Deep Learning

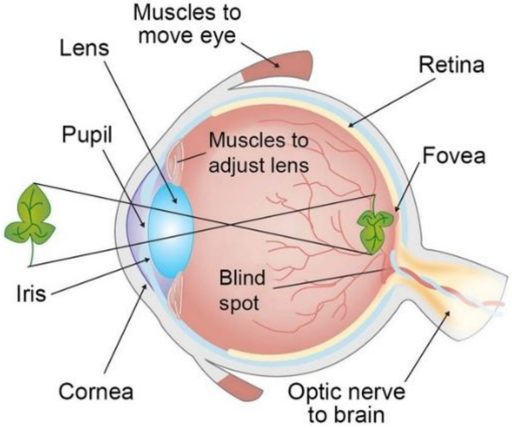
CNNs

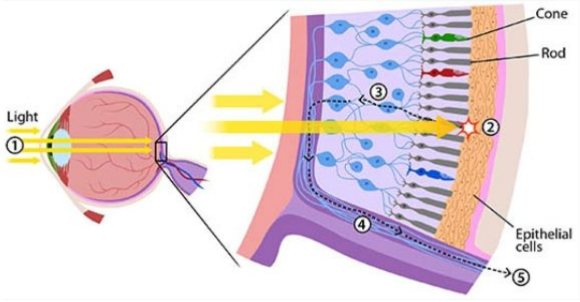
Bryan Cardenas
Thomas van Osch

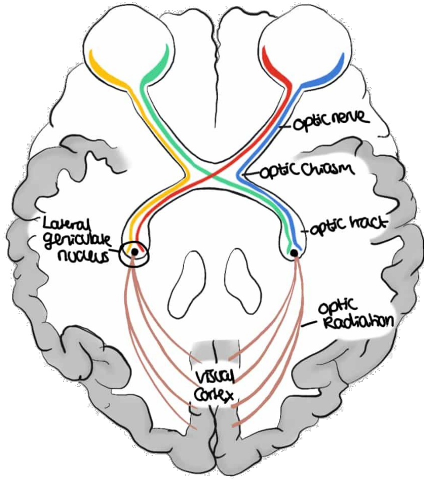
How do Computers see?

How do Humans see?

Mammalian Visual System



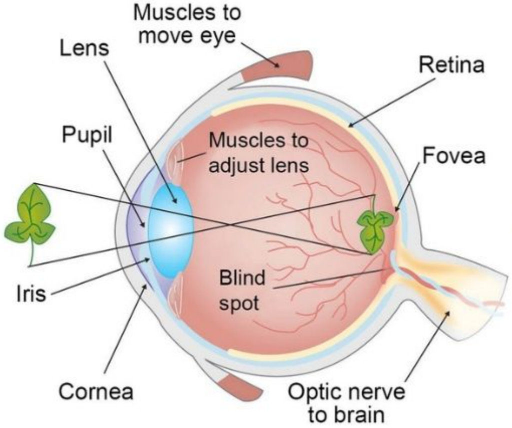


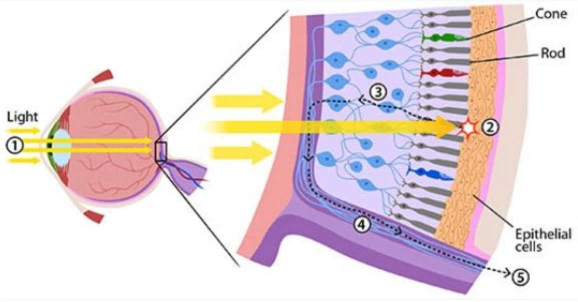


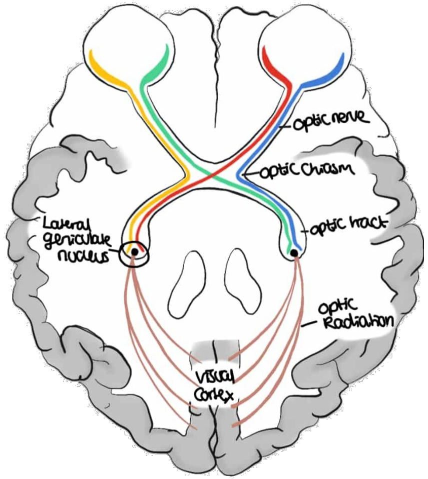
Mammalian Visual System

Cones and rods picks up photons and propagate the signal to the back of our brain

Along the way, the signal gets processed in stages





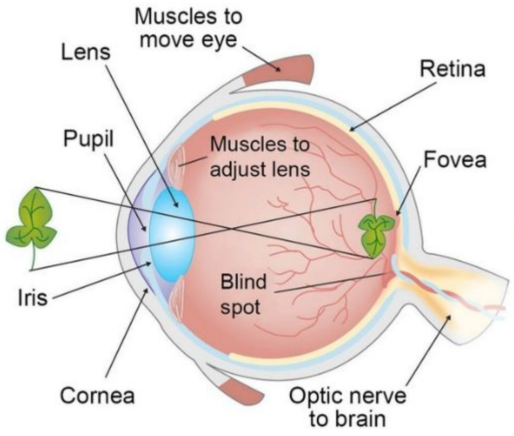


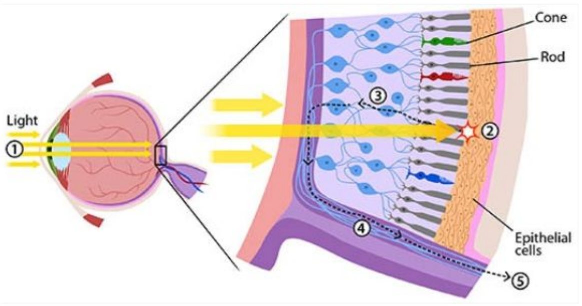
Mammalian Visual System

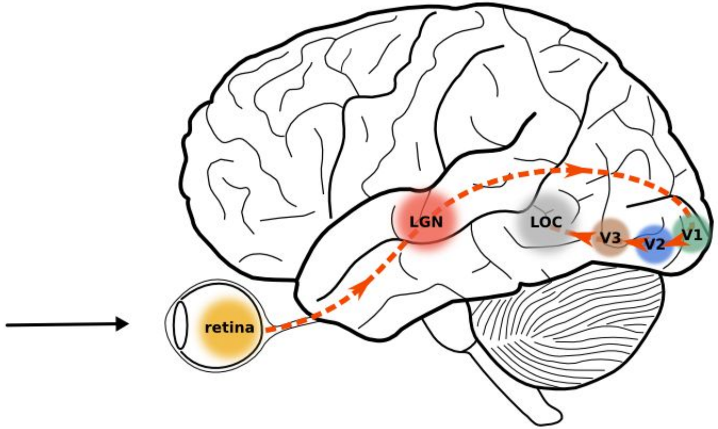
Cones and rods picks up photons and propagate the signal to the back of our brain

Along the way, the signal gets processed in stages

The extracted information gets aggregated and formed into an image





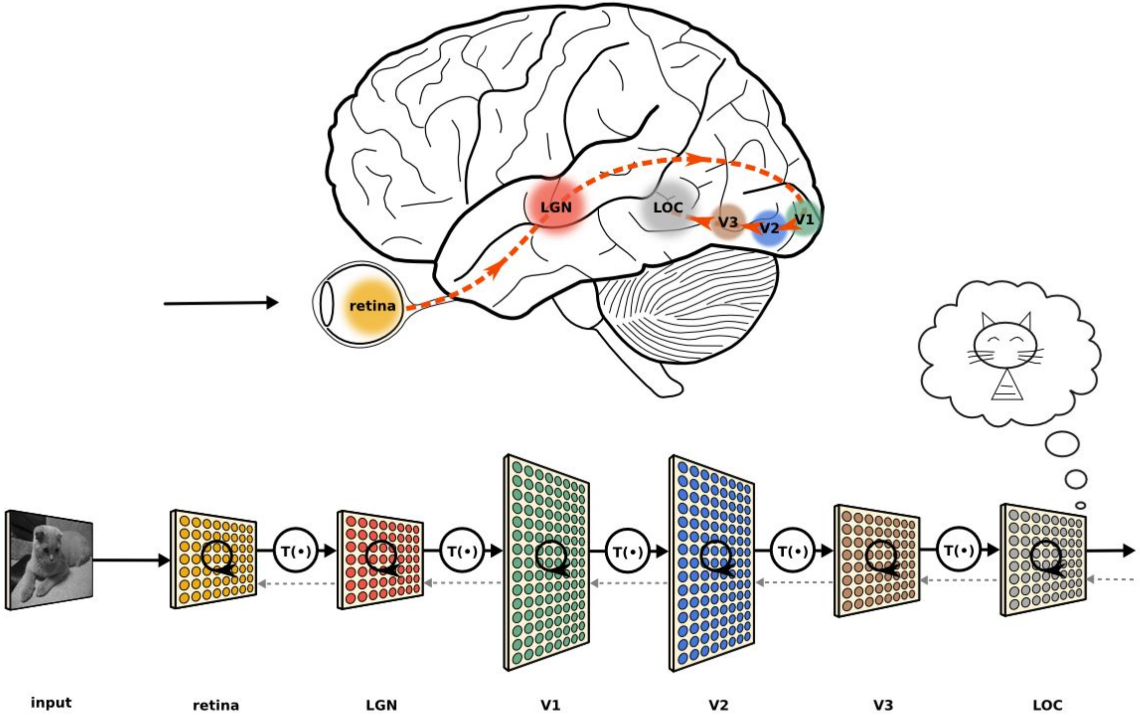


Mammalian Visual System

Cones and rods picks up photons and propagate the signal to the back of our brain

Along the way, the signal gets processed in stages

The extracted information gets aggregated and formed into an image



SURF

11/01/2023

Convolutional Neural Networks

High Performance Machine Learning Group

Image Representation

01. Is a matrix or grid of intensity values

02. integers [0, 255] or float points [0,1]

03. Each element in the matrix is a **pixel**

04. Can have 1 greyscale channel or multiple colour **channels: RGB**

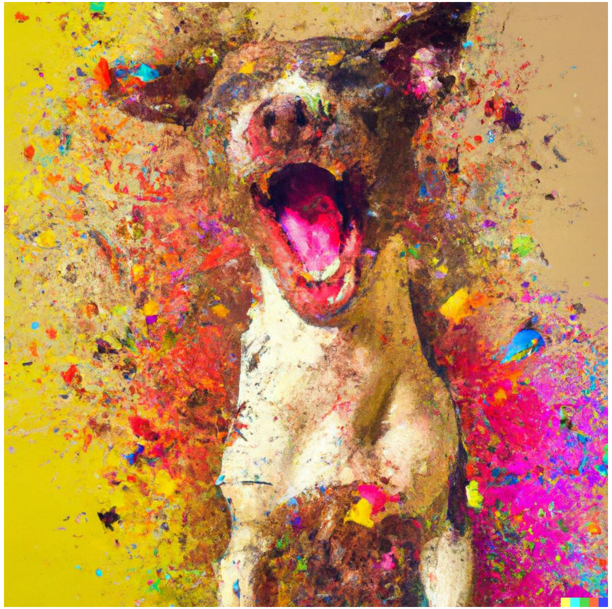


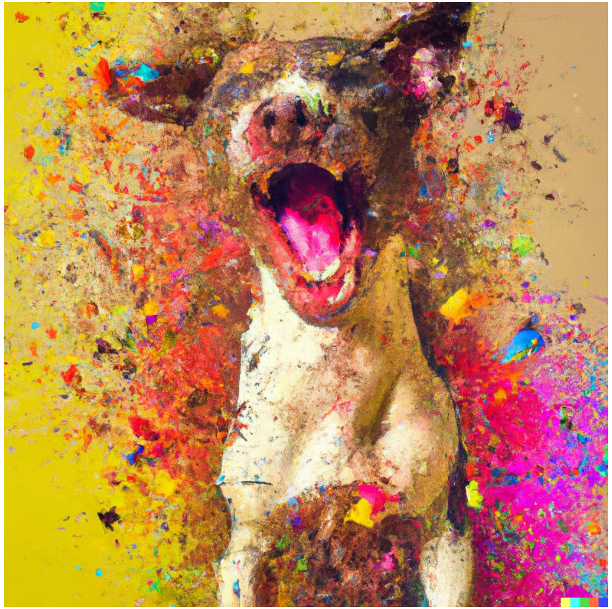
Image Representation

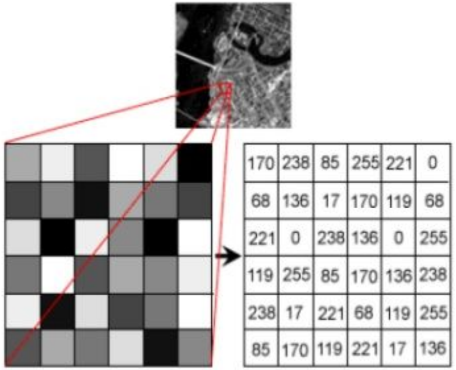
01. Is a matrix or grid of intensity values

02. integers [0, 255] or float points [0,1]

03. Each element in the matrix is a **pixel**

04. Can have 1 greyscale channel or multiple colour **channels: RGB**





170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

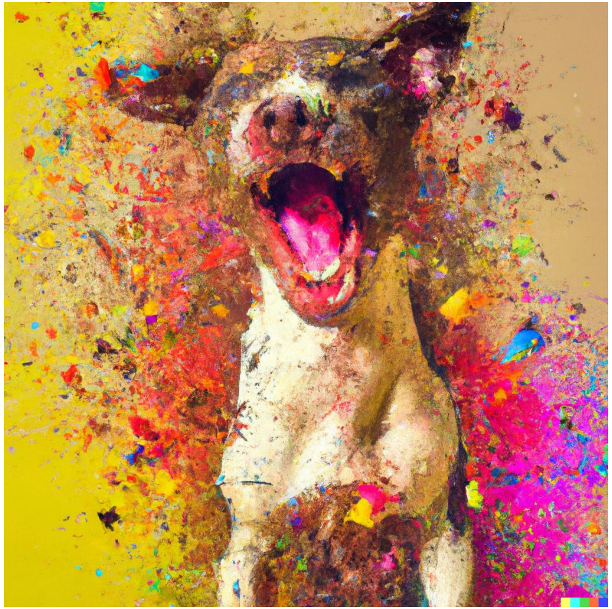
Image Representation

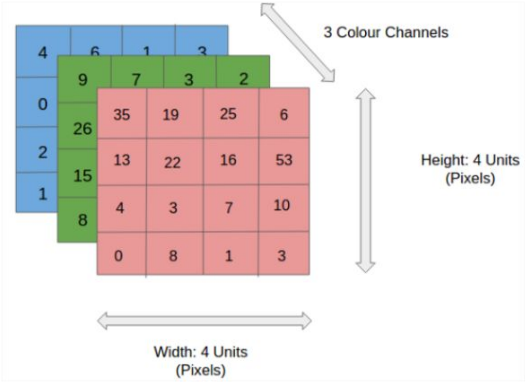
01. Is a matrix or grid of intensity values

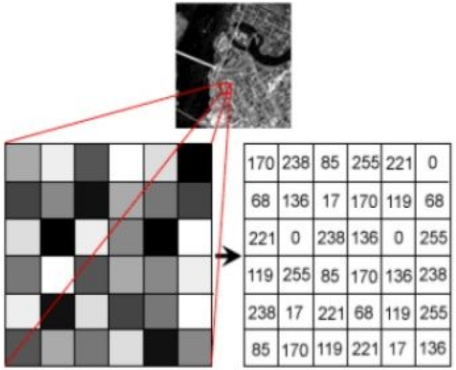
02. integers [0, 255] or float points [0,1]

03. Each element in the matrix is a **pixel**

04. Can have 1 greyscale channel or multiple colour **channels: RGB**

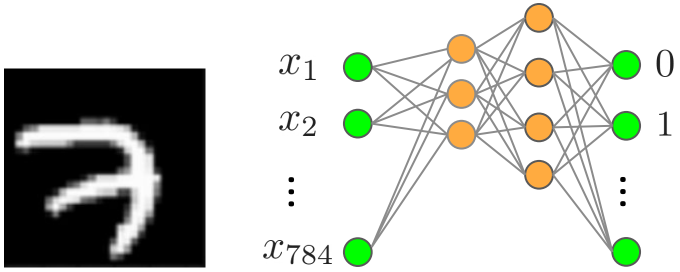






SURF

Correlated structures

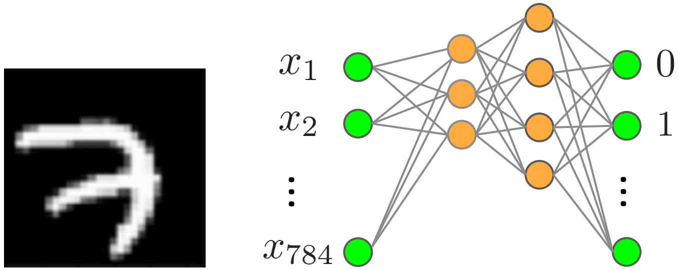


Images are **correlated spatially**

Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

Correlated structures

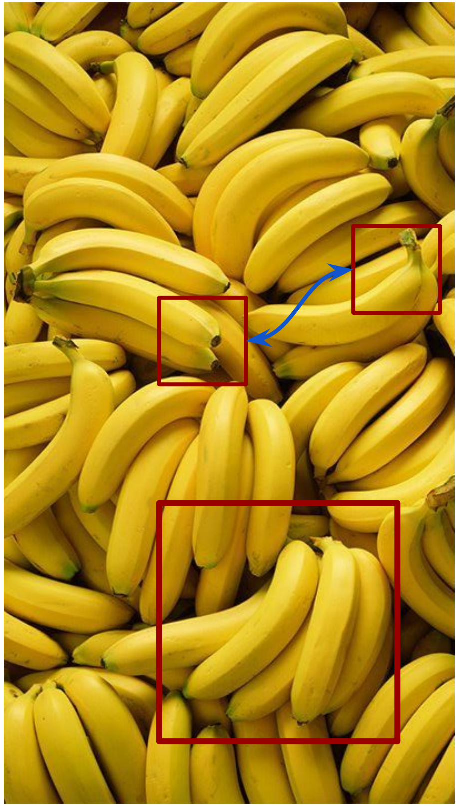


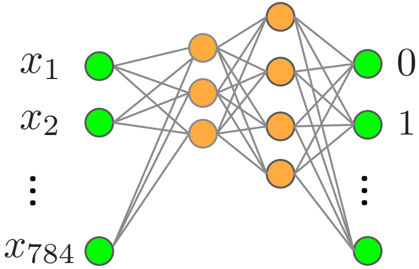
Images are **correlated spatially**

Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

Correlated structures



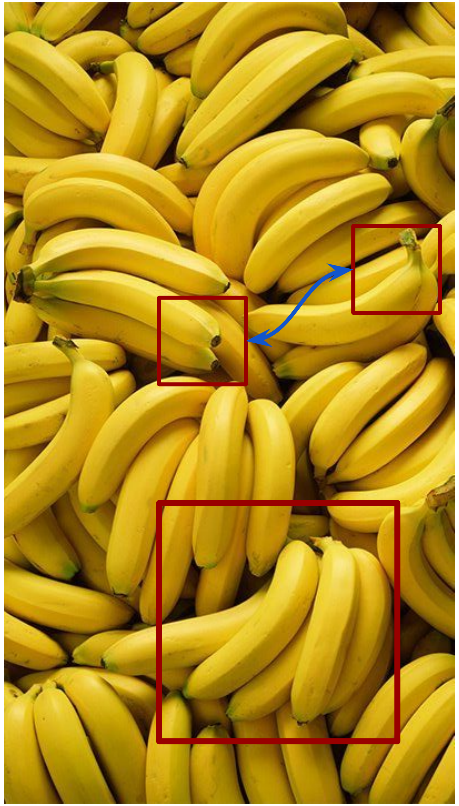


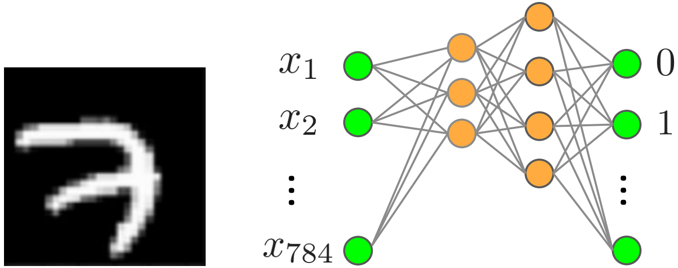
Images are **correlated spatially**

Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

Correlated structures



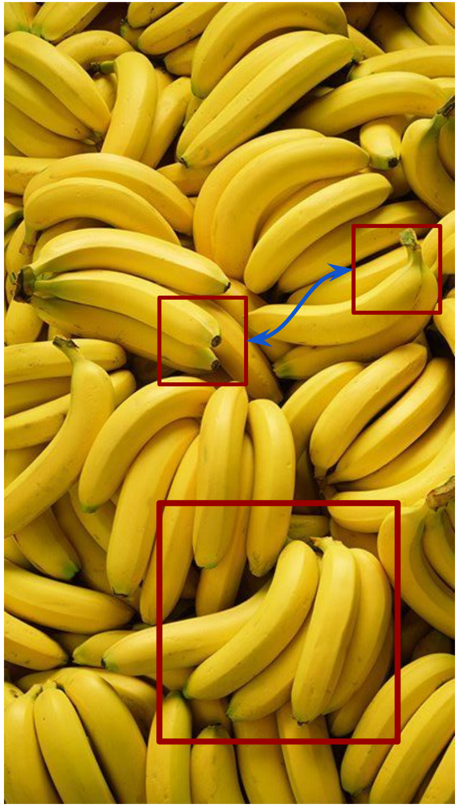


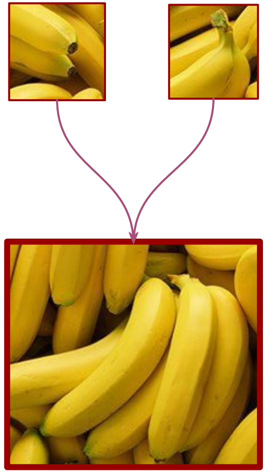
Images are **correlated spatially**

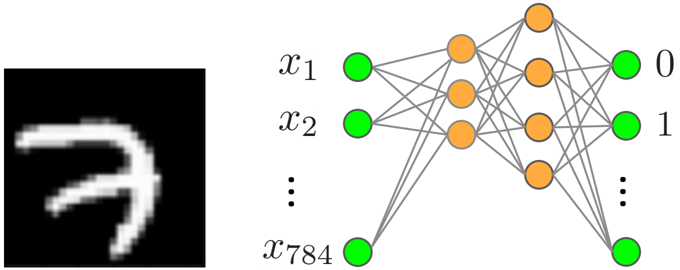
Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

Correlated structures





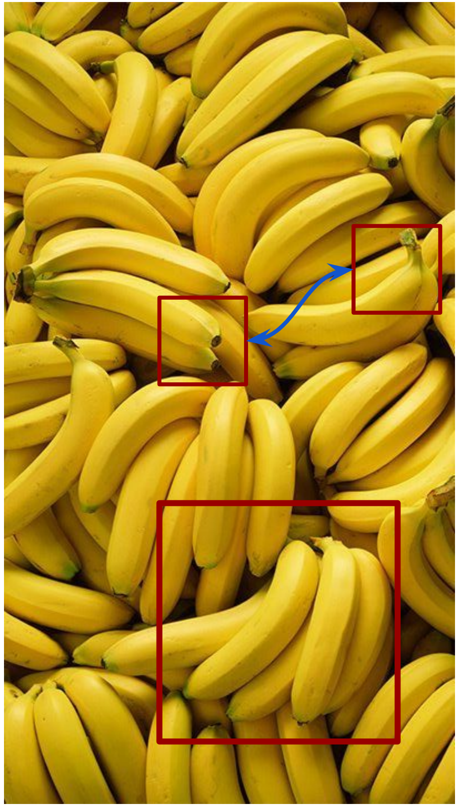


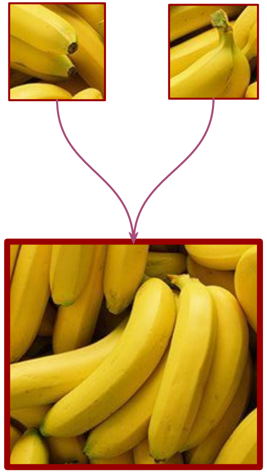
Images are **correlated spatially**

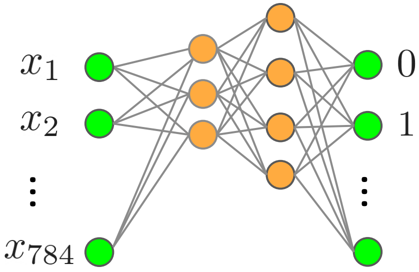
Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

Correlated structures







Detect the features across an image and aggregate them

Form complete features higher up in the **hierarchy**

Images are **correlated spatially**

Strong correlation between neighbouring pixels

Shifting an object will not alter its essence

ImageNet Challenge

Computer Vision Benchmark	1.4M Images, 1000 classes
Image Classification	Difficult until 2012

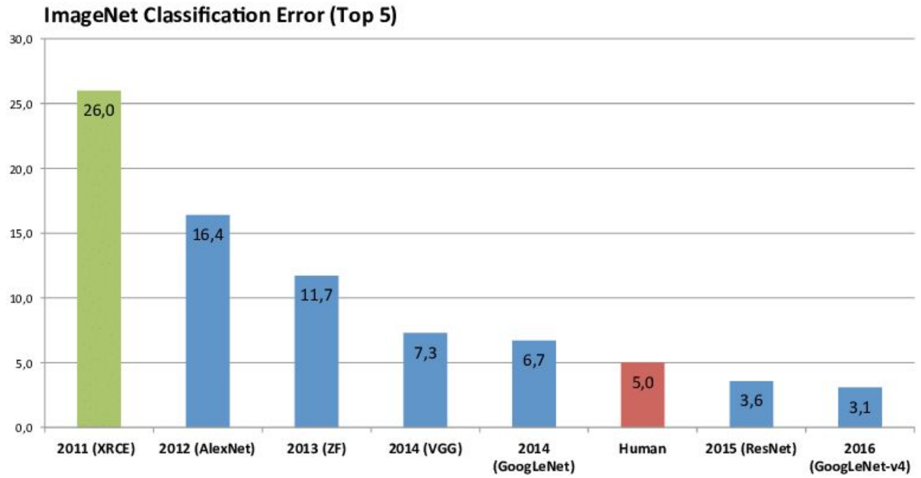
ImageNet Challenge

Computer Vision Benchmark

1.4M Images, 1000 classes

Image Classification

Difficult until 2012



Convolutions

Kernel (filter): small matrix that we use to convolve an image

Convolution:

An operation that “blends” one function with another.

Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

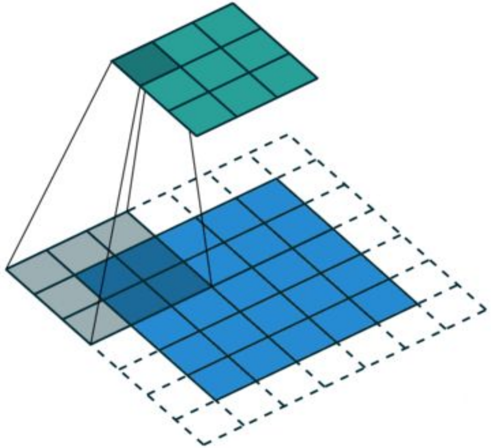
Convolutions

Kernel (filter): small matrix that we use to convolve an image

Convolution:

An operation that “blends” one function with another.

A filter applies a **convolution** operation on an image



Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

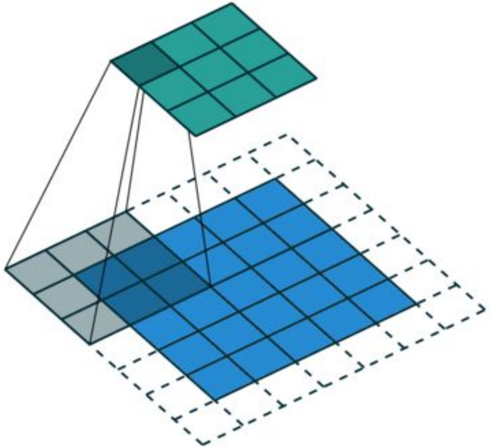
Convolutions

Kernel (filter): small matrix that we use to convolve an image

Convolution:

An operation that “blends” one function with another.

A filter applies a **convolution** operation on an image



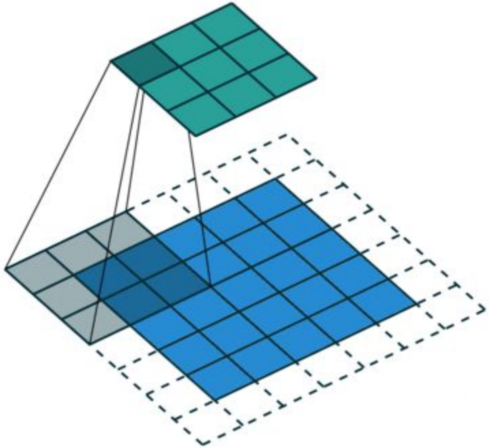
Convolutions

Kernel (filter): small matrix that we use to convolve an image

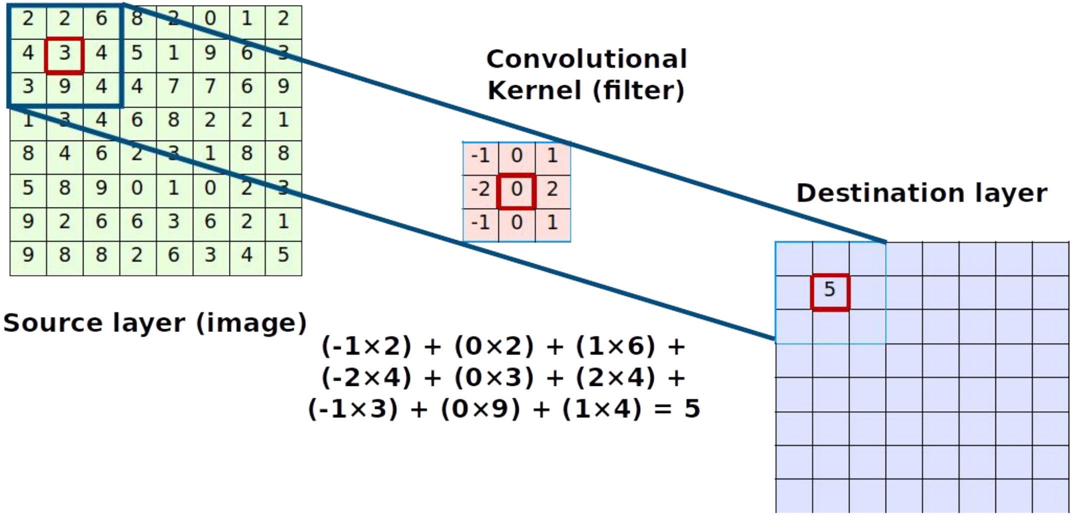
Convolution:

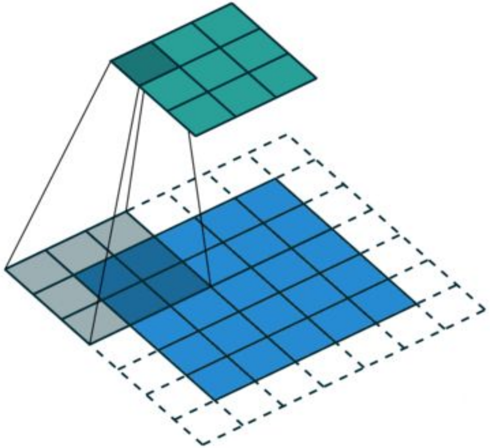
An operation that “blends” one function with another.

A filter applies a **convolution** operation on an image

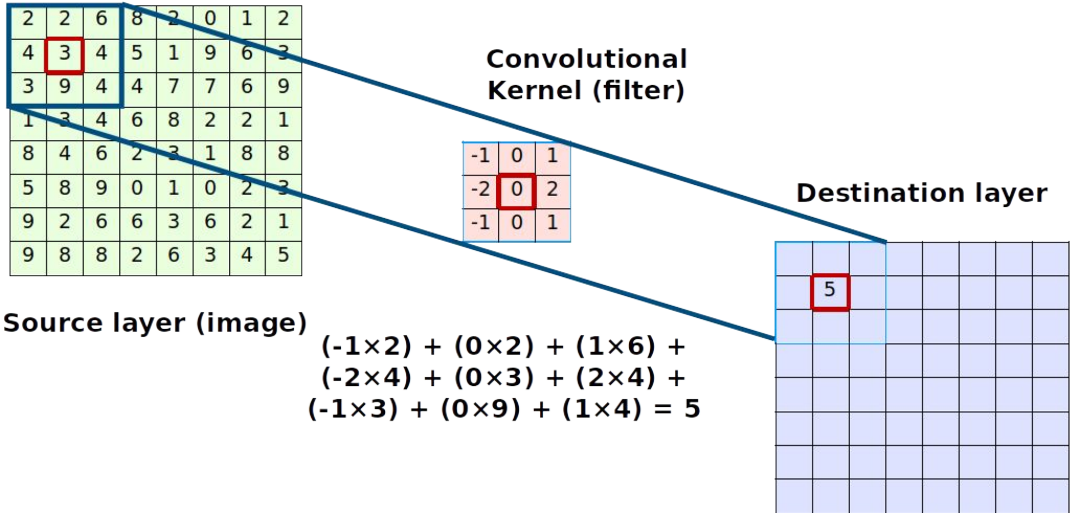


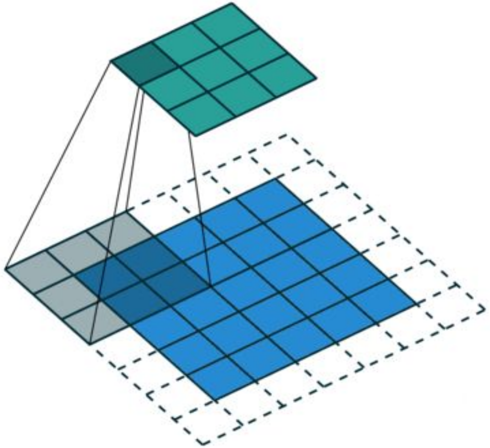
Convolutions Compute





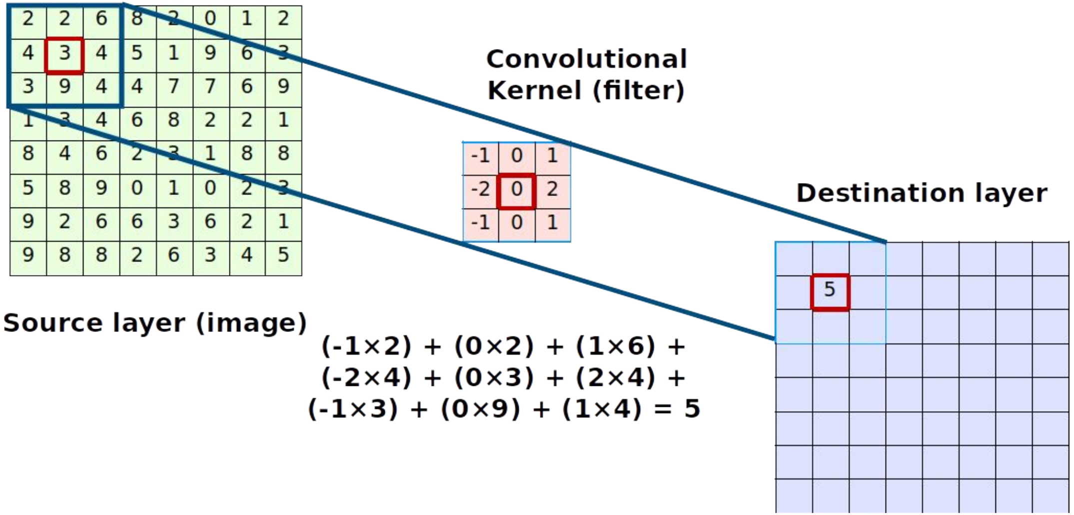
Convolutions Compute

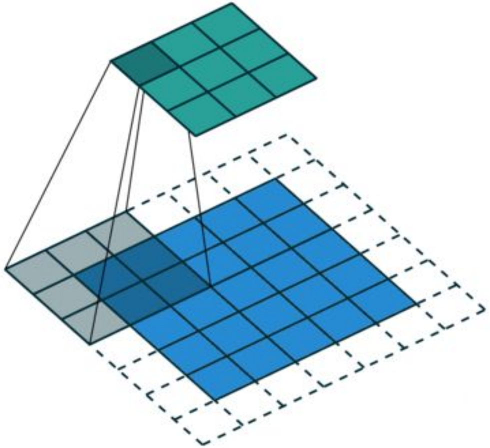




The kernel is **shifted** across the image and produces a point value
The step size in which it shifts is the **stride**
The output is always smaller, we use **padding** to preserve dimensions

Convolutions Compute



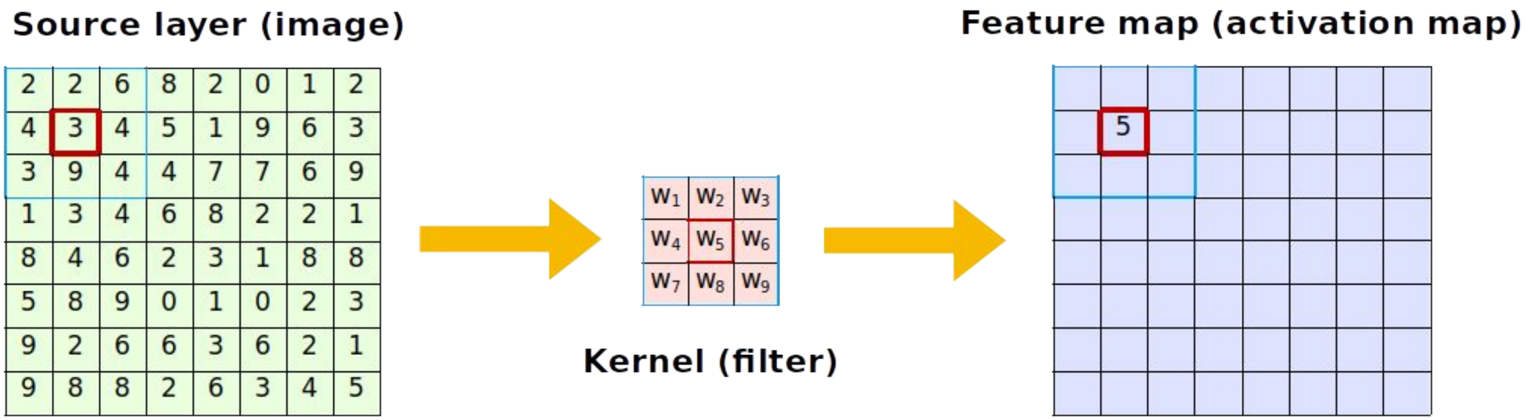


The kernel is **shifted** across the image and produces a point value
The step size in which it shifts is the **stride**
The output is always smaller, we use **padding** to preserve dimensions

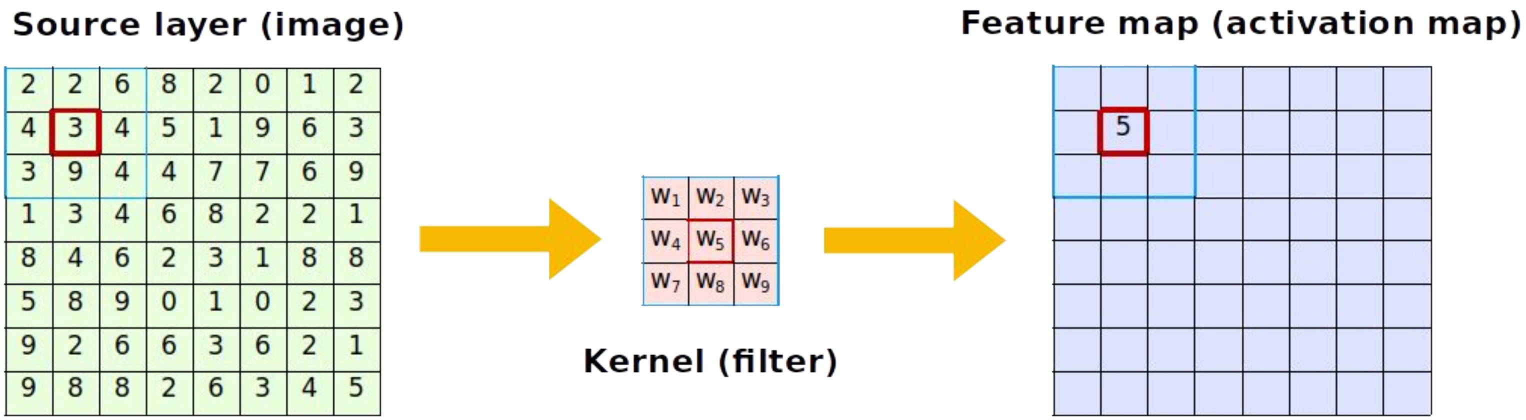
The image shrinks according to

Output_size = inputSize -(KernelSize - 1)

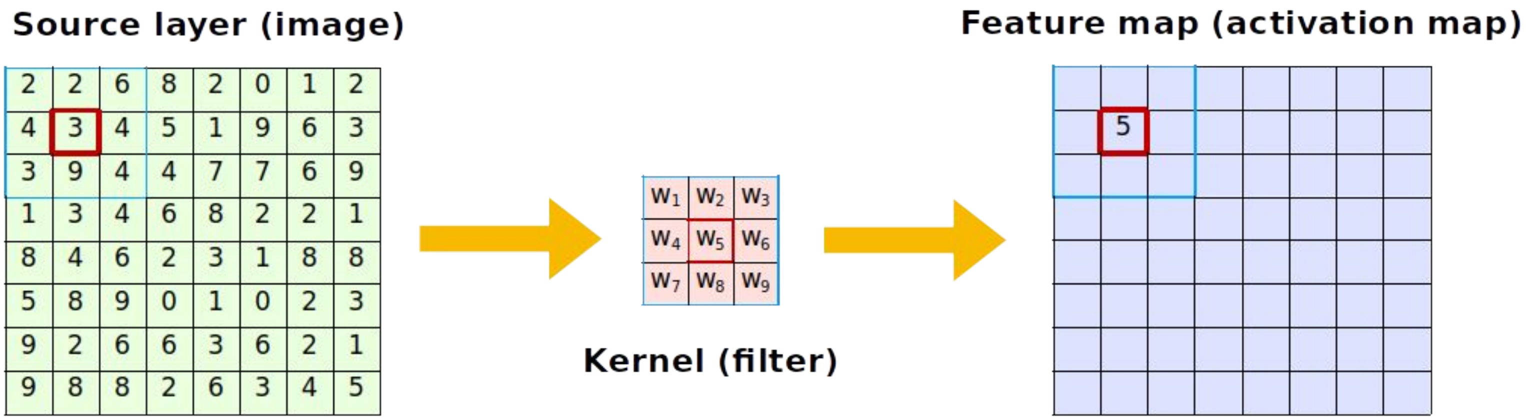
Convolutions



Convolutions



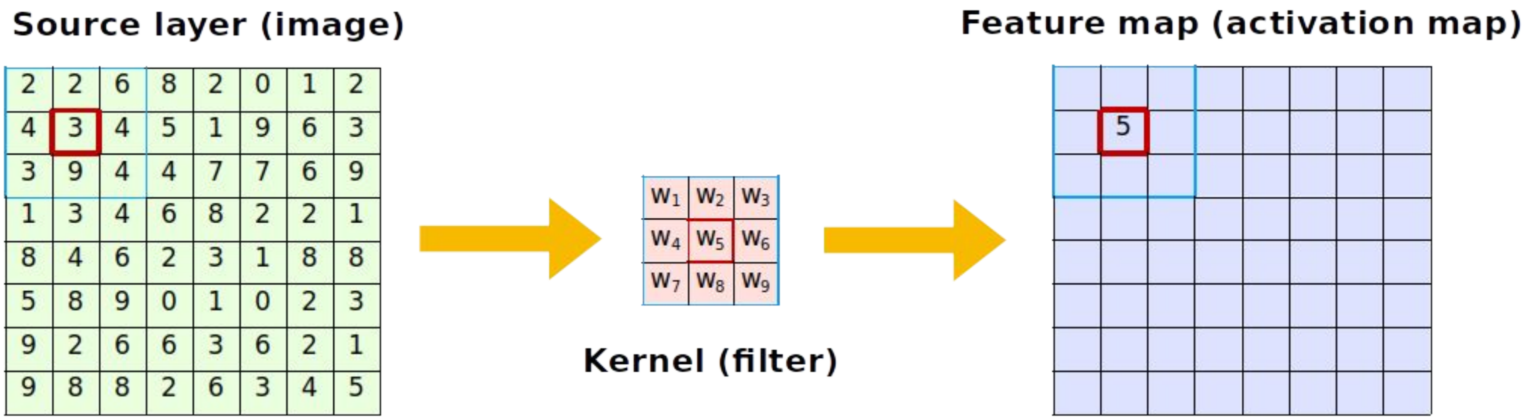
Convolutions



How do we know which kernels to use?

Kernels are learnt and initialized randomly
during training the CNN learns spatial features

Convolutions



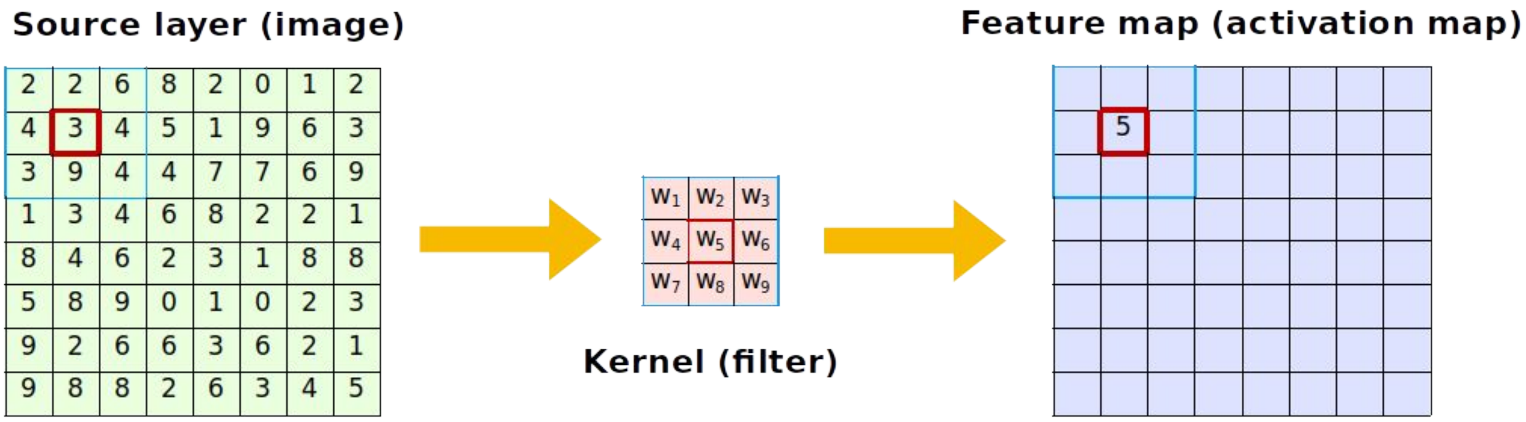
How do we know which kernels to use?

Kernels are learnt and initialized randomly
during training the CNN learns spatial features

Fully connected

$$\sum_{i \in \mathrm{image}}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

Convolutions



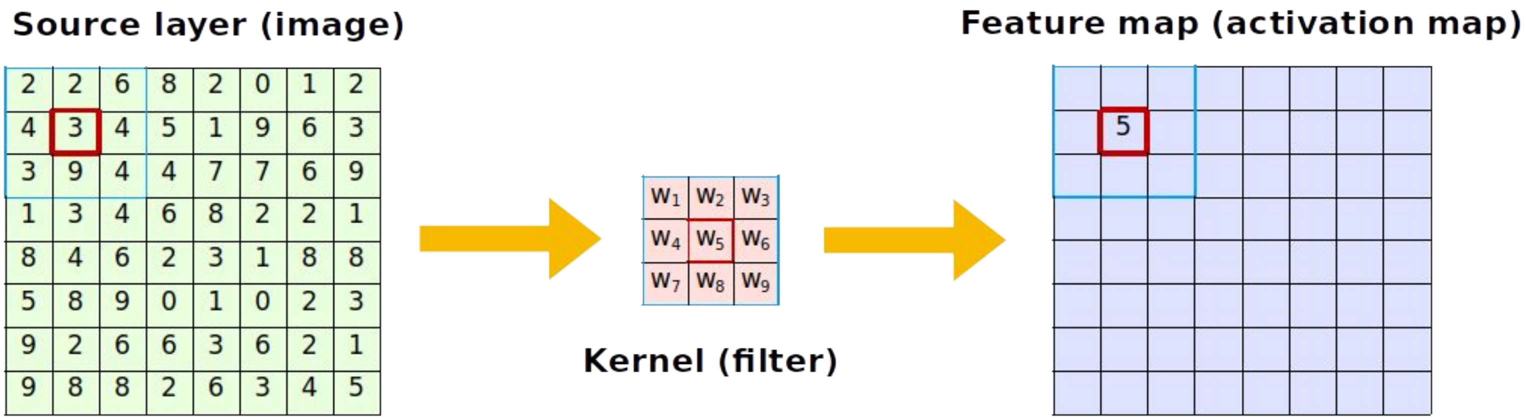
How do we know which kernels to use?

Kernels are learnt and initialized randomly
during training the CNN learns spatial features

Fully connected

$$\sum_{i \in \text{image}}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

Convolutions



How do we know which kernels to use?

Kernels are learnt and initialized randomly
during training the CNN learns spatial features

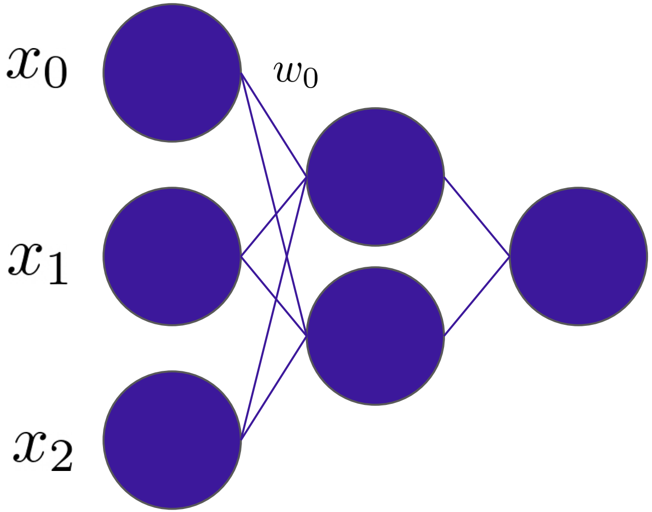
Fully connected

$$\sum_{i \in \text{image}}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

Locally connected shared weights

$$\sum_{i \in 3 \times 3}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

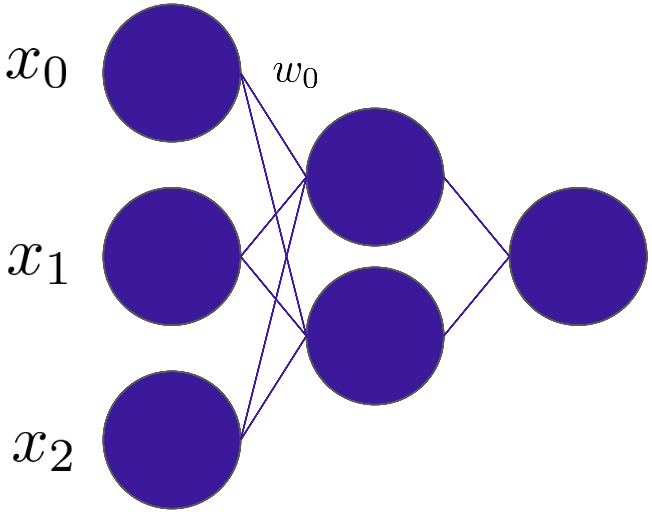
Anatomy of an Convolutional Neural Network



CNNs actually look a lot like normal Dense Neural Networks

Interpretation: Every node is a kernel

Anatomy of an Convolutional Neural Network



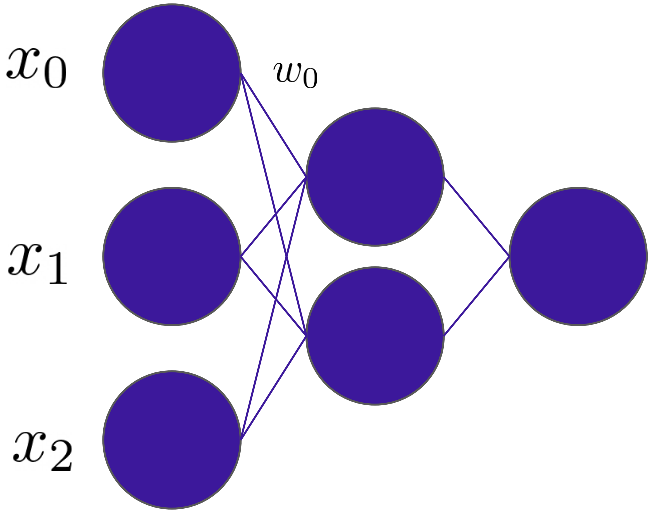
See the nodes as moving across the image!

We say these are **shared weights**

CNNs actually look a lot like normal Dense Neural Networks

Interpretation: Every node is a kernel

Anatomy of an Convolutional Neural Network



Fully connected

$$\sum_{i \in \text{image}}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

$\rightarrow$

Locally connected shared weights

$$\sum_{i \in 3 \times 3}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

See the nodes as moving across the image!

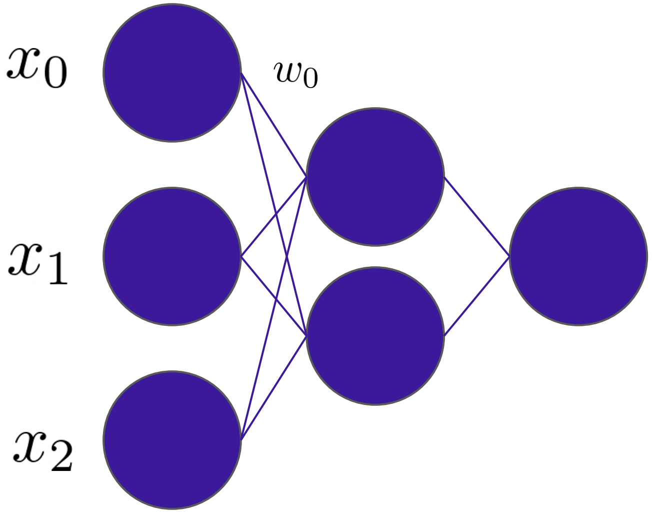
We say these are **shared weights**

CNNs actually look a lot like normal Dense Neural Networks

Interpretation: Every node is a kernel

SURF

Anatomy of an Convolutional Neural Network



Fully connected

$$\sum_{i \in \text{image}}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

→

Locally connected shared weights

$$\sum_{i \in 3 \times 3}^{W \times H \times C} \mathbf{x}_i \mathbf{w}_i$$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

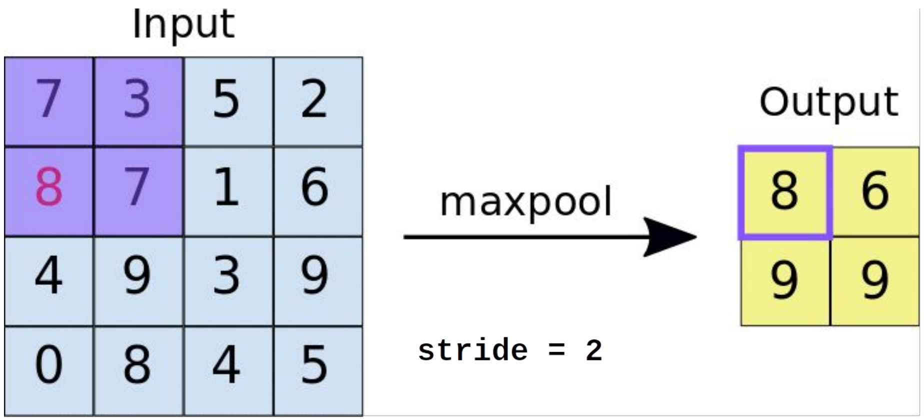
See the nodes as moving across the image!

We say these are **shared weights**

CNNs actually look a lot like normal Dense Neural Networks

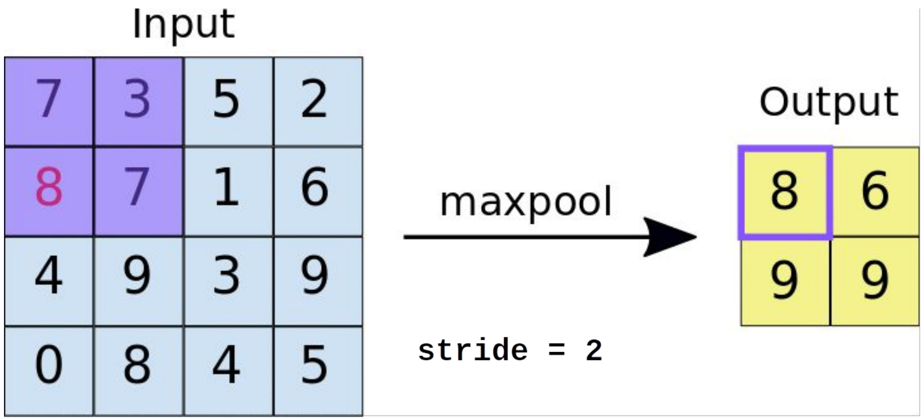
Interpretation: Every node is a kernel

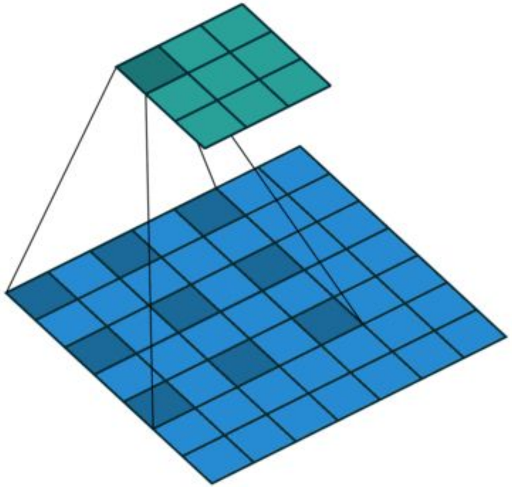
Pooling and dilated convolutions



We could discard information gradually by max pooling
Keep the strongest signal, works better than average pooling
Nowadays, we use a bigger stride for dimensionality reduction

Pooling and dilated convolutions





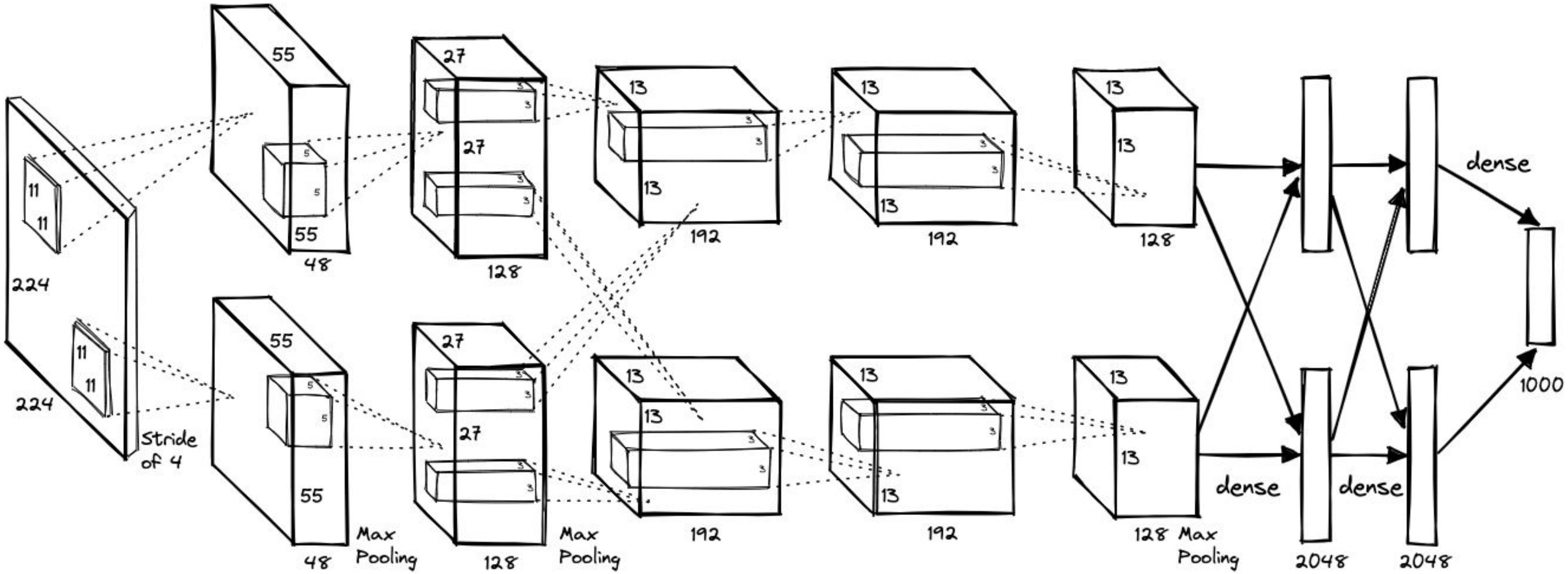
We could discard information gradually by max pooling
Keep the strongest signal, works better than average pooling
Nowadays, we use a bigger stride for dimensionality reduction

Dilated convolutions can be used if you expect your images to have information which are spatially far from each other

CNN Cases

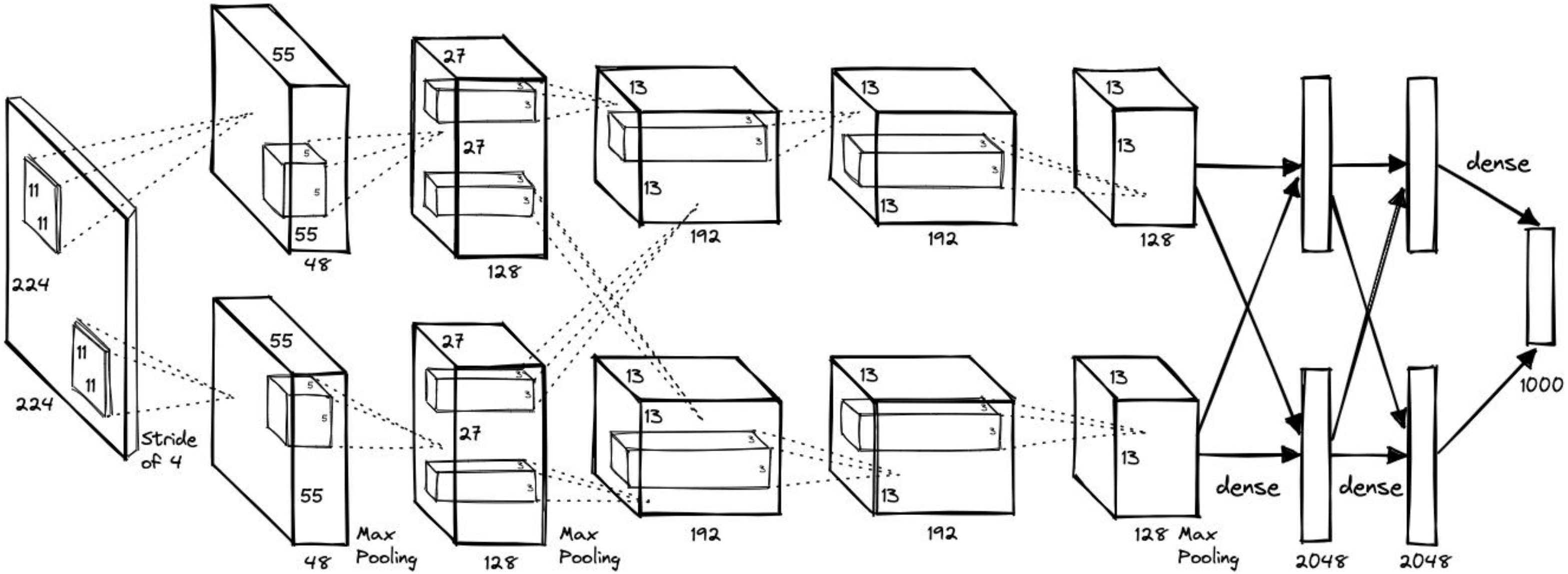
High Performance Machine Learning Group

AlexNet (2012)



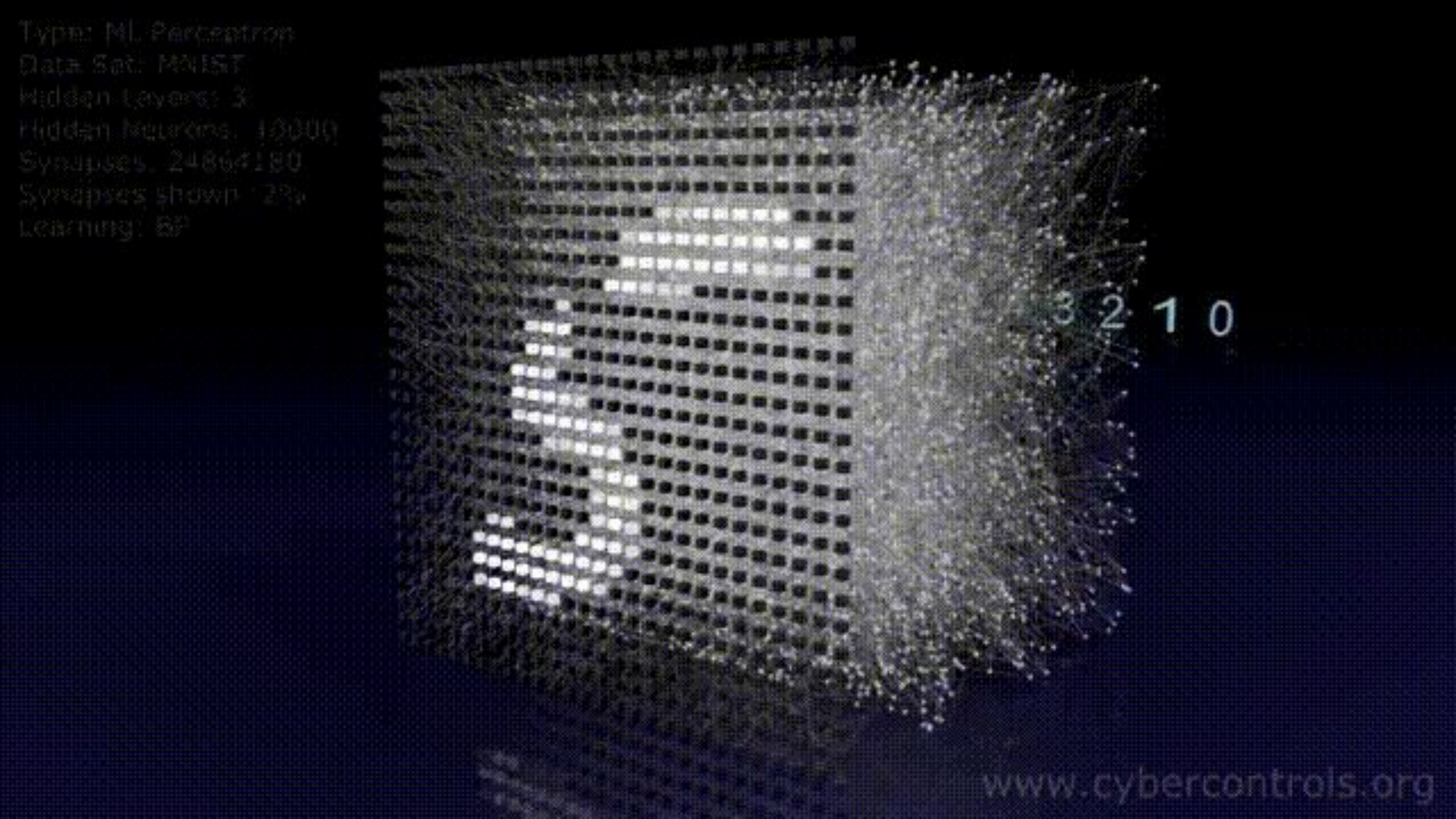
SURF

AlexNet (2012)

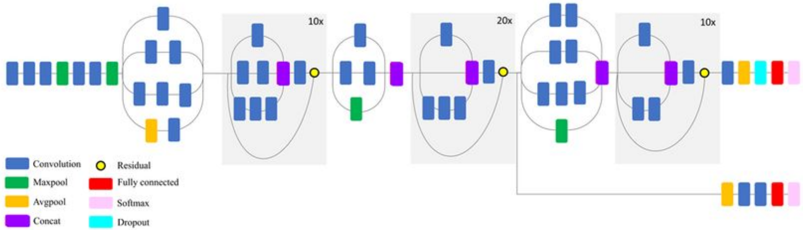


Images become smaller and number of filters increases as we go deeper

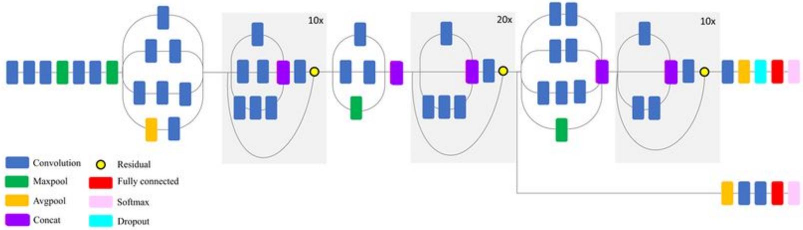
8 layers with dropout and ReLU activations trained for 6 days on 2 GPUs

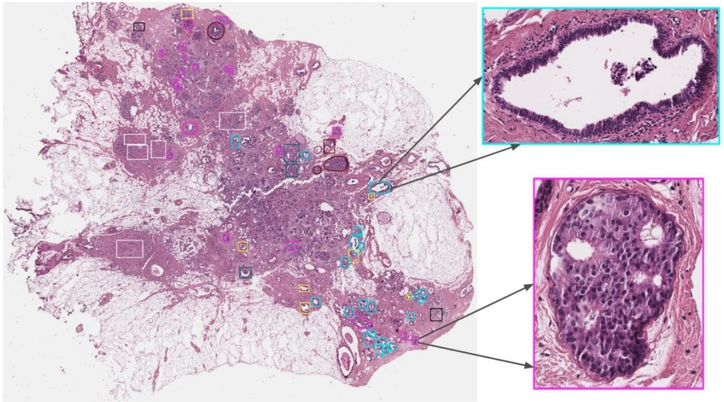

Type: ML Perceptron
Data Set: MNIST
Hidden Layers: 3
Hidden Neurons: 10000
Synapses: 24864180
Learning: BP
3 2 1 0
www.cybercontrols.org

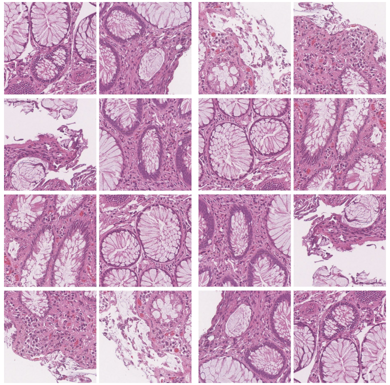
Transfer Learning



Transfer Learning

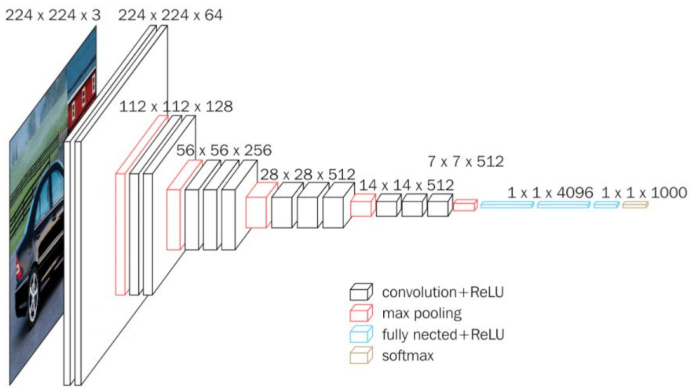






Thank You

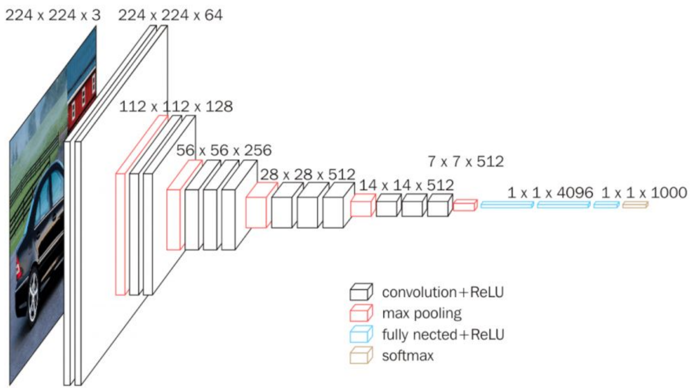
VGG16 and ResNet (2014, 2015)

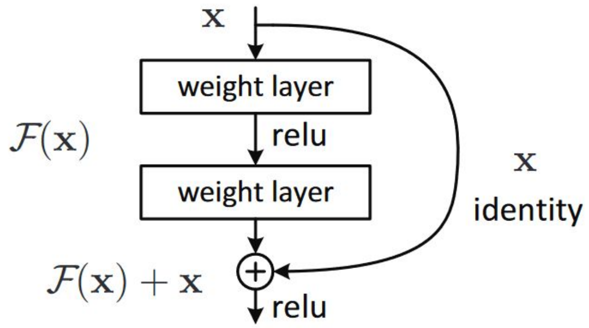


VGG16: construct large and deep models (120M params)

Vanishing gradient problem

VGG16 and ResNet (2014, 2015)

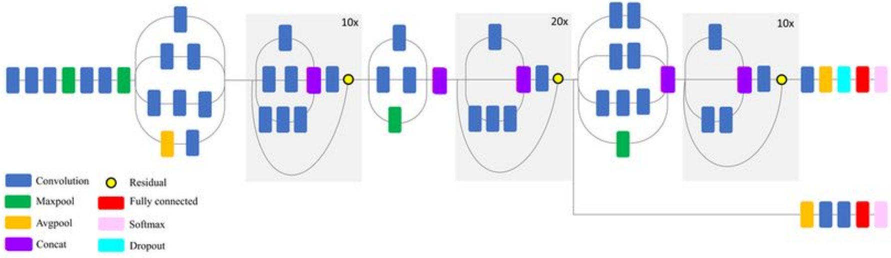




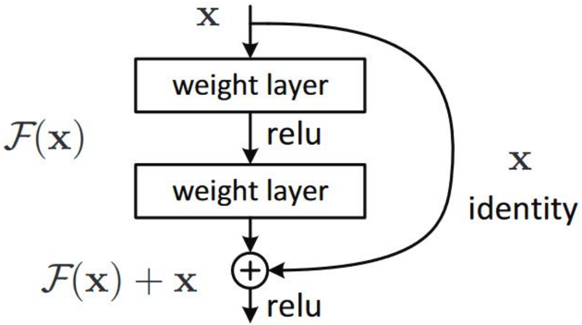
VGG16: construct large and deep models (120M params)

Vanishing gradient problem

ResNet (2015)

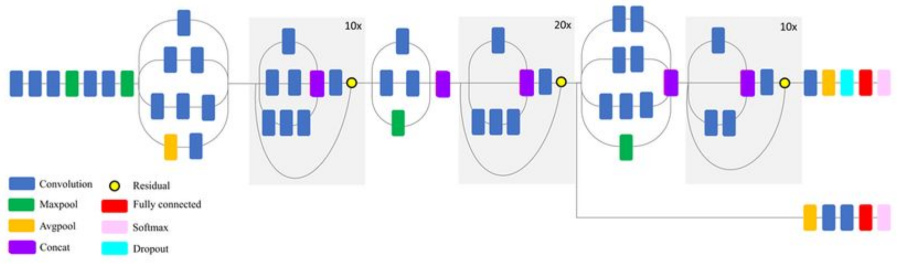


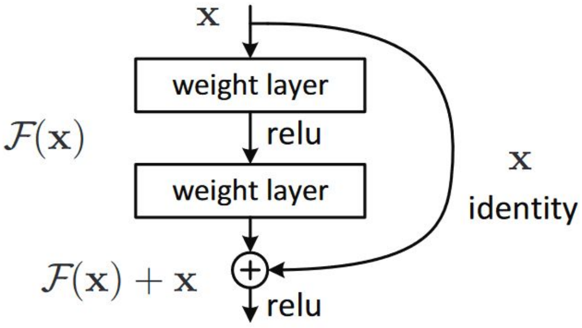
10x
20x
10x
Convolution
Maxpool
Avgpool
Concat
Residual
Fully connected
Softmax
Dropout



x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

ResNet (2015)

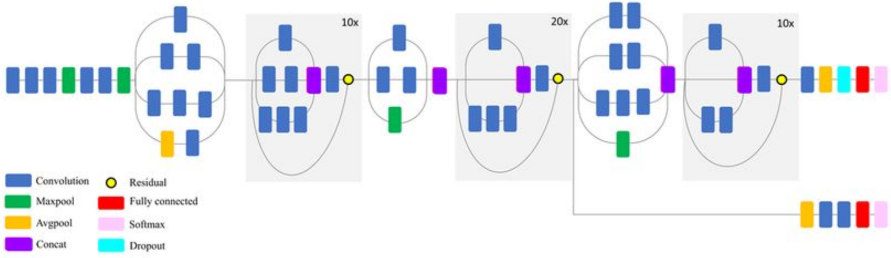


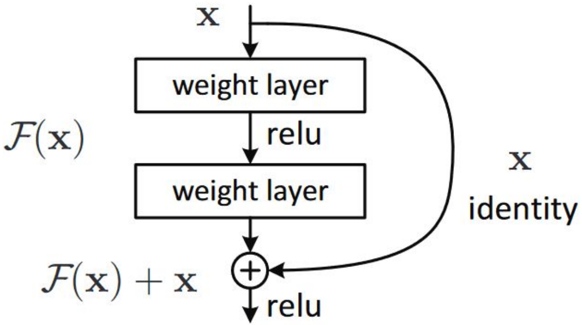


ResNets: construct large and deep models with skip connections, able to train up to 152 layers (!)

Higher abstraction and less nuisance from vanishing or exploding gradients

ResNet (2015)



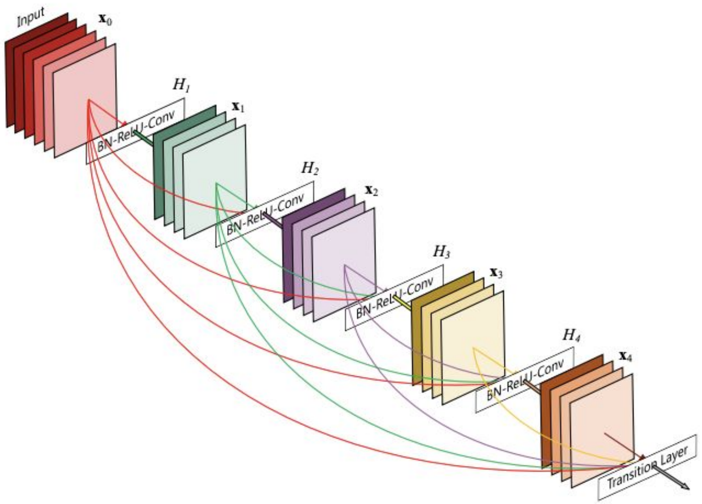


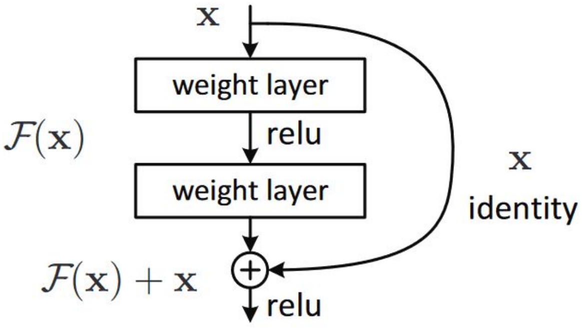
ResNets: construct large and deep models with skip connections, able to train up to 152 layers (!)

Higher abstraction and less nuisance from vanishing or exploding gradients

Early layers learn features that get progressively more abstract, we can **preserve** and **control** the flow of information with **skip connections**

DenseNet (2016)



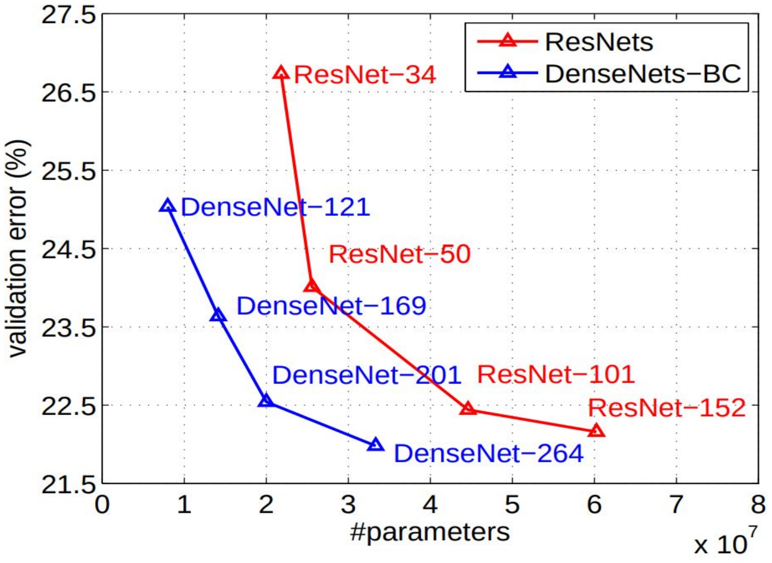


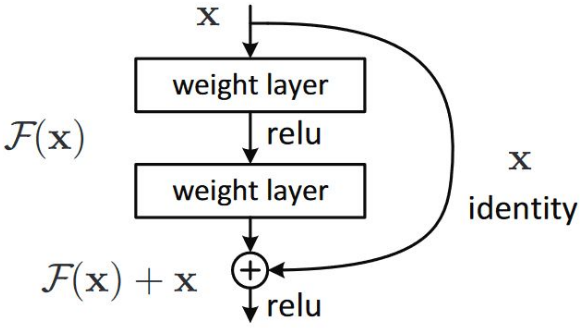
ResNets: construct large and deep models with skip connections, able to train up to 152 layers (!)

Higher abstraction and less nuisance from vanishing or exploding gradients

Early layers learn features that get progressively more abstract, we can **preserve** and **control** the flow of information with **skip connections**

DenseNet (2016)





ResNets: construct large and deep models with skip connections, able to train up to 152 layers (!)

Higher abstraction and less nuisance from vanishing or exploding gradients

Early layers learn features that get progressively more abstract, we can **preserve** and **control** the flow of information with **skip connections**