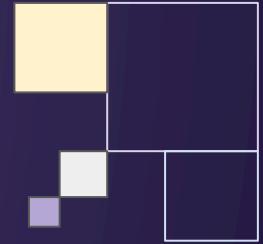


Deep Learning

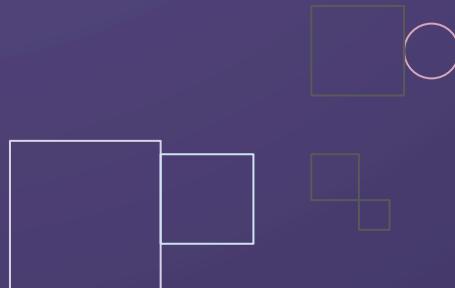
CNNs

Bryan Cardenas
Caspar van Leeuwen
Monica Rotulo
Robert Jan Schlimbach

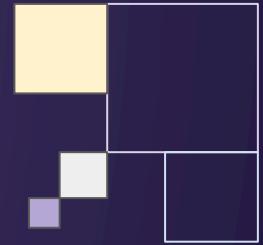




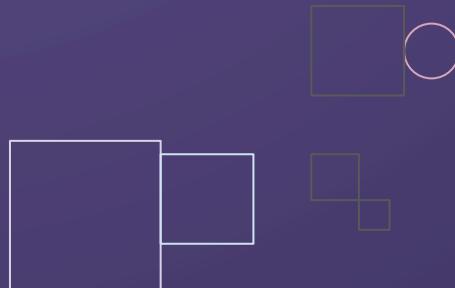
How do Computers see?



SURF

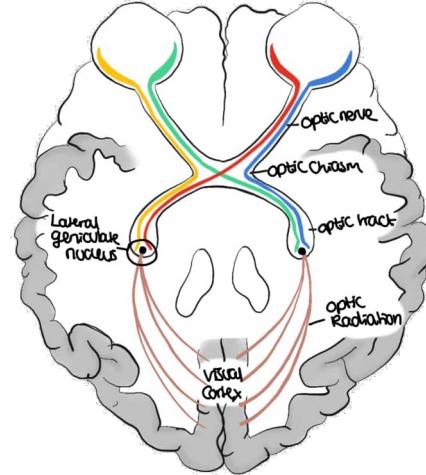
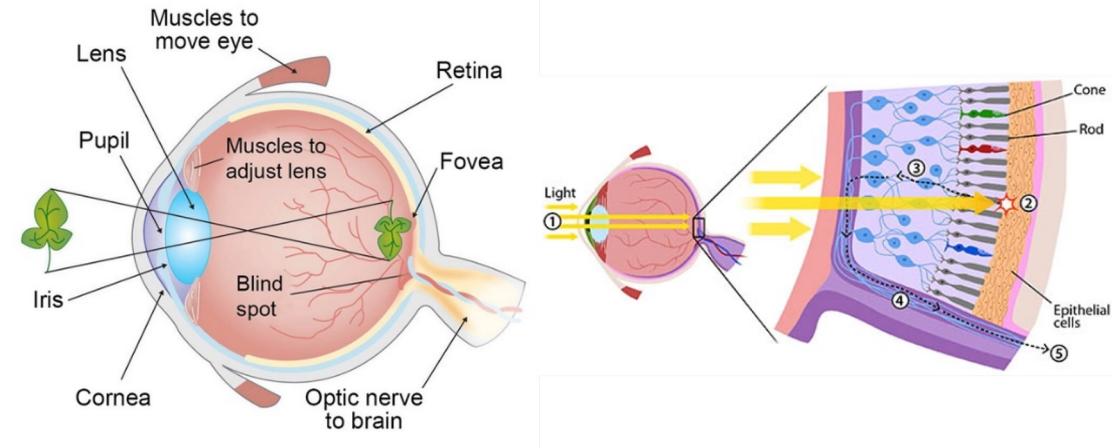


How do Humans see?



SURF

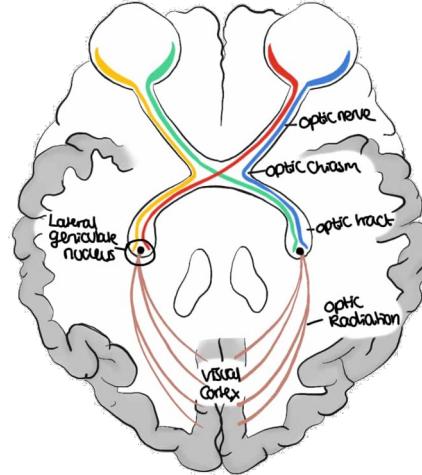
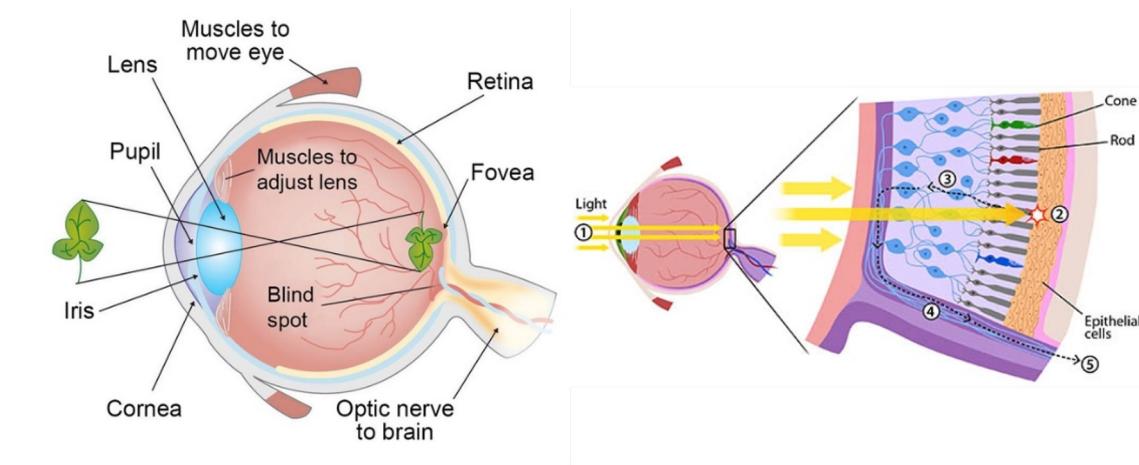
Mammalian Visual System



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

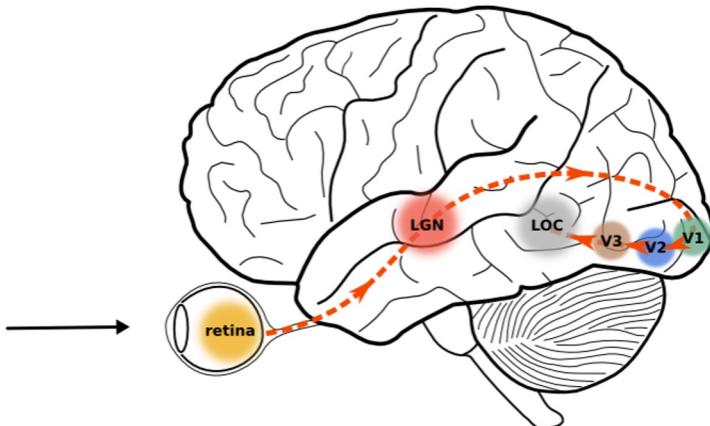
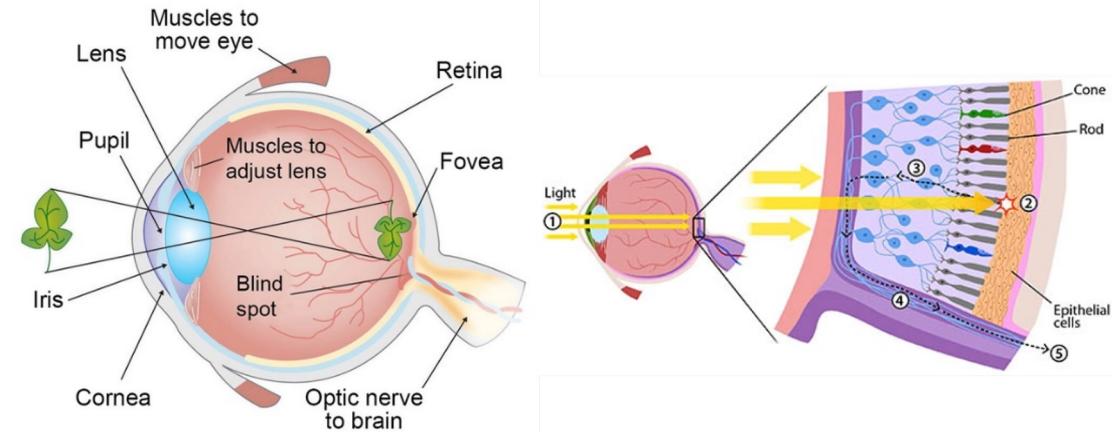


Mammalian Visual System

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

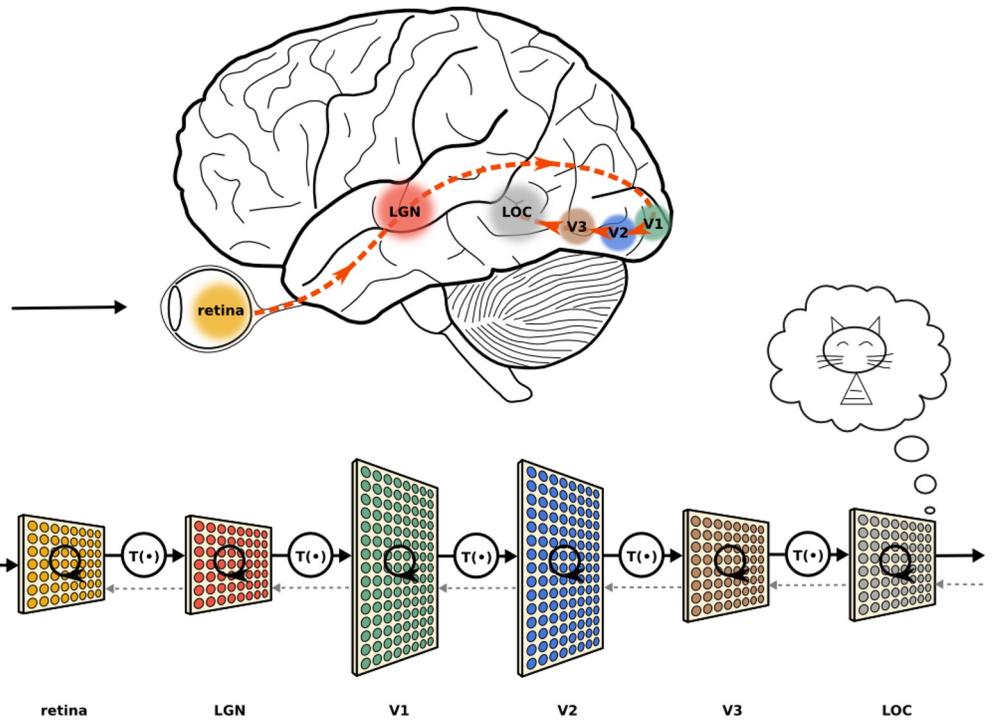


Mammalian Visual System

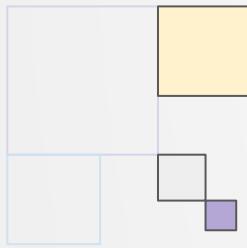
Cones and rods picks up photons and propagate the signal to the back of our brain

Along the way, the signal gets processed in stages

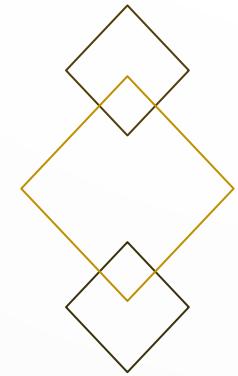
The extracted information gets aggregated and formed into an image



SURF



Convolutional Neural Networks



High Performance Machine Learning Group



Image Representation

- 1. Is a matrix or grid of intensity values
- 1. integers [0, 255] or float points [0,1]
- 1. Each element in the matrix is a pixel
- 1. Can have 1 greyscale channel or multiple colour channels: RGB

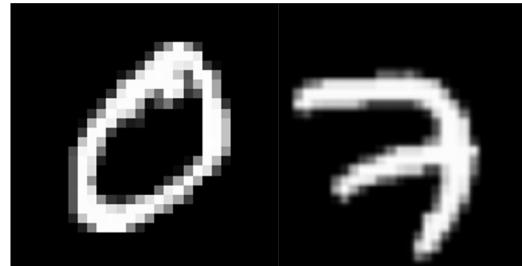
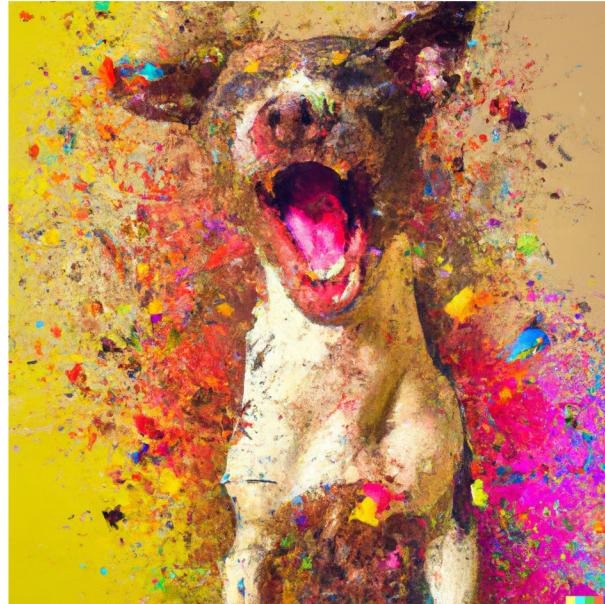
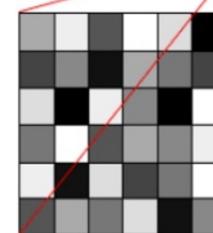
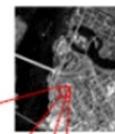
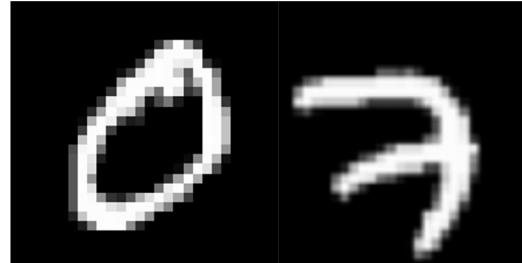


Image Representation

- 1. Is a matrix or grid of intensity values
- 1. integers [0, 255] or float points [0,1]
- 1. Each element in the matrix is a **pixel**
- 1. Can have 1 greyscale channel or multiple colour channels: **RGB**

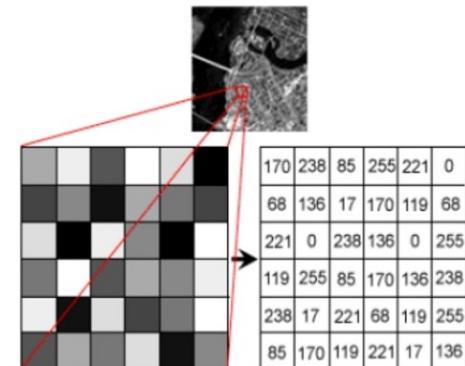
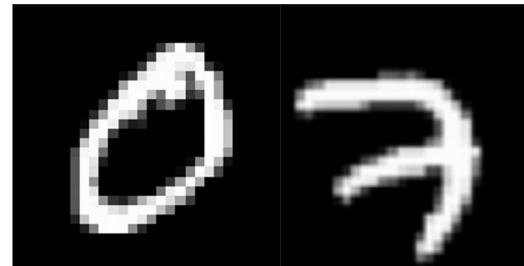
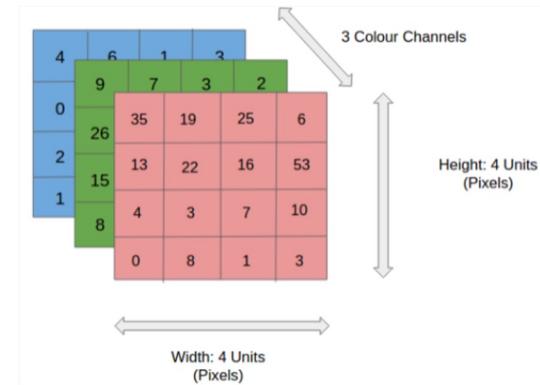


170	238	85	255	221	0	
68	136	17	170	119	68	
221	0	238	136	0	255	
119	255	85	170	136	238	
238	17	221	68	119	255	
85	170	119	221	17	136	

SURF

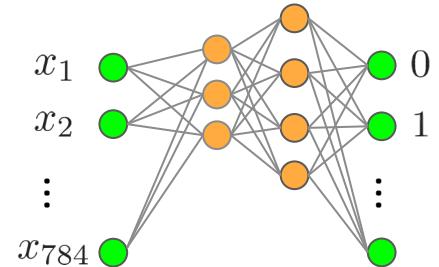
Image Representation

1. Is a matrix or grid of intensity values
1. integers [0, 255] or float points [0,1]
1. Each element in the matrix is a **pixel**
1. Can have 1 greyscale channel or multiple colour channels: **RGB**



SURF

Correlated structures

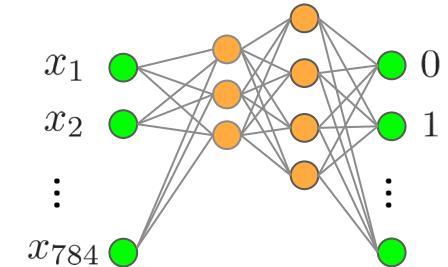


Images are **correlated spatially**

Strong correlation between
neighbouring pixels

Shifting an object will not alter its
essence

Correlated structures

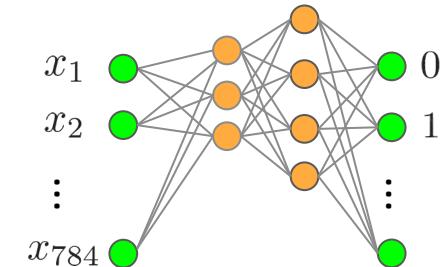
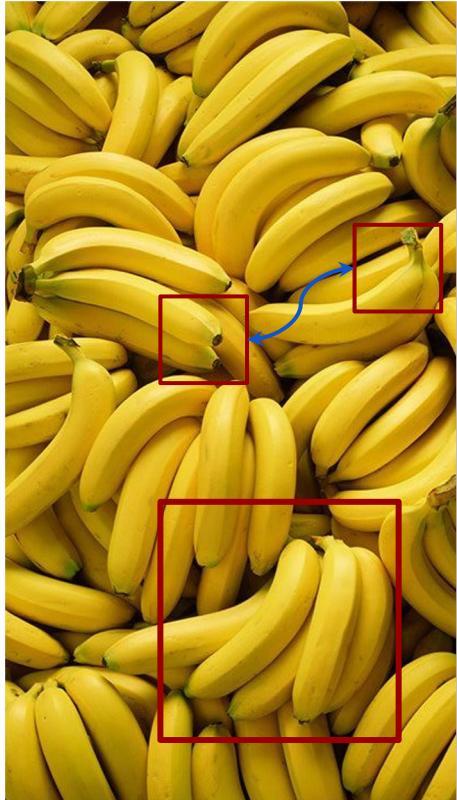


Images are **correlated spatially**

Strong correlation between
neighbouring pixels

Shifting an object will not alter its
essence

Correlated structures

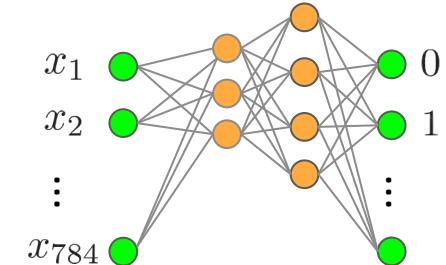
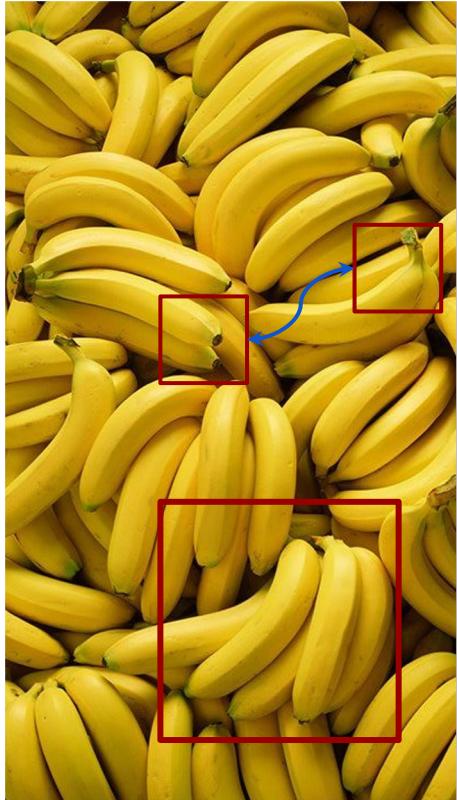


Images are **correlated spatially**

Strong correlation between
neighbouring pixels

Shifting an object will not alter its
essence

Correlated structures

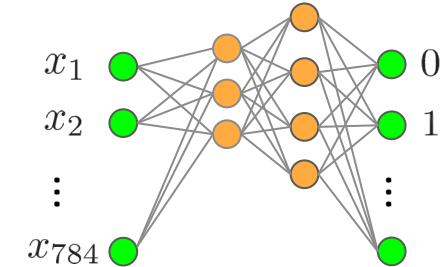
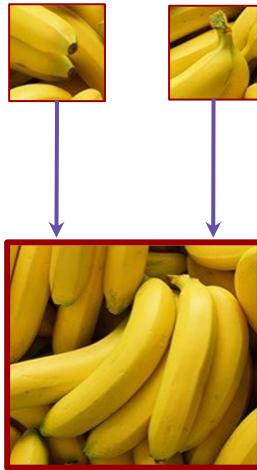
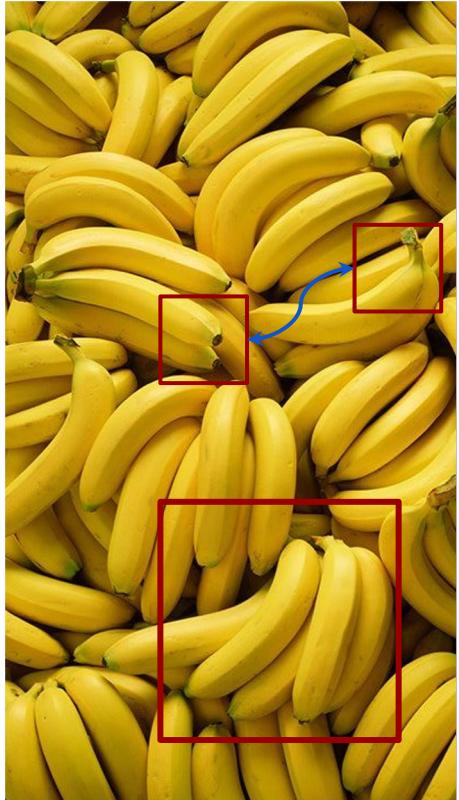


Images are **correlated spatially**

Strong correlation between
neighbouring pixels

Shifting an object will not alter its
essence

Correlated structures

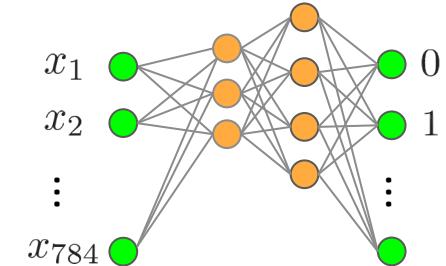
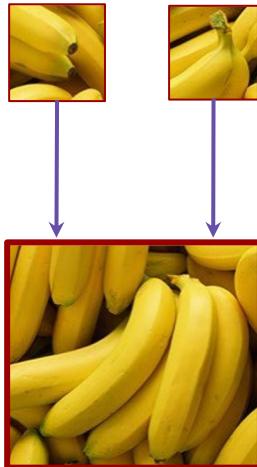
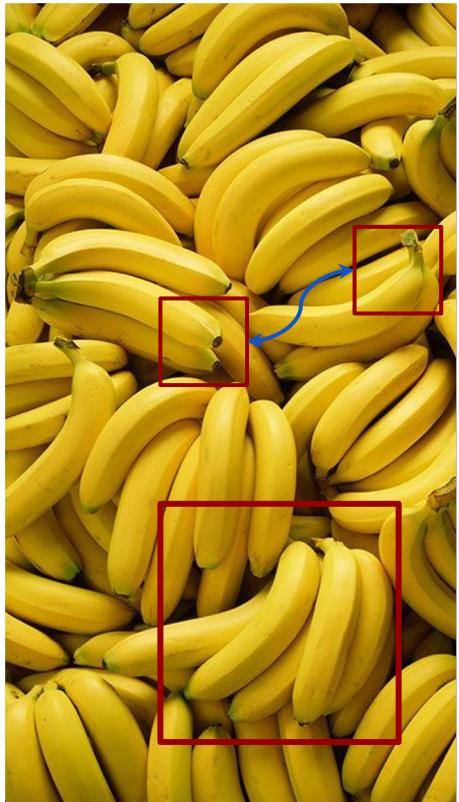


Images are **correlated spatially**

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

ImageNet Challenge

Computer Vision Benchmark

1.4M Images, 1000 classes

Image Classification

Difficult until 2012

SURF



ImageNet Challenge

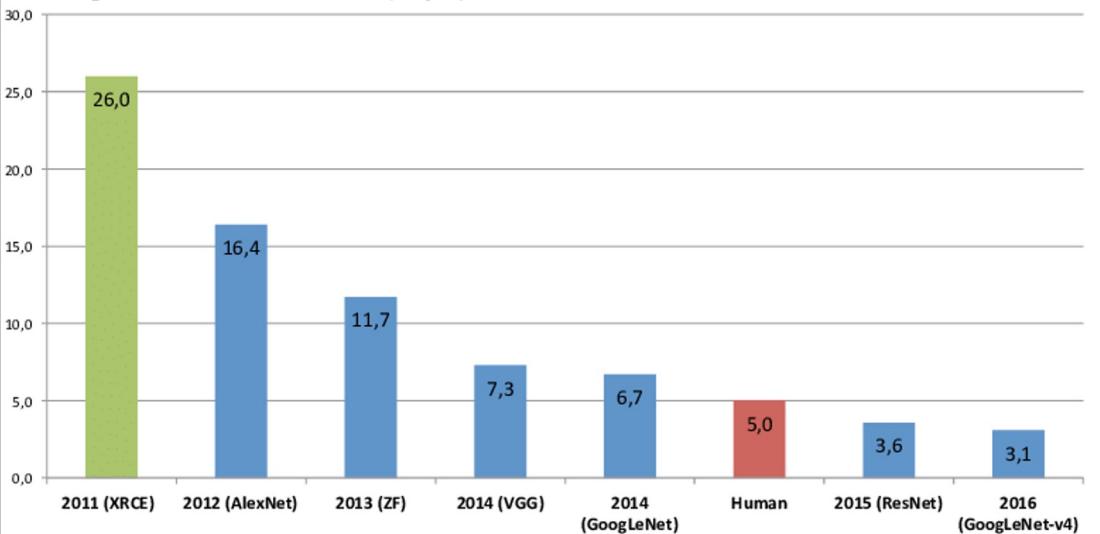
Computer Vision Benchmark

1.4M Images, 1000 classes

Image Classification

Difficult until 2012

ImageNet Classification Error (Top 5)



SURF

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

$$S(t) = (E * R)(t) = \int_{-\infty}^{\infty} E(\tau)R(t - \tau)d\tau$$

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

Impulse
Response

$$S(t) = (E * R)(t) = \int_{-\infty}^{\infty} E(\tau)R(t - \tau)d\tau$$

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

Source Impulse
Signal Response

$$S(t) = (E * R)(t) = \int_{-\infty}^{\infty} E(\tau)R(t - \tau)d\tau$$

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

Seismogram Source Impulse
 Signal Response

$$S(t) = (E * R)(t) = \int_{-\infty}^{\infty} E(\tau)R(t - \tau)d\tau$$

The Convolution Operation

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m]$$

The Convolution Operation

Initial signal gets modified by the system's response

One signal **slides** across the other

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m]$$

Convolutions

Kernel (filter): small matrix that we use to convolve 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)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

SURF

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

SURF

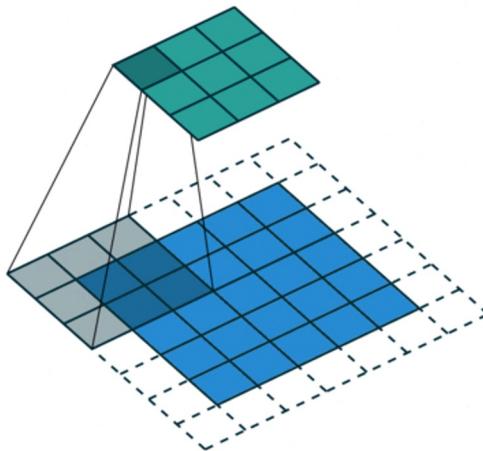
Convolutions

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	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
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

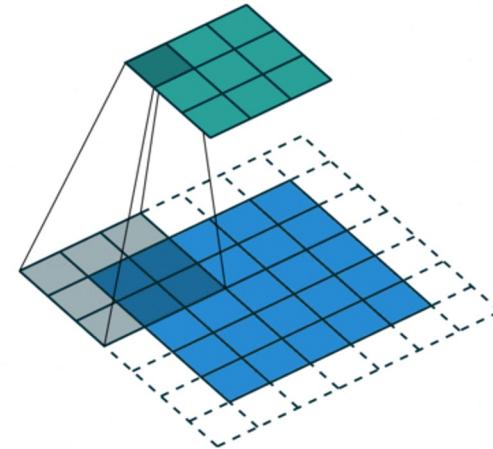


Convolution:

An operation that “blends” one function with another.

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

○



Convolutions

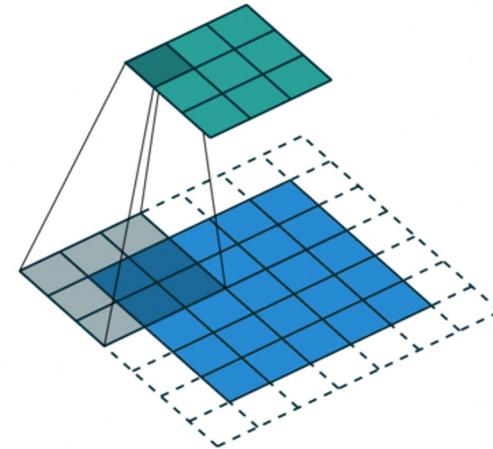
Kernel (filter): small matrix that we use to convolve an image



Convolution:

An operation that “blends” one function with another.

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



Convolutions

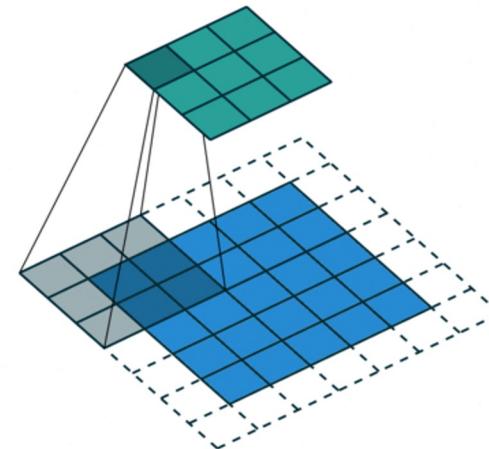
Kernel (filter): small matrix that we use to convolve an image

Convolution:

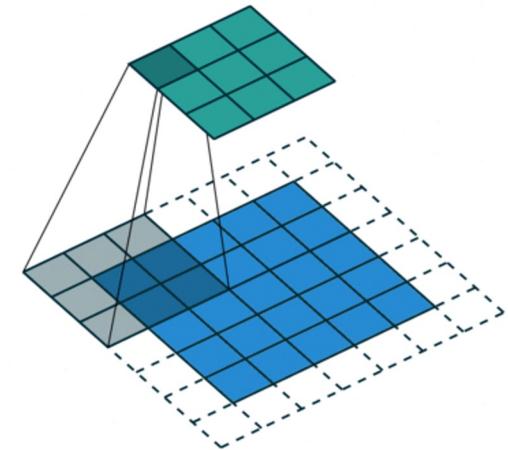
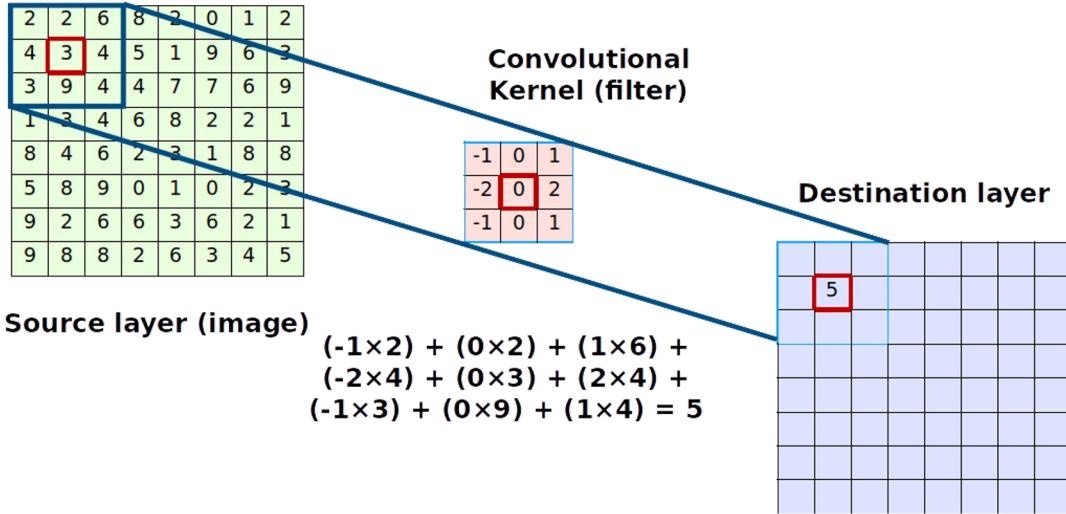
An operation that “blends” one function with another.

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

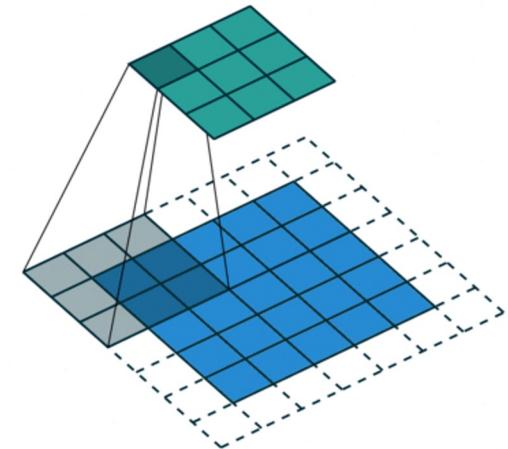
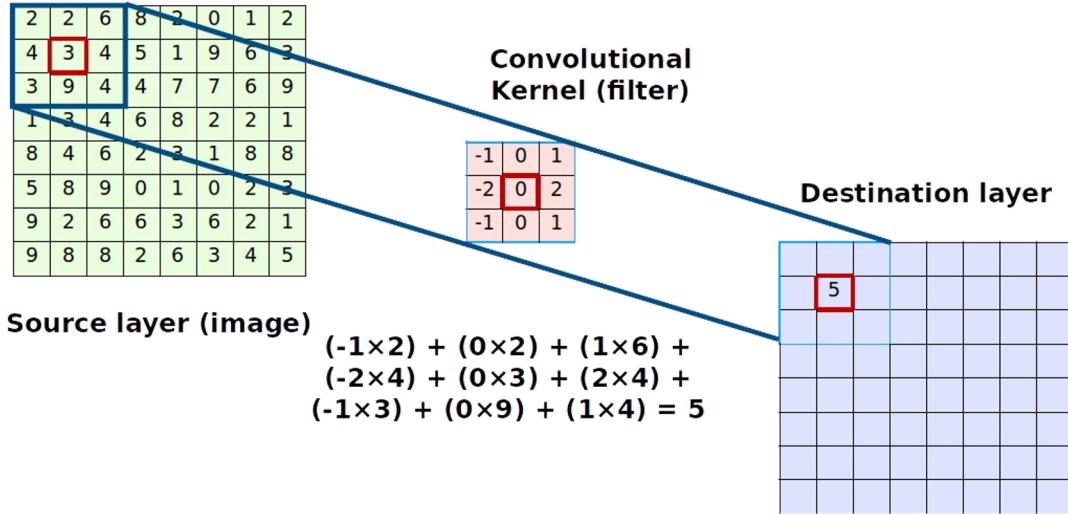
$$\sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(i, j) \cdot K(x - i, y - j)$$



Convolutions Compute



Convolutions Compute

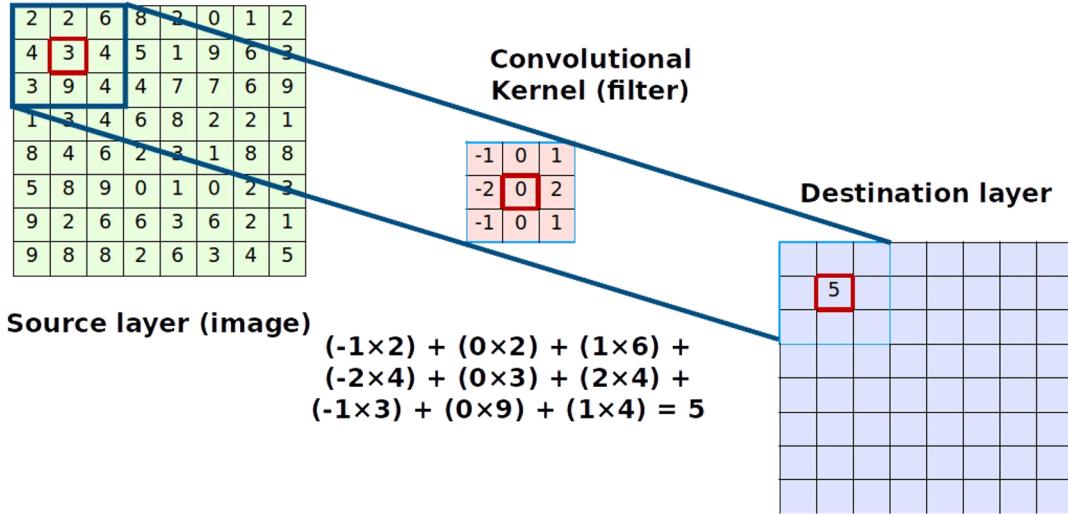


The kernel is **shifted** across the image and produces a point value

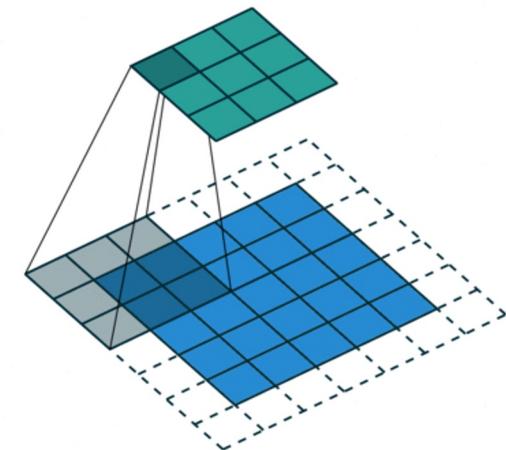
The step size in which it shifts is the **stride**

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
 $\text{Output_size} = \text{inputSize} - (\text{KernelSize} - 1)$

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

5								

Kernel (filter)

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

Kernel (filter)

How do we know which kernels to use?

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

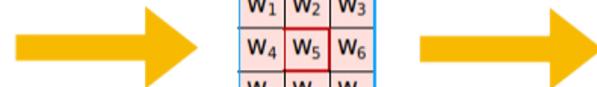
Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

5								



Kernel (filter)

How do we know which kernels to use?

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

Fully connected

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

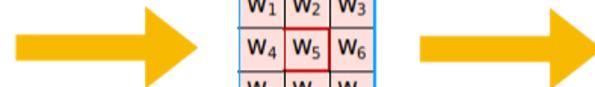
Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

5								



Kernel (filter)

How do we know which kernels to use?

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

Fully connected

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

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

		5		

Kernel (filter)

How do we know which kernels to use?

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

Fully connected

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

Locally connected shared weights

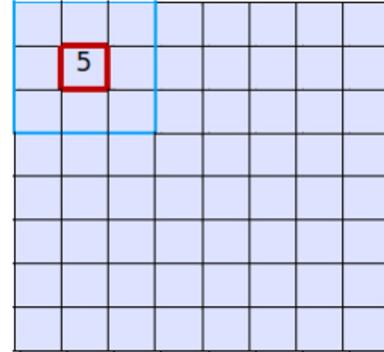
$$W \times H \times C \sum_{i \in 3 \times 3} \mathbf{x}_i \mathbf{w}_i$$

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)



Kernel (filter)

How do we know which kernels to use?

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

Fully connected

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

Locally connected shared weights

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

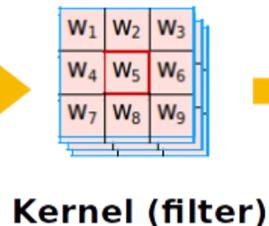
Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)

5								



How do we know which kernels to use?

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

Fully connected

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

Locally connected shared weights

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

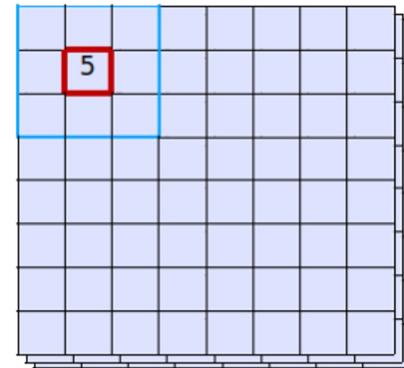
SURF

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)



Kernel (filter)

How do we know which kernels to use?

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

Fully connected

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

Locally connected shared weights

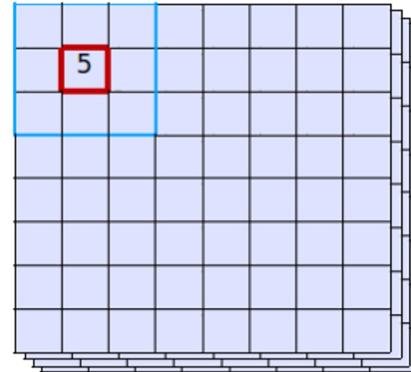
$$W \times H \times C \sum_{i \in 3 \times 3} \mathbf{x}_i \mathbf{w}_i$$

Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Feature map (activation map)



Kernel (filter)

How do we know which kernels to use?

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

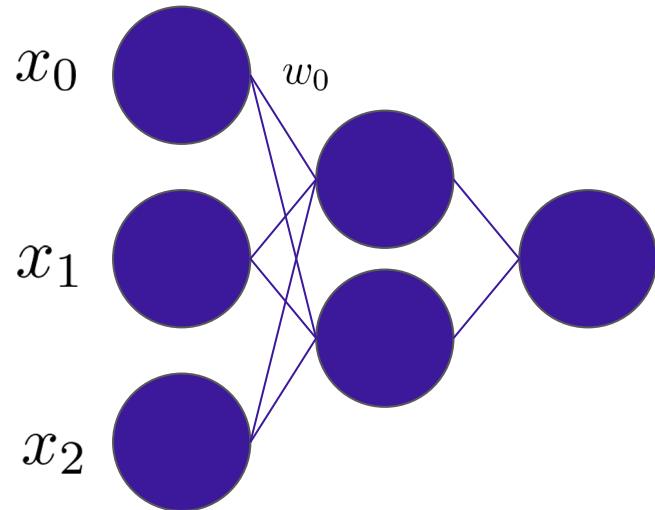
Fully connected

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

Locally connected shared weights

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

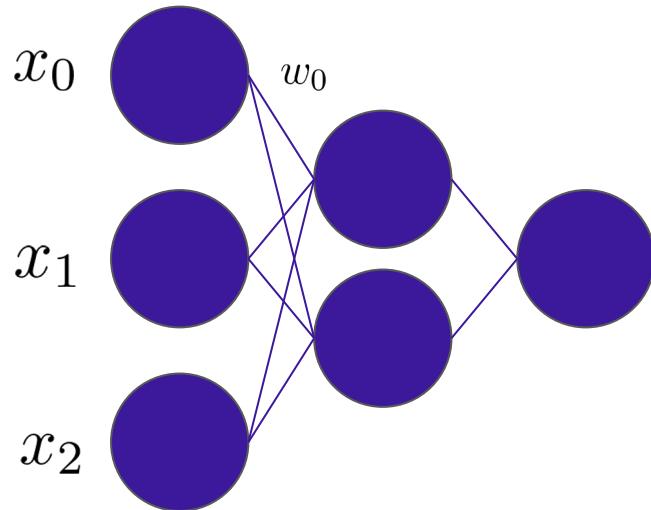
Anatomy of an Convolutional Neural Network



CNNs actually look a lot like
normal Dense Neural Networks

Interpretation: Every node is a
kernel

Anatomy of an Convolutional Neural Network



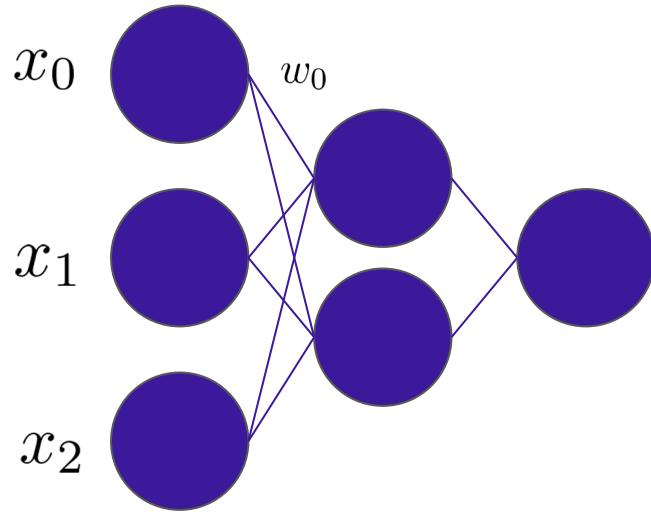
See the nodes as moving across the image!

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

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

Locally connected
shared weights

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

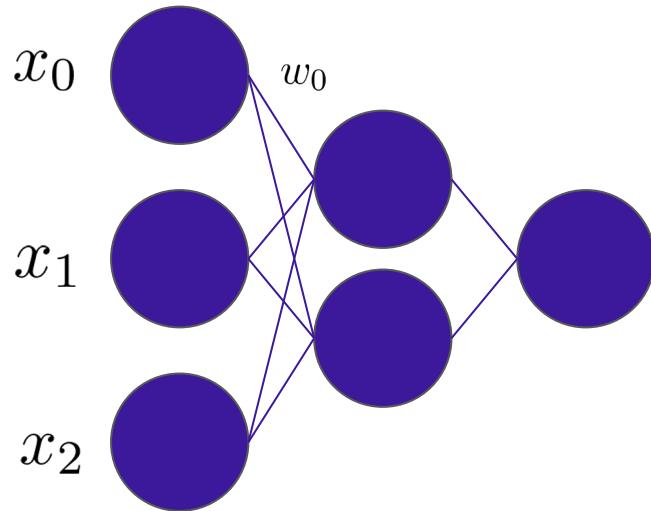
See the nodes as moving across
the image!

We say these are **shared weights**

CNNs actually look a lot like
normal Dense Neural Networks

Interpretation: Every node is a
kernel

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

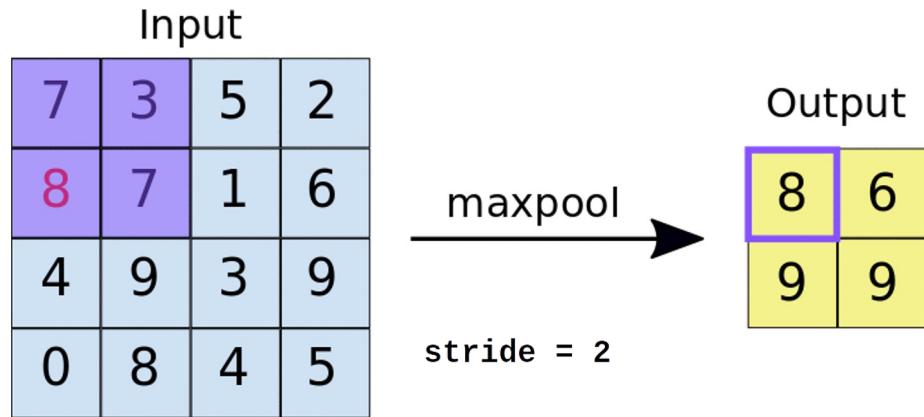
See the nodes as moving across the image!

We say these are **shared weights**

CNNs actually look a lot like normal Dense Neural Networks

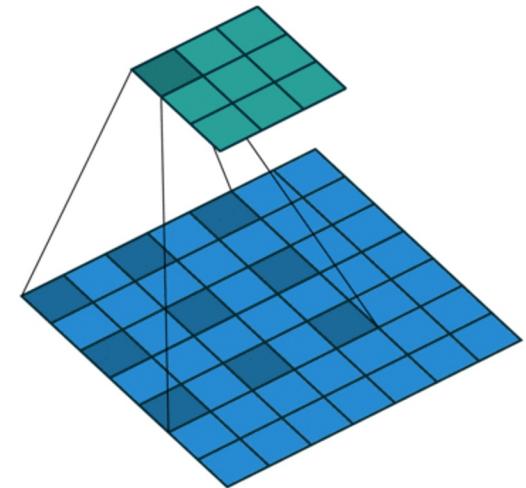
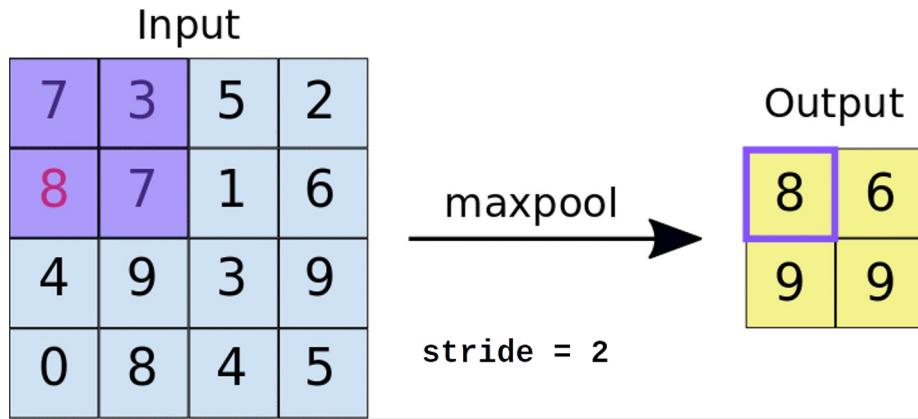
Interpretation: Every node is a kernel

Pooling and dilated convolutions



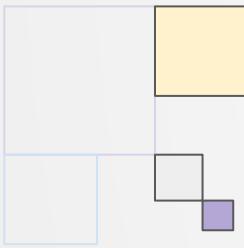
We could discard information gradually by max pooling
Keep the strongest signal, works better than average pooling
Nowadays, we use a bigger stride for dimensionality reduction

Pooling and dilated convolutions

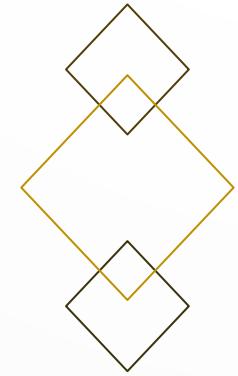


We could discard information gradually by max pooling
Keep the strongest signal, works better than average pooling
Nowadays, we use a bigger stride for dimensionality reduction

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



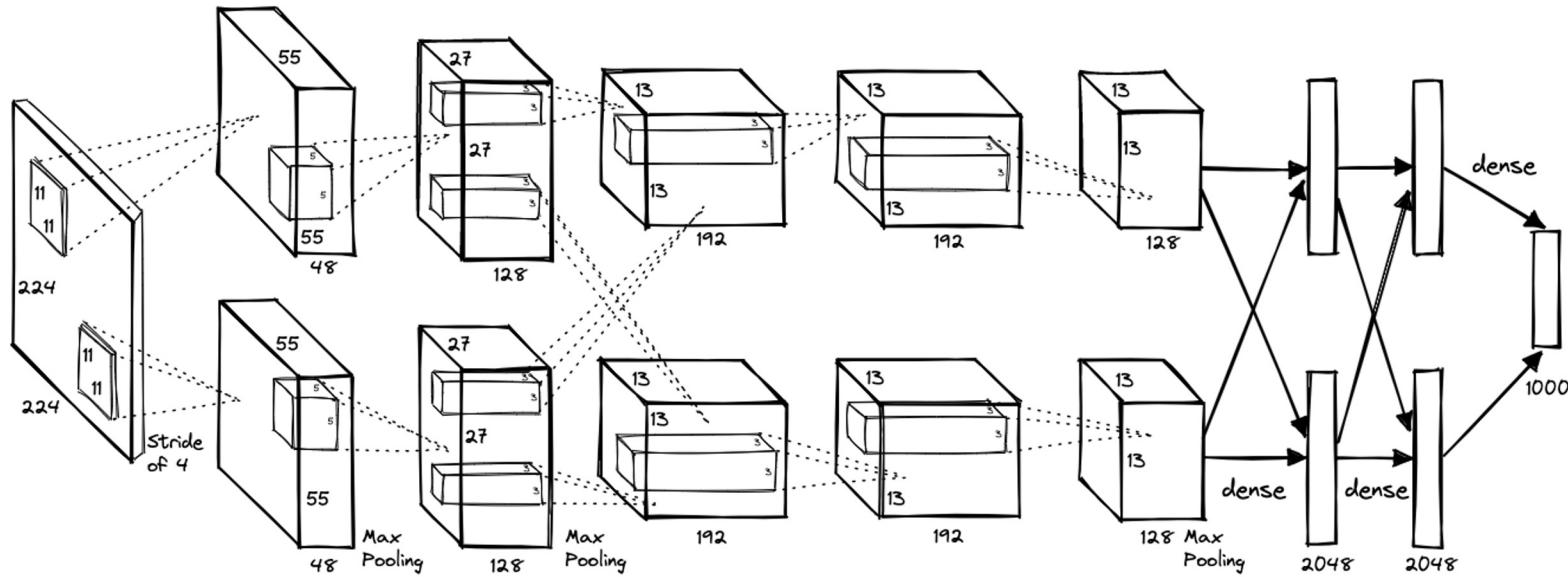
CNN Cases



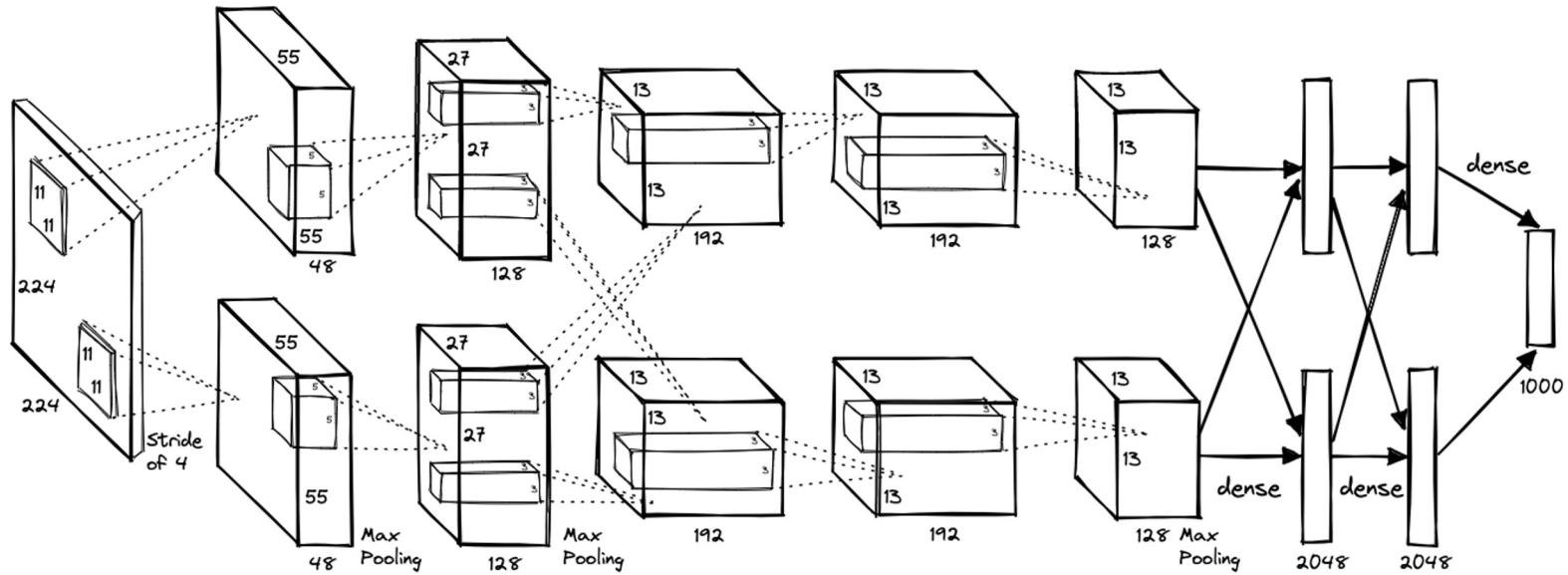
High Performance Machine Learning Group



AlexNet (2012)



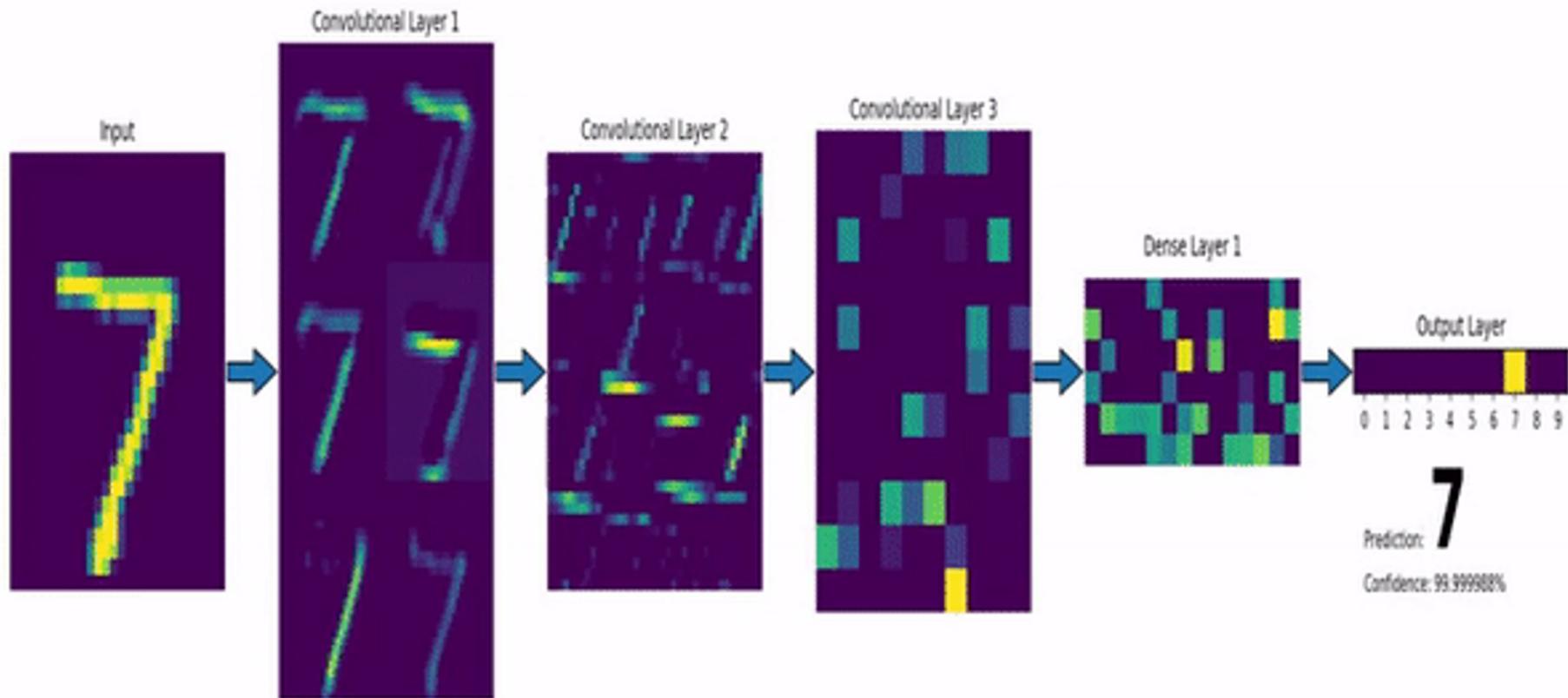
AlexNet (2012)



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**

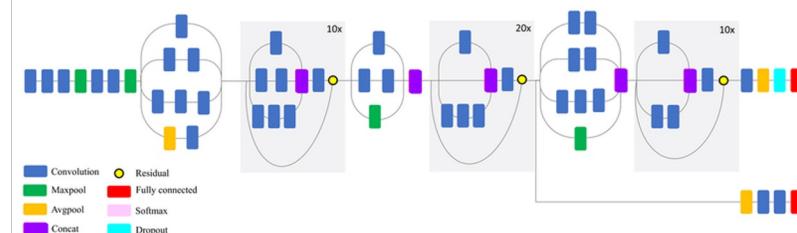
CNN with a Dense Layer output





IMAGENET

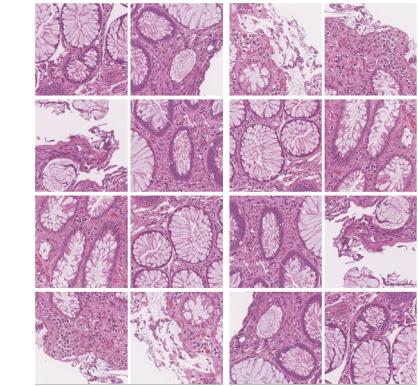
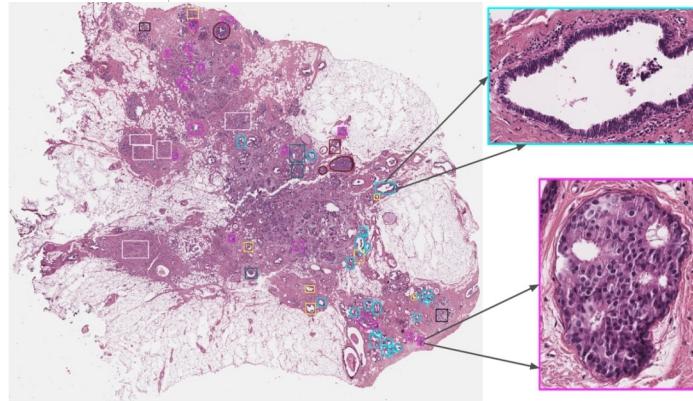
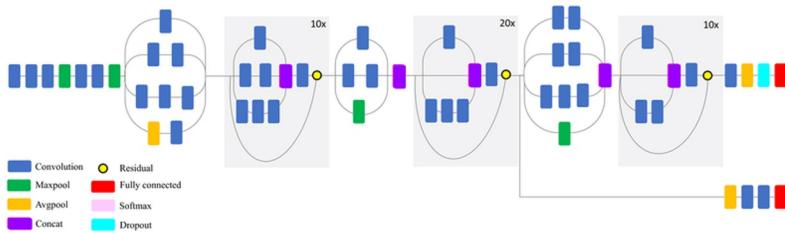
Transfer Learning





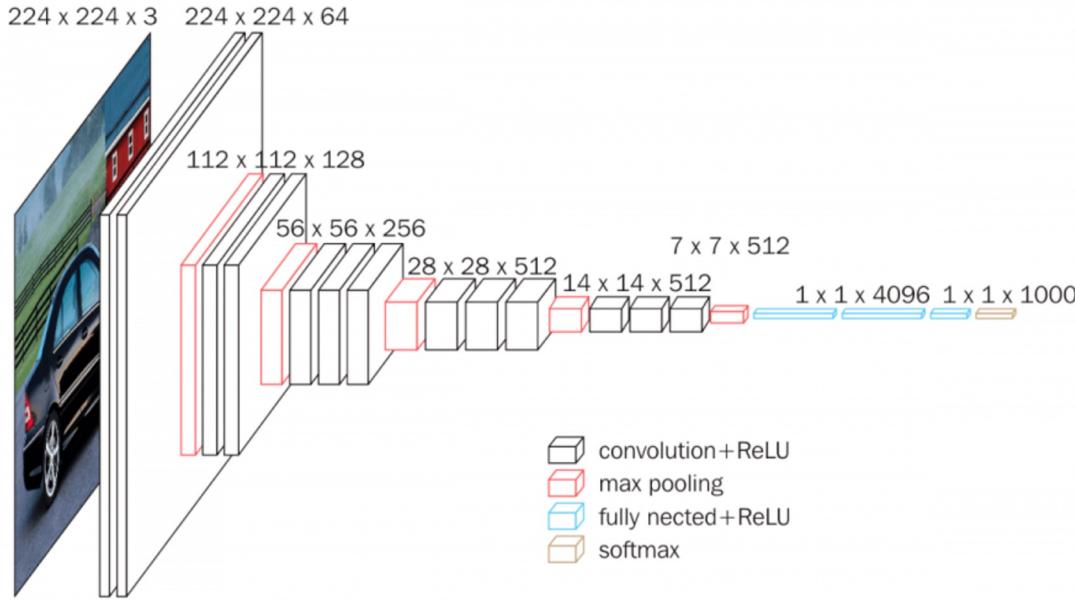
IMAGENET

Transfer Learning



SURF

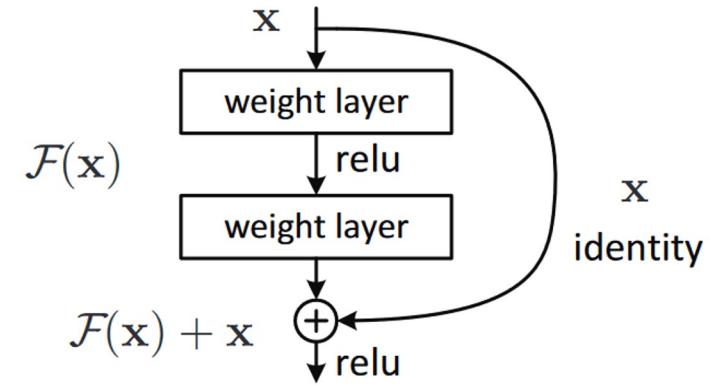
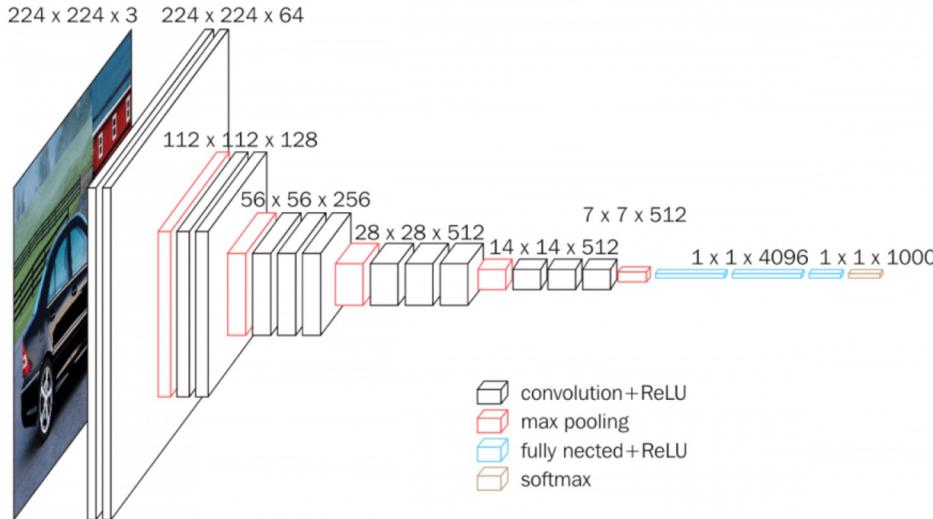
VGG16 and ResNet (2014, 2015)



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

Vanishing gradient problem

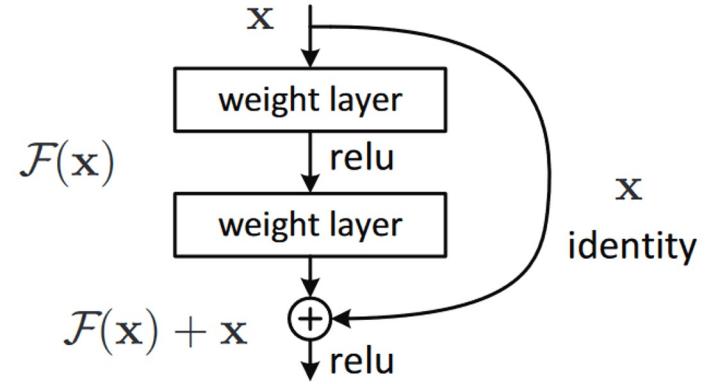
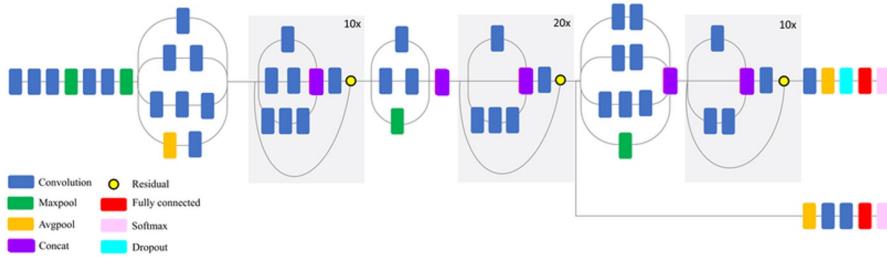
VGG16 and ResNet (2014, 2015)



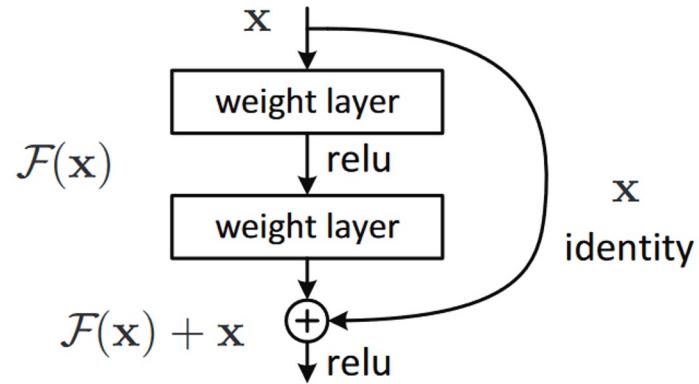
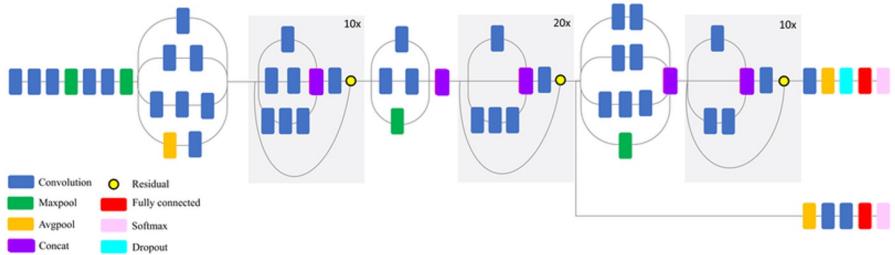
VGG16: construct large and deep models (120M params)

Vanishing gradient problem

ResNet (2015)



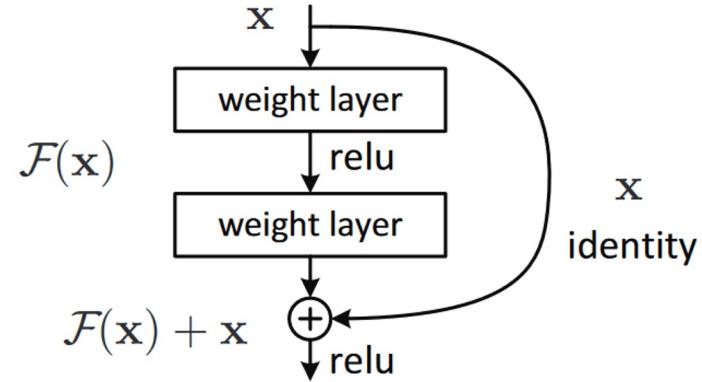
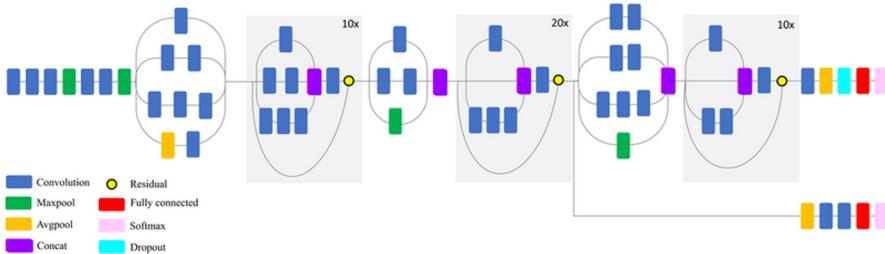
ResNet (2015)



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)

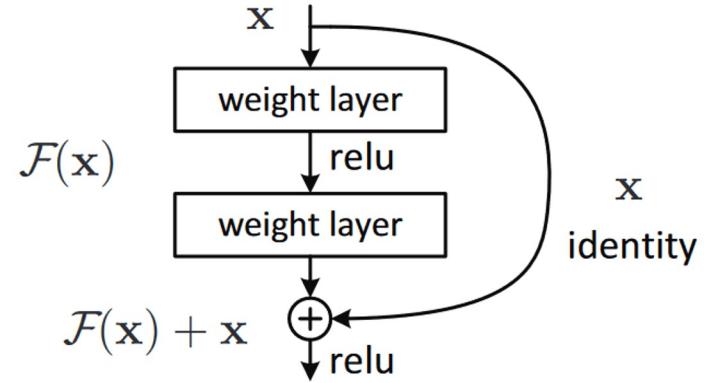
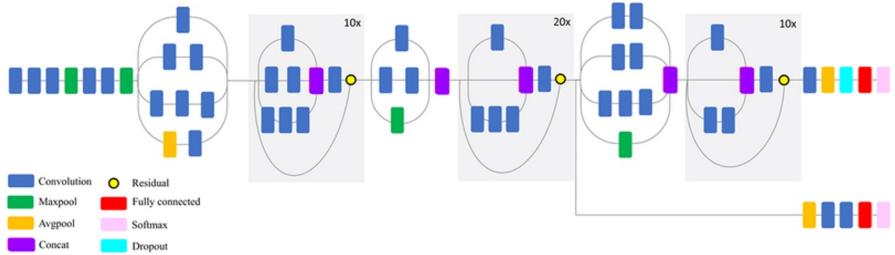


ResNets: construct large and deep models with **skip connections**, able to train up to **152 layers** (!)

Higher abstraction and less nuisance from vanishing or exploding gradients

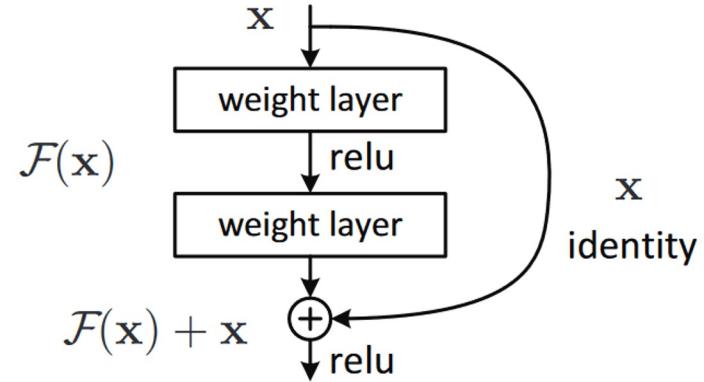
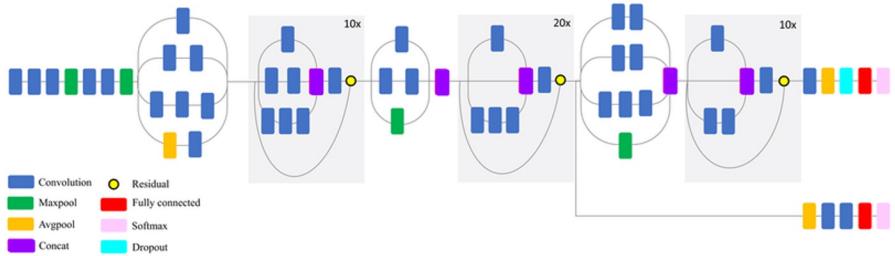
Early layers learn features that get progressively more abstract, we can **preserve** and **control** the flow of information with **skip connections**

ResNet (2015)



$$G(x) = \mathcal{F}(x) + x$$

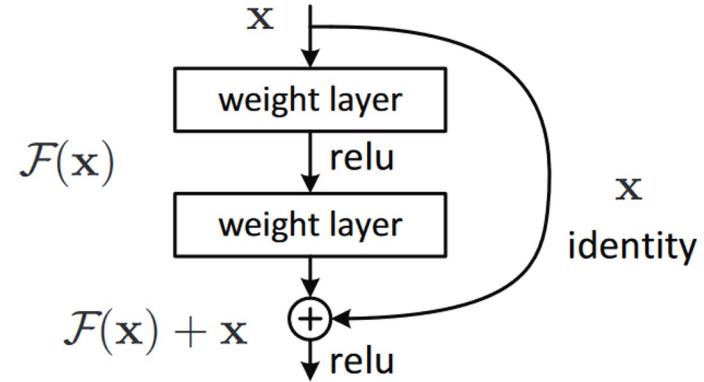
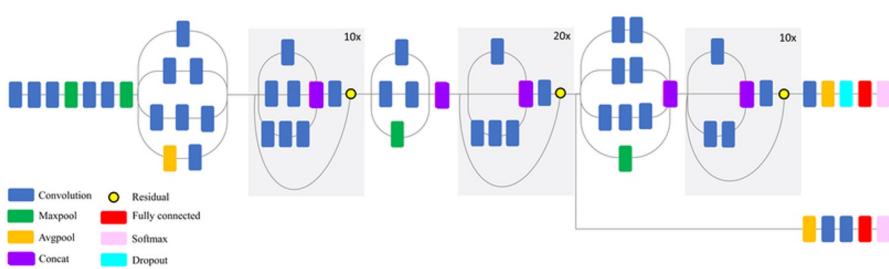
ResNet (2015)



$$G(x) = \mathcal{F}(x) + x$$

$$\frac{\partial \mathcal{L}}{\partial G} \frac{\partial F(\mathbf{x}) + \mathbf{x}}{\partial \mathbf{x}}$$

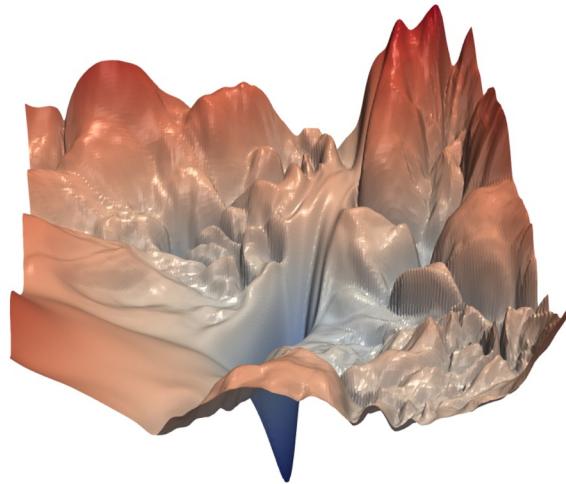
ResNet (2015)



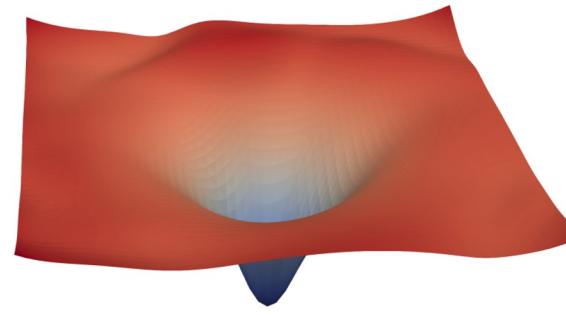
$$G(x) = \mathcal{F}(x) + x$$

$$\frac{\partial \mathcal{L}}{\partial G} \frac{\partial F(\mathbf{x}) + \mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathcal{L}}{\partial G} \frac{\partial F}{\partial \mathbf{x}} + \frac{\partial \mathcal{L}}{\partial G}$$

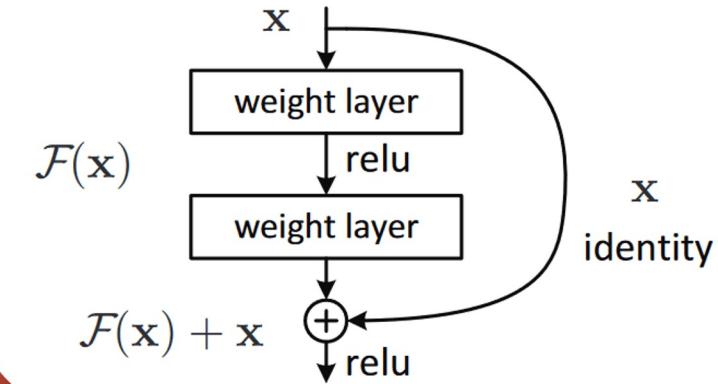
ResNet (2015)



(a) without skip connections

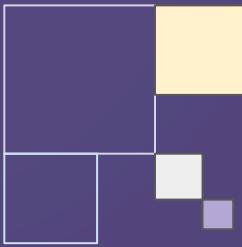


(b) with skip connections



$$\frac{\partial \mathcal{L}}{\partial G} \frac{\partial F(\mathbf{x}) + \mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathcal{L}}{\partial G} \frac{\partial F}{\partial \mathbf{x}} + \frac{\partial \mathcal{L}}{\partial G}$$



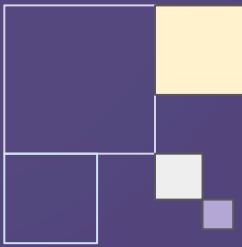


Thank You



High Performance Machine Learning Group

SURF



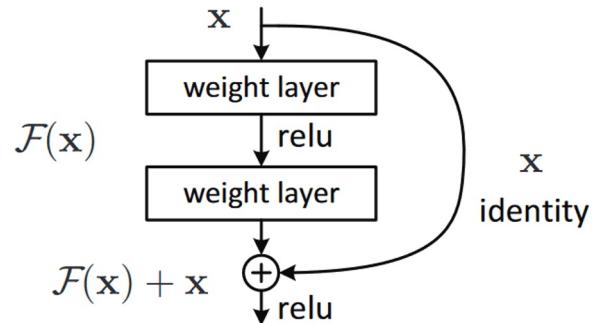
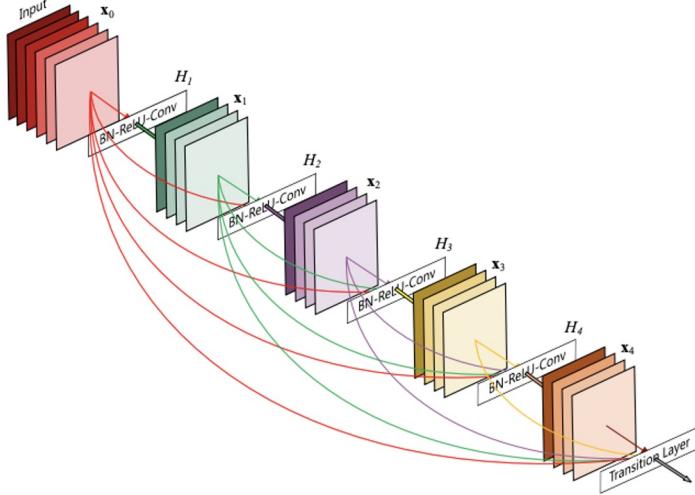
Slides Graveyard



High Performance Machine Learning Group

SURF

DenseNet (2016)

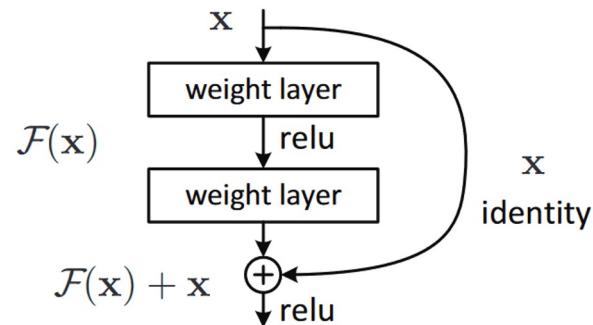
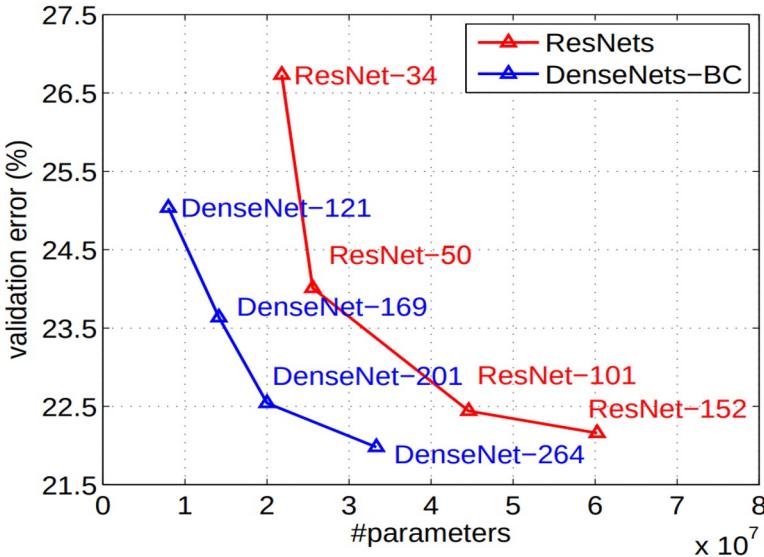


ResNets: construct large and deep models with skip connections, able to train up to 152 layers (!)

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**

Type: ML Perceptron

Data Set: MNIST

Hidden Layers: 3

Hidden Neurons: 10000

Synapses: 24864180

Synapses shown: 2%

Learning: BP

