

# Convolutional Neural Networks (8DC00)

Friso G. Heslinga

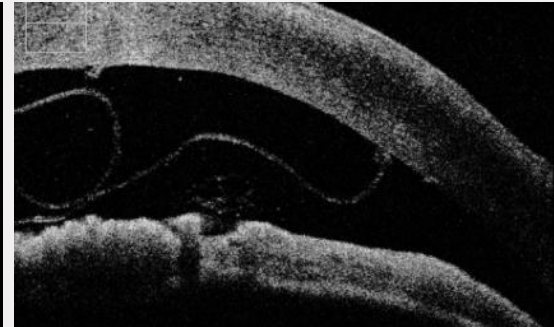
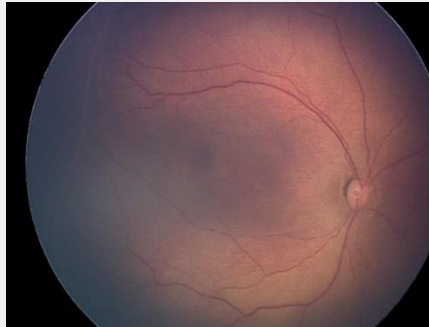
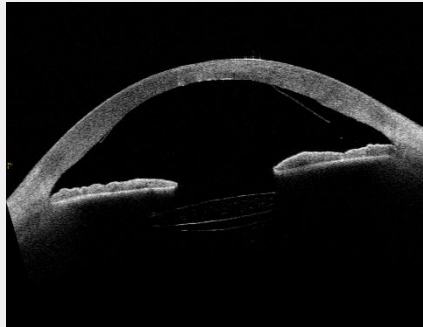
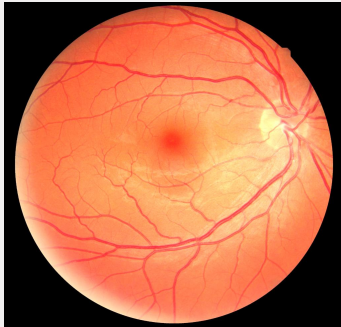
# Friso G. Heslinga MSc.



**Background:** BSc and MSc in Biomedical Engineering, MSc in Health Sciences (University of Twente).

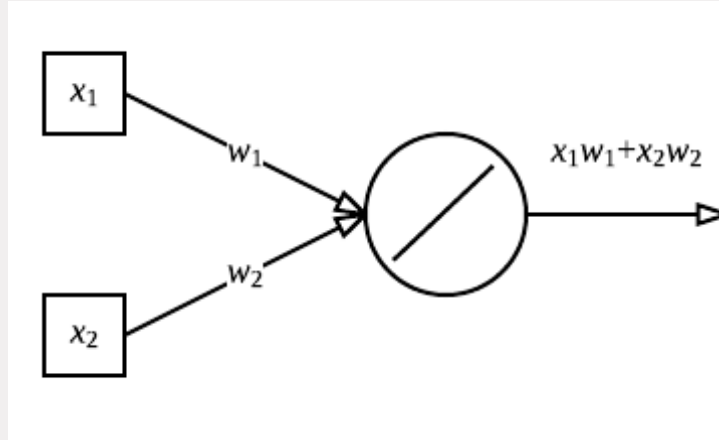
**Work/internship experience:** University of Western Australia, University of California - Berkeley, Harvard Medical School

**PhD Research:** Deep learning, Medical image analysis, Ophthalmology

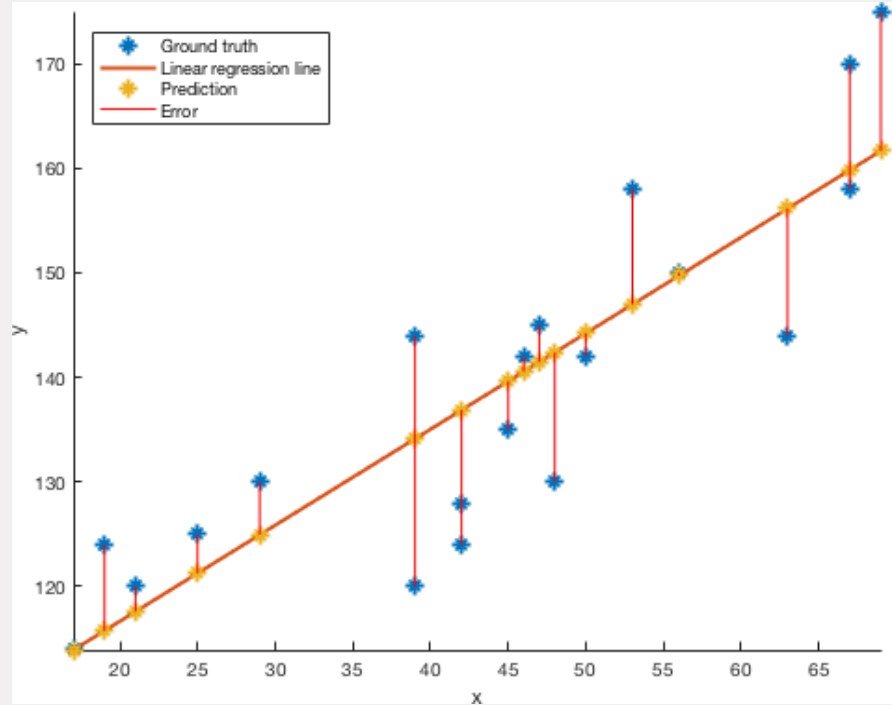


Week	Date	Lecturer	Topics
1	1 Sept.	Maureen	Course introduction; Software demo; Image registration (1)
	3 Sept.	Maureen	Image registration (2); Geometrical transformations
2	8 Sept.	Maureen	Point-based registration
	10 Sept.	Maureen	Intensity-based registration; Evaluation metrics
3	15 Sept.	<i>Catch-up day (no lecture)</i>	
	17 Sept.	Cornel Zachiu (UMCU)	Guest lecture 1: Image analysis for adaptive radiotherapy
4	22 Sept.	Mitko	Introduction to CAD; k-NN; Decision trees
	24 Sept.	Mitko	Generalization and overfitting
5	29 Sept.	Mitko	Logistic regression; Neural networks
	1 Oct.	Friso	Convolutional neural networks
6	6 Oct.	Friso	Deep learning frameworks and applications
	8 Oct.	Friso	Unsupervised machine learning
7	13 Oct.	Maureen	Deep learning for deformable image registration
	15 Oct.	Geert-Jan Rutten (ETZ)	Guest lecture 2: Image analysis in neurosurgery applications
8	20 Oct	<i>Self-study (no lecture)</i>	Active shape models
	22 Oct	<i>Self-study (no lecture)</i>	Active shape models

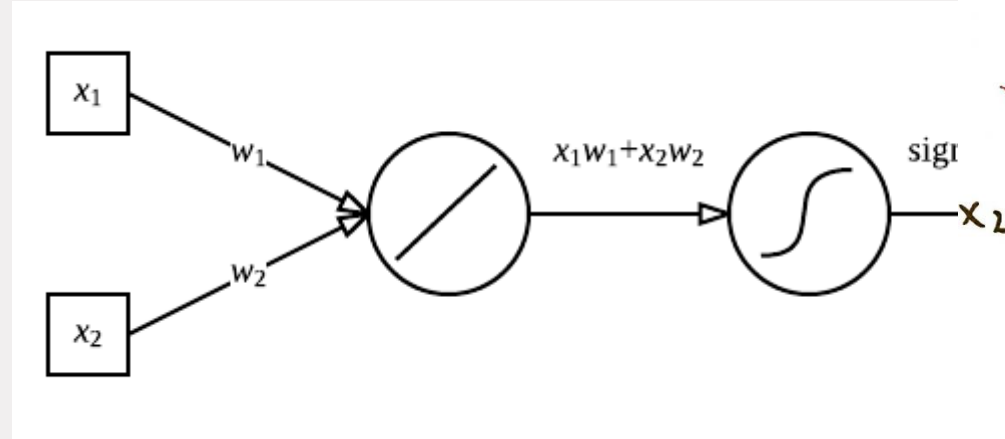
## Previously – Linear regression



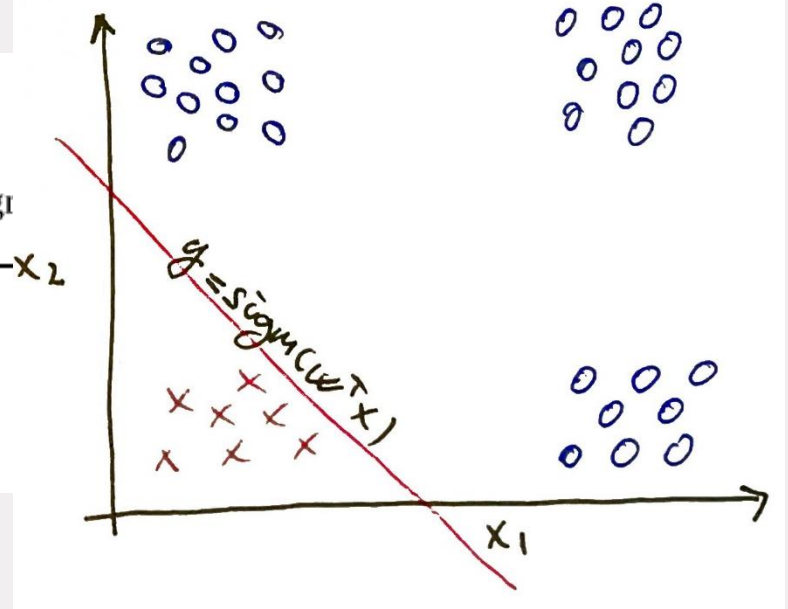
$$\hat{y} = \theta^T \mathbf{x}$$



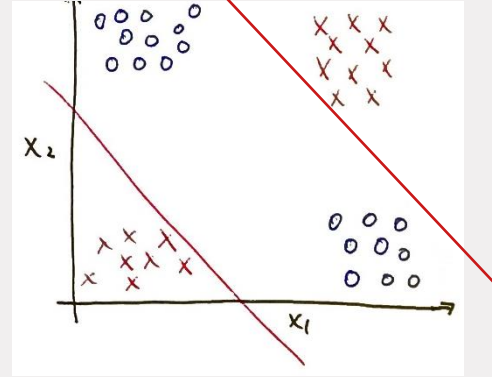
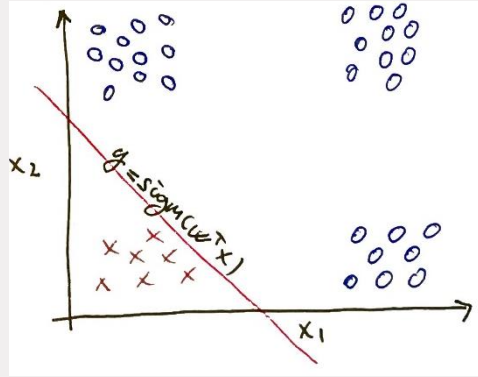
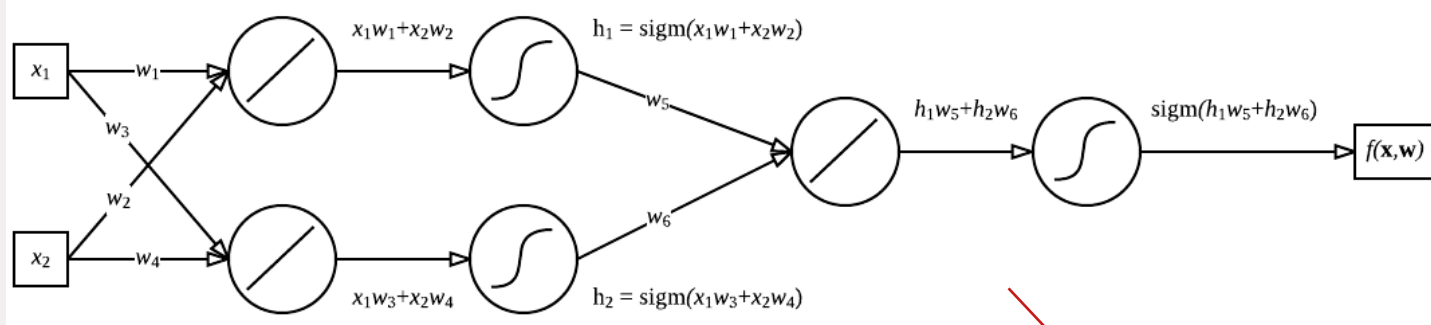
## Previously – Logistic regression



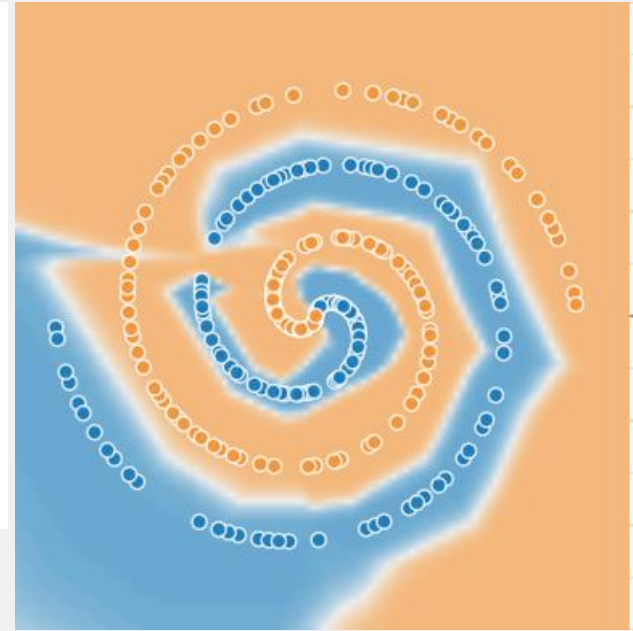
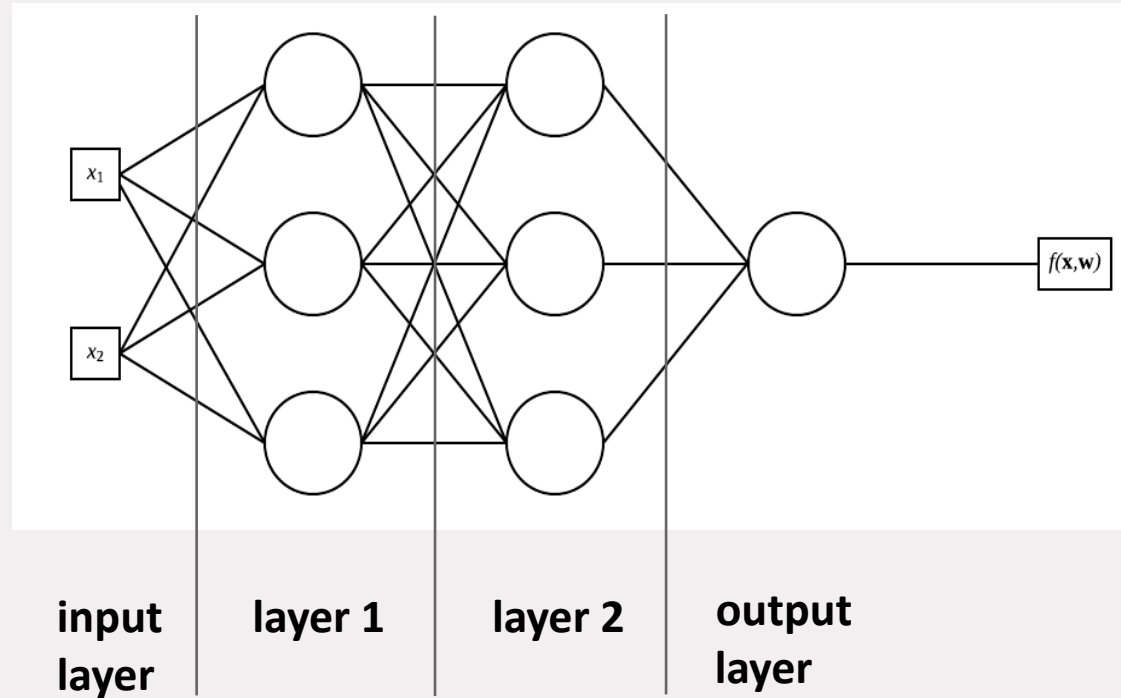
$$p(y = 1 \mid \mathbf{x}) = \sigma(\boldsymbol{\theta}^T \mathbf{x})$$



# Previously – Neural networks

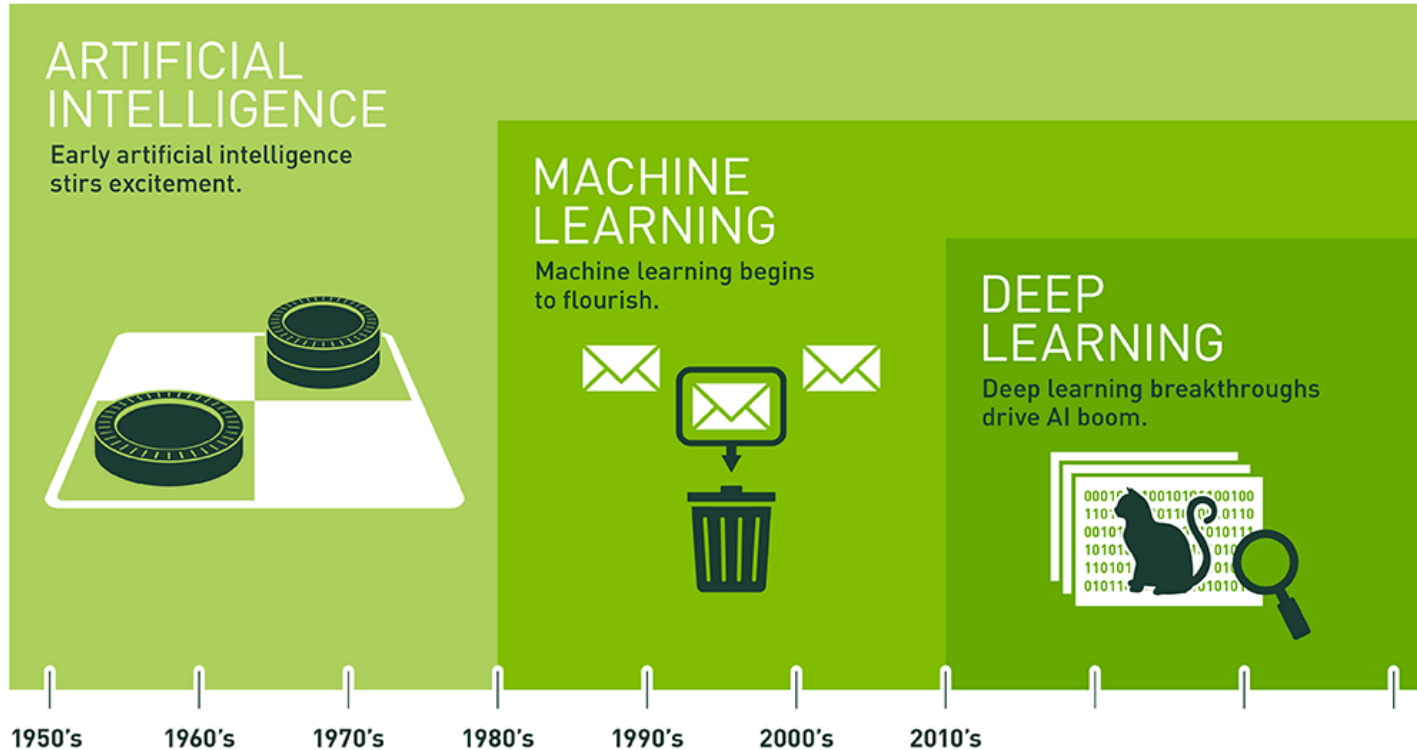


# Previously – Neural networks



# Previously – Machine learning

Figure source: nvidia.com





# Today: Convolutional neural networks (CNN)

- Neural networks → Convolutional neural networks

Building blocks for deep learning models for image analysis:

- Convolutional layer
- Max-pooling layer
- Not needed for the project, but will be on the exam

# Learning outcomes

- Student can explain the concept of **convolutions** in a neural network
- Student can describe why we can use a **convolutional approach** for (medical) **images**
- Student can explain why convolutions enable development of **deep** (and large) **neural networks**
- Students can explain and apply the **max-pooling layer** in a convolutional neural network
- Students can motivate the choice for a **kernel size**

# Lecture outline

- Images as input to neural network
- Reducing # of weights
- 1D convolutions
- 2D convolutions
- *Break (15 mins)*
- Kernels
- Max-pooling
- Interactive example

# Images as input to neural networks

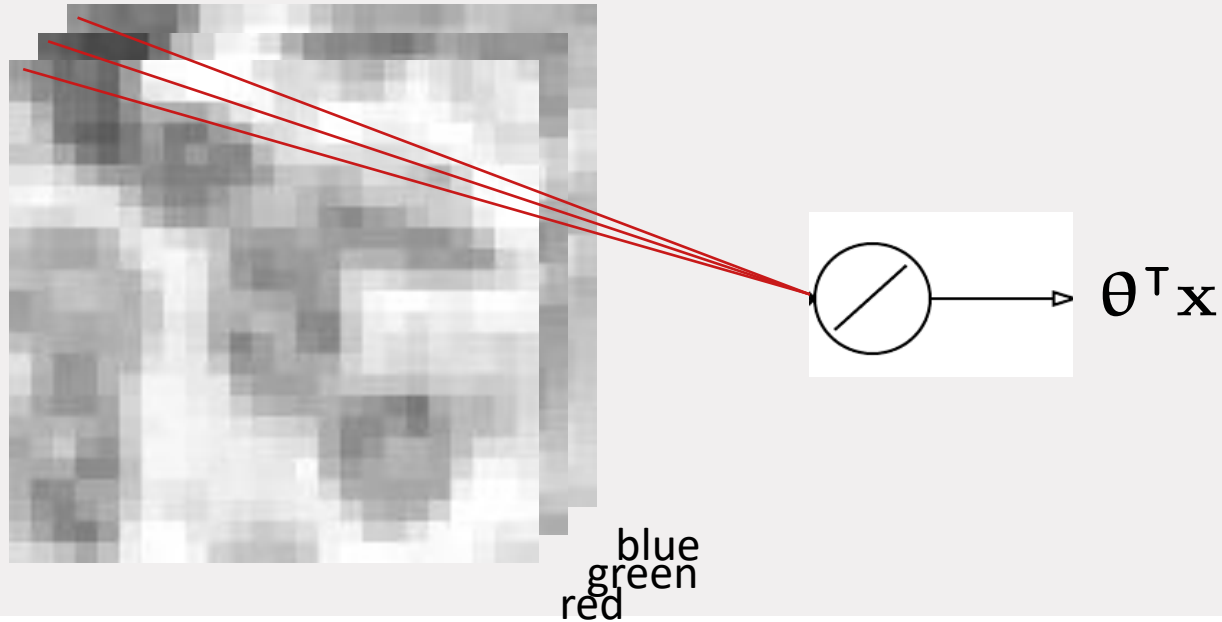


= 1728 features ( $\mathbf{x}_i$  are 1728 dimensional vectors)

If we train linear regression with these inputs (such as in the first practical), we will have 1728 weights  $w$  and a bias  $b$ .

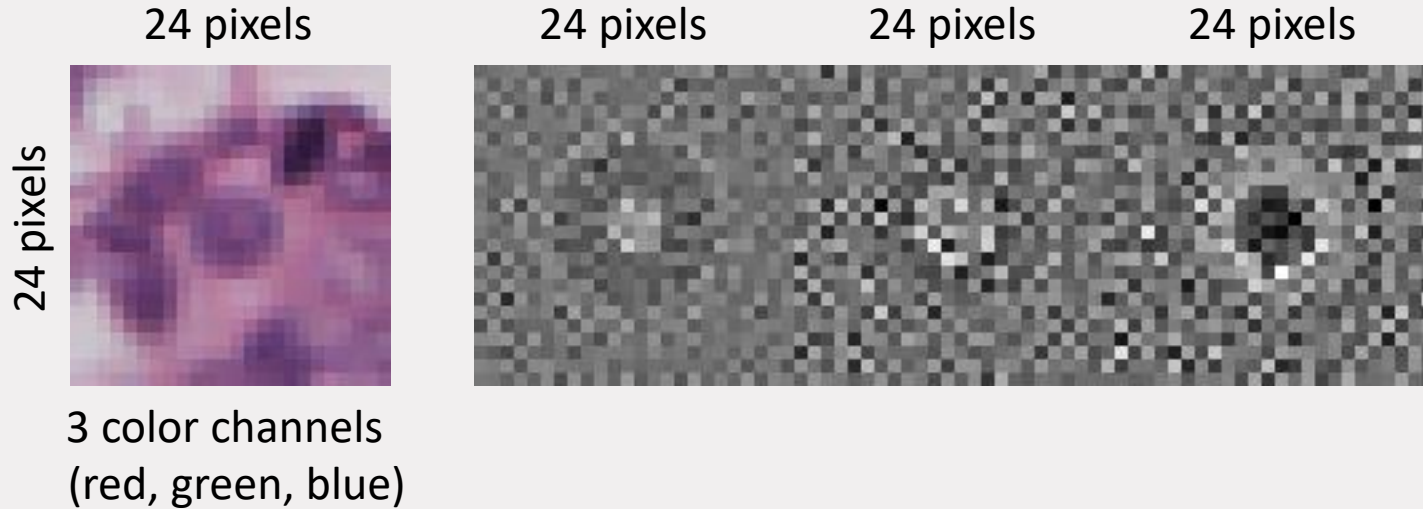
# Every pixel is an input

Every pixel from every color channel is multiplied by a weight



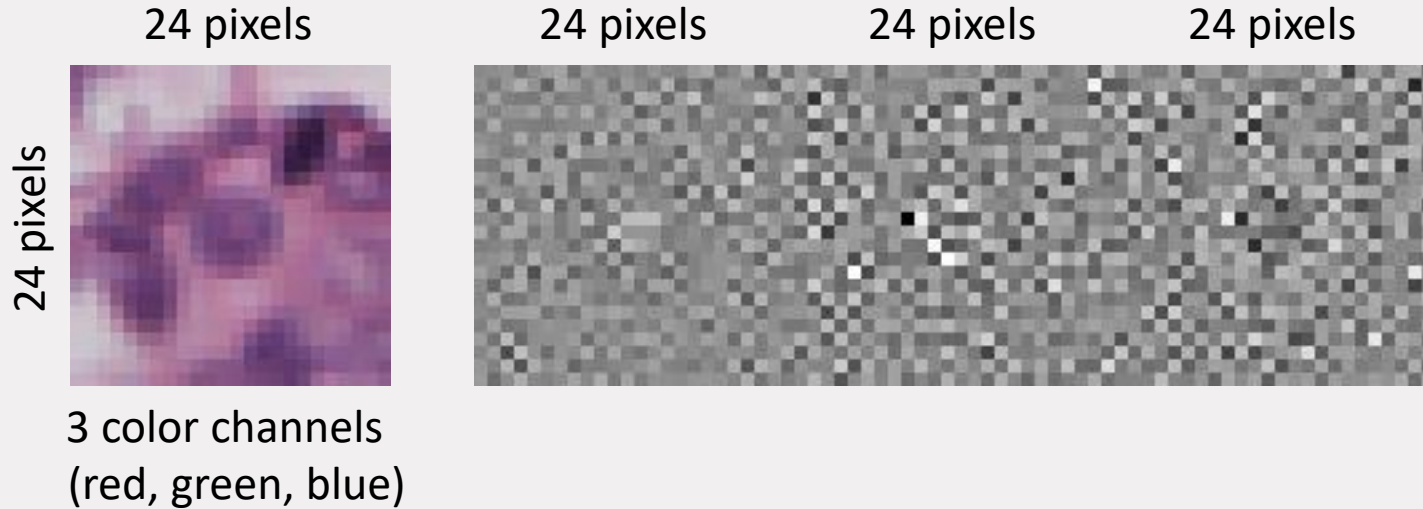
# Post-training: visualizing what model has learned

Reshape vector of parameters into 24x24x3 image



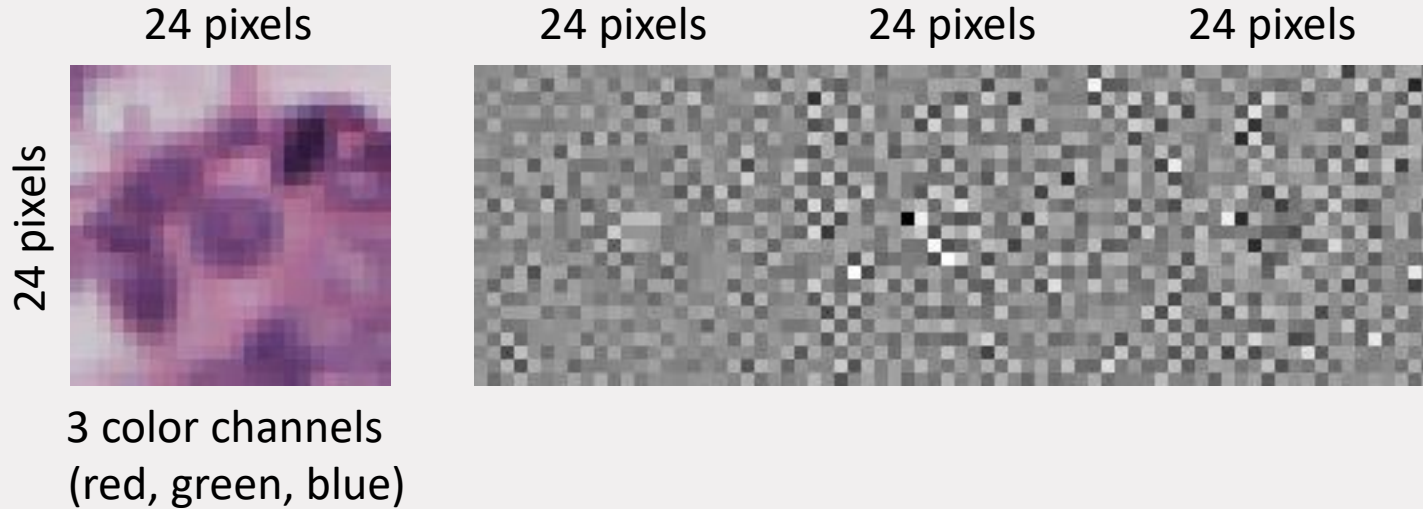
# Post-training: visualizing what model has learned

Only 25% of training samples.. Looks noisy!



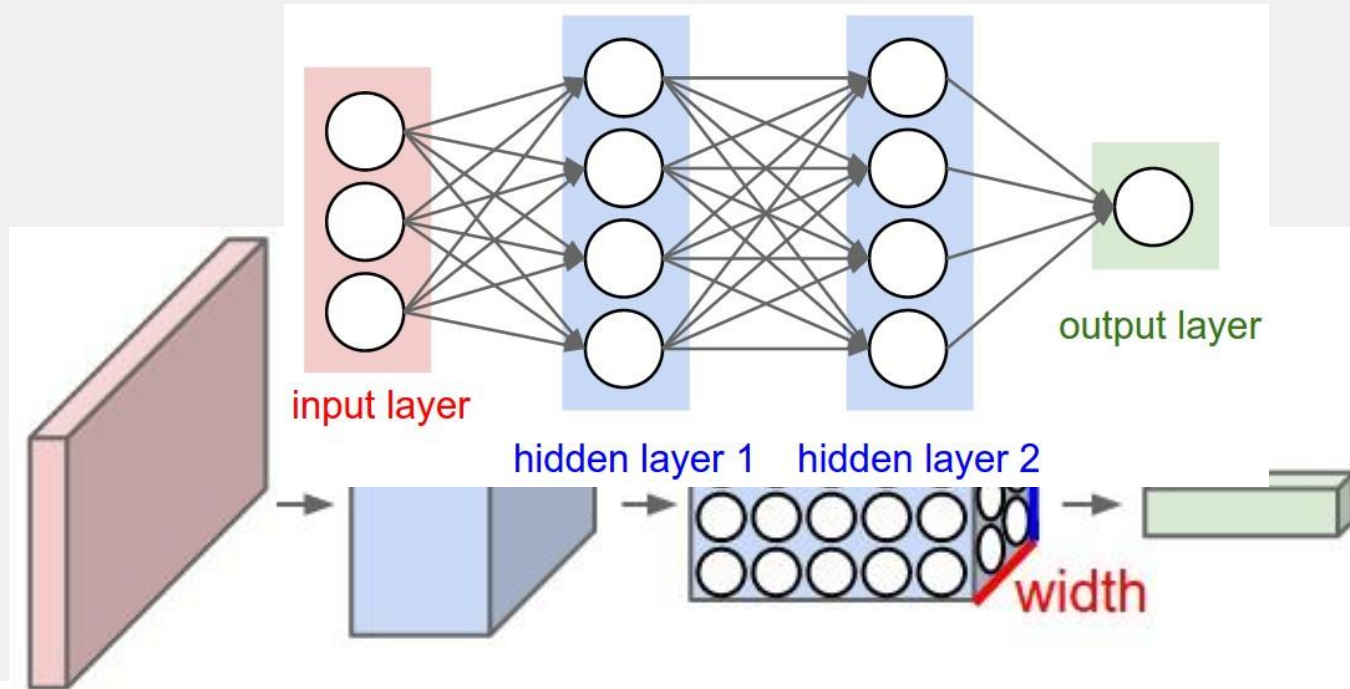
# Post-training: visualizing what model has learned

You can think of it as “there is not enough training data to reliably estimate all model weights”.

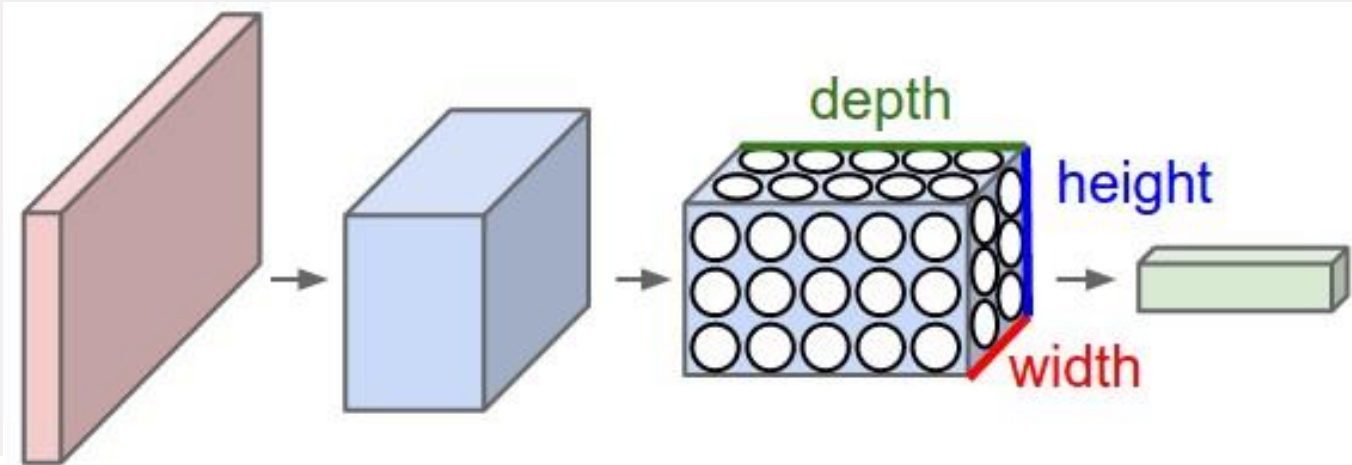




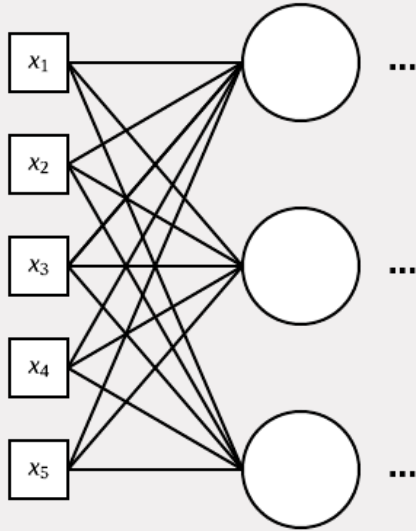
# Many weights



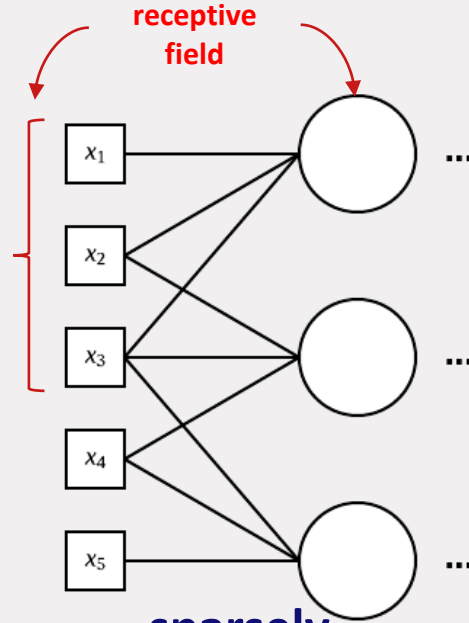
With large inputs such as images and deep networks, the number of weights "explodes". We need a way to reduce the number of weights, without sacrificing performance.



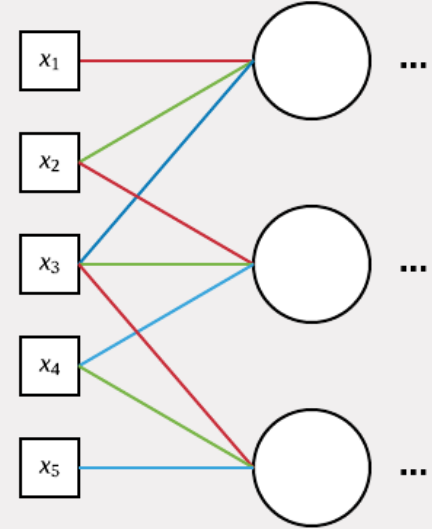
## How many weights?



**“regular” NN**  
15 weights

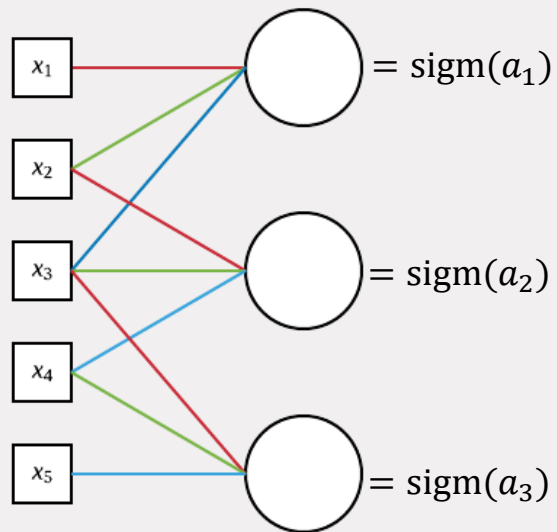


**sparsely  
connected NN**  
9 weights



**shared weights**  
3 weights

# Shared weights



**shared weights**  
3 weights

$$a_1 = x_1 w_1 + x_2 w_2 + x_3 w_3$$

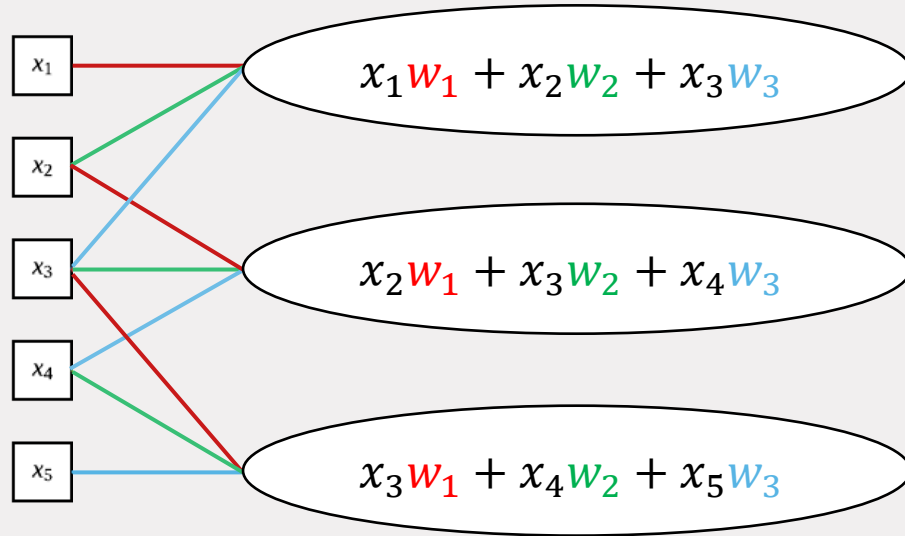
$$a_2 = x_2 w_1 + x_3 w_2 + x_4 w_3$$

$$a_3 = x_3 w_1 + x_4 w_2 + x_5 w_3$$

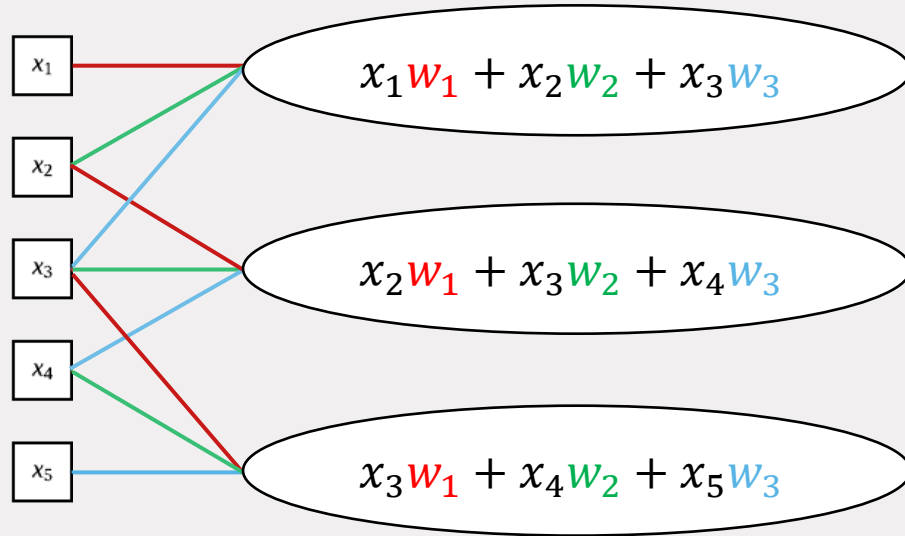
$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix} * \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix}$$

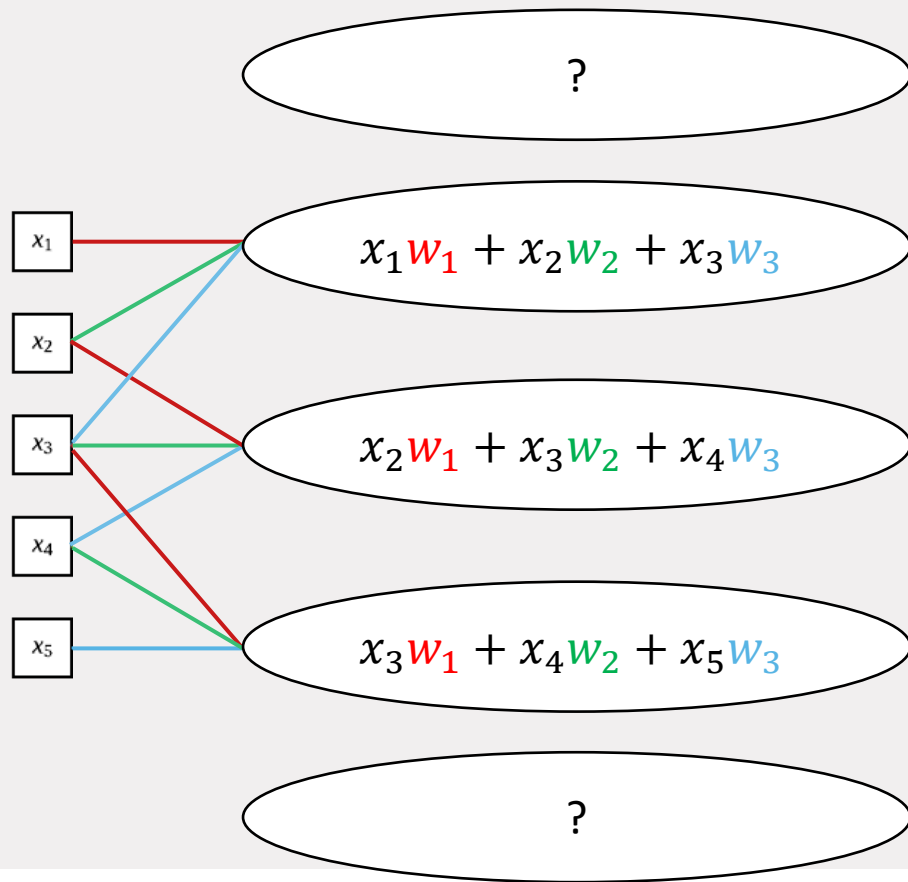
**convolution, thus convolutional NN**

# 1-D Convolution



# 1-D Convolution



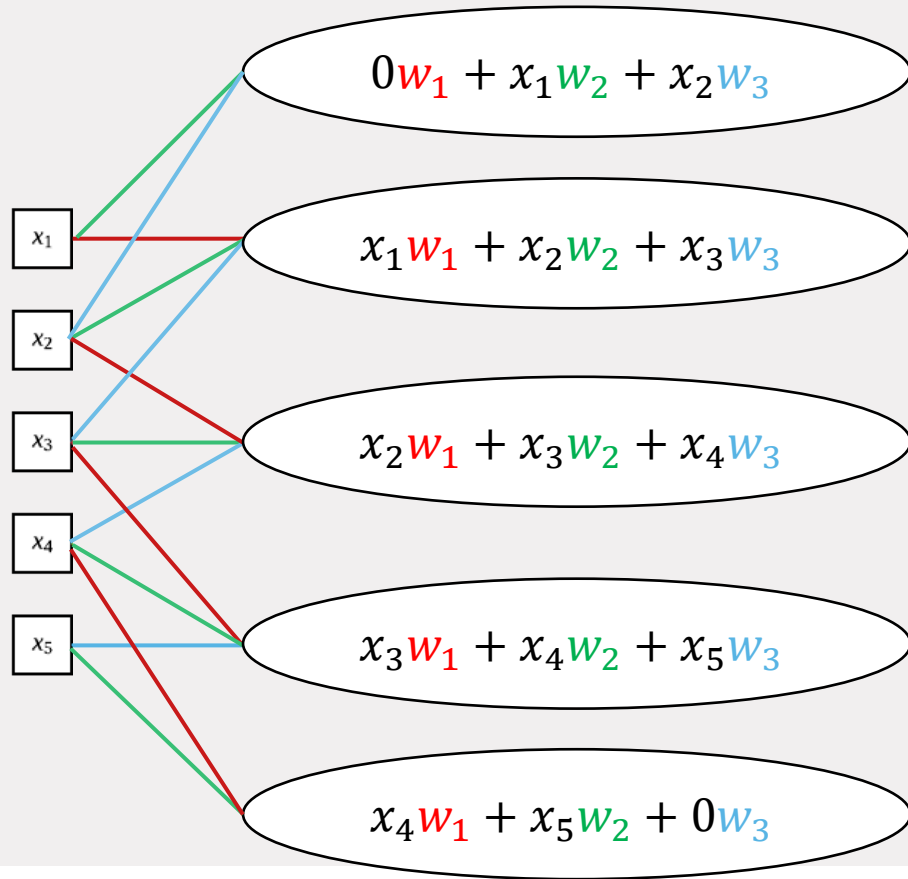


## 1-D Convolution

How can we keep the same number of features in hidden layer 1?

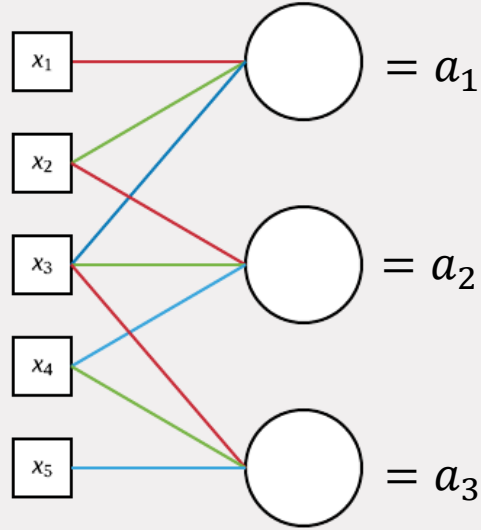
# 1-D Convolution

Zero-padding!





# 1-D Convolution



$$a_1 = x_1 w_1 + x_2 w_2 + x_3 w_3$$

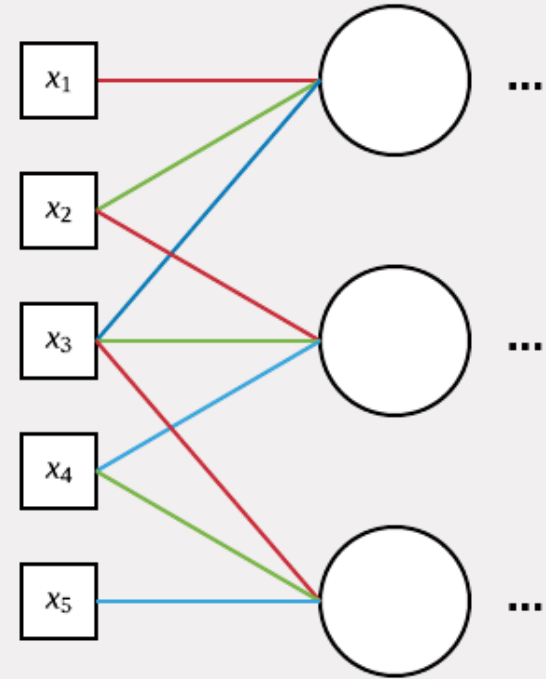
$$a_2 = x_2 w_1 + x_3 w_2 + x_4 w_3$$

$$a_3 = x_3 w_1 + x_4 w_2 + x_5 w_3$$

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix} * \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix}$$

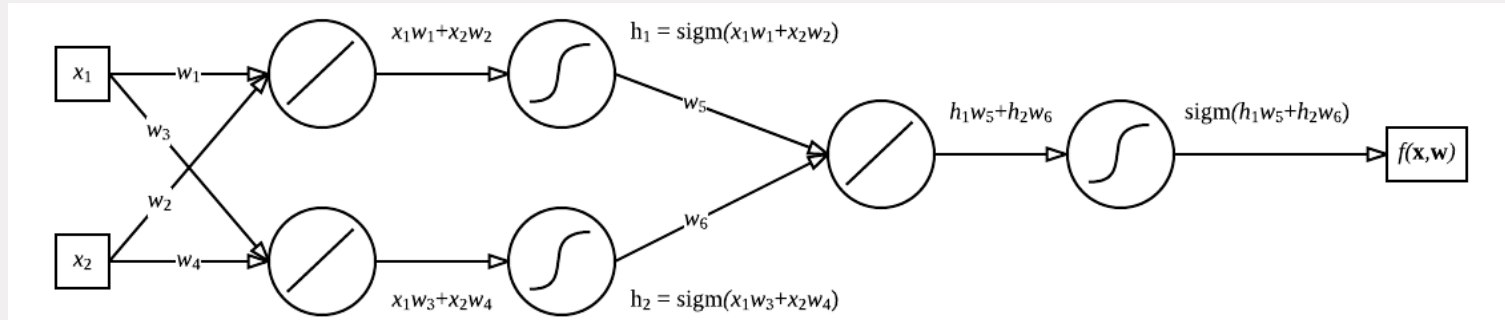
# Properties of convolutional neural networks

- Sparse connectivity
- Weight sharing
- Parallel computations



# Why does this work?

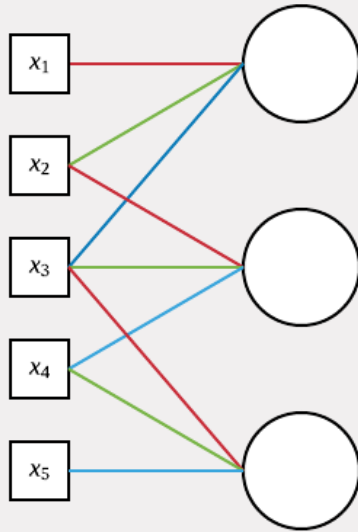
- Multiple layers
- Hidden layers contain features calculated from previous layers



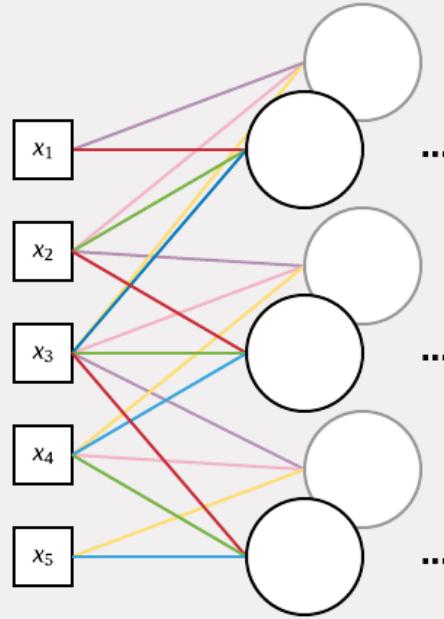
# Why does this work?

- Different layers contain different transformation
  - Simple (e.g. edges, colors)
  - Complex (final layers)

One added benefit is that the learned transformations will be equivariant with translation (if the features/image is shifted up/down the features will still be detected).



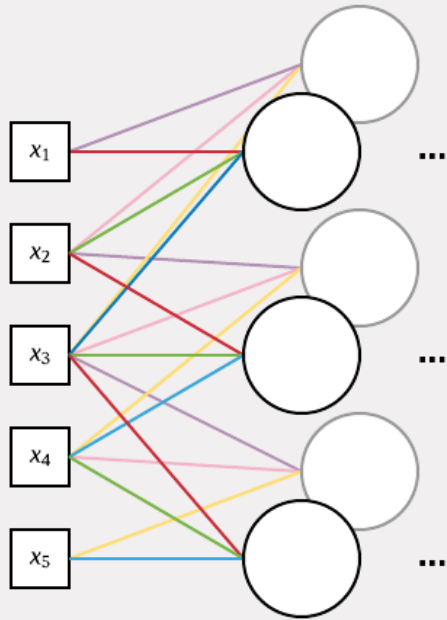
**shared weights**  
3 weights



**two sets shared weights**  
6 weights

We can add additional sets of weights that can learn additional interesting transformation of the input.

Note that the added neurons are not a new layer. They are part of layer 1.



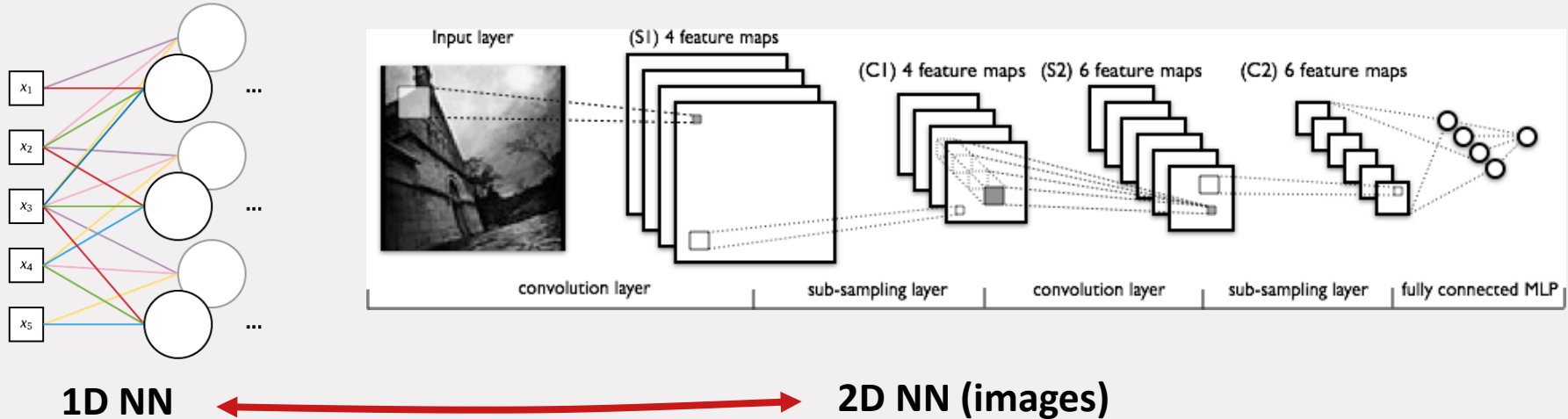
**two sets shared weights**  
6 weights

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix} * \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}$$

$$\begin{bmatrix} a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix} * \begin{bmatrix} w_{2,1} & w_{2,2} & w_{2,3} \end{bmatrix}$$

$\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}$ , and  $\begin{bmatrix} w_{2,1} & w_{2,2} & w_{2,3} \end{bmatrix}$  are **convolution kernels**. They extract features. However, they are not hand-designed features – they were learned by the neural network.

# Convolutional neural networks are ideal for images



Because of the weight sharing, convolutional neural networks only work with structured data (such as images) as inputs.

$$\begin{array}{|c|c|c|c|c|} \hline I_1 & I_2 & I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 & I_9 & I_{10} \\ \hline I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \\ \hline I_{16} & I_{17} & I_{18} & I_{19} & I_{20} \\ \hline I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline w_1 & w_2 & w_3 \\ \hline w_4 & w_5 & w_6 \\ \hline w_7 & w_8 & w_9 \\ \hline \end{array} =$$

$$\begin{array}{|c|c|c|} \hline I_1 w_1 + I_2 w_2 + I_3 w_3 & & \\ \hline + & & \\ I_6 w_4 + I_7 w_5 + I_8 w_6 & & \\ \hline + & & \\ I_{11} w_7 + I_{12} w_8 + I_{13} w_9 & & \\ \hline \end{array}$$



$I_1$	$I_2$	$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$	$I_9$	$I_{10}$
$I_{11}$	$I_{12}$	$I_{13}$	$I_{14}$	$I_{15}$
$I_{16}$	$I_{17}$	$I_{18}$	$I_{19}$	$I_{20}$
$I_{21}$	$I_{22}$	$I_{23}$	$I_{24}$	$I_{25}$

\*

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

=

$I_1 w_1 + I_2 w_2 + I_3 w_3$ $+$ $I_6 w_4 + I_7 w_5 + I_8 w_6$ $+$ $I_{11} w_7 + I_{12} w_8 + I_{13} w_9$	$I_2 w_1 + I_3 w_2 + I_4 w_3$ $+$ $I_7 w_4 + I_8 w_5 + I_9 w_6$ $+$ $I_{12} w_7 + I_{13} w_8 + I_{14} w_9$	

$$\begin{bmatrix} I_1 & I_2 & I_3 & I_4 & I_5 \\ I_6 & I_7 & I_8 & I_9 & I_{10} \\ I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \\ I_{16} & I_{17} & I_{18} & I_{19} & I_{20} \\ I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \end{bmatrix} * \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} =$$

$$\begin{bmatrix} I_1 w_1 + I_2 w_2 + I_3 w_3 + I_6 w_4 + I_7 w_5 + I_8 w_6 + I_{11} w_7 + I_{12} w_8 + I_{13} w_9 & I_2 w_1 + I_3 w_2 + I_4 w_3 + I_7 w_4 + I_8 w_5 + I_9 w_6 + I_{12} w_7 + I_{13} w_8 + I_{14} w_9 & I_3 w_1 + I_4 w_2 + I_5 w_3 + I_8 w_4 + I_9 w_5 + I_{10} w_6 + I_{13} w_7 + I_{14} w_8 + I_{15} w_9 \\ & & \\ & & \\ & & \\ & & \end{bmatrix}$$

$$\begin{array}{|c|c|c|c|c|} \hline I_1 & I_2 & I_3 & I_4 & I_5 \\ \hline I_6 & I_7 & I_8 & I_9 & I_{10} \\ \hline I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \\ \hline I_{16} & I_{17} & I_{18} & I_{19} & I_{20} \\ \hline I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \\ \hline \end{array}
 *
 \begin{array}{|c|c|c|} \hline w_1 & w_2 & w_3 \\ \hline w_4 & w_5 & w_6 \\ \hline w_7 & w_8 & w_9 \\ \hline \end{array}
 =$$

$$\begin{array}{|c|c|c|} \hline I_1 w_1 + I_2 w_2 + I_3 w_3 + I_6 w_4 + I_7 w_5 + I_8 w_6 + I_{11} w_7 + I_{12} w_8 + I_{13} w_9 \\ \hline I_2 w_1 + I_3 w_2 + I_4 w_3 + I_7 w_4 + I_8 w_5 + I_9 w_6 + I_{12} w_7 + I_{13} w_8 + I_{14} w_9 \\ \hline I_3 w_1 + I_4 w_2 + I_5 w_3 + I_8 w_4 + I_9 w_5 + I_{10} w_6 + I_{13} w_7 + I_{14} w_8 + I_{15} w_9 \\ \hline I_6 w_1 + I_7 w_2 + I_8 w_3 + I_{11} w_4 + I_{12} w_5 + I_{13} w_6 + I_{16} w_7 + I_{17} w_8 + I_{18} w_9 \\ \hline \\ \hline \\ \hline \\ \hline \end{array}$$

$I_1$	$I_2$	$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$	$I_9$	$I_{10}$
$I_{11}$	$I_{12}$	$I_{13}$	$I_{14}$	$I_{15}$
$I_{16}$	$I_{17}$	$I_{18}$	$I_{19}$	$I_{20}$
$I_{21}$	$I_{22}$	$I_{23}$	$I_{24}$	$I_{25}$

\*

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

=

$  \begin{aligned}  &I_1 w_1 + I_2 w_2 + I_3 w_3 \\  &+ \\  &I_6 w_4 + I_7 w_5 + I_8 w_6 \\  &+ \\  &I_{11} w_7 + I_{12} w_8 + I_{13} w_9  \end{aligned}  $	$  \begin{aligned}  &I_2 w_1 + I_3 w_2 + I_4 w_3 \\  &+ \\  &I_7 w_4 + I_8 w_5 + I_9 w_6 \\  &+ \\  &I_{12} w_7 + I_{13} w_8 + I_{14} w_9  \end{aligned}  $	$  \begin{aligned}  &I_3 w_1 + I_4 w_2 + I_5 w_3 \\  &+ \\  &I_8 w_4 + I_9 w_5 + I_{10} w_6 \\  &+ \\  &I_{13} w_7 + I_{14} w_8 + I_{15} w_9  \end{aligned}  $
$  \begin{aligned}  &I_6 w_1 + I_7 w_2 + I_8 w_3 \\  &+ \\  &I_{11} w_4 + I_{12} w_5 + I_{13} w_6 \\  &+ \\  &I_{16} w_7 + I_{17} w_8 + I_{18} w_9  \end{aligned}  $	$  \begin{aligned}  &I_7 w_1 + I_8 w_2 + I_9 w_3 \\  &+ \\  &I_{12} w_4 + I_{13} w_5 + I_{14} w_6 \\  &+ \\  &I_{17} w_7 + I_{18} w_8 + I_{19} w_9  \end{aligned}  $	<div style="border: 1px solid black; height: 100px; width: 100%;"></div>
<div style="border: 1px solid black; height: 100px; width: 100%;"></div>	<div style="border: 1px solid black; height: 100px; width: 100%;"></div>	<div style="border: 1px solid black; height: 100px; width: 100%;"></div>

$$\begin{bmatrix} I_1 & I_2 & I_3 & I_4 & I_5 \\ I_6 & I_7 & I_8 & I_9 & I_{10} \\ I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \\ I_{16} & I_{17} & I_{18} & I_{19} & I_{20} \\ I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \end{bmatrix} * \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} =$$

$$\begin{bmatrix} I_1 w_1 + I_2 w_2 + I_3 w_3 + I_6 w_4 + I_7 w_5 + I_8 w_6 + I_{11} w_7 + I_{12} w_8 + I_{13} w_9 \\ I_2 w_1 + I_3 w_2 + I_4 w_3 + I_7 w_4 + I_8 w_5 + I_9 w_6 + I_{12} w_7 + I_{13} w_8 + I_{14} w_9 \\ I_3 w_1 + I_4 w_2 + I_5 w_3 + I_8 w_4 + I_9 w_5 + I_{10} w_6 + I_{13} w_7 + I_{14} w_8 + I_{15} w_9 \\ I_6 w_1 + I_7 w_2 + I_8 w_3 + I_{11} w_4 + I_{12} w_5 + I_{13} w_6 + I_{16} w_7 + I_{17} w_8 + I_{18} w_9 \\ I_7 w_1 + I_8 w_2 + I_9 w_3 + I_{12} w_4 + I_{13} w_5 + I_{14} w_6 + I_{17} w_7 + I_{18} w_8 + I_{19} w_9 \\ I_8 w_1 + I_9 w_2 + I_{10} w_3 + I_{13} w_4 + I_{14} w_5 + I_{15} w_6 + I_{18} w_7 + I_{19} w_8 + I_{20} w_9 \end{bmatrix}$$

$$\begin{array}{ccccc}
 \boxed{I_1} & \boxed{I_2} & \boxed{I_3} & \boxed{I_4} & \boxed{I_5} \\
 \boxed{I_6} & \boxed{I_7} & \boxed{I_8} & \boxed{I_9} & \boxed{I_{10}} \\
 \boxed{I_{11}} & \boxed{I_{12}} & \boxed{I_{13}} & \boxed{I_{14}} & \boxed{I_{15}} \\
 \boxed{I_{16}} & \boxed{I_{17}} & \boxed{I_{18}} & \boxed{I_{19}} & \boxed{I_{20}} \\
 \boxed{I_{21}} & \boxed{I_{22}} & \boxed{I_{23}} & \boxed{I_{24}} & \boxed{I_{25}}
 \end{array}
 *
 \begin{array}{ccc}
 \boxed{w_1} & \boxed{w_2} & \boxed{w_3} \\
 \boxed{w_4} & \boxed{w_5} & \boxed{w_6} \\
 \boxed{w_7} & \boxed{w_8} & \boxed{w_9}
 \end{array}
 =$$

$$\begin{array}{ccc}
 \begin{array}{c}
 I_1 w_1 + I_2 w_2 + I_3 w_3 \\
 + \\
 I_6 w_4 + I_7 w_5 + I_8 w_6 \\
 + \\
 I_{11} w_7 + I_{12} w_8 + I_{13} w_9
 \end{array}
 &
 \begin{array}{c}
 I_2 w_1 + I_3 w_2 + I_4 w_3 \\
 + \\
 I_7 w_4 + I_8 w_5 + I_9 w_6 \\
 + \\
 I_{12} w_7 + I_{13} w_8 + I_{14} w_9
 \end{array}
 &
 \begin{array}{c}
 I_3 w_1 + I_4 w_2 + I_5 w_3 \\
 + \\
 I_8 w_4 + I_9 w_5 + I_{10} w_6 \\
 + \\
 I_{13} w_7 + I_{14} w_8 + I_{15} w_9
 \end{array} \\
 \begin{array}{c}
 I_6 w_1 + I_7 w_2 + I_8 w_3 \\
 + \\
 I_{11} w_4 + I_{12} w_5 + I_{13} w_6 \\
 + \\
 I_{16} w_7 + I_{17} w_8 + I_{18} w_9
 \end{array}
 &
 \begin{array}{c}
 I_7 w_1 + I_8 w_2 + I_9 w_3 \\
 + \\
 I_{12} w_4 + I_{13} w_5 + I_{14} w_6 \\
 + \\
 I_{17} w_7 + I_{18} w_8 + I_{19} w_9
 \end{array}
 &
 \begin{array}{c}
 I_8 w_1 + I_9 w_2 + I_{10} w_3 \\
 + \\
 I_{13} w_4 + I_{14} w_5 + I_{15} w_6 \\
 + \\
 I_{18} w_7 + I_{19} w_8 + I_{20} w_9
 \end{array} \\
 \begin{array}{c}
 I_{11} w_1 + I_{12} w_2 + I_{13} w_3 \\
 + \\
 I_{16} w_4 + I_{17} w_5 + I_{18} w_6 \\
 + \\
 I_{21} w_7 + I_{22} w_8 + I_{23} w_9
 \end{array}
 &
 \begin{array}{c}
 \\
 \\
 \\
 \end{array}
 &
 \begin{array}{c}
 \\
 \\
 \\
 \end{array}
 \end{array}$$

$I_1$	$I_2$	$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$	$I_9$	$I_{10}$
$I_{11}$	$I_{12}$	$I_{13}$	$I_{14}$	$I_{15}$
$I_{16}$	$I_{17}$	$I_{18}$	$I_{19}$	$I_{20}$
$I_{21}$	$I_{22}$	$I_{23}$	$I_{24}$	$I_{25}$

 $\ast$ 

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

 $=$

$I_1 w_1 + I_2 w_2 + I_3 w_3$ $+$ $I_6 w_4 + I_7 w_5 + I_8 w_6$ $+$ $I_{11} w_7 + I_{12} w_8 + I_{13} w_9$	$I_2 w_1 + I_3 w_2 + I_4 w_3$ $+$ $I_7 w_4 + I_8 w_5 + I_9 w_6$ $+$ $I_{12} w_7 + I_{13} w_8 + I_{14} w_9$	$I_3 w_1 + I_4 w_2 + I_5 w_3$ $+$ $I_8 w_4 + I_9 w_5 + I_{10} w_6$ $+$ $I_{13} w_7 + I_{14} w_8 + I_{15} w_9$
$I_6 w_1 + I_7 w_2 + I_8 w_3$ $+$ $I_{11} w_4 + I_{12} w_5 + I_{13} w_6$ $+$ $I_{16} w_7 + I_{17} w_8 + I_{18} w_9$	$I_7 w_1 + I_8 w_2 + I_9 w_3$ $+$ $I_{12} w_4 + I_{13} w_5 + I_{14} w_6$ $+$ $I_{17} w_7 + I_{18} w_8 + I_{19} w_9$	$I_8 w_1 + I_9 w_2 + I_{10} w_3$ $+$ $I_{13} w_4 + I_{14} w_5 + I_{15} w_6$ $+$ $I_{18} w_7 + I_{19} w_8 + I_{20} w_9$
$I_{11} w_1 + I_{12} w_2 + I_{13} w_3$ $+$ $I_{16} w_4 + I_{17} w_5 + I_{18} w_6$ $+$ $I_{21} w_7 + I_{22} w_8 + I_{23} w_9$	$I_{12} w_1 + I_{13} w_2 + I_{14} w_3$ $+$ $I_{17} w_4 + I_{18} w_5 + I_{19} w_6$ $+$ $I_{22} w_7 + I_{23} w_8 + I_{24} w_9$	

$I_1$	$I_2$	$I_3$	$I_4$	$I_5$
$I_6$	$I_7$	$I_8$	$I_9$	$I_{10}$
$I_{11}$	$I_{12}$	$I_{13}$	$I_{14}$	$I_{15}$
$I_{16}$	$I_{17}$	$I_{18}$	$I_{19}$	$I_{20}$
$I_{21}$	$I_{22}$	$I_{23}$	$I_{24}$	$I_{25}$

 $\ast$ 

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

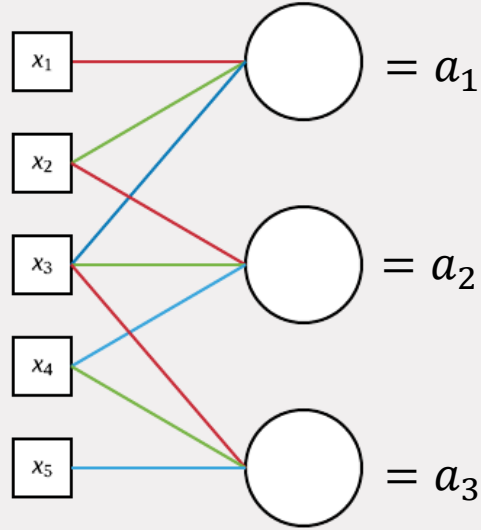
 $+ \mathbf{b} =$

$I_1 w_1 + I_2 w_2 + I_3 w_3$ + $I_6 w_4 + I_7 w_5 + I_8 w_6$ + $I_{11} w_7 + I_{12} w_8 + I_{13} w_9$	$I_2 w_1 + I_3 w_2 + I_4 w_3$ + $I_7 w_4 + I_8 w_5 + I_9 w_6$ + $I_{12} w_7 + I_{13} w_8 + I_{14} w_9$	$I_3 w_1 + I_4 w_2 + I_5 w_3$ + $I_8 w_4 + I_9 w_5 + I_{10} w_6$ + $I_{13} w_7 + I_{14} w_8 + I_{15} w_9$
$I_6 w_1 + I_7 w_2 + I_8 w_3$ + $I_{11} w_4 + I_{12} w_5 + I_{13} w_6$ + $I_{16} w_7 + I_{17} w_8 + I_{18} w_9$	$I_7 w_1 + I_8 w_2 + I_9 w_3$ + $I_{12} w_4 + I_{13} w_5 + I_{14} w_6$ + $I_{17} w_7 + I_{18} w_8 + I_{19} w_9$	$I_8 w_1 + I_9 w_2 + I_{10} w_3$ + $I_{13} w_4 + I_{14} w_5 + I_{15} w_6$ + $I_{18} w_7 + I_{19} w_8 + I_{20} w_9$
$I_{11} w_1 + I_{12} w_2 + I_{13} w_3$ + $I_{16} w_4 + I_{17} w_5 + I_{18} w_6$ + $I_{21} w_7 + I_{22} w_8 + I_{23} w_9$	$I_{12} w_1 + I_{13} w_2 + I_{14} w_3$ + $I_{17} w_4 + I_{18} w_5 + I_{19} w_6$ + $I_{22} w_7 + I_{23} w_8 + I_{24} w_9$	$I_{13} w_1 + I_{14} w_2 + I_{15} w_3$ + $I_{18} w_4 + I_{19} w_5 + I_{20} w_6$ + $I_{23} w_7 + I_{24} w_8 + I_{25} w_9$

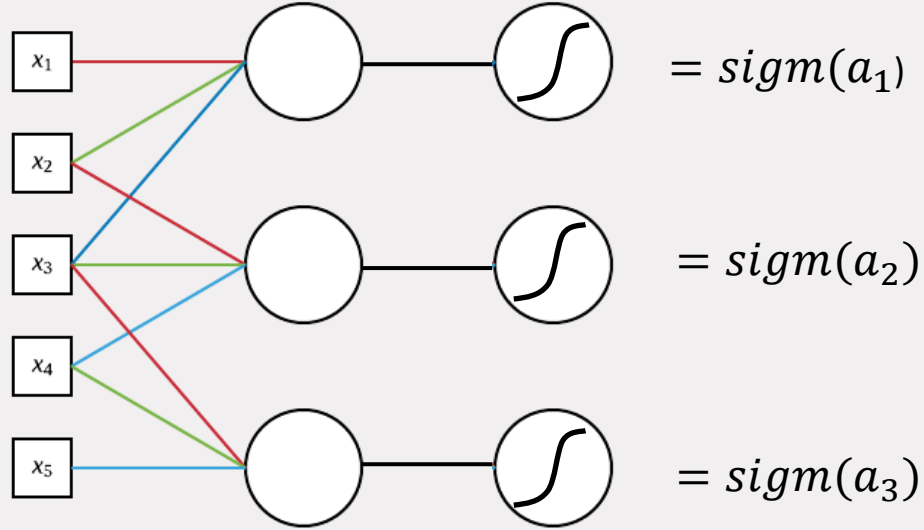
 $+ \mathbf{b}$



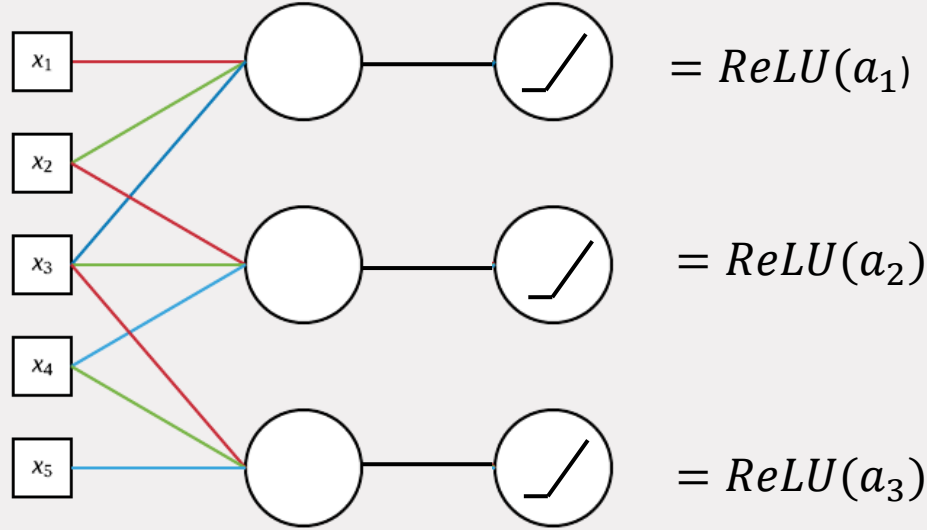
# Convolutions + non-linearity



# Convolutions + non-linearity (sigmoid)



# Convolutions + non-linearity (ReLU)



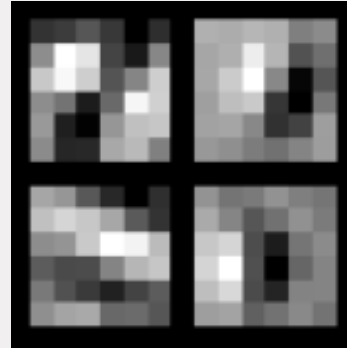
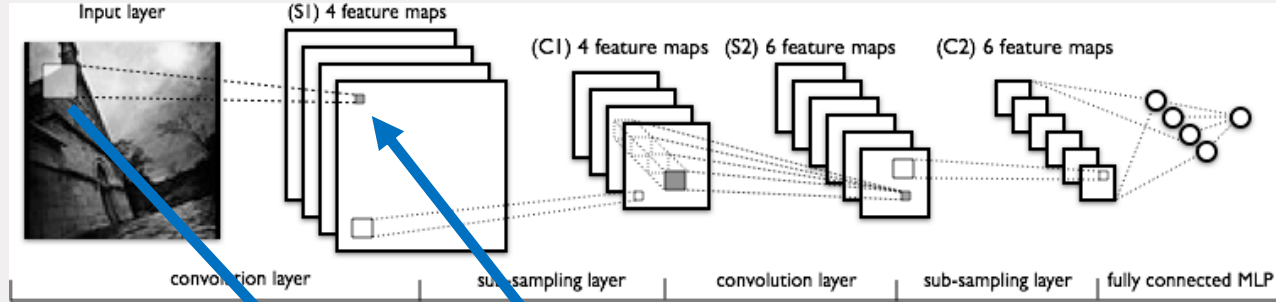
# Break!

**Quokka (Australia)**

<https://imgur.com/r/aww/GqJi0il>



# Convolutional kernels



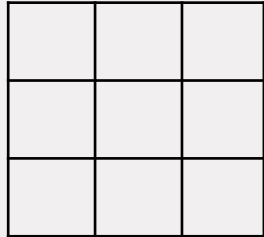
these are the weights  
(convolutional kernels)  
that produce the four  
feature maps

# Kernel size

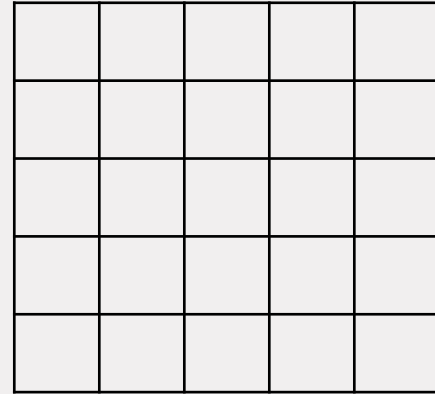
1 x 1



3 x 3

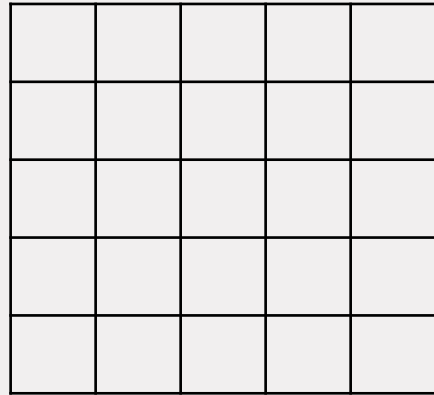


5 x 5



- More weights → more information
- More computations / memory
- Receptive field

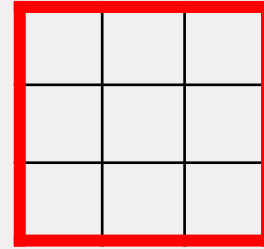
# Receptive field



5 x 5

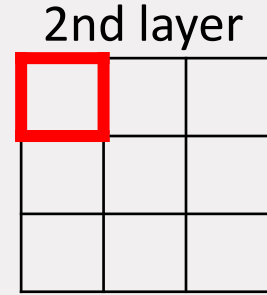
25 weights

Same receptive  
field



3 x 3

9 weights

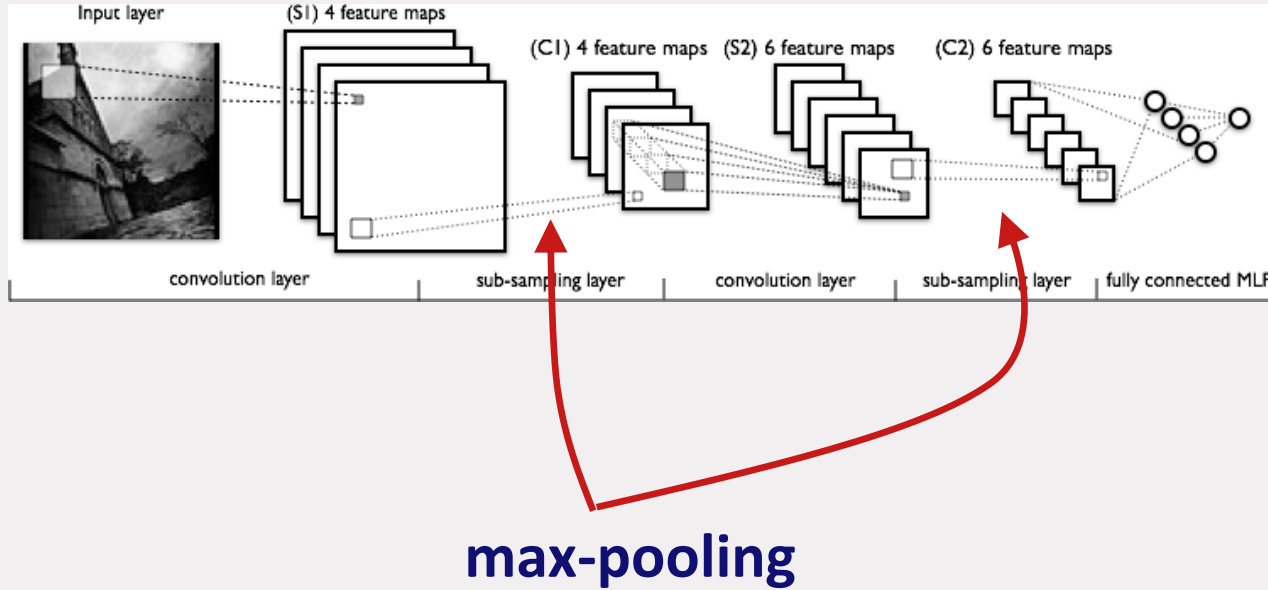


3 x 3

9 weights

= 18 weights

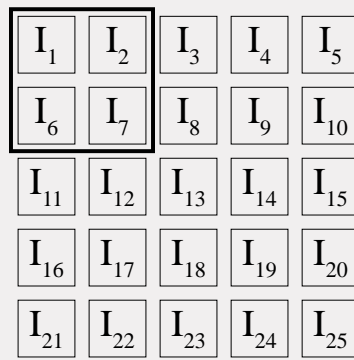
# Max-pooling



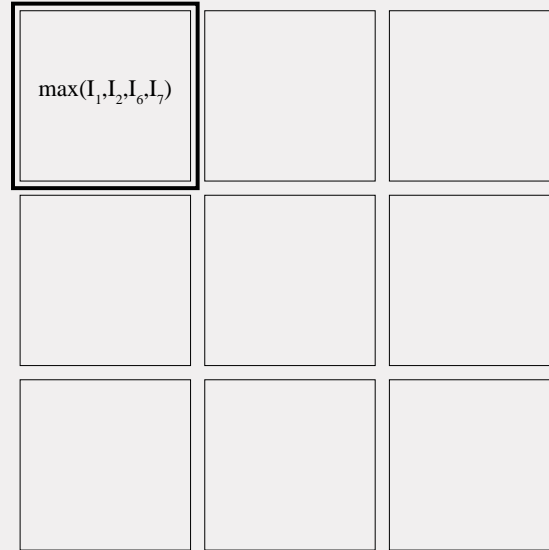


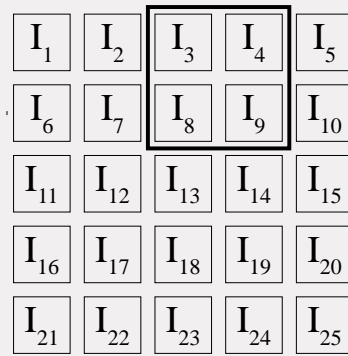
# Max-pooling

- Reduce size of feature space
- Maximum of features
- Typical kernel size =  $2 \times 2$

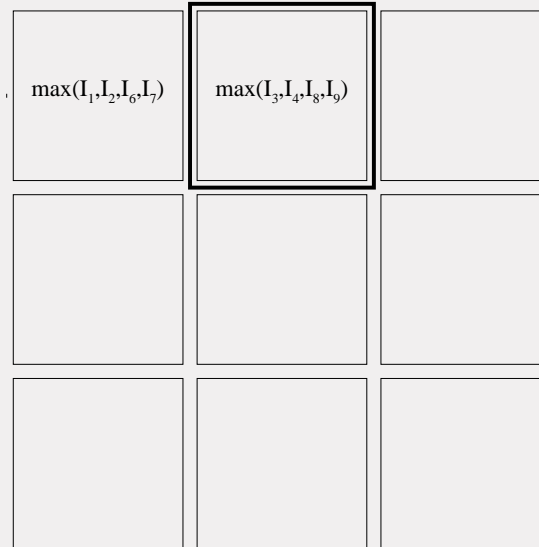


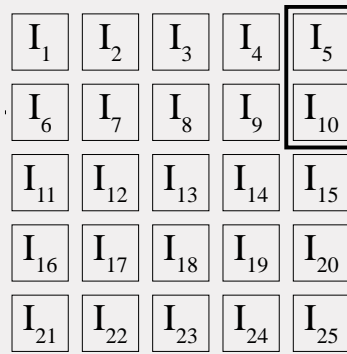
2×2 max pooling



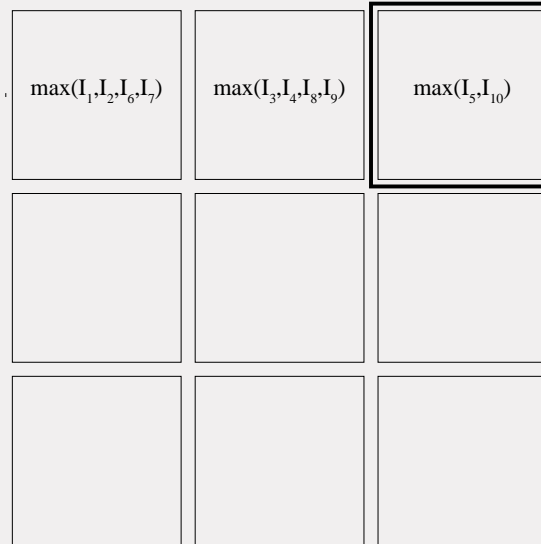


2×2 max pooling

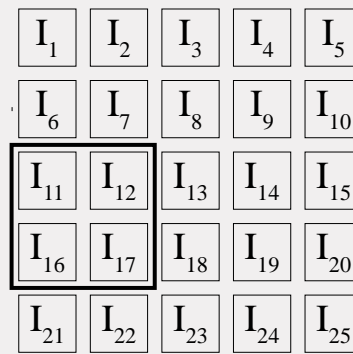




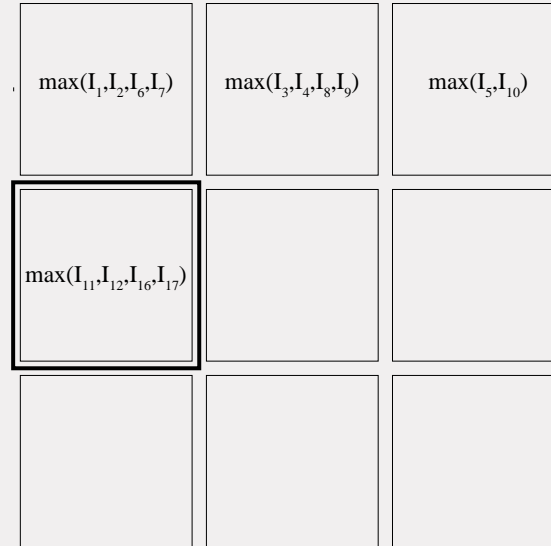
2×2 max pooling

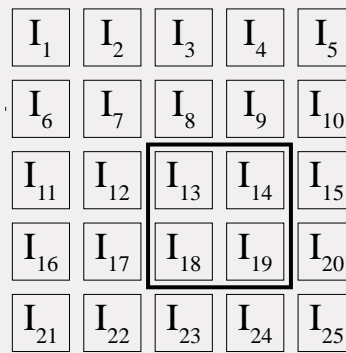


In this example:  
no zero padding

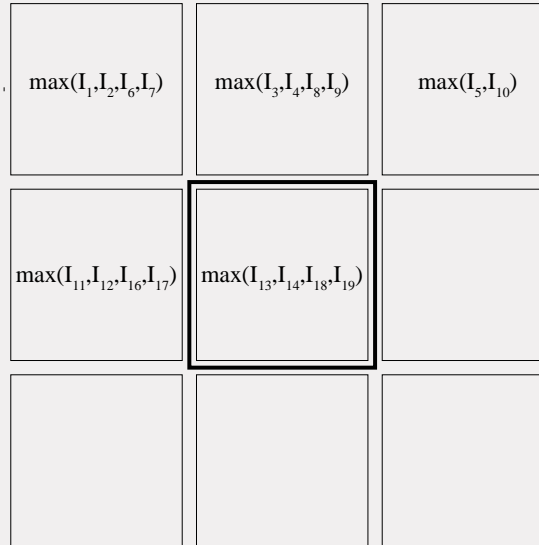


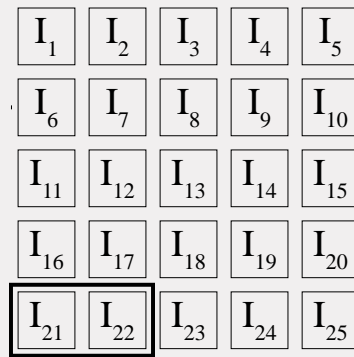
2×2 max pooling



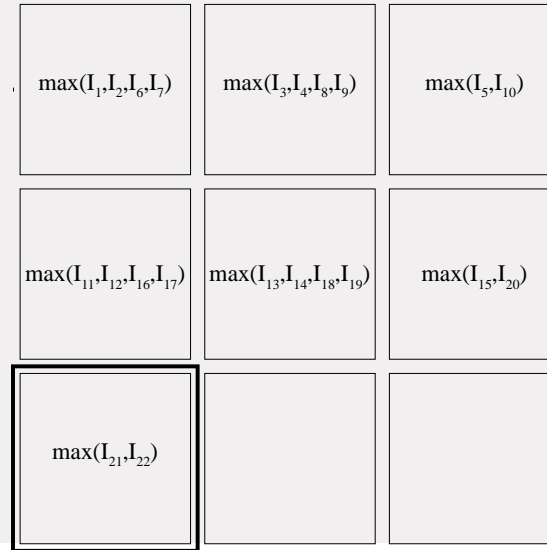


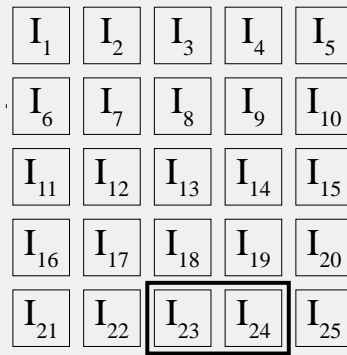
2×2 max pooling



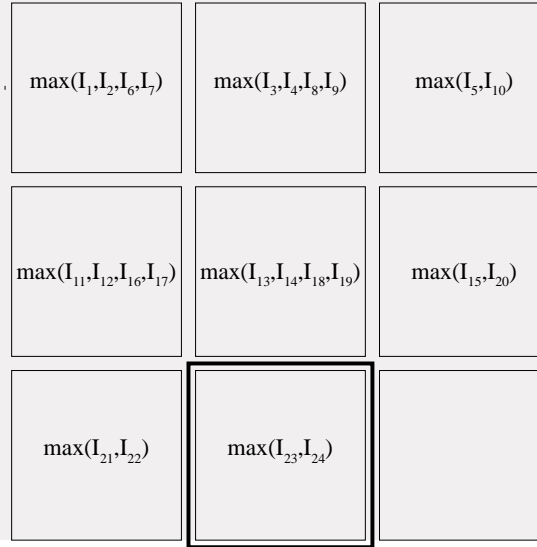


2×2 max pooling

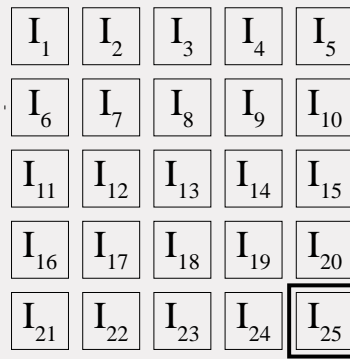




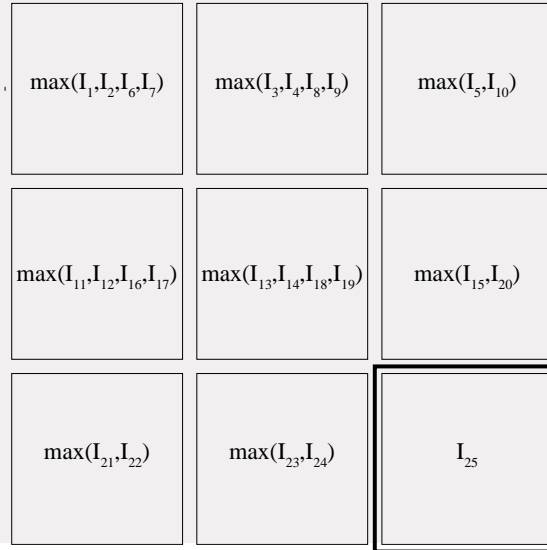
2×2 max pooling







2×2 max pooling



# Benefits of max-pooling

- “Quickly” reduces the number size of the feature maps
- Introduces translation invariance (slightly translated version of the input image will result in the same output)

Alternative: Average Pooling

# Let try it out!

<http://scs.ryerson.ca/~aharley/vis/conv/flat.html>

## An Interactive Node-Link Visualization of Convolutional Neural Networks

Adam W. Harley<sup>(✉)</sup>

Department of Computer Science, Ryerson University,  
Toronto, ON M5B 2K3, Canada  
[aharley@scs.ryerson.ca](mailto:aharley@scs.ryerson.ca)

# Training of Convolutional Neural Networks

- Similar to training of 'fully connected' neural networks
- Choose some (random) initial values for network weights
- Optimize networks weights with respect to a loss function that describes difference between network output and label/annotation
- Update networks weights iteratively

Through a process called 'backpropagation'. A good explanation can be found here: <https://www.youtube.com/watch?v=i94OvYb6noo> (5:10 - 28:00, not exam material)

- Keep track of model performance on train & validation set

# Summary

- Concept of **convolutions** in a neural network
- Why can we use a **convolutional approach** for (medical) **images**
- Convolutions enable development of **deep** (and large) **neural networks**
- **Max-pooling layer** in a convolutional neural network
- **Kernel size**
- **Receptive field**

