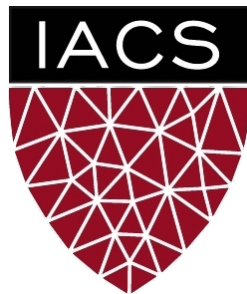


# Lecture 16: Convolution Networks

CS 109B, STAT 121B, AC 209B, CSE 109B

Mark Glickman and Pavlos Protopapas



# Image Data

- Fully-connected nets infeasible
- E.g. 1K x 1K image (1 million pixels)
  - No. of parameters blow up
- *Convolution Nets*
  - Exploit image structure
  - Use sparse weights between layers

# Images are Local and Hierarchical



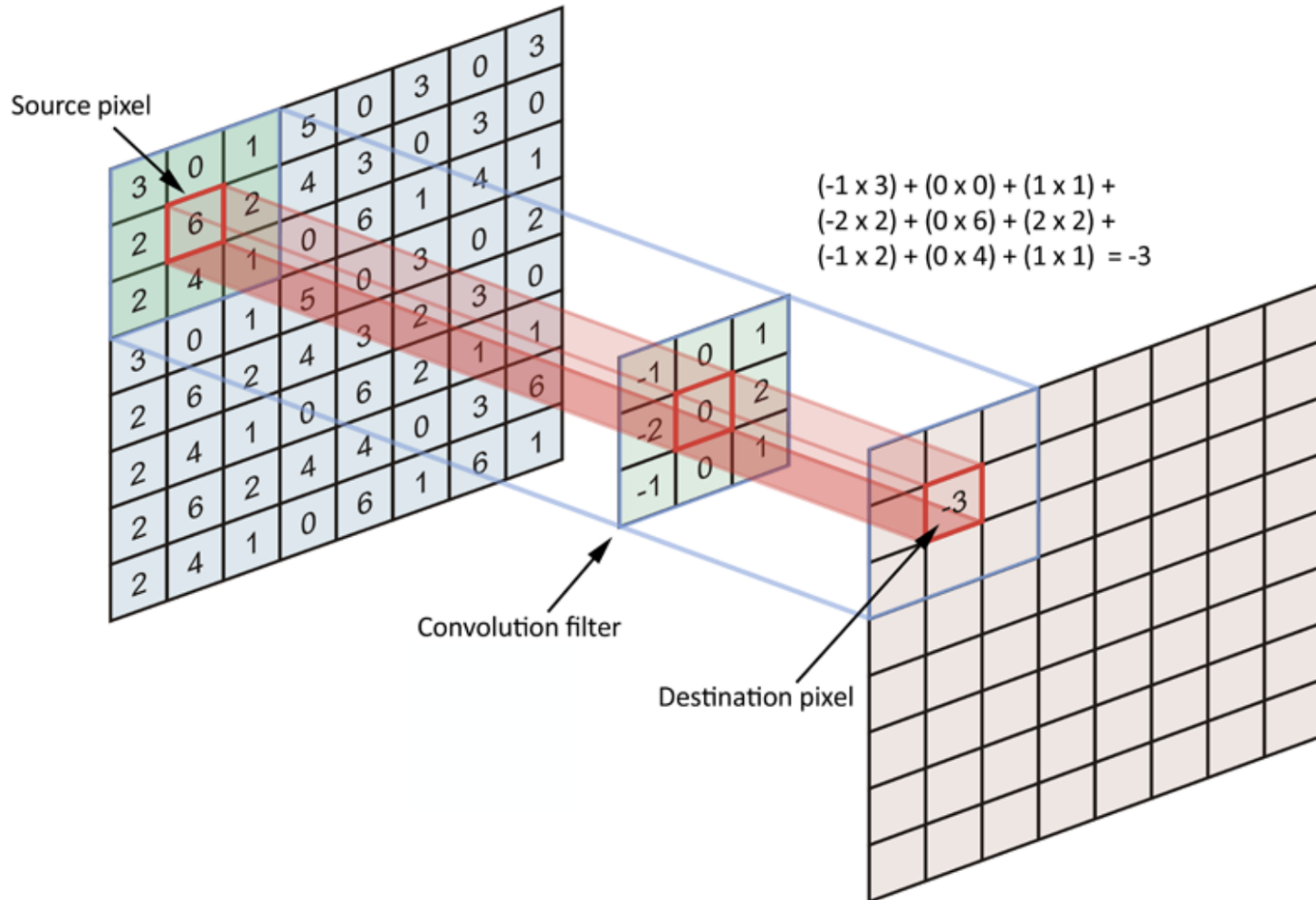
Nearby pixels are more strongly related than distant ones.

Objects are built up out of smaller parts.

# Images are Invariant

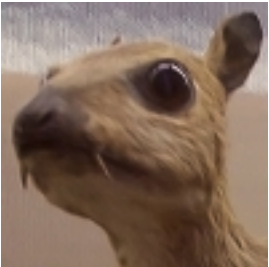


# “Convolution” Operation



# “Convolution” Operation

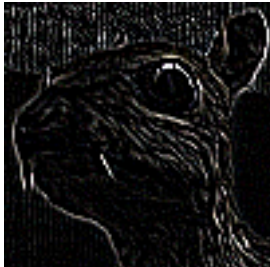
*Edge detection*



\*

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

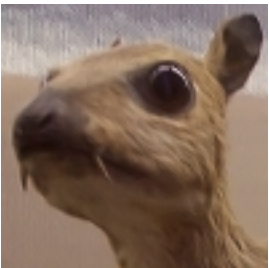
=



Kernel

The diagram illustrates the edge detection convolution operation. It shows the original image of a squirrel's head on the left, followed by a multiplication symbol and a 3x3 kernel matrix. The kernel matrix is: 
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$
 This is followed by an equals sign and the resulting edge detection image, which highlights the boundaries of the squirrel's head. An arrow labeled "Kernel" points to the kernel matrix.

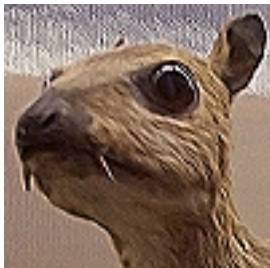
*Sharpen*



\*

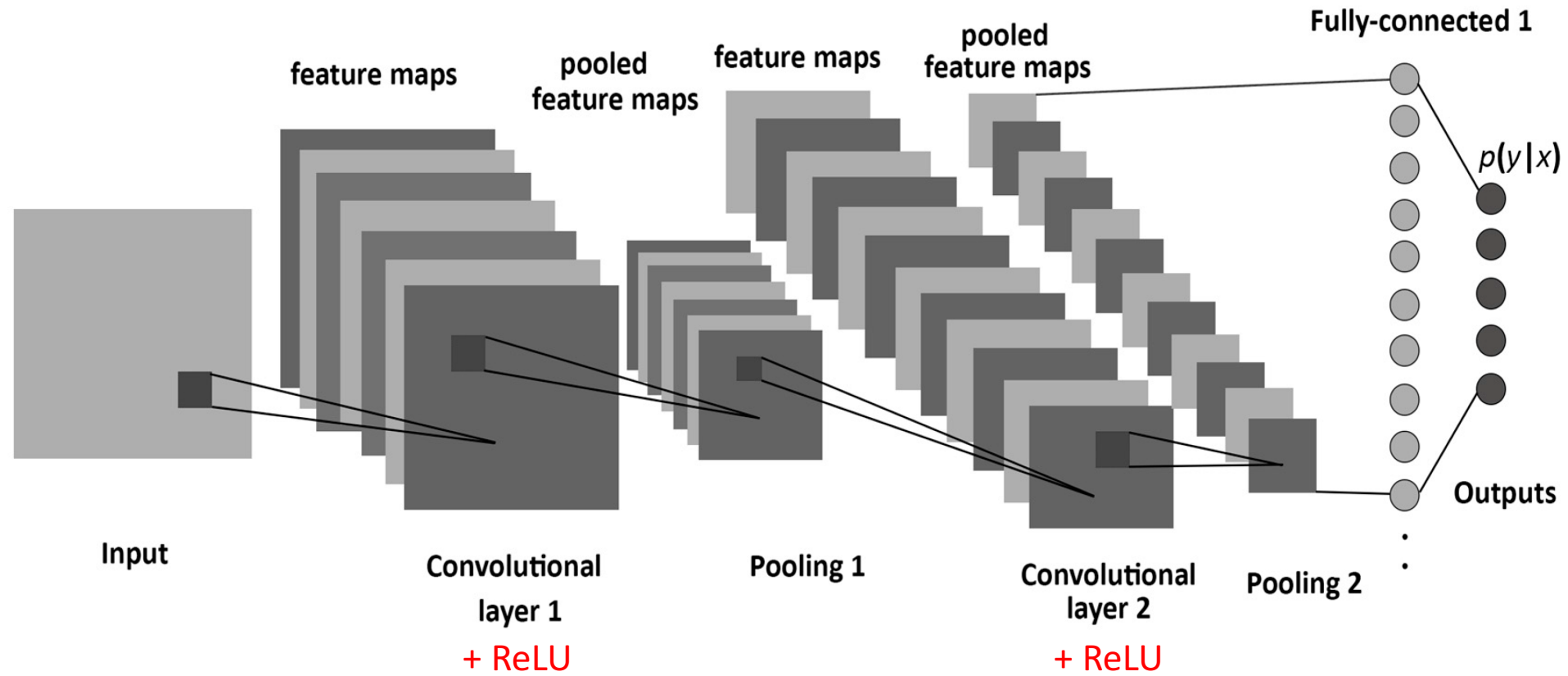
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

=



The diagram illustrates the sharpening convolution operation. It shows the original image of a squirrel's head on the left, followed by a multiplication symbol and a 3x3 kernel matrix. The kernel matrix is: 
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$
 This is followed by an equals sign and the resulting sharpened image, which has enhanced edges and contrast.

# A Convolutional Network

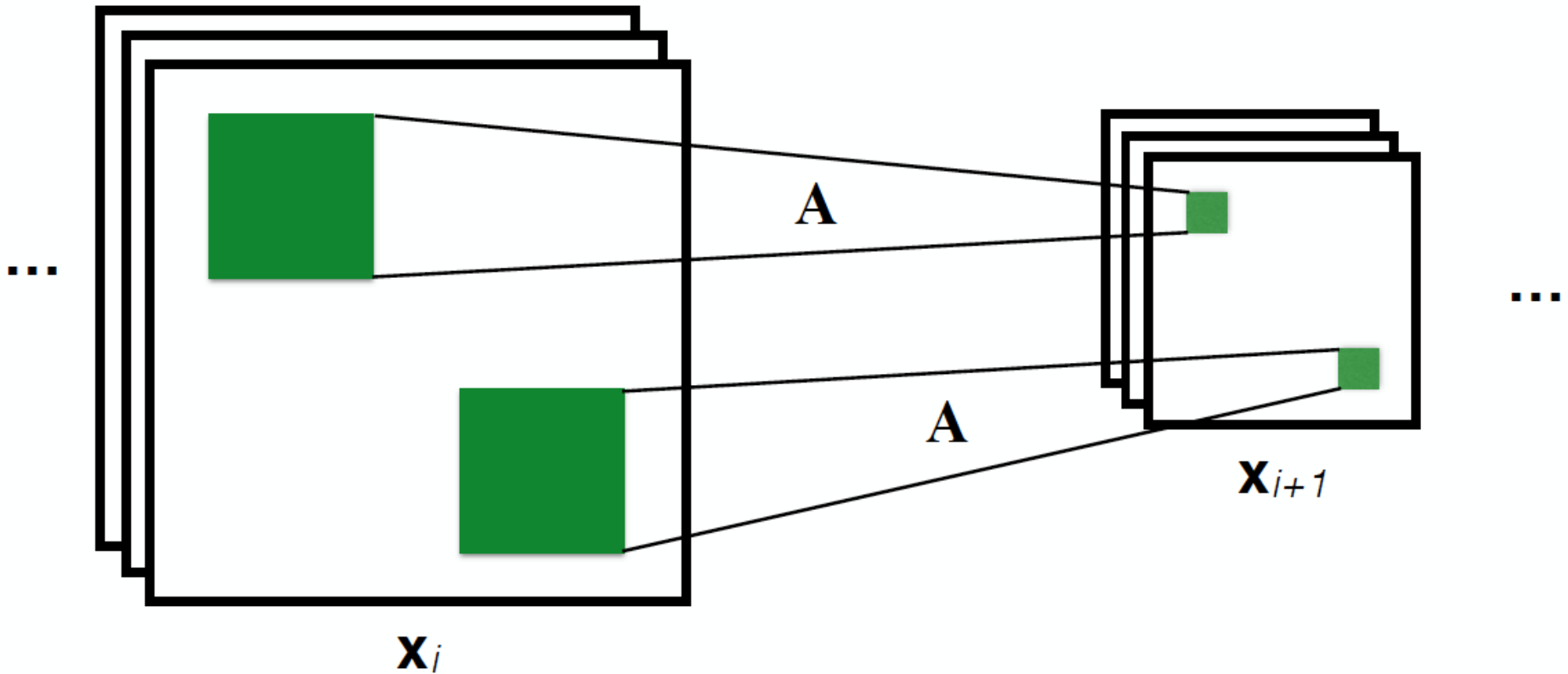


# Key Features

- Sparse interaction
- Parameter sharing
- Pooling
- Zero padding



# Sparse Interactions



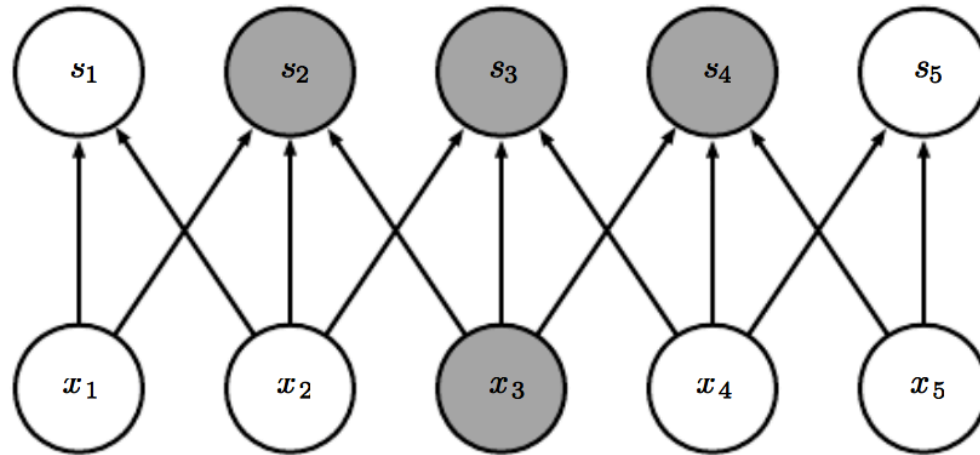
$$\mathbf{x}_{i+1} = \sigma_i(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i)$$

Each output interacts to a small region of inputs

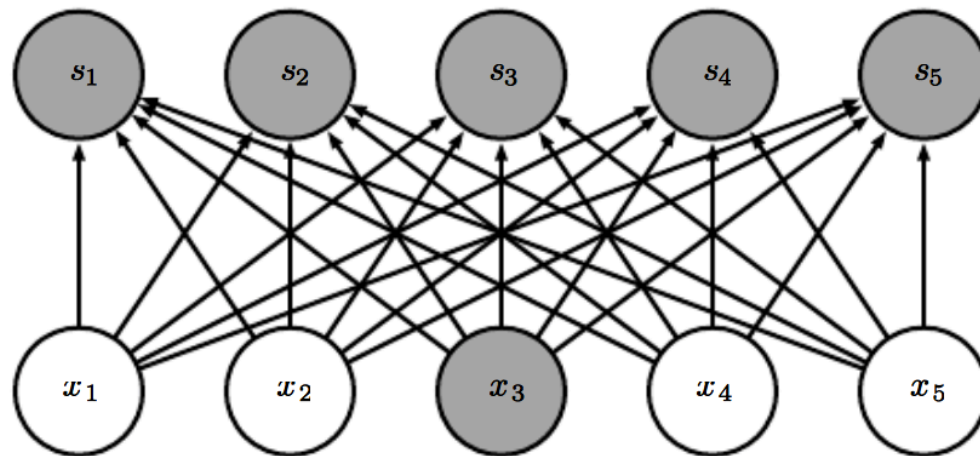
# Sparse Interactions

- Kernel size smaller than input
  - Image: usually millions of pixels
  - Kernel: usually 100s or 1000s of parameters
- Reduced memory and run-time, improved statistical efficiency
  - $O(m \times n) \Rightarrow O(k \times n)$  parameters

# Sparse Interactions



**Convolutional**



**Fully-connected**

# Computational Costs

Input size: 320 by 280

Kernel size: 2 by 1

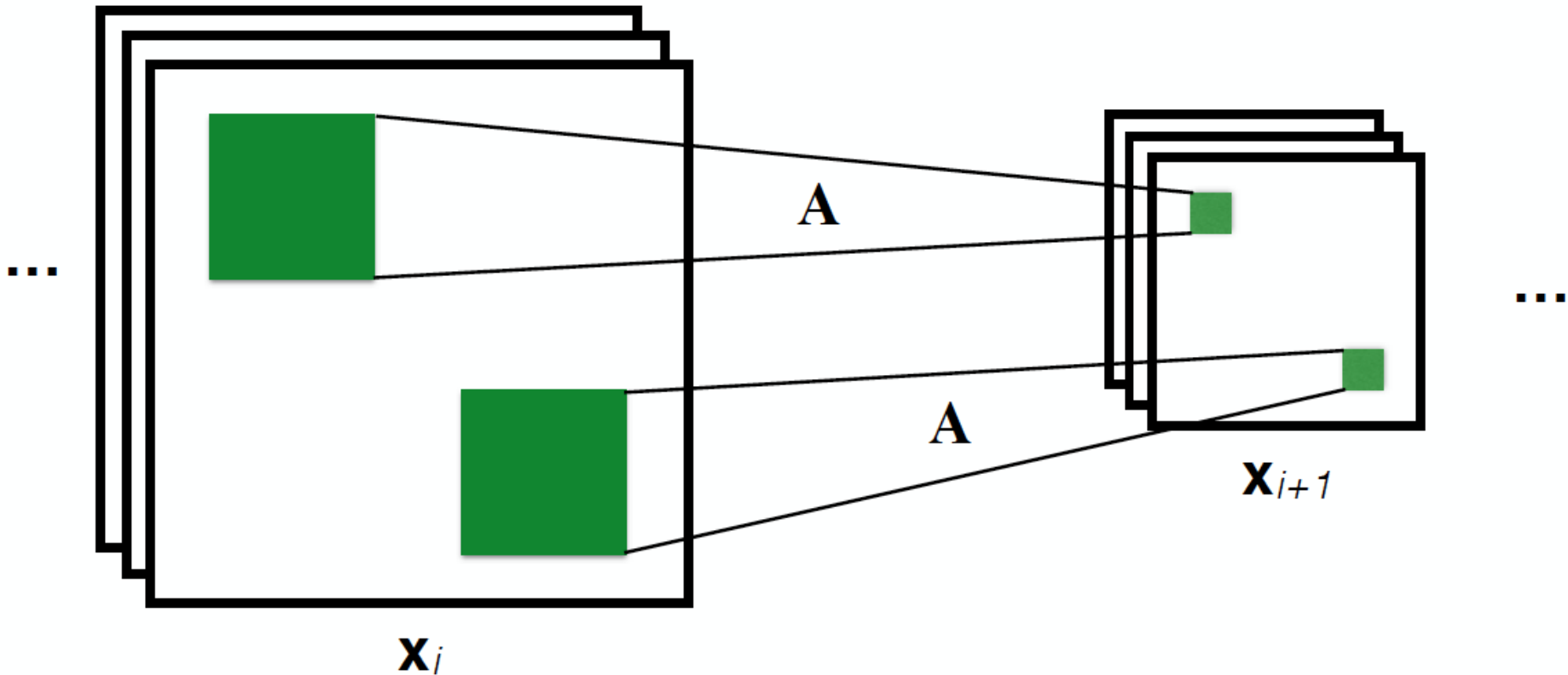
Output size: 319 by 280

	Convolution	Dense matrix	Sparse matrix
Stored floats	2	$319 \cdot 280 \cdot 320 \cdot 280$ > 8e9	$2 \cdot 319 \cdot 280 =$ 178,640
Float muls or adds	$319 \cdot 280 \cdot 3 =$ 267,960	> 16e9	Same as convolution (267,960)

# Key Features

- Sparse interaction
- Parameter sharing
- Pooling
- Zero padding

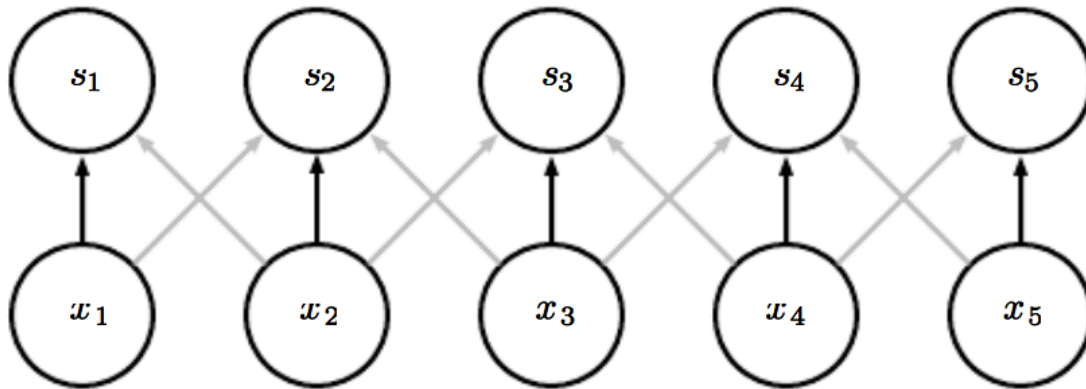
# Parameter Sharing



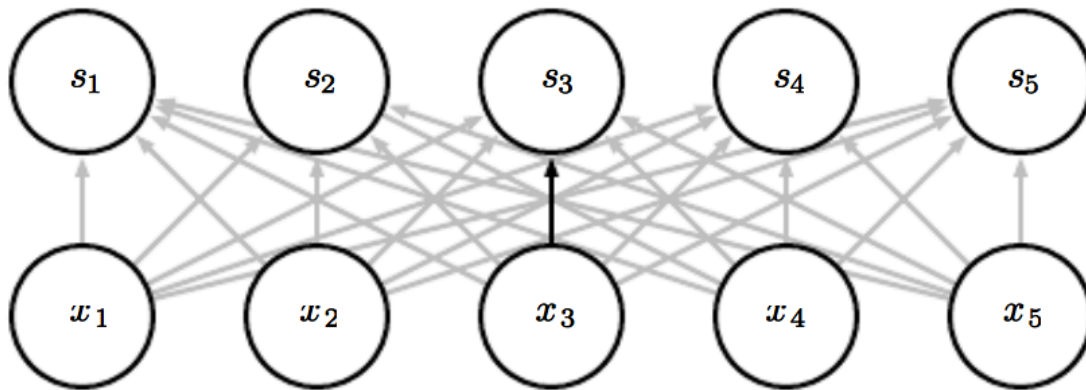
$$\mathbf{x}_{i+1} = \sigma_i(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i)$$

*Same  $\mathbf{A}$  in different locations – only learn  $\mathbf{A}$*

# Parameter Sharing

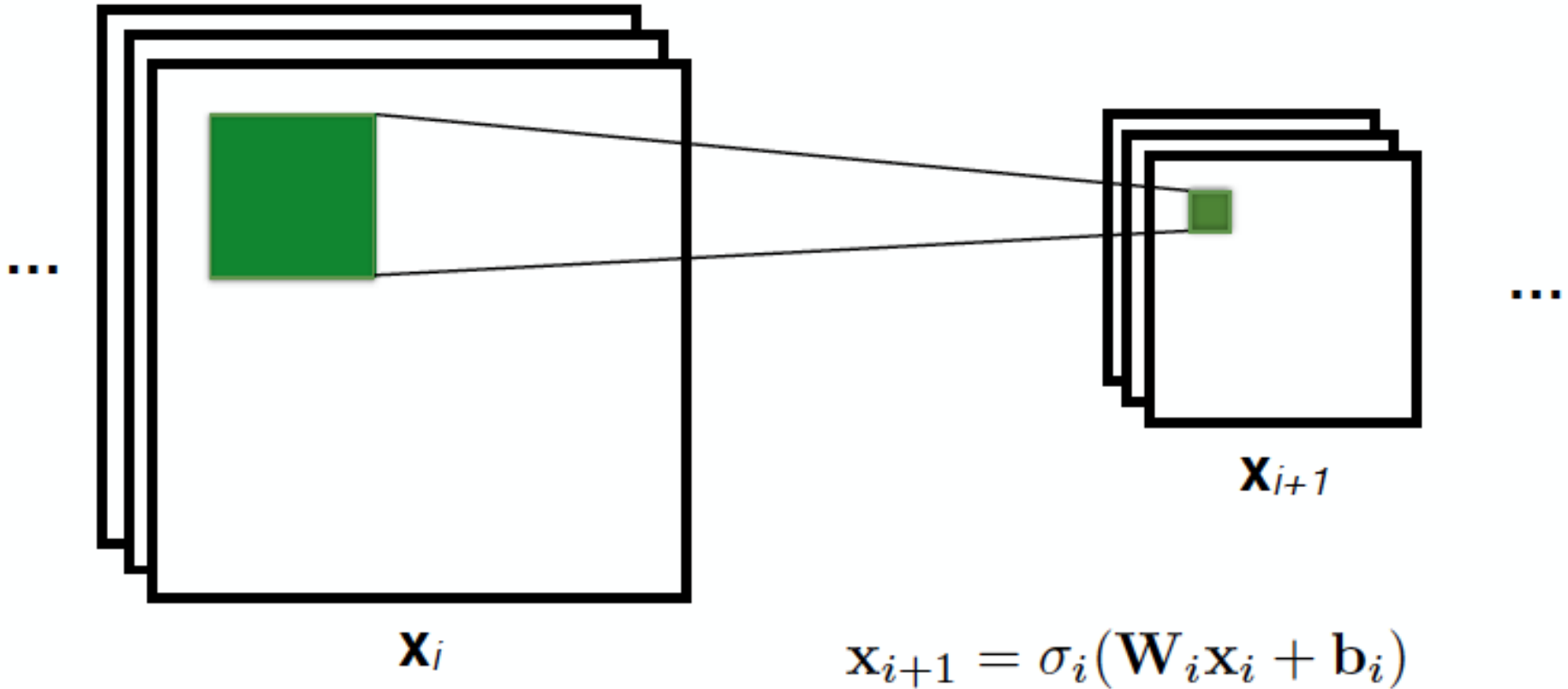


**Convolutional**



**Fully-connected**

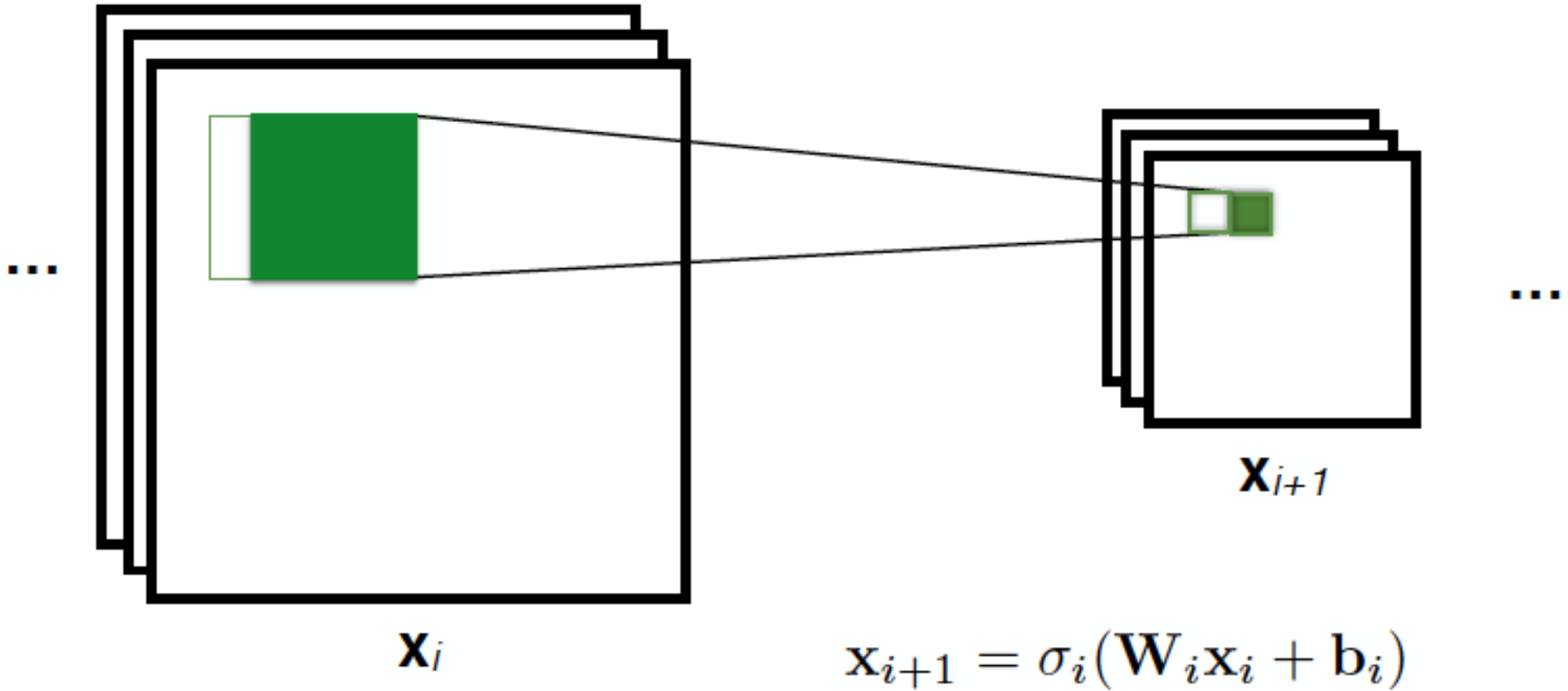
# Parameter Sharing



*Equivariant to translation*

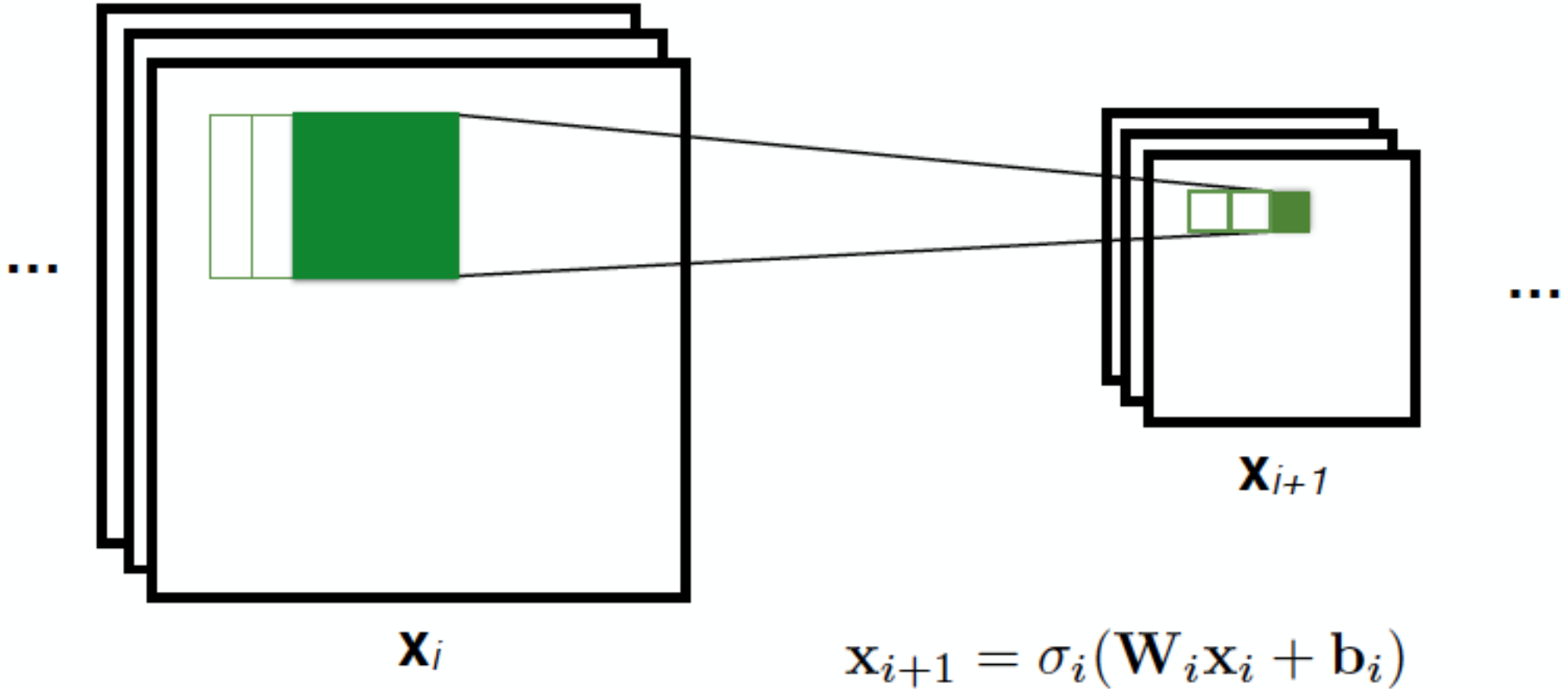


# Parameter Sharing

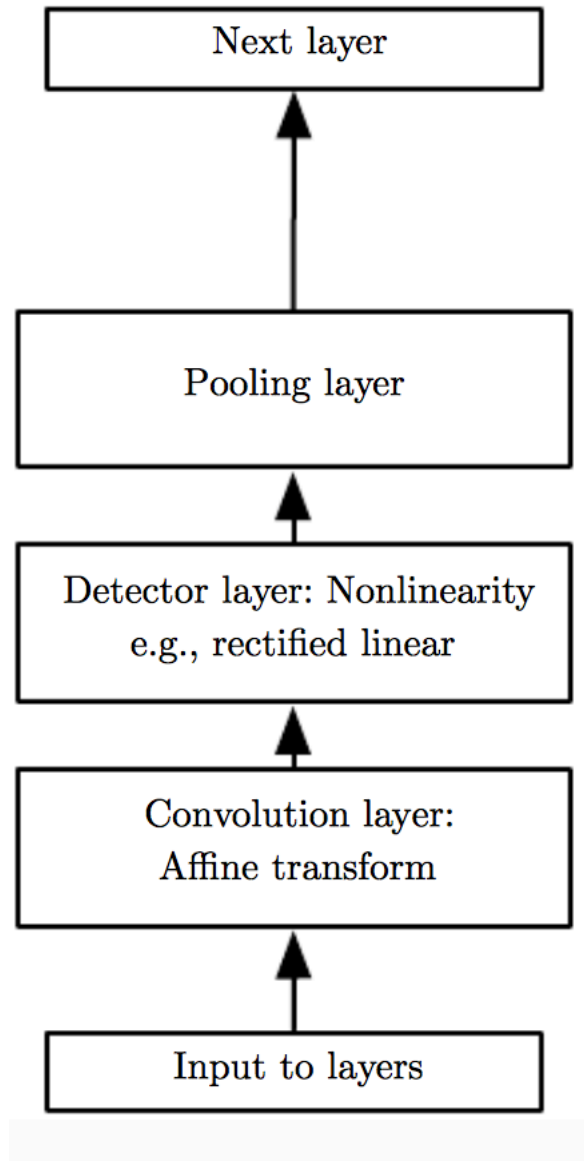


*Equivariant to translation*

# Parameter Sharing



*Equivariant to translation*

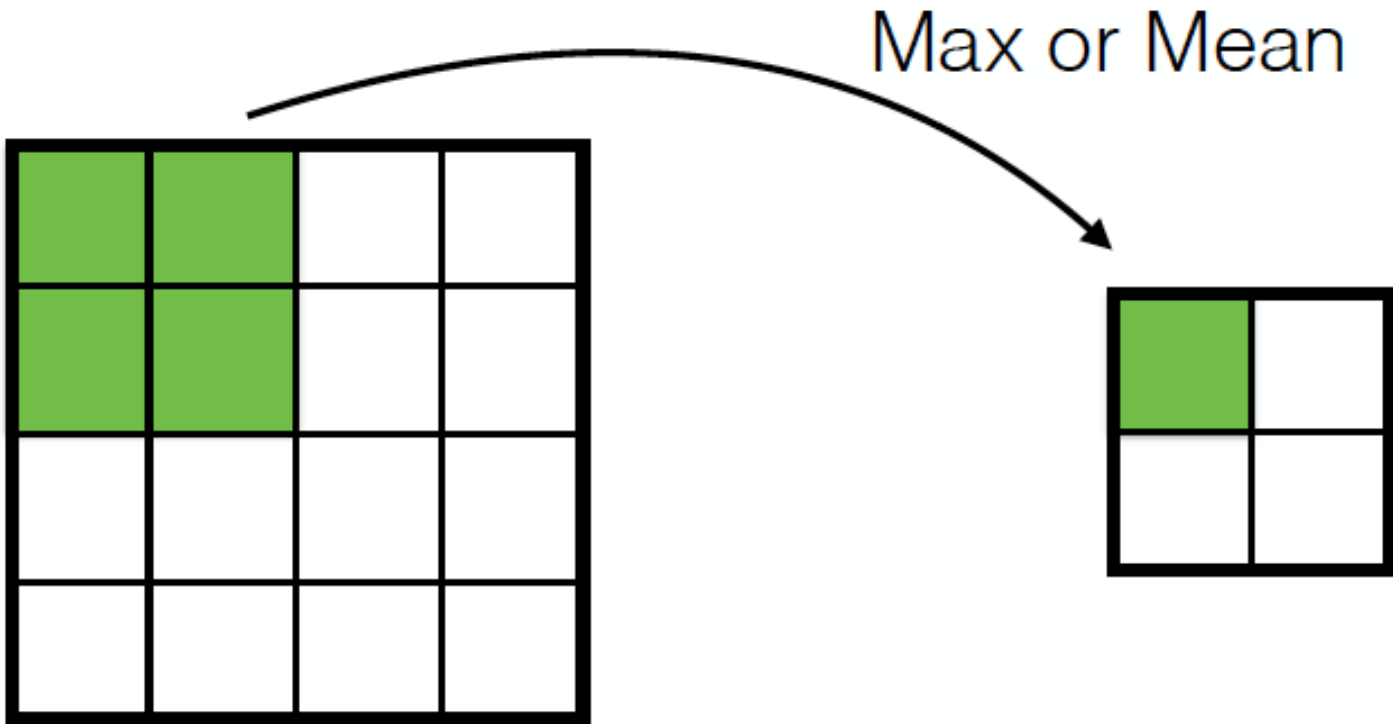


Non-linear  
activation

# Key Features

- Sparse interaction
- Parameter sharing
- Pooling
- Zero padding

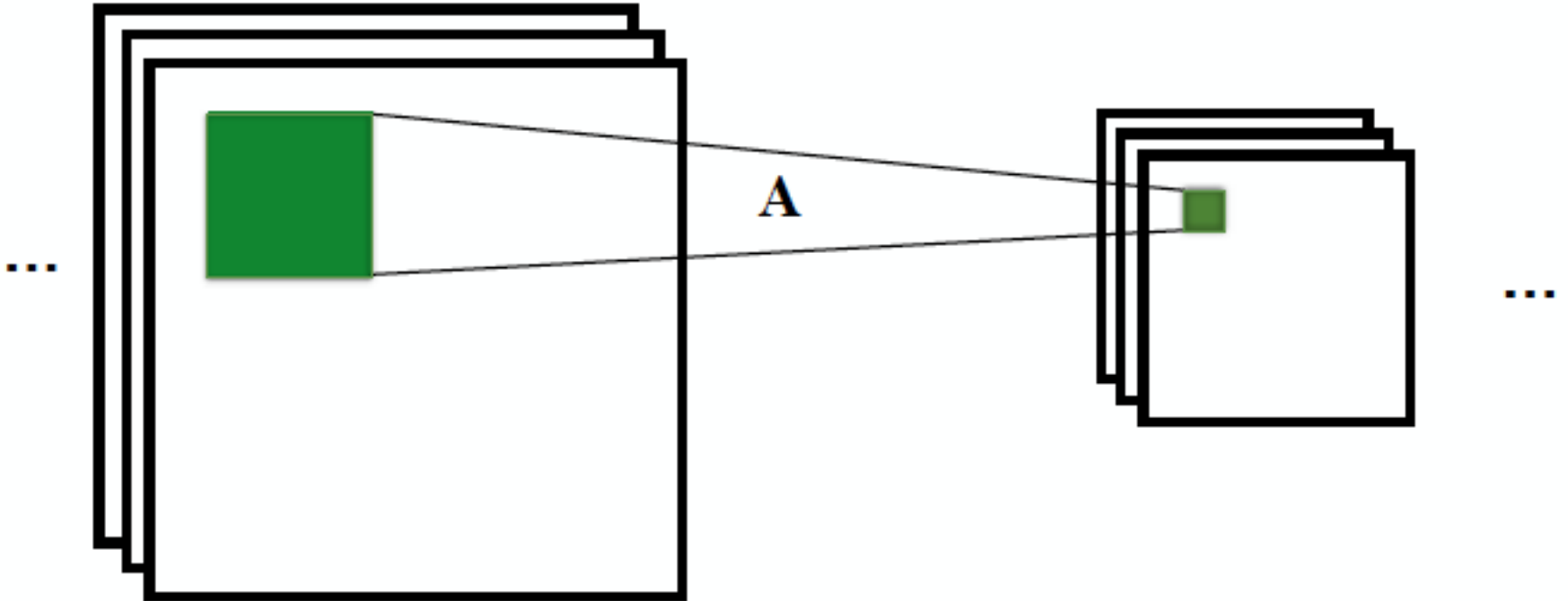
# Pooling



# Pooling

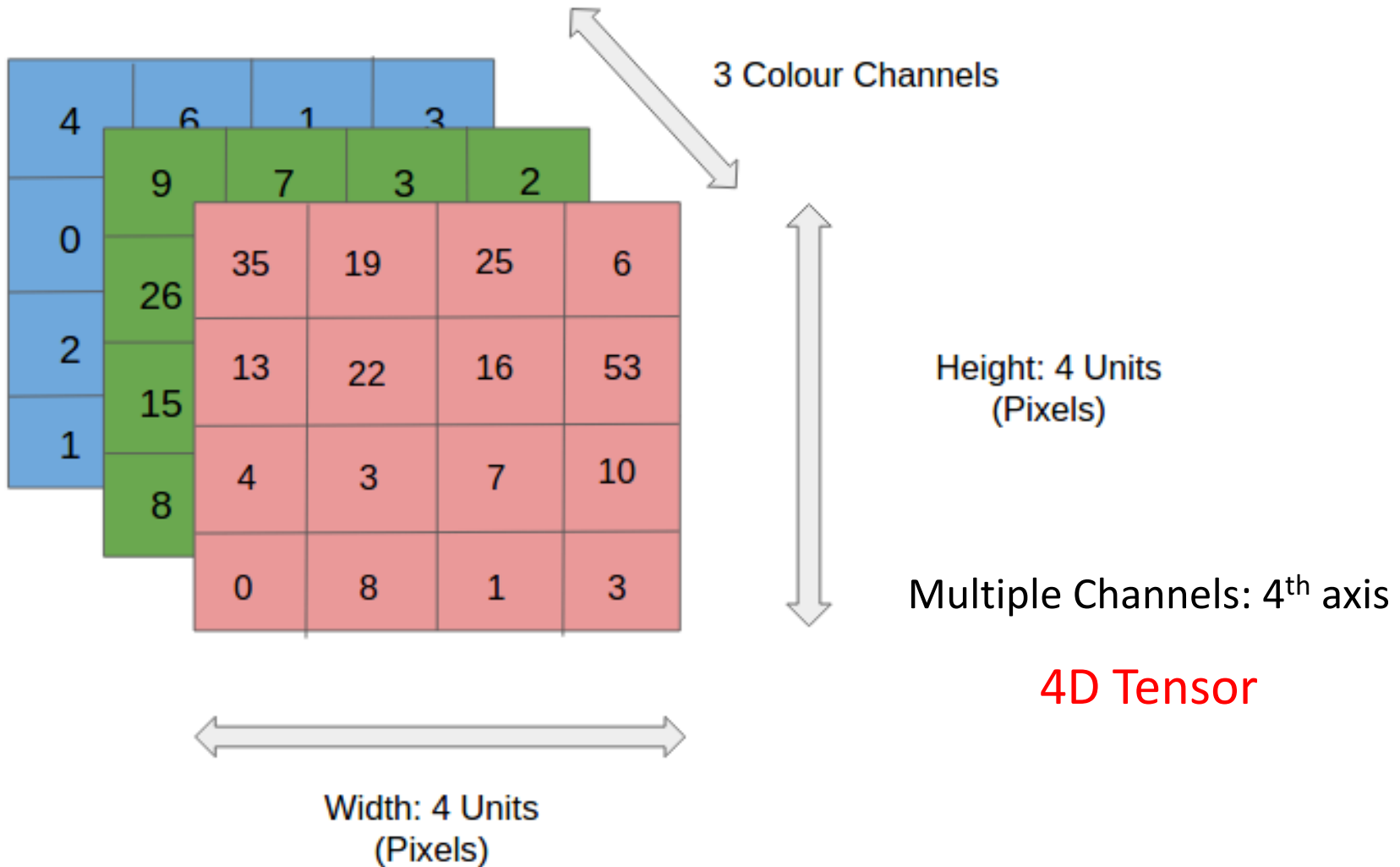
- Invariant to small, “local transitions”
  - Face detection: enough to check the presence of eyes, not their precise location
- Fewer pooling units than detector units
  - Reduces input size to final fully connected layers

# Multiple Channels



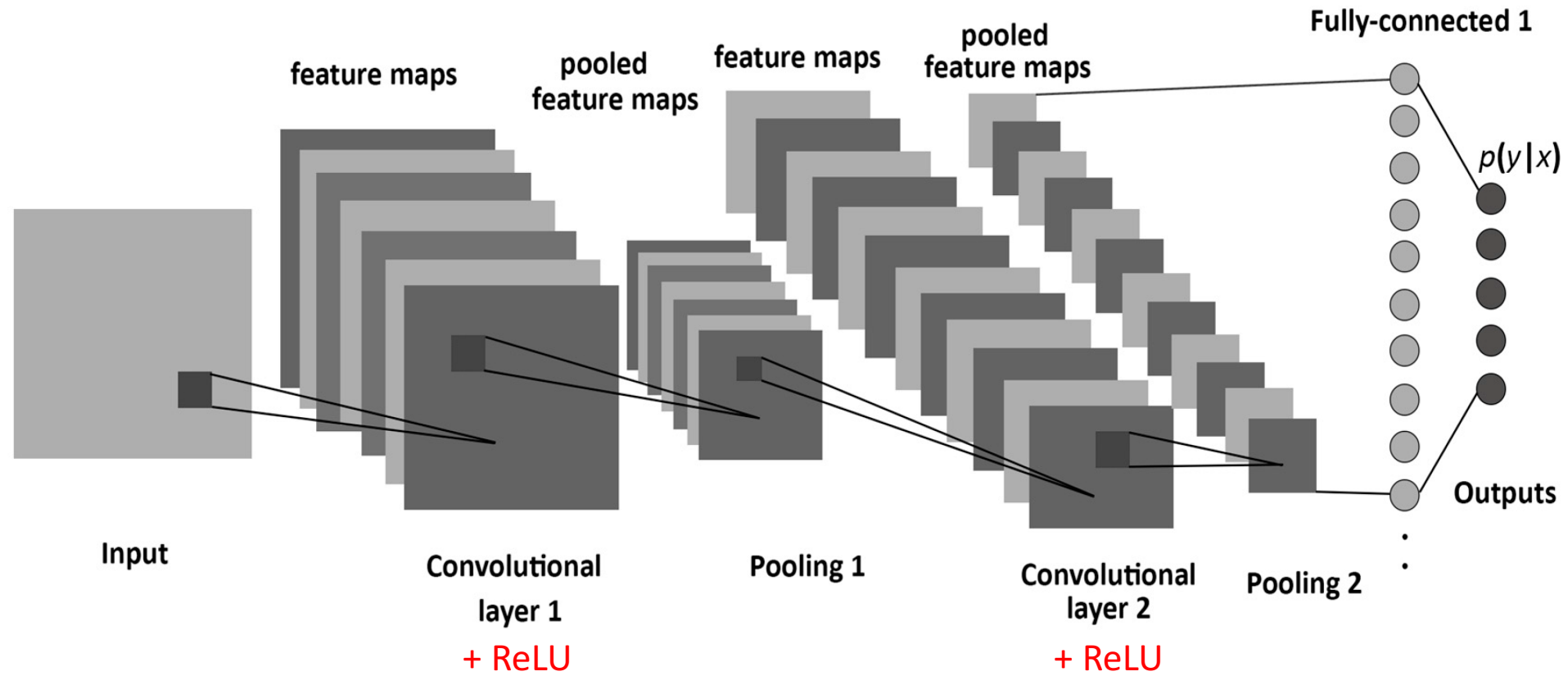
ConvNets can have multiple channels a.k.a. **feature maps**,  
each with separate parameters

# Multiple Dimensions





# Overall Architecture

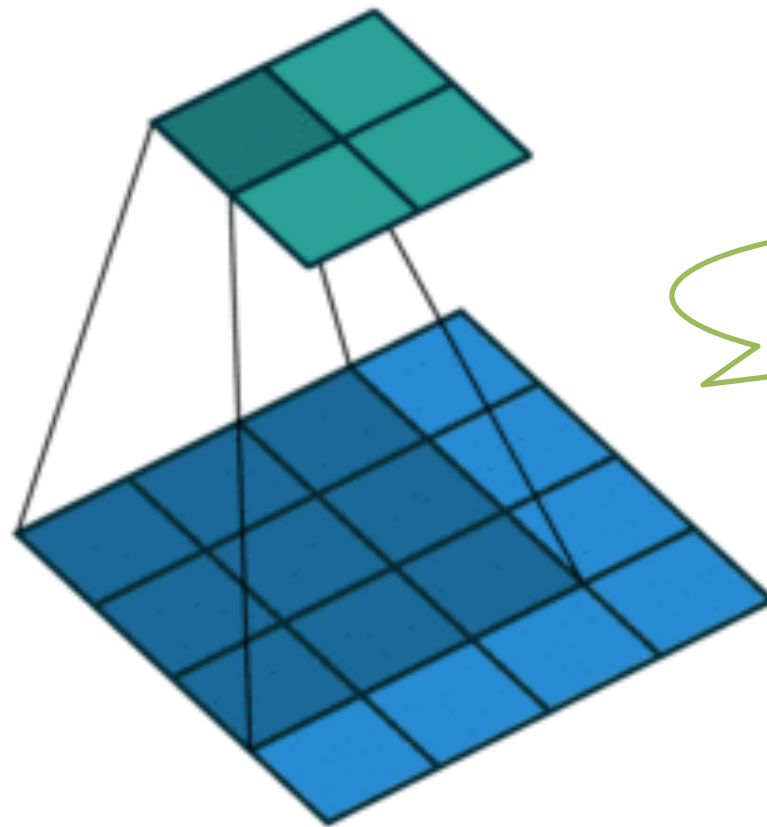


# Key Features

- Sparse interaction
- Parameter sharing
- Pooling
- Zero padding

# No Padding

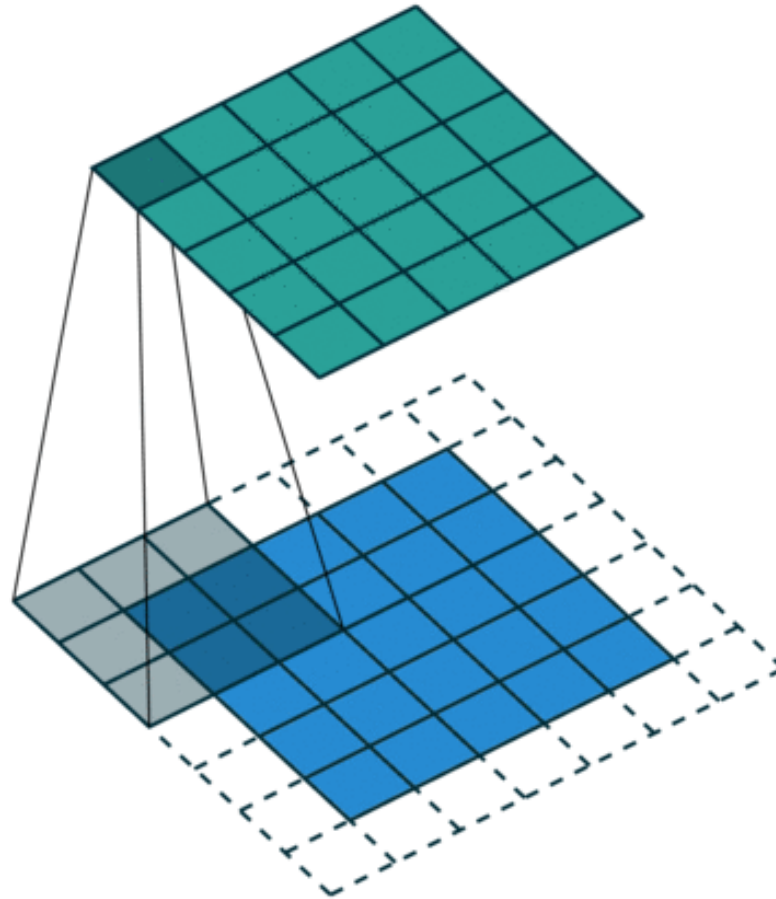
- Representation size **shrinks by 'kernel width - 1'** after each layer



Ignores large portion  
of input image

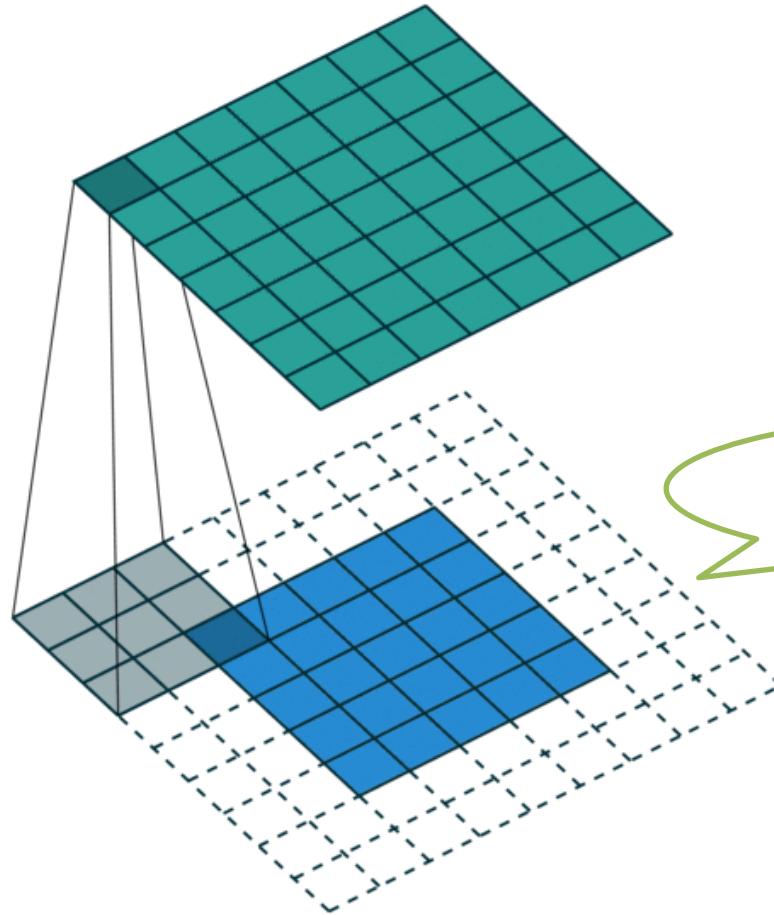
# Same Padding

- Pad zeros so that **output is same size as input**



# Full Padding

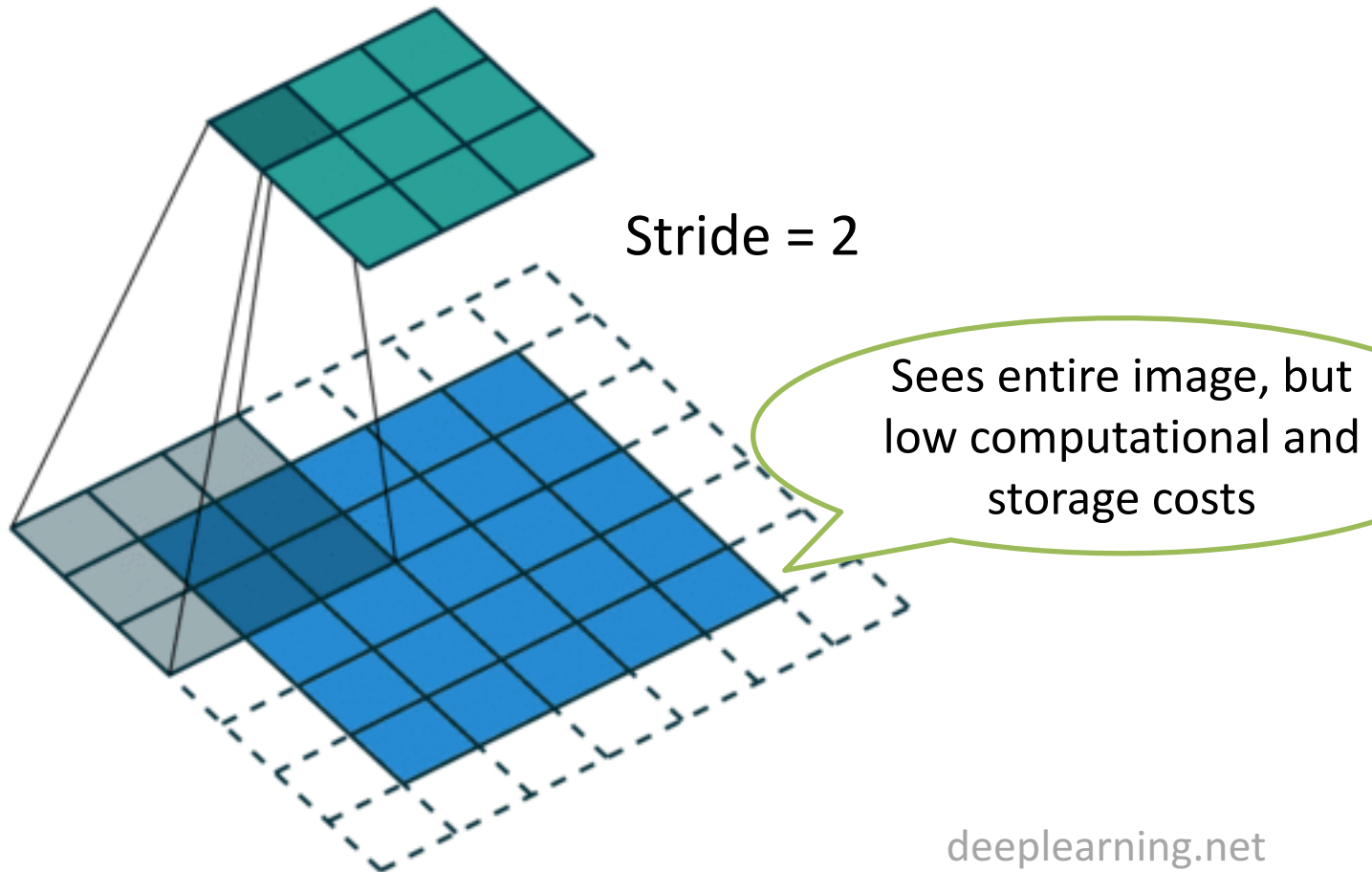
- Pad zeros so that every pixel is visited **same number of times**



Increases computation and storage cost

# Padding with Strides

- Shift the kernel not by 1 pixel, but by  $k$  pixels (stride)



# Key Features

- Sparse interaction
- Parameter sharing
- Pooling
- Zero padding

# Back-propagation

Propagate image forward



feature maps

pooled  
feature maps

feature maps

pooled  
feature maps

Fully-connected 1

$p(y|x)$

Outputs

Input

Convolutional  
layer 1

+ ReLU

Pooling 1

Convolutional  
layer 2

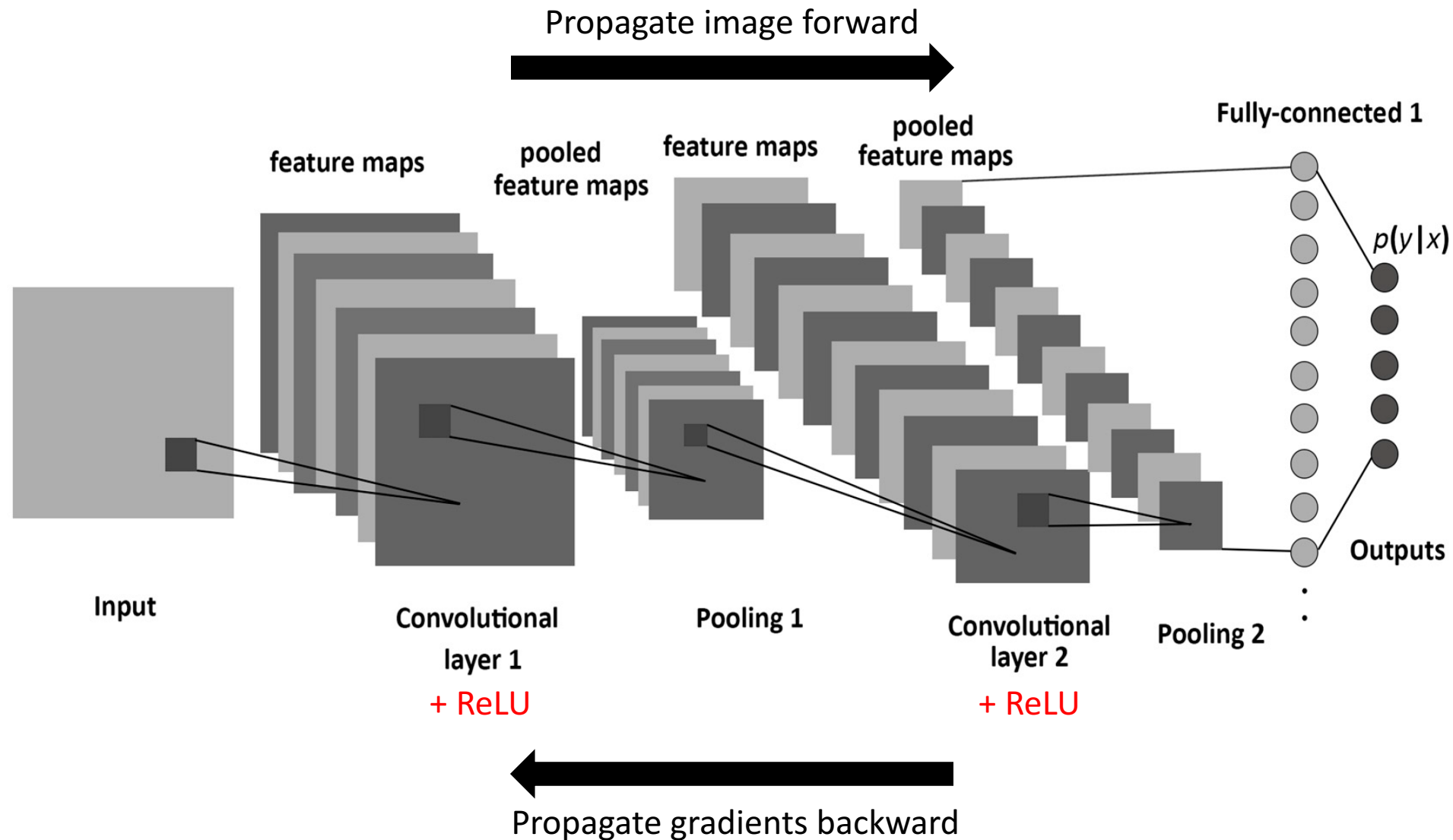
+ ReLU

Pooling 2

⋮



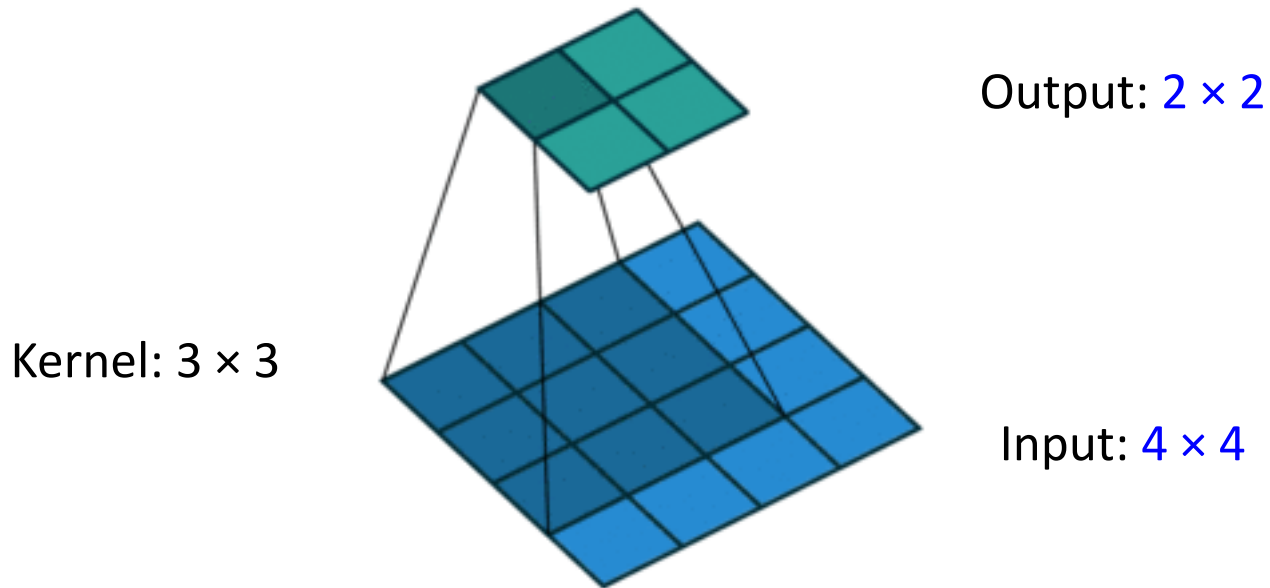
Propagate gradients backward





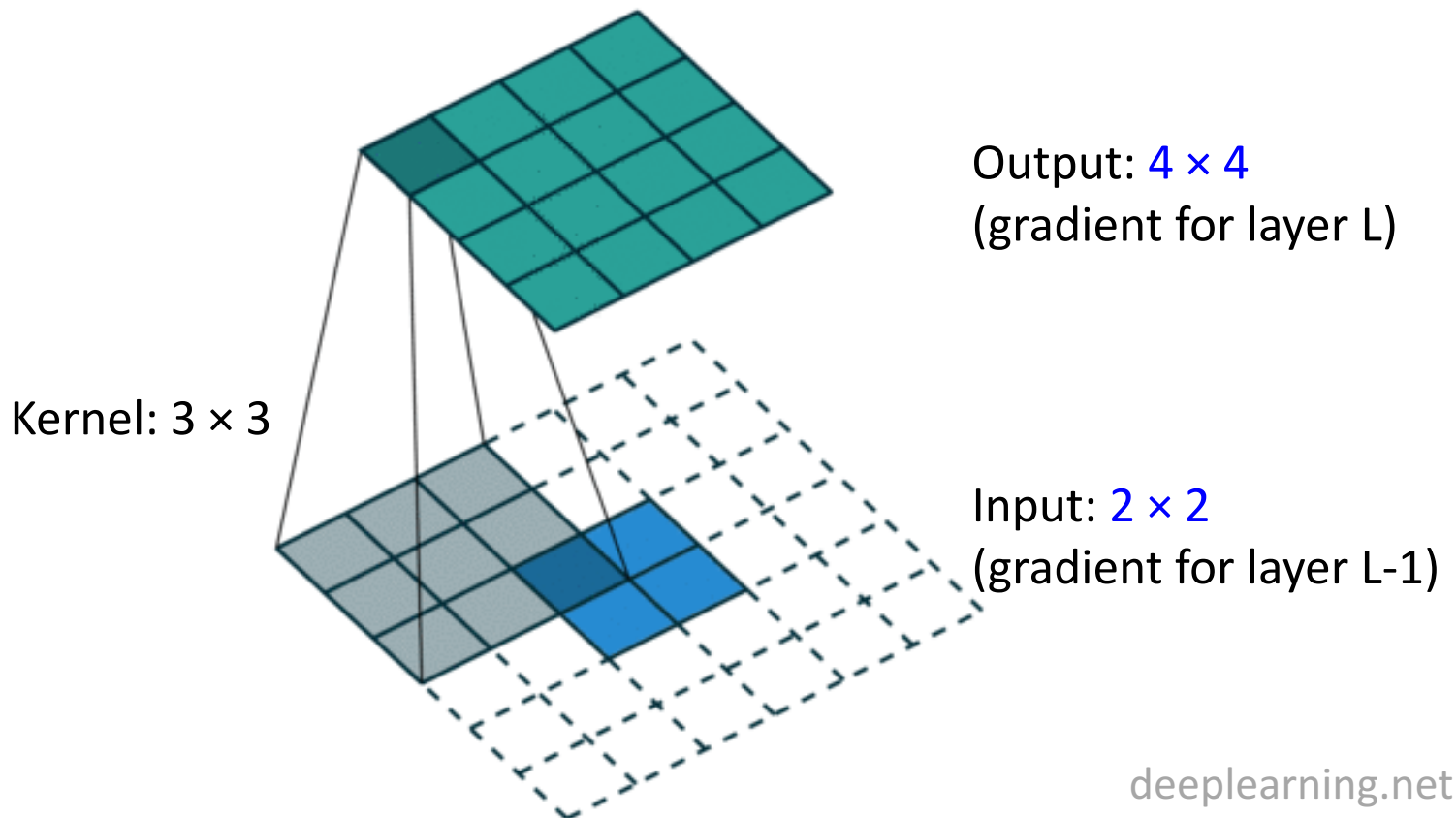
# Back-propagation

- Forward pass through a **convolution unit**



# Back-propagation

- Backward pass = **transpose convolution**



# Back-propagation

- Parameter matrix  $\mathbf{W}$  contains **offset copies of the kernel** matrix (Toeplitz matrix)

$$\mathbf{x}^L = \mathbf{W}\mathbf{x}^{L-1} \quad (\text{forward})$$

- Parameter matrix for the transpose convolution is  $\mathbf{W}^T$

$$\mathbf{g}^{L-1} = \mathbf{W}^T\mathbf{g}^L \quad (\text{backward})$$

# ConvNet Variants

- **Locally-connected Layers**
  - No parameter sharing, learn **separate weights** for each spatial location
  - Useful when each output needs to be a function of a small part of the input
- **Tiled Convolution**
  - Learn **set of kernels**, rotate them through space
  - Neighboring locations have different filters