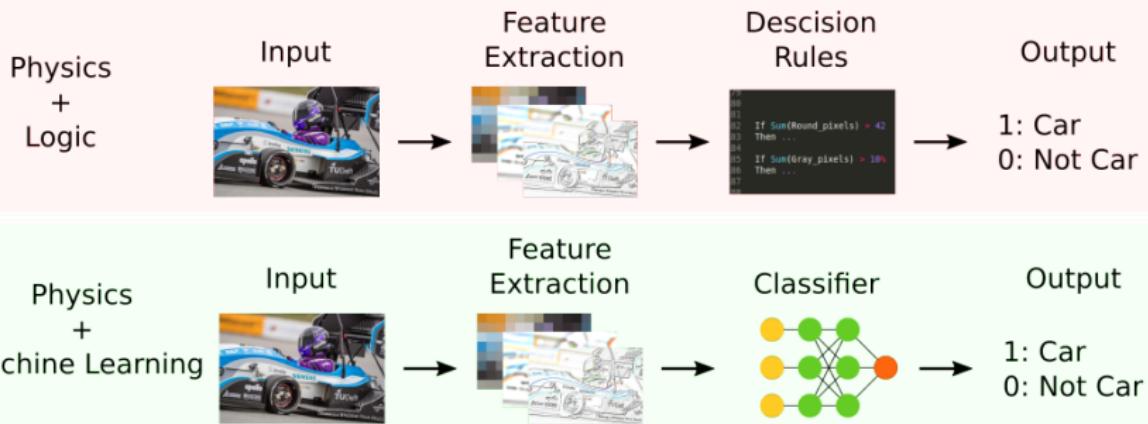
Delft University of Technology

Lecturer: Jan van Gemert

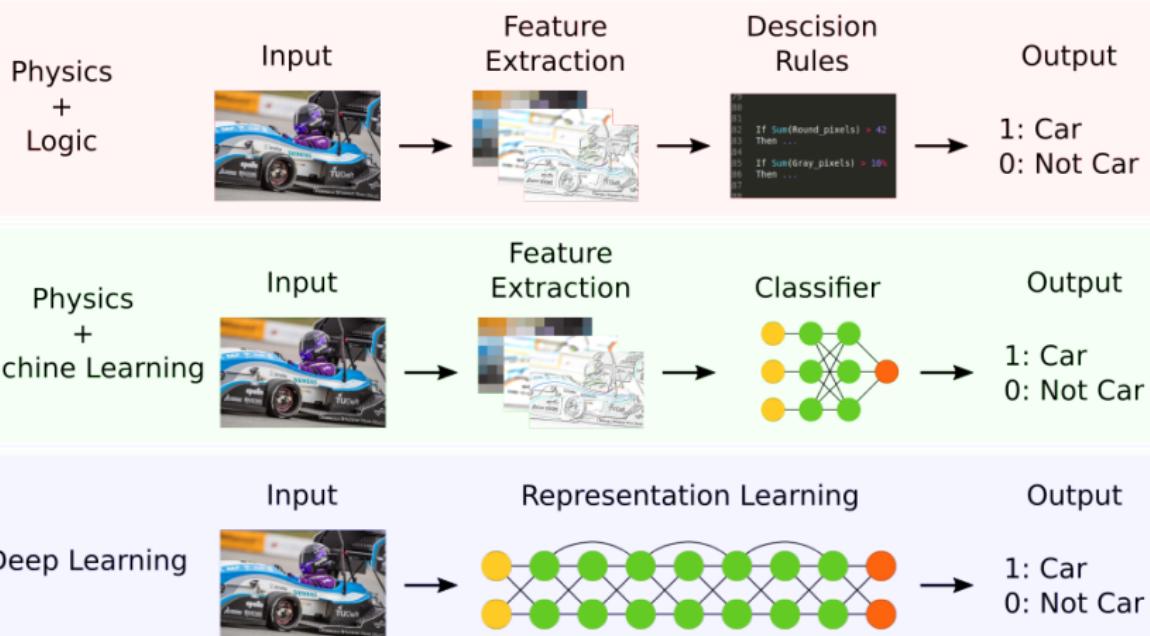
# Feature extraction vs deep learning



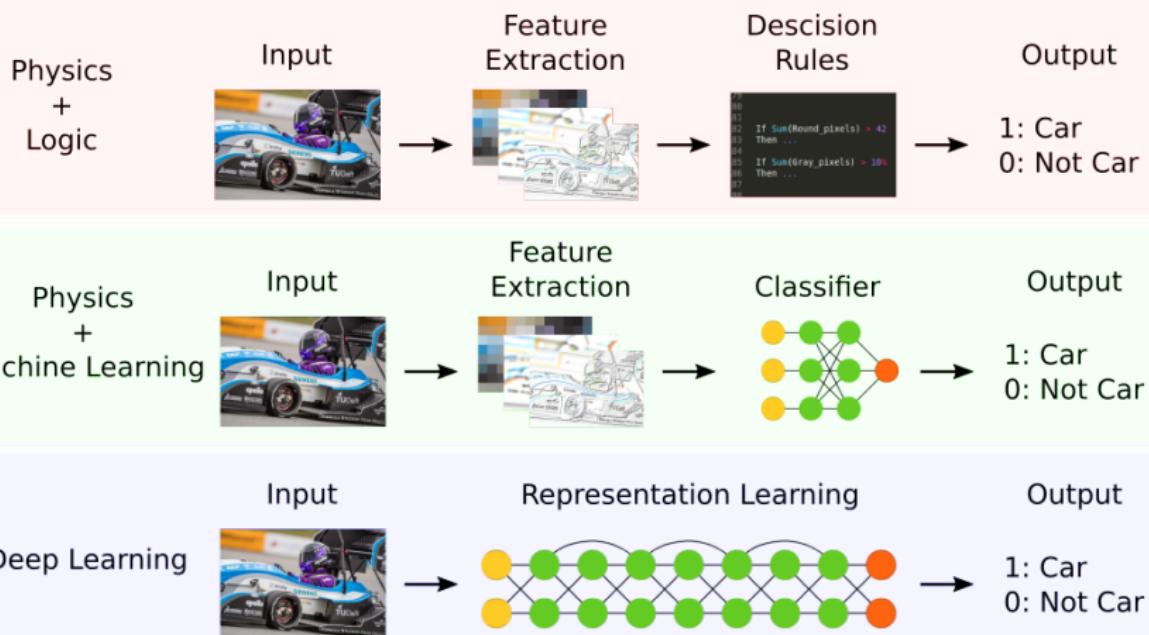
# Feature extraction vs deep learning



# 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

$F(x, y)$

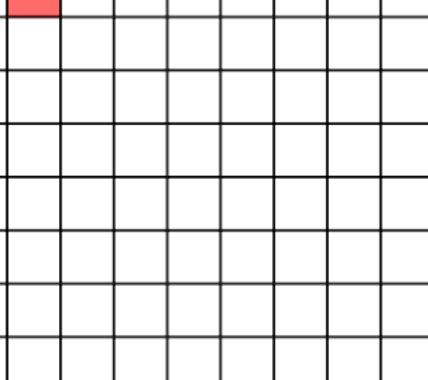
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$G(x, y)$


## Moving Neighborhood Average

$$F(x, y)$$

$$G(x,y)$$

A 10x10 grid of squares. The first column contains 10 white squares. The second column contains 9 white squares, with the top one being red. The remaining 8 columns each contain 10 white squares.

## Moving Neighborhood Average

$$F(x, y)$$

$$G(x, y)$$

# 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	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	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	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	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	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	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	0	0	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	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	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	0	0	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	0	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	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	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	0	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
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	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?

$F(x, y)$

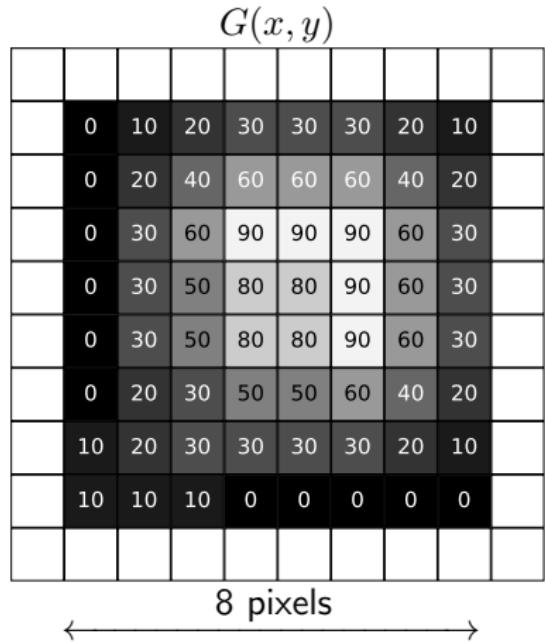
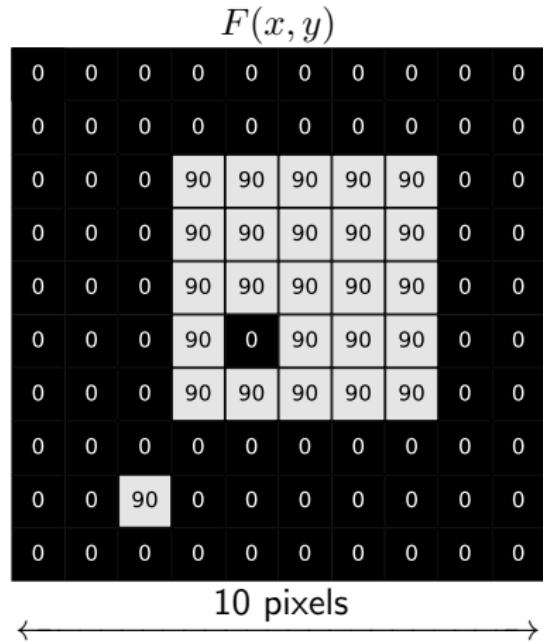
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	90	0	90	90	90	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

$G(x, y)$

	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)

# Convolution

## Chapter 9.1

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

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

The diagram illustrates the convolution operation between a feature map  $F(x, y)$  and a kernel  $H(x, y)$  to produce a result  $G(x, y)$ .

**Feature Map  $F(x, y)$ :**

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

**Kernel  $H(x, y)$ :**

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

**Result  $G(x, y)$ :**

			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**

$$\begin{matrix} F(x,y) & \star & H(x,y) & = & G(x,y) \end{matrix}$$

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

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	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

			0	10	20	30			

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

# Practice with kernels

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

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

# Practice with kernels

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

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

 $=$   
 $G(x, y)$ 

# Practice with kernels

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

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

# Practice with kernels

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

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

 $=$   
 $G(x, y)$ 

# Practice with kernels

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

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

# Practice with kernels

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

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

 $=$   
 $G(x, y)$ 

# Practice with kernels

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

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

# Practice with kernels

$F(x, y)$



$\star$   
 $H(x, y)$

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

$=$   
 $G(x, y)$



# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



\*

$$H(x, y)$$



=

$$G(x, y)$$

# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



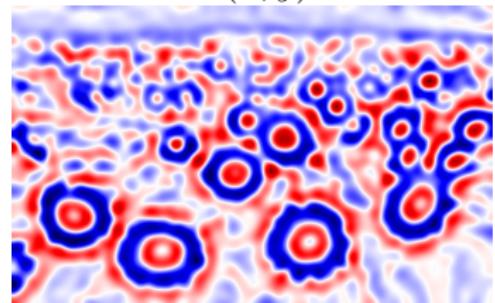
\*

$$H(x, y)$$



=

$$G(x, y)$$



# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



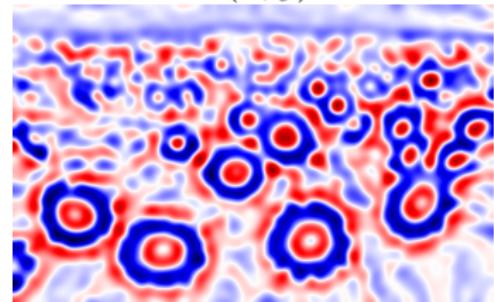
\*

$$H(x, y)$$



=

$$G(x, y)$$



Q: How about a smaller kernel?



# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



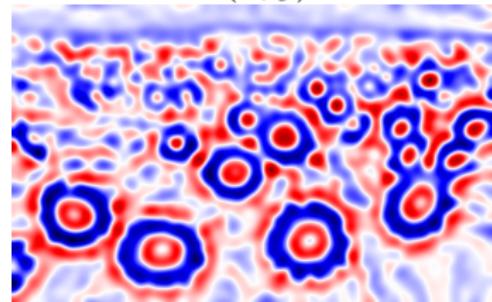
\*

$$H(x, y)$$

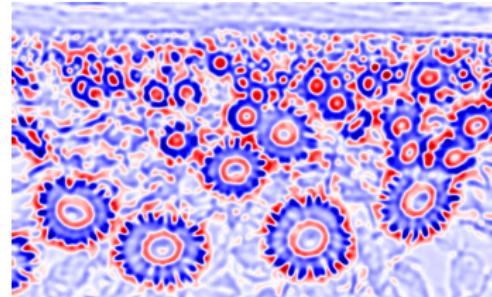


=

$$G(x, y)$$



Q: How about a smaller kernel?



# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



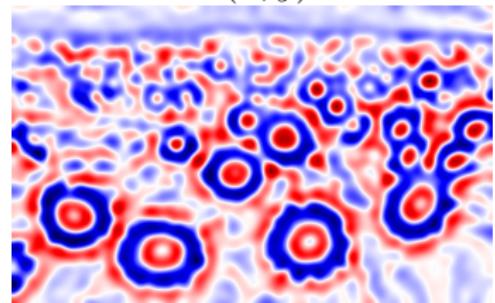
\*

$$H(x, y)$$

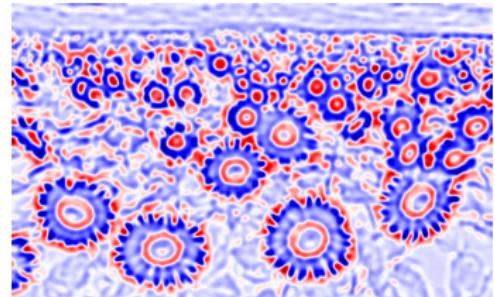


=

$$G(x, y)$$



Q: How about a smaller kernel?



Q: How about a smaller input image?

# Practice with kernels

(Red = positive; blue is negative)

$$F(x, y)$$



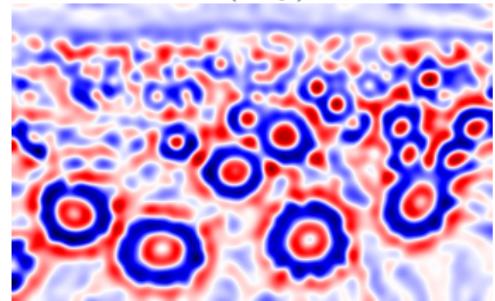
\*

$$H(x, y)$$

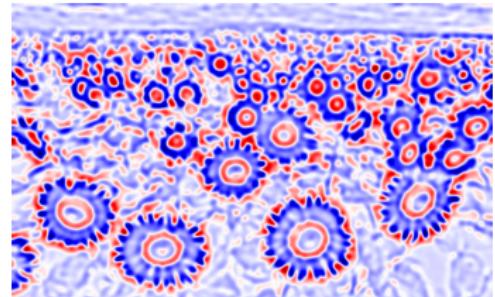


=

$$G(x, y)$$



Q: How about a smaller kernel?



Q: How about a smaller input image?

A: Smaller image = larger kernel.

# Where is Waldo?



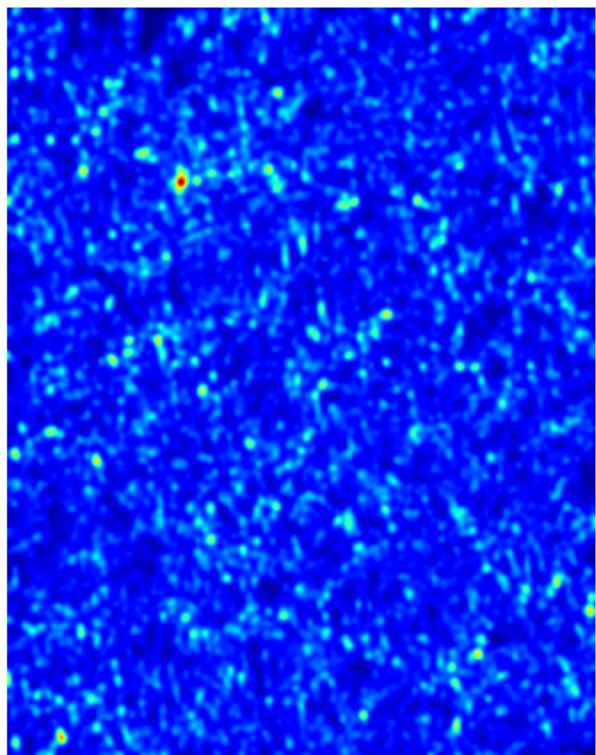
# Where is Waldo?



Use this normalized kernel:



# Where is Waldo?



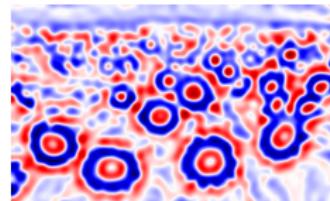
# Representation learning



\*



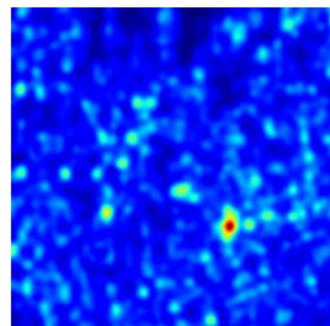
=



\*



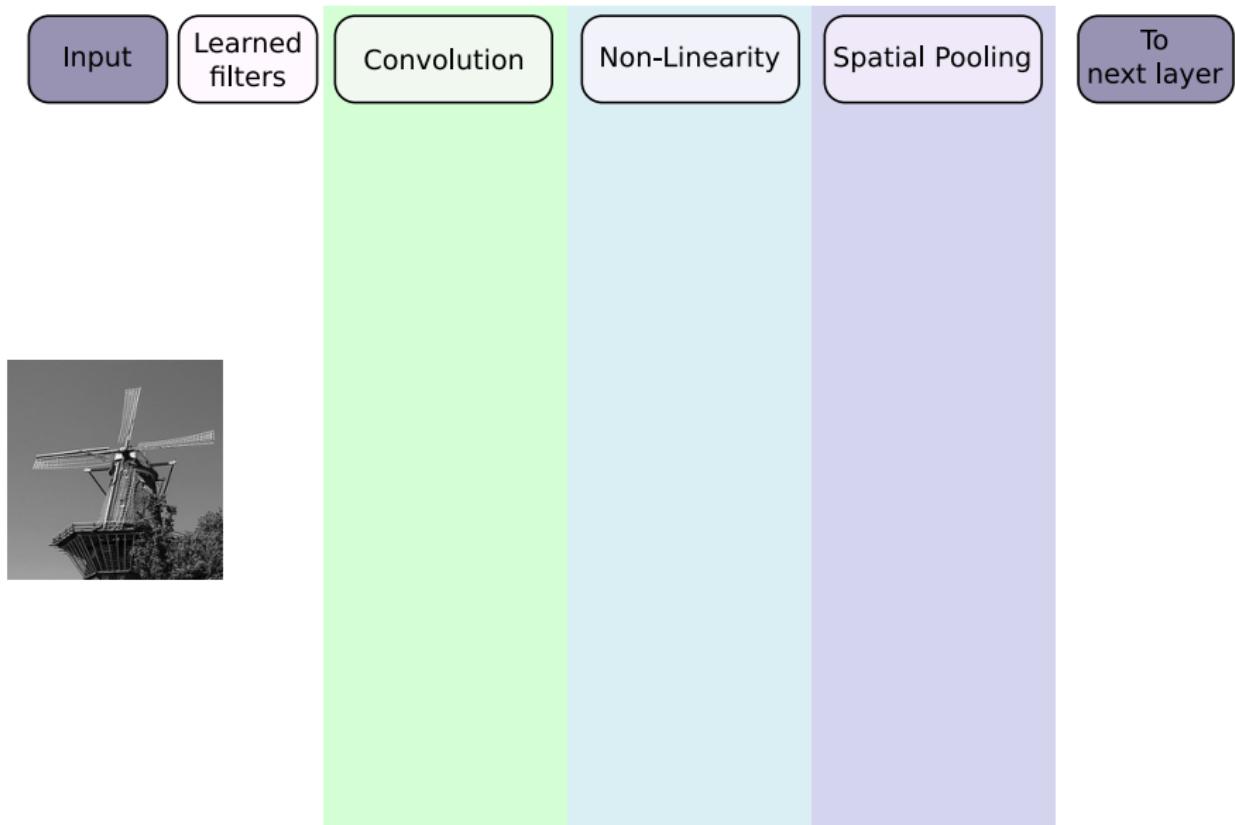
=



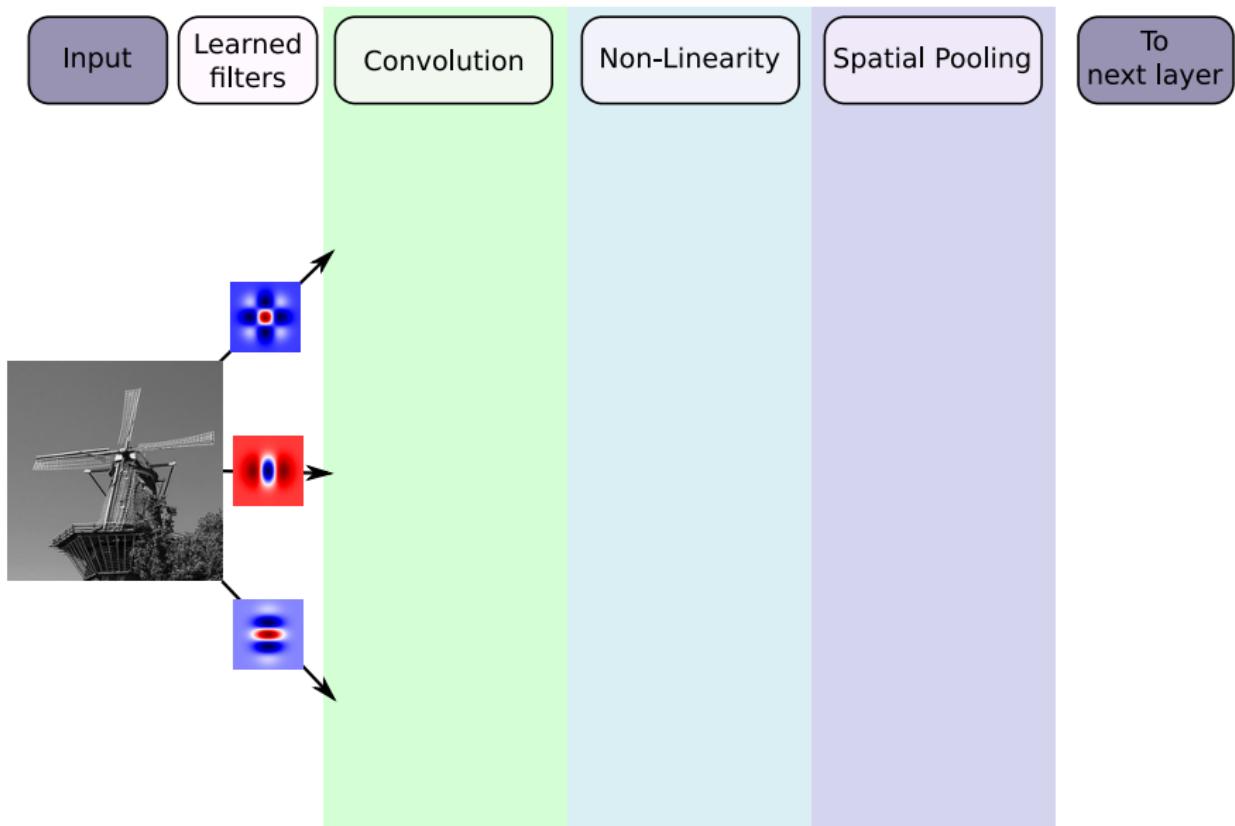
- Kernel weights are feature detectors
- Learning weights = Learning features
- Convnet learns the feature representation

# Questions?

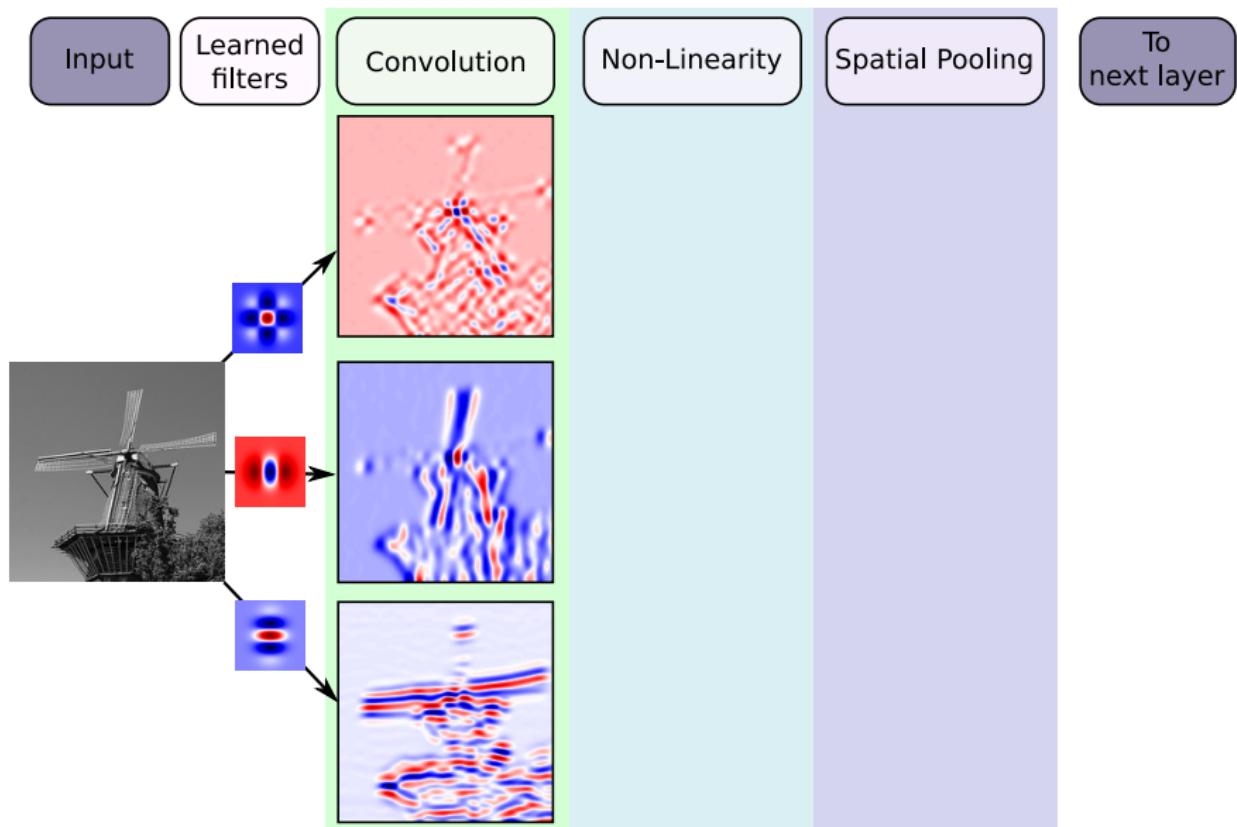
# Convolutional Network



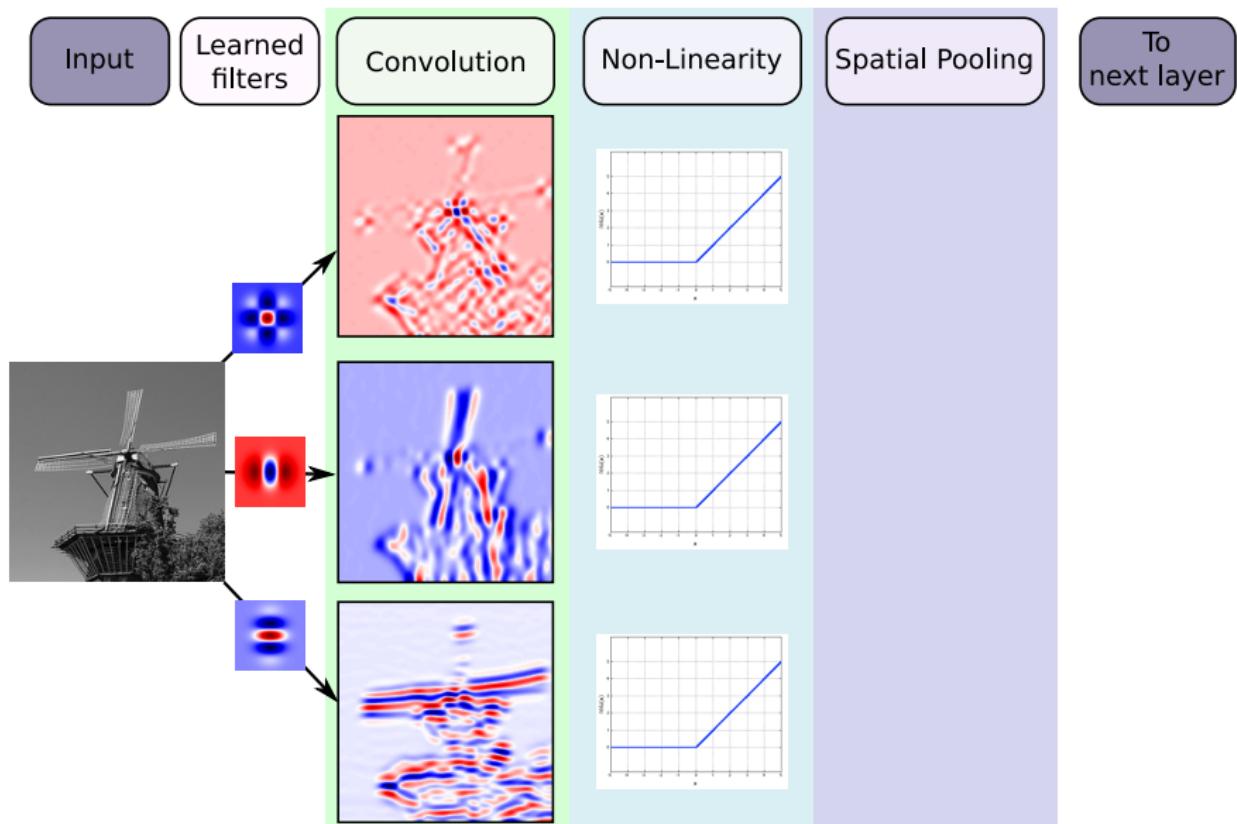
# Convolutional Network: Filter



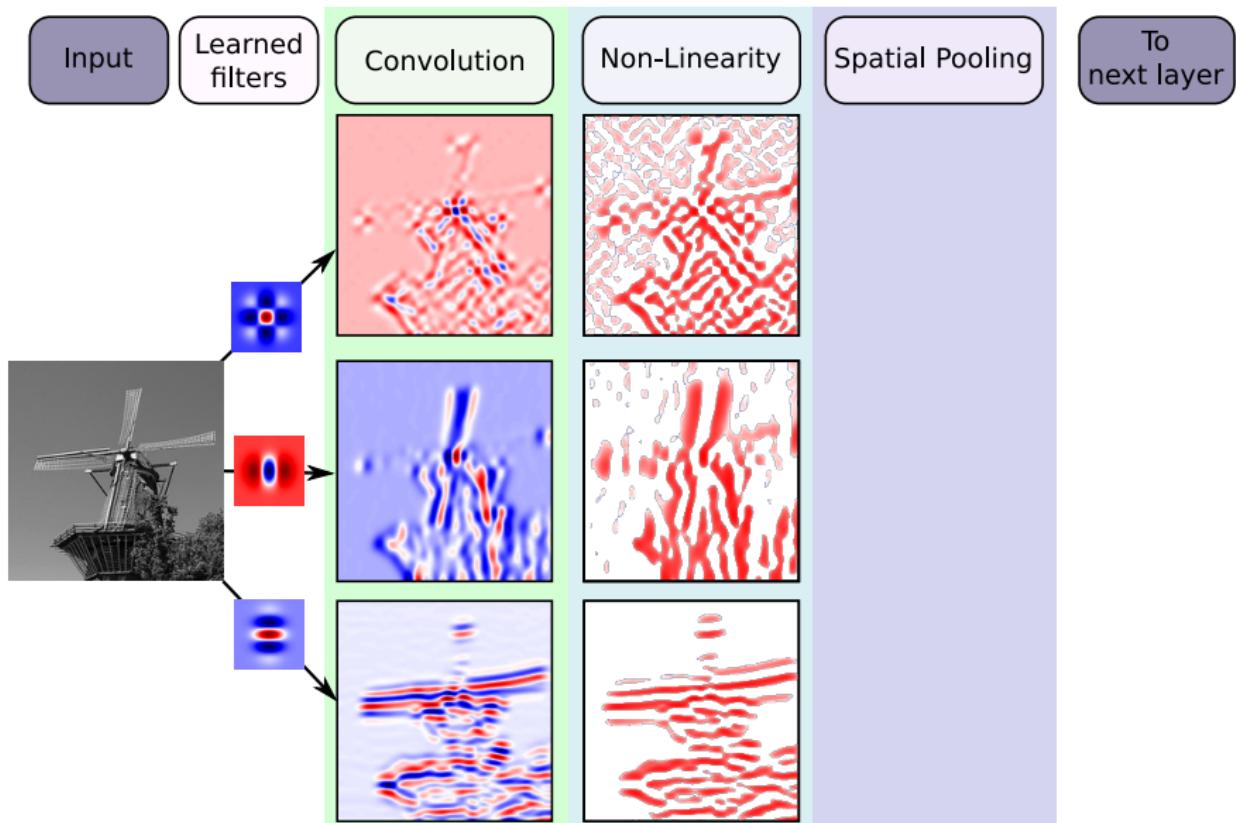
# Convolutional Network: Featuremaps



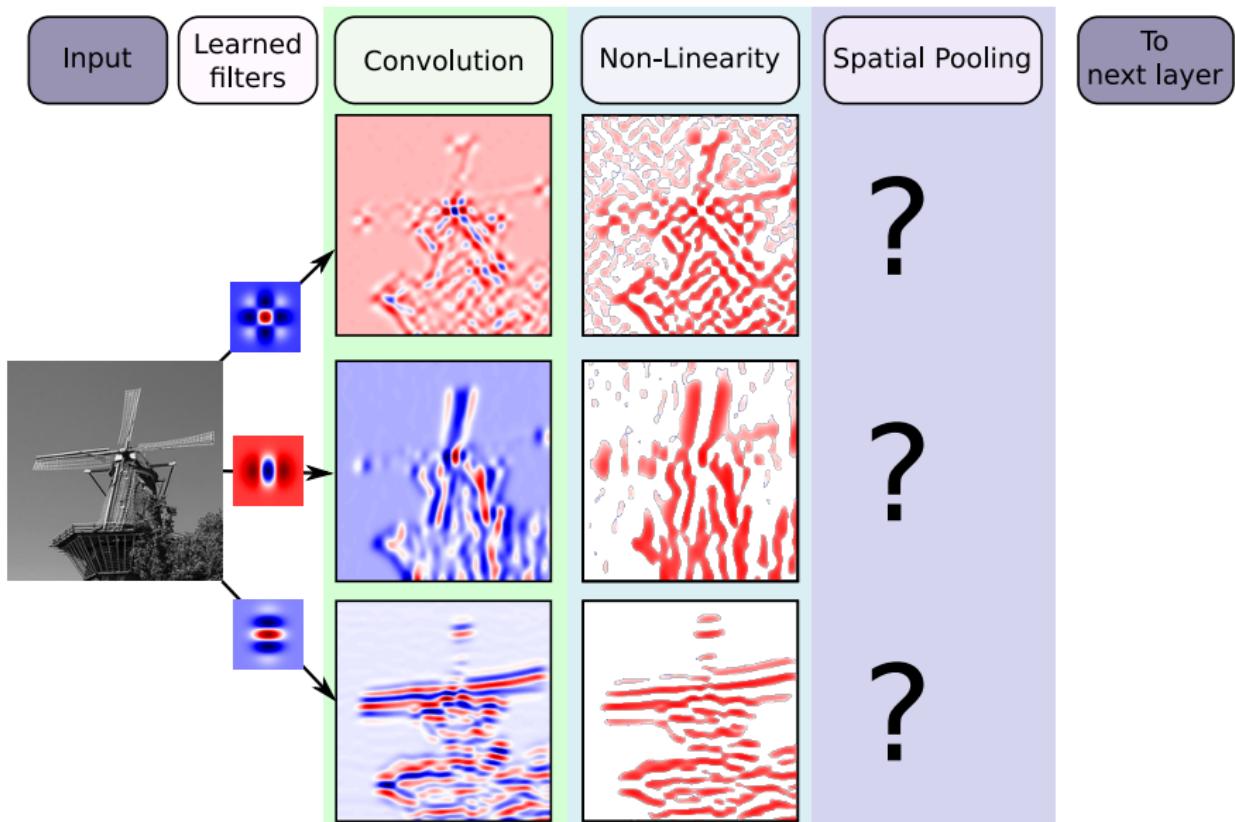
# Convolutional Network: ReLu ( $f(x) = \max(0, x)$ )



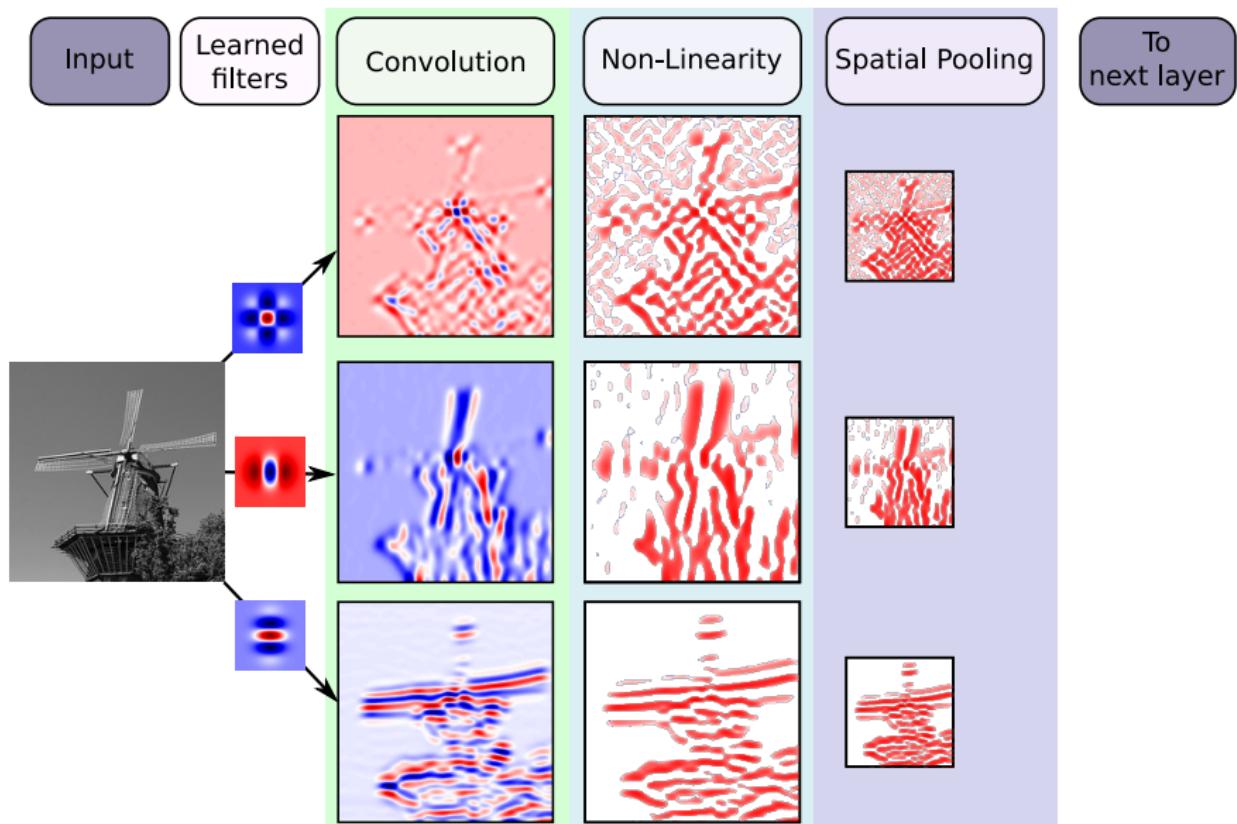
# Convolutional Network: All negative values removed



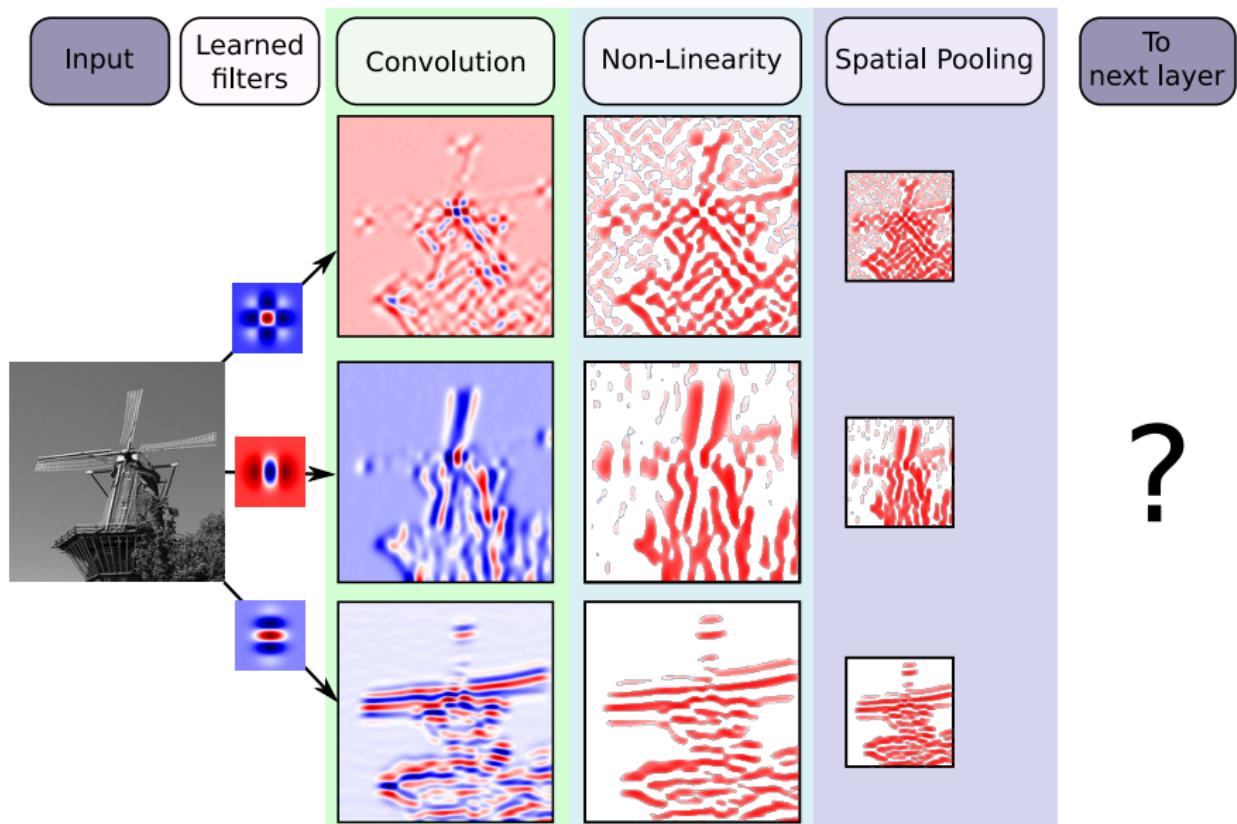
# Convolutional Network: $2 \times 2$ max, $2 \times 2$ sub-sample



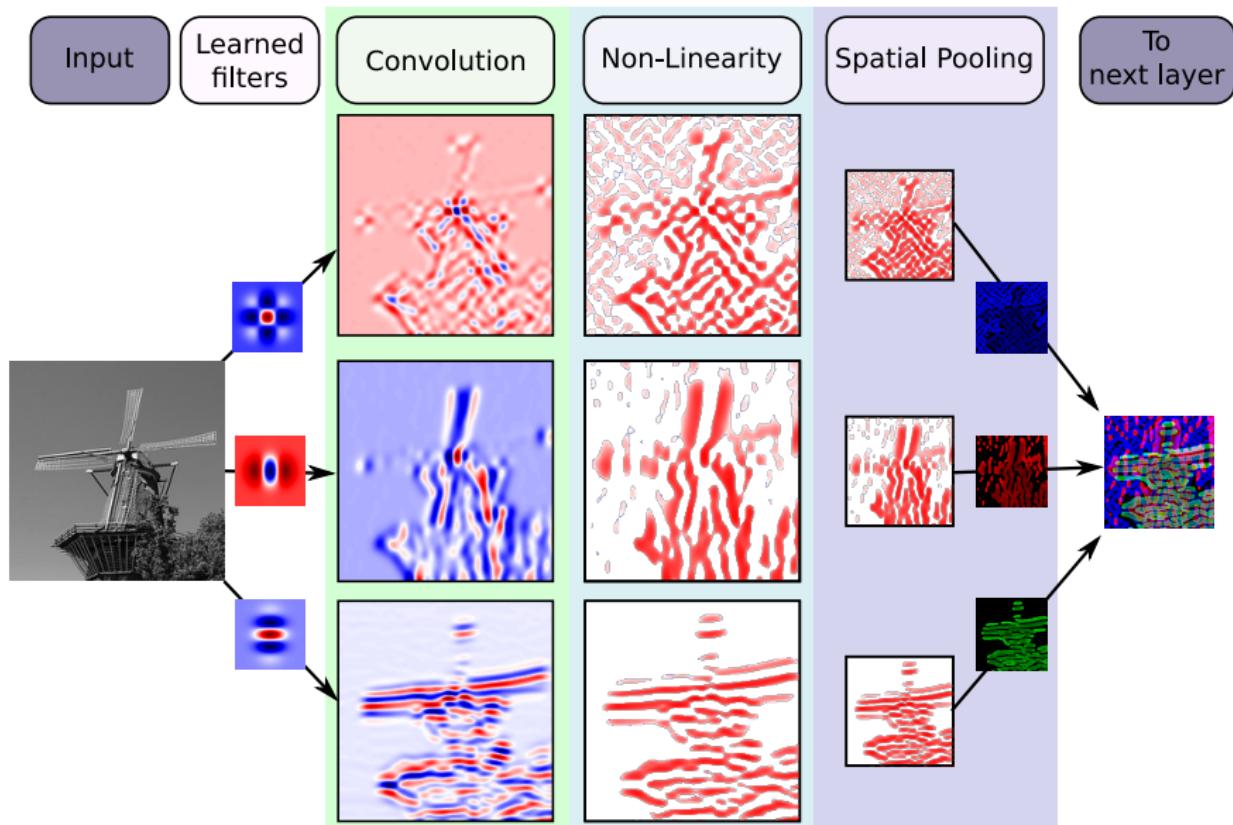
# Convolutional Network: Smaller feature maps



# Convolutional Network: To the next layer

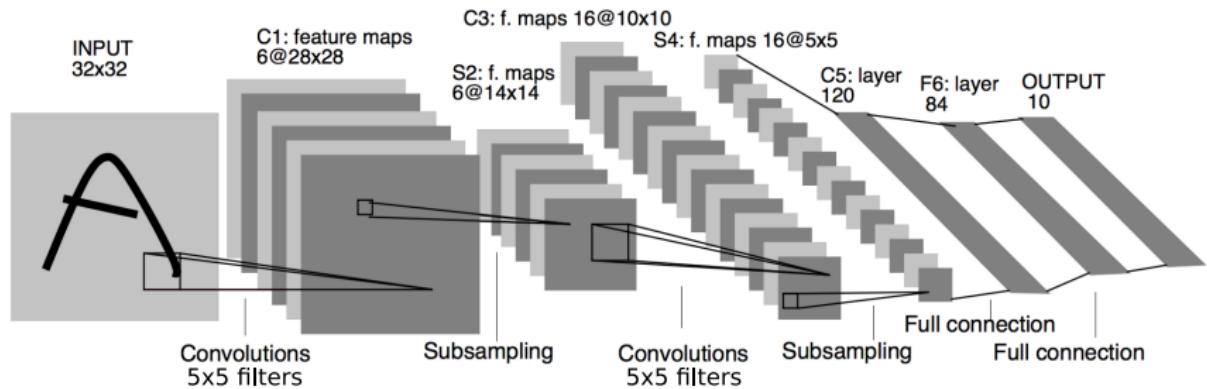


# Convolutional Network: Featuremaps as new channels



# Questions?

# Questions?



Q: Can you explain this CNN?