Deep Learning Approaches for Medical Imaging

Wintertagung der AG Thoraxpathologie DGPath

Elvis Murina

Institute of Data Analysis and Process Design
Zurich University of Applied Sciences

03.02.2018

Overview

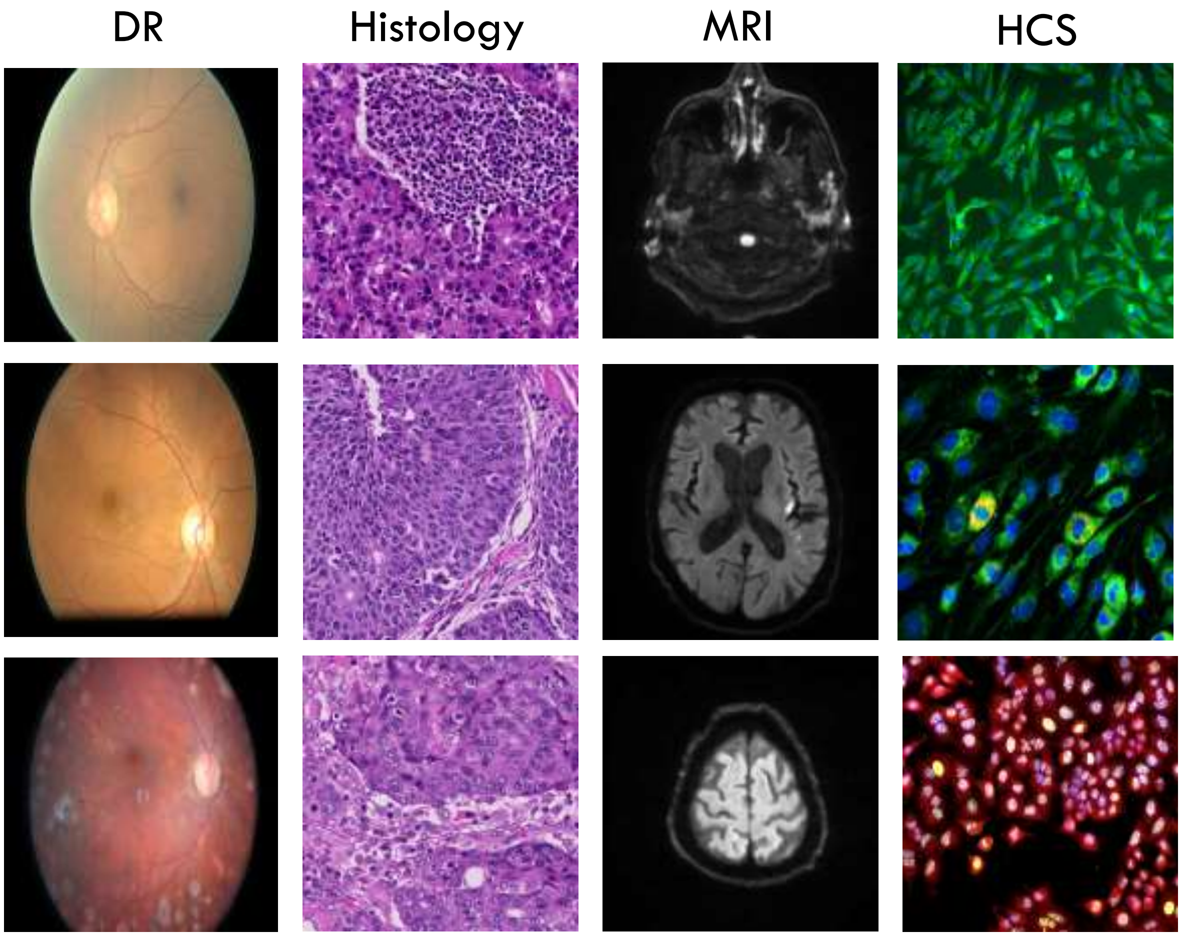
- Motivation
- Traditional image analysis vs. Deep Learning
- ImageNet competition
- Artificial neural networks
- Convolutional neural networks
- Deep Learning for feature extraction

Motivation

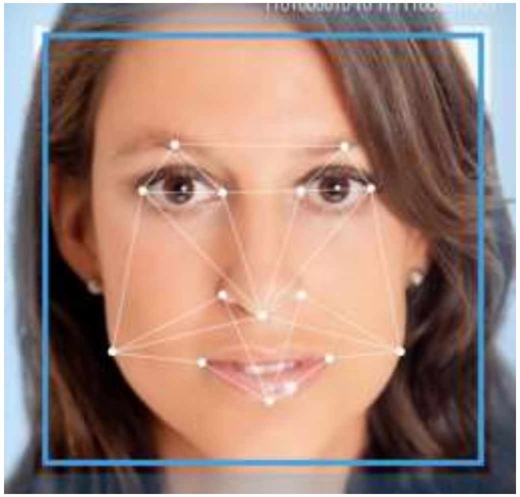
- Medical field produces a lot of images
- Goal is often some kind of classification in types or stages
- Time-consuming and manual process
- Expert needed

Motivation

- Histology
  - adeno vs sqcc
- Eye fundus
  - five DR stages
- MRI
  - stroke/ no stroke
- High content screening
  - diffrent pheotypes
- X-ray
- CT
- …



Traditional image analysis vs. Deep Learning

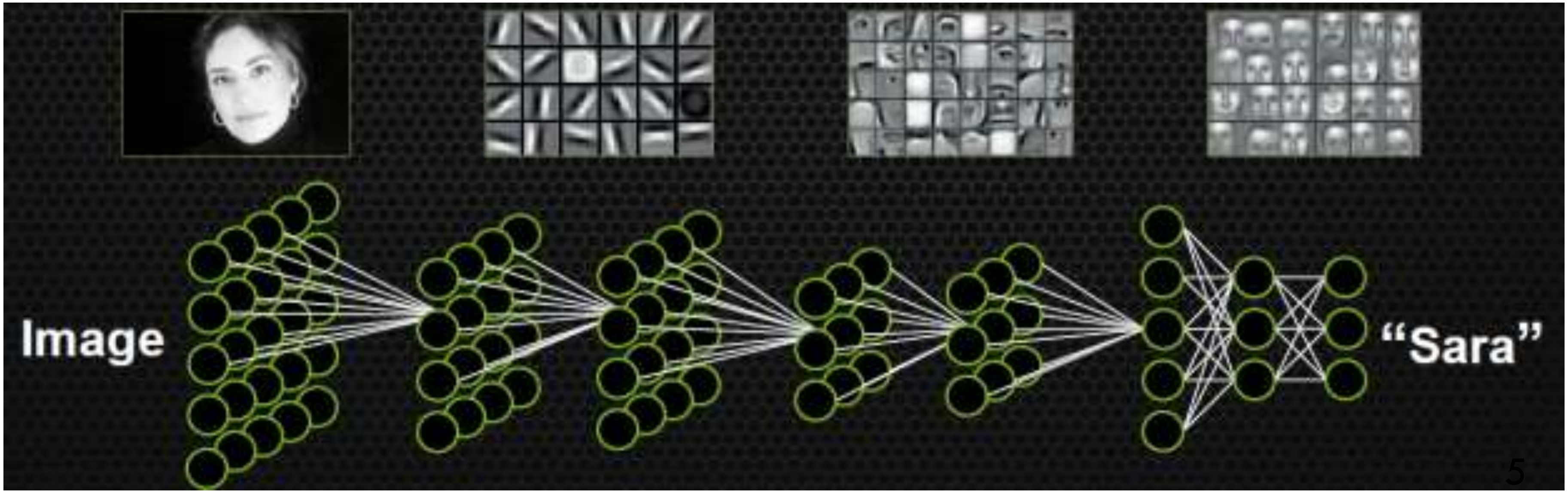


Traditional:

Extract handcrafted features and use these features to train/fit a model (SVM, RF) and use fitted model to perform classification/prediction.

Deep learning:

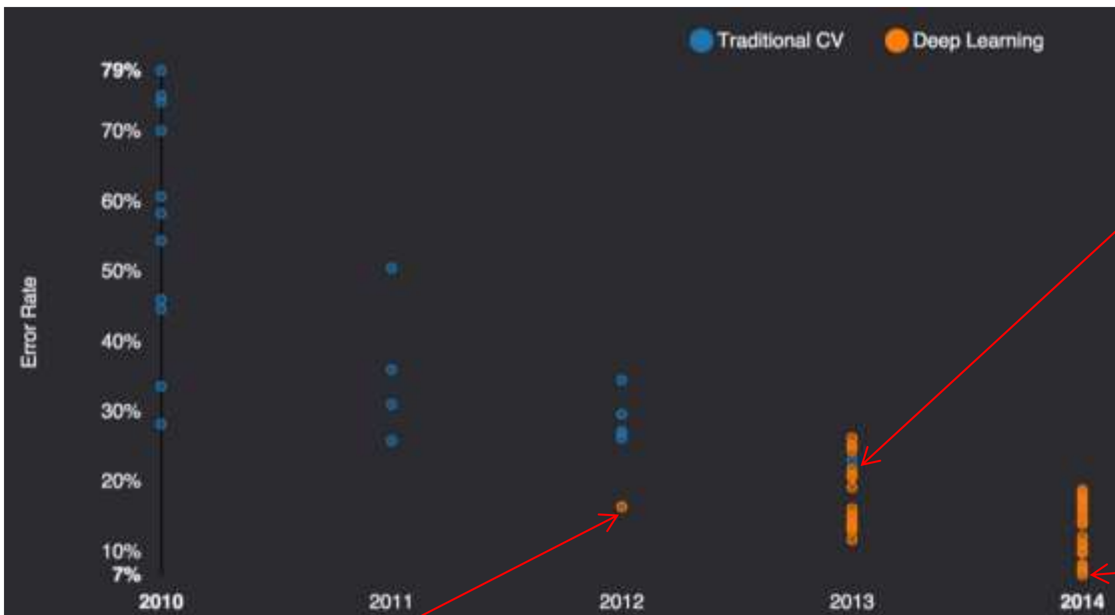
In deep neural networks start with raw data and learn during training/fitting to extract appropriate hierarchical features and to use them for classification/prediction.



ImageNet competition

1000 classes
1 Mio samples

Human: 5% misclassification



Only one non-CNN approach in 2013

GoogLeNet 6.7%

A. Krizhevsky
first CNN in 2012

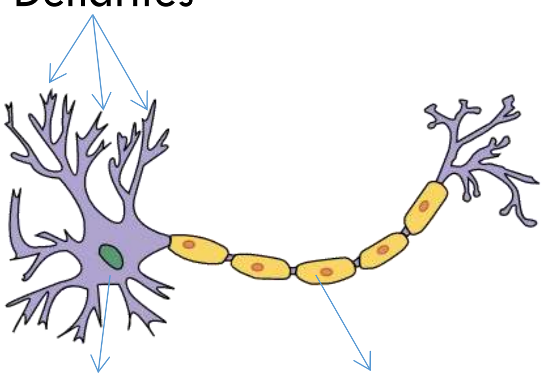
<u>2015</u>
4.95% Microsoft (surpassing human performance 5.1%)
4.8% Google (further improved to 3.6%)
3.57% Microsoft (Resnet winner 2015)

Artificial neural networks

$$z_1 = \sum x_1 * w_{11} + x_2 * w_{21} + x_3 * w_{31} + b$$

Weighted sum of x and W

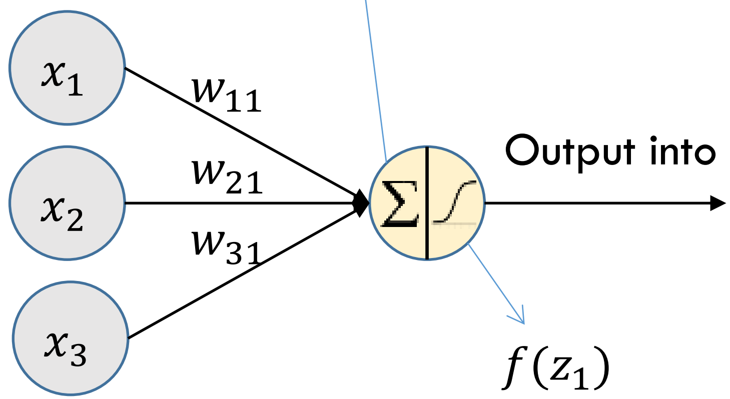
Dendrites



Cell body

Axon

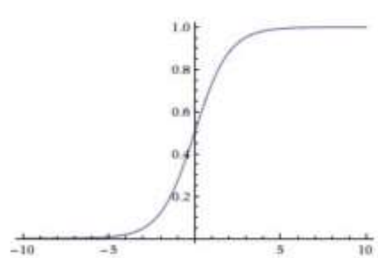
Input



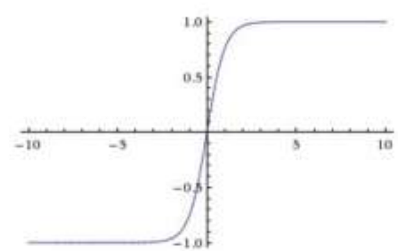
Output into the next layer

$f(z_1)$

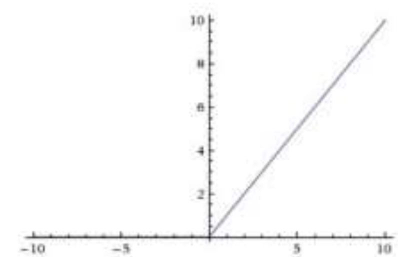
Non–linear activation function for z



Sigmoid

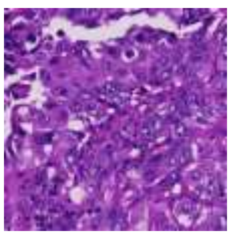


tanh

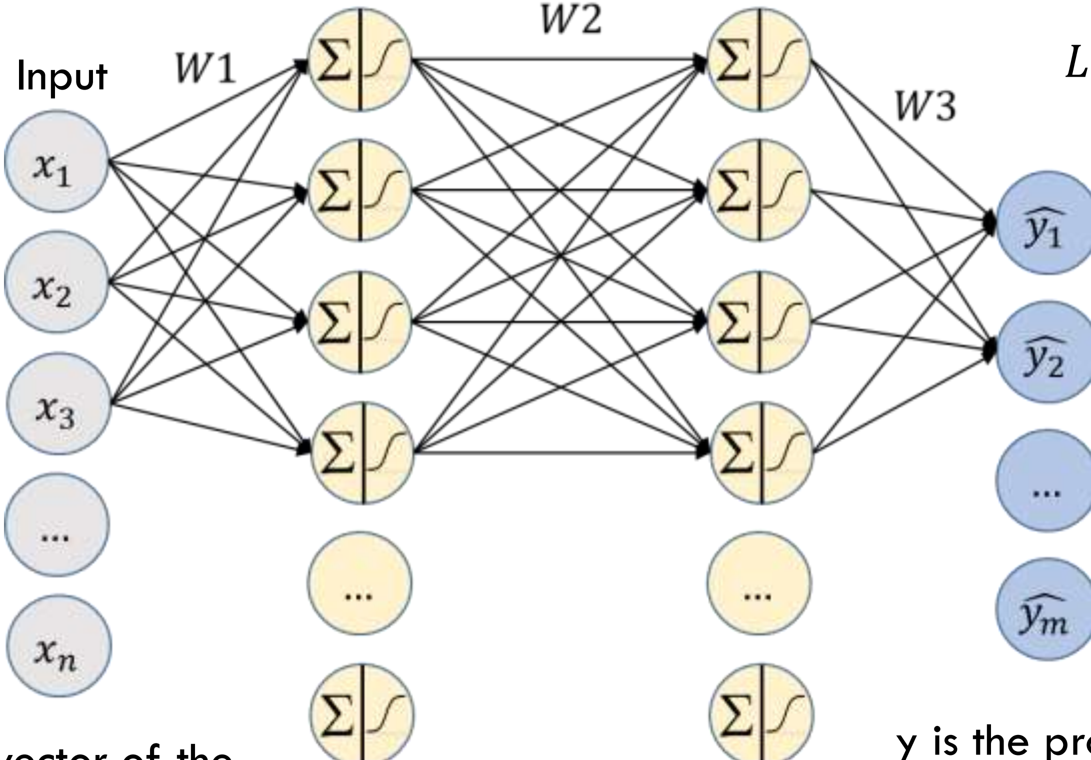


Relu

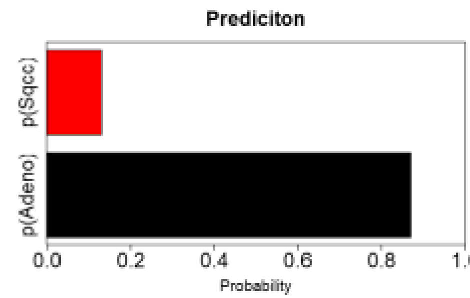
Artificial neural networks



(100x100x3)



$$Loss = -\frac{1}{N}\sum_i^N y_i\,(\log(\hat{y}_i))$$



Example for two classes

x is the vector of the flattened input image

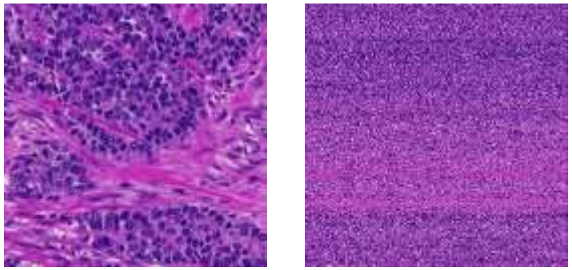
W are the weight matrices that are learned during the training phase of the network

y is the probability vector for each class

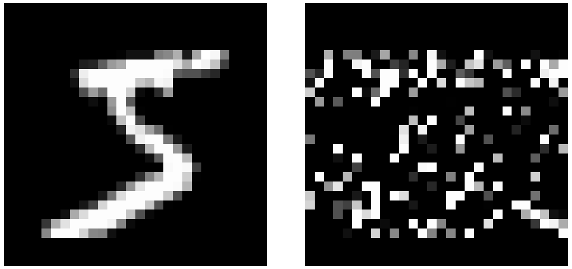
Artificial neural networks

- Two main disadvantages
  - Images are often big and therefore you need a lot of weights (also for the hidden layers)
  - Spatial information of the image gets lost, because of flatten inputs
  - Images left and right are the same for the network

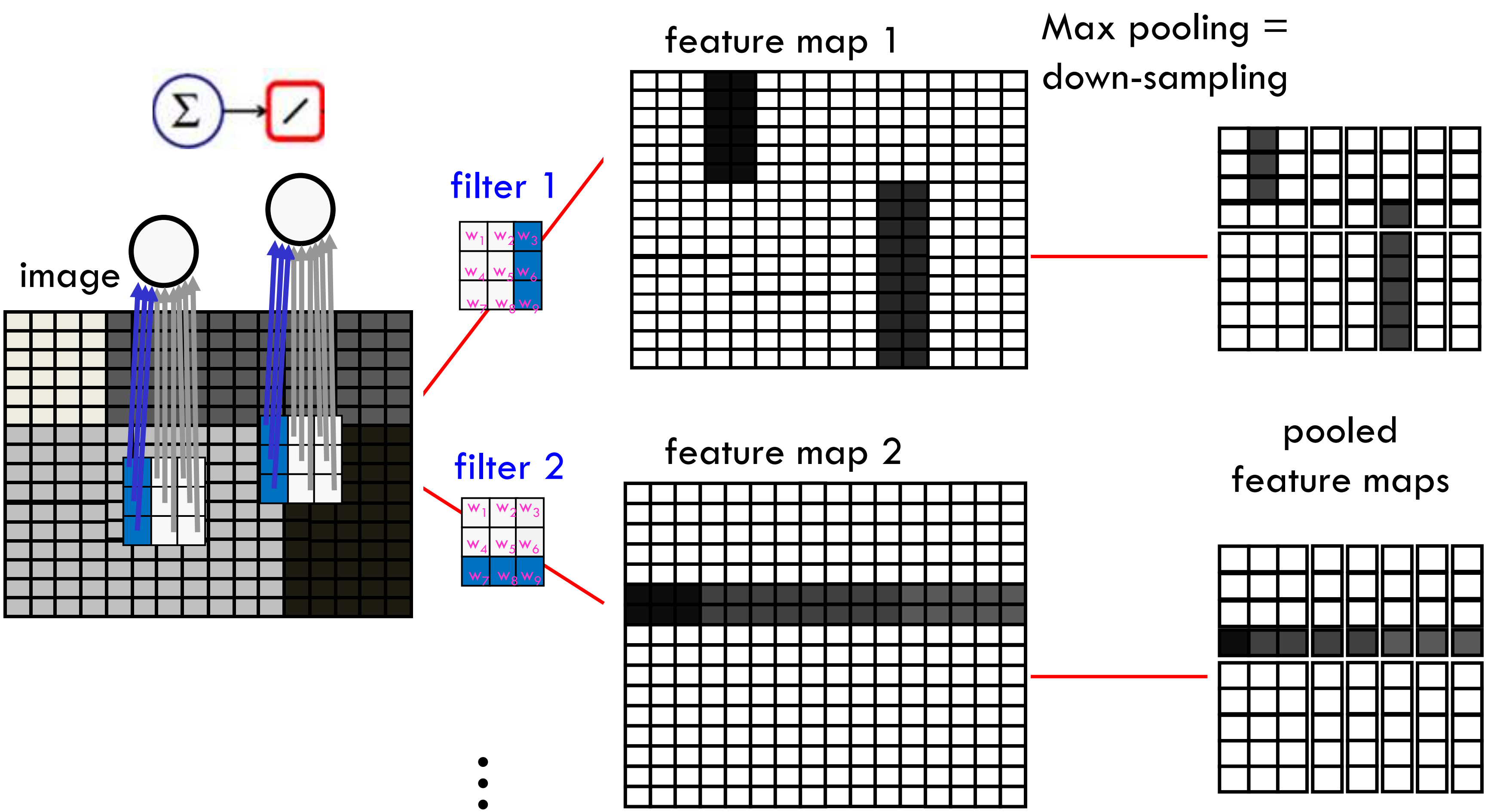
Adeno vs Sqcc example



Mnist example



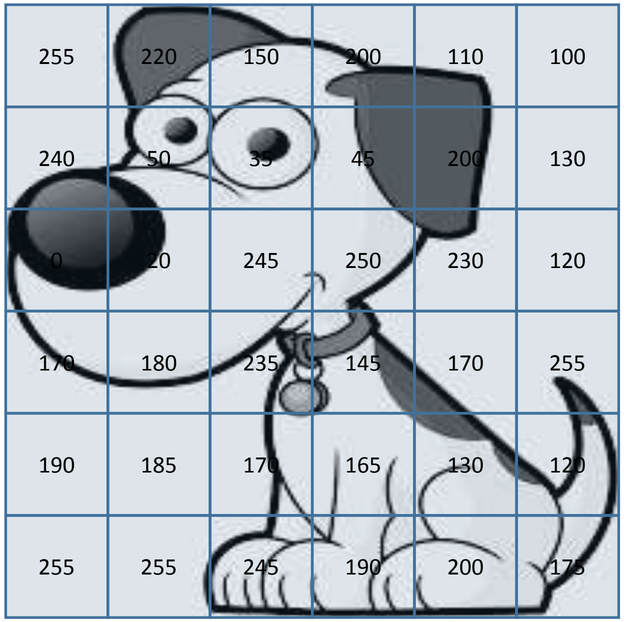
Convolutional neural networks



The weights of each filter are randomly initiated and then adapted during the training.

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	145	170	255
190	185	170	165	130	120
255	255	245	190	200	175

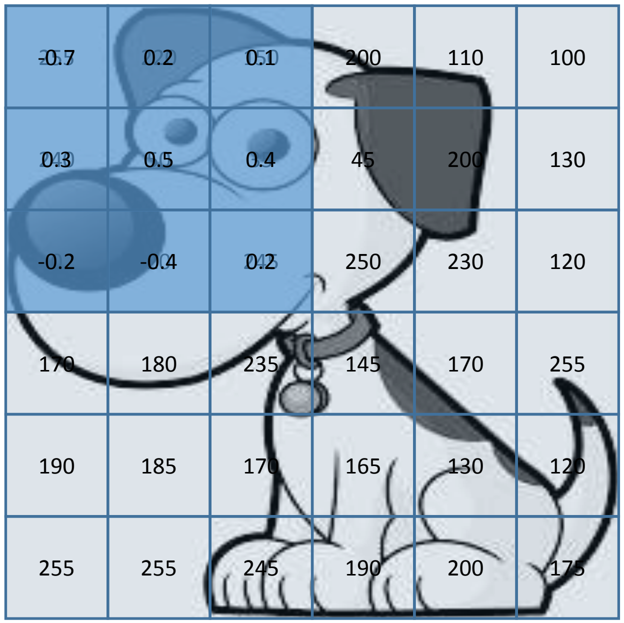
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



Feature map 4x4x1

32.5

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	-0.7	0.2	0.1	110	100
240	0.3	0.5	0.4	200	130
0	-0.2	-0.4	0.2	230	120
170	180	235	145	170	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	-0.7	0.2	0.1	100
240	50	0.3	0.5	0.4	130
0	20	-0.2	-0.4	0.2	120
170	180	235	145	170	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5

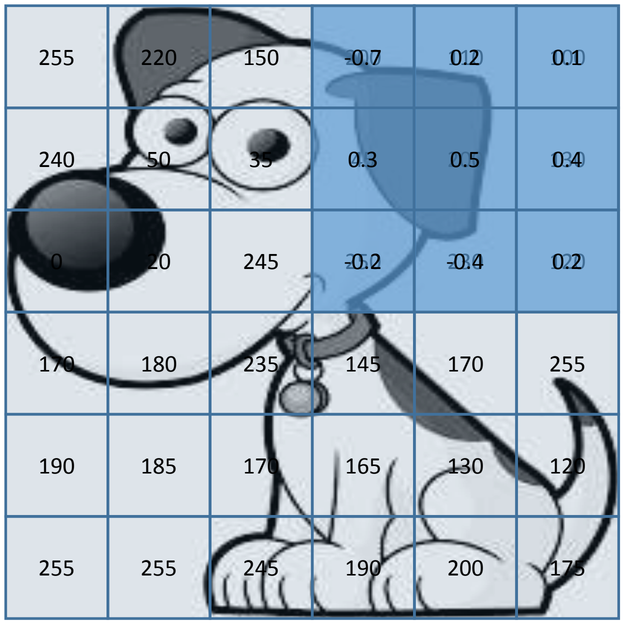
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	-0.7	0.2	0.1
240	50	35	0.3	0.5	0.4
0	20	245	-0.2	-0.4	0.2
170	180	235	145	170	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54

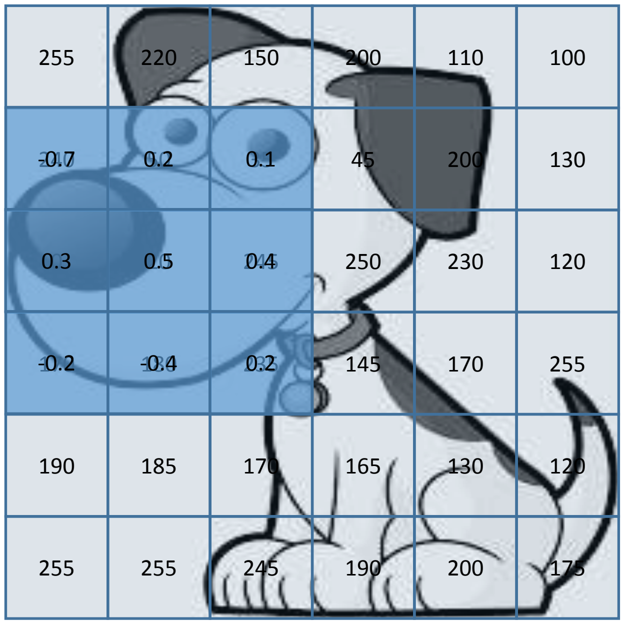
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
-0.7	0.2	0.1	45	200	130
0.3	0.5	0.4	250	230	120
-0.2	-0.4	0.2	145	170	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5			

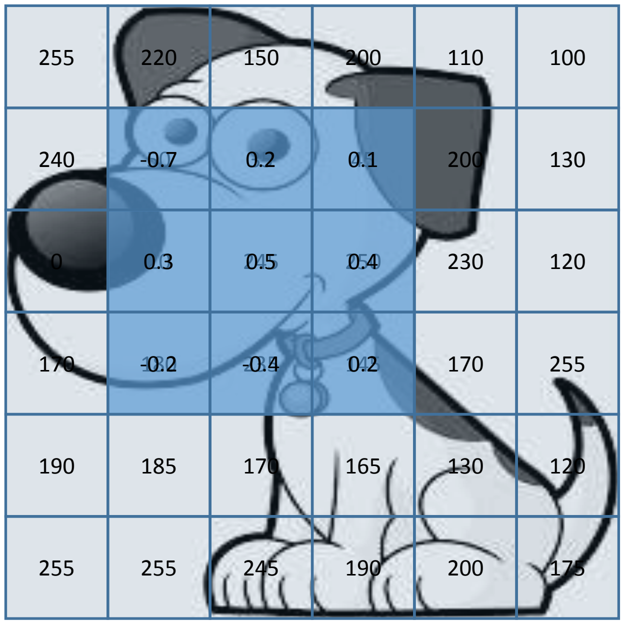
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	-0.7	0.2	0.1	200	130
0	0.3	0.5	0.4	230	120
170	-0.2	-0.4	0.2	170	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104		

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	-0.7	0.2	0.1	130
0	20	0.3	0.5	0.4	120
170	180	-0.2	-0.4	0.2	255
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	

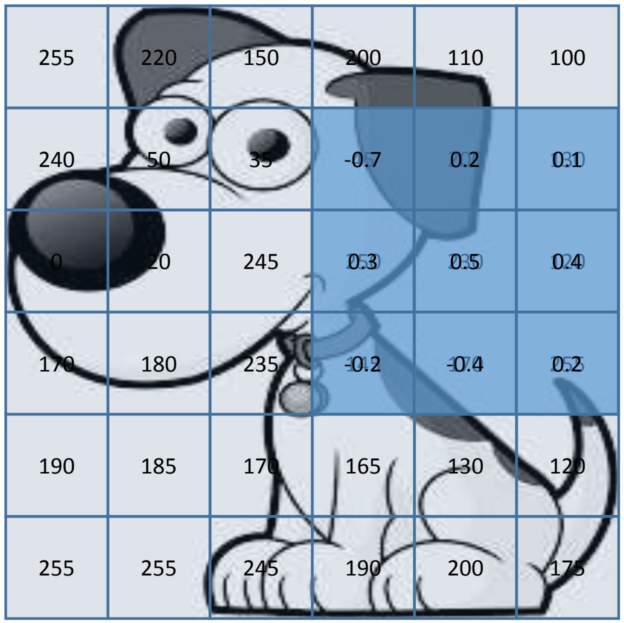
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	-0.7	0.2	0.1
0	20	245	0.3	0.5	0.4
170	180	235	-0.2	-0.4	0.2
190	185	170	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31

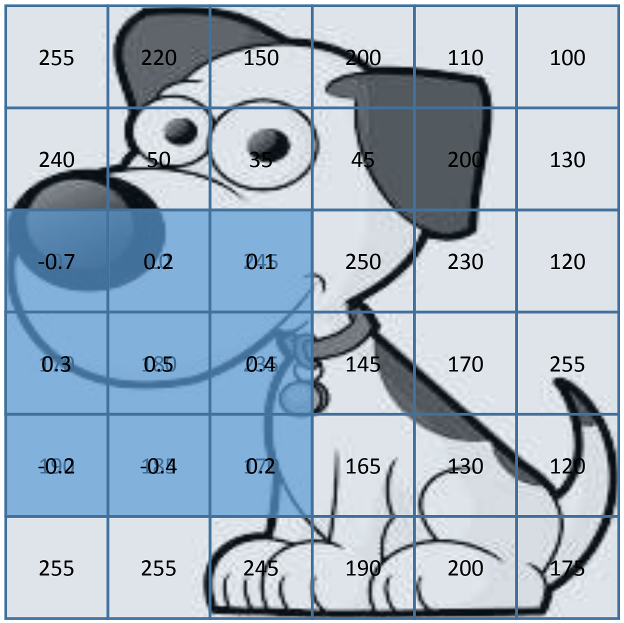
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
-0.7	0.2	0.1	250	230	120
0.3	0.5	0.4	145	170	255
-0.2	-0.4	0.2	165	130	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44			

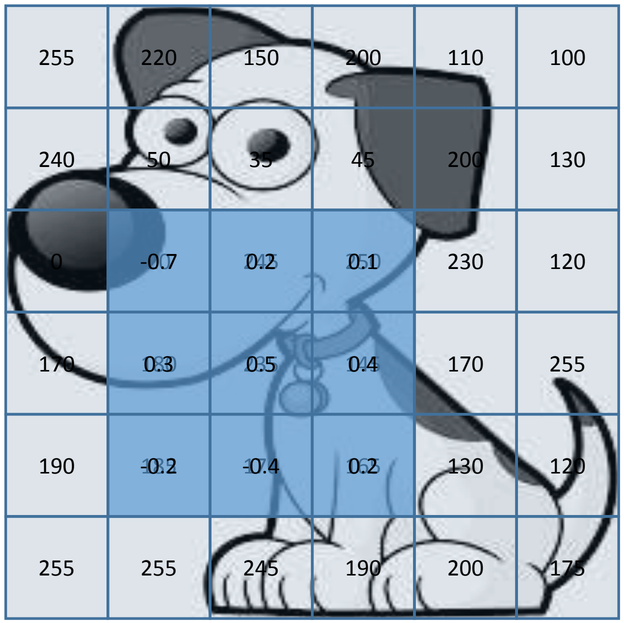
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224		

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	35	45	200	130
0	20	-0.7	0.2	0.1	120
170	180	0.3	0.5	0.4	255
190	185	-0.2	-0.4	0.2	120
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	

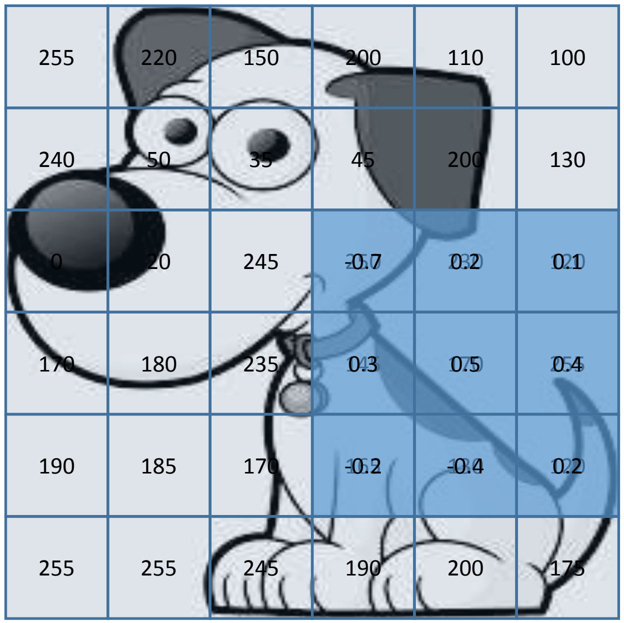
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	-0.7	0.2	0.1
170	180	235	0.3	0.5	0.4
190	185	170	-0.2	-0.4	0.2
255	255	245	190	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18

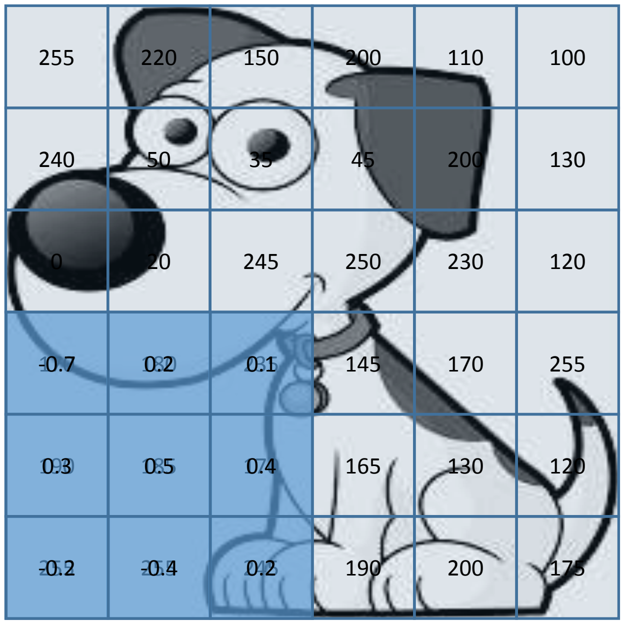
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5			

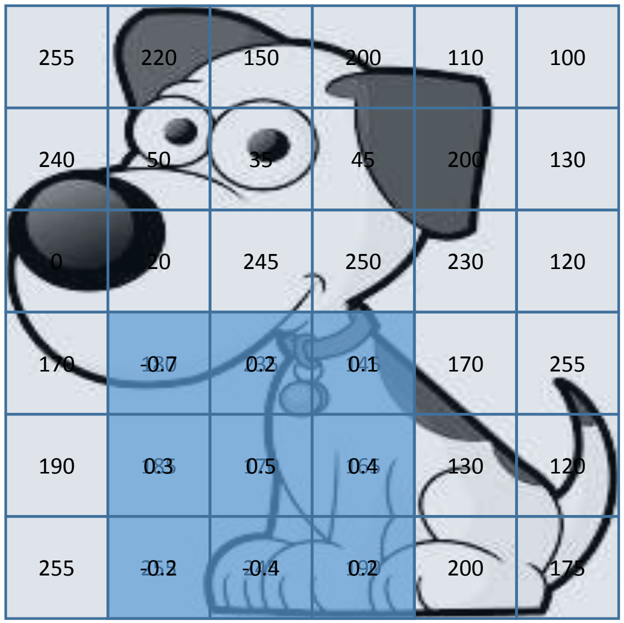
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	-0.7	0.2	0.1	170	255
190	0.3	0.5	0.4	130	120
255	-0.2	-0.4	0.2	200	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5		

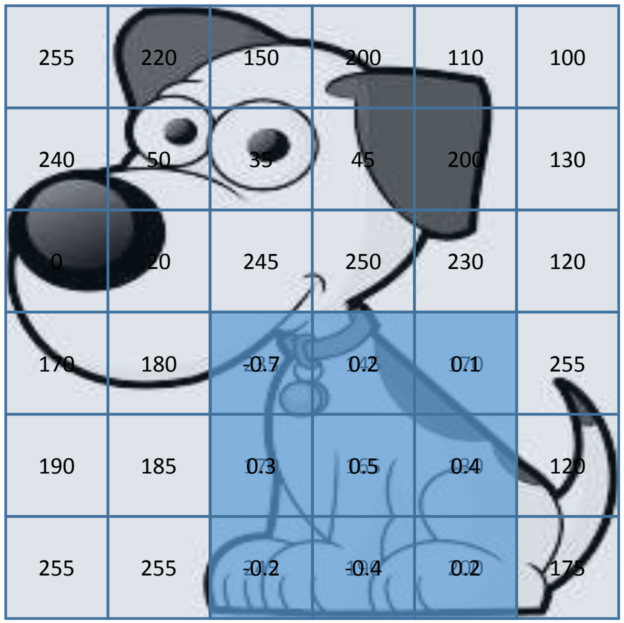
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	-0.7	0.2	0.1	255
190	185	0.3	0.5	0.4	120
255	255	-0.2	-0.4	0.2	175

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	

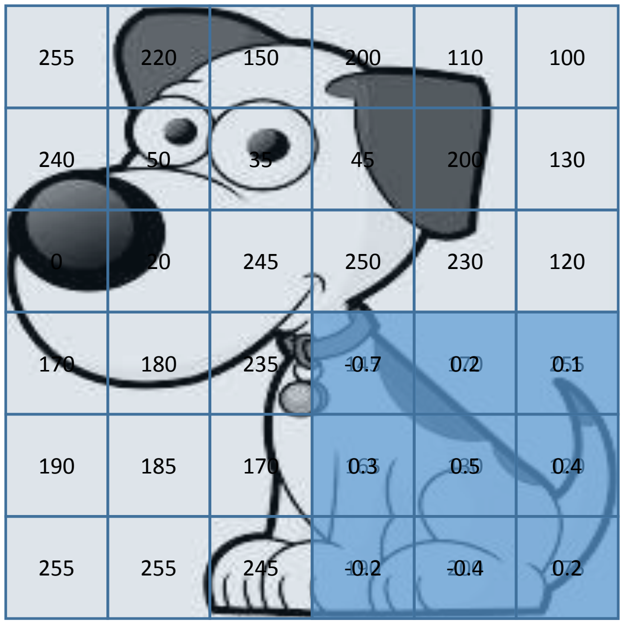
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

Relu

32.5	0	185.5	54
0	104	217.5	31
0	224	38.5	0
0	213.5	52.5	37.5

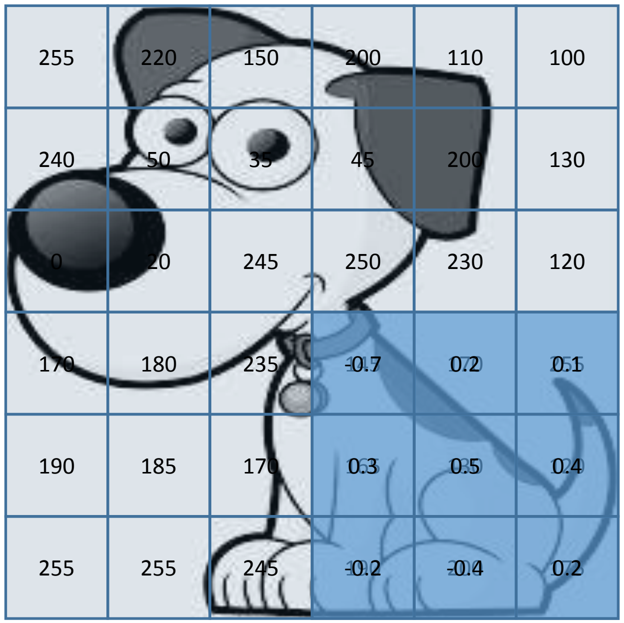
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1



255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

Relu

32.5	0	185.5	54
0	104	217.5	31
0	224	38.5	0
0	213.5	52.5	37.5

Maxpool (2x2x1)

104

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

Relu

32.5	0	185.5	54
0	104	217.5	31
0	224	38.5	0
0	213.5	52.5	37.5

Maxpool (2x2x1)

104	217.5

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

Relu

32.5	0	185.5	54
0	104	217.5	31
0	224	38.5	0
0	213.5	52.5	37.5

Maxpool (2x2x1)

104	217.5
224	

-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Input image 6x6x1

255	220	150	200	110	100
240	50	35	45	200	130
0	20	245	250	230	120
170	180	235	-0.7	0.2	0.1
190	185	170	0.3	0.5	0.4
255	255	245	-0.2	-0.4	0.2

Feature map 4x4x1

32.5	-105.5	185.5	54
-105.5	104	217.5	31
-44	224	38.5	-18
-60.5	213.5	52.5	37.5

Relu

32.5	0	185.5	54
0	104	217.5	31
0	224	38.5	0
0	213.5	52.5	37.5

Maxpool (2x2x1)

104	217.5
224	52.5

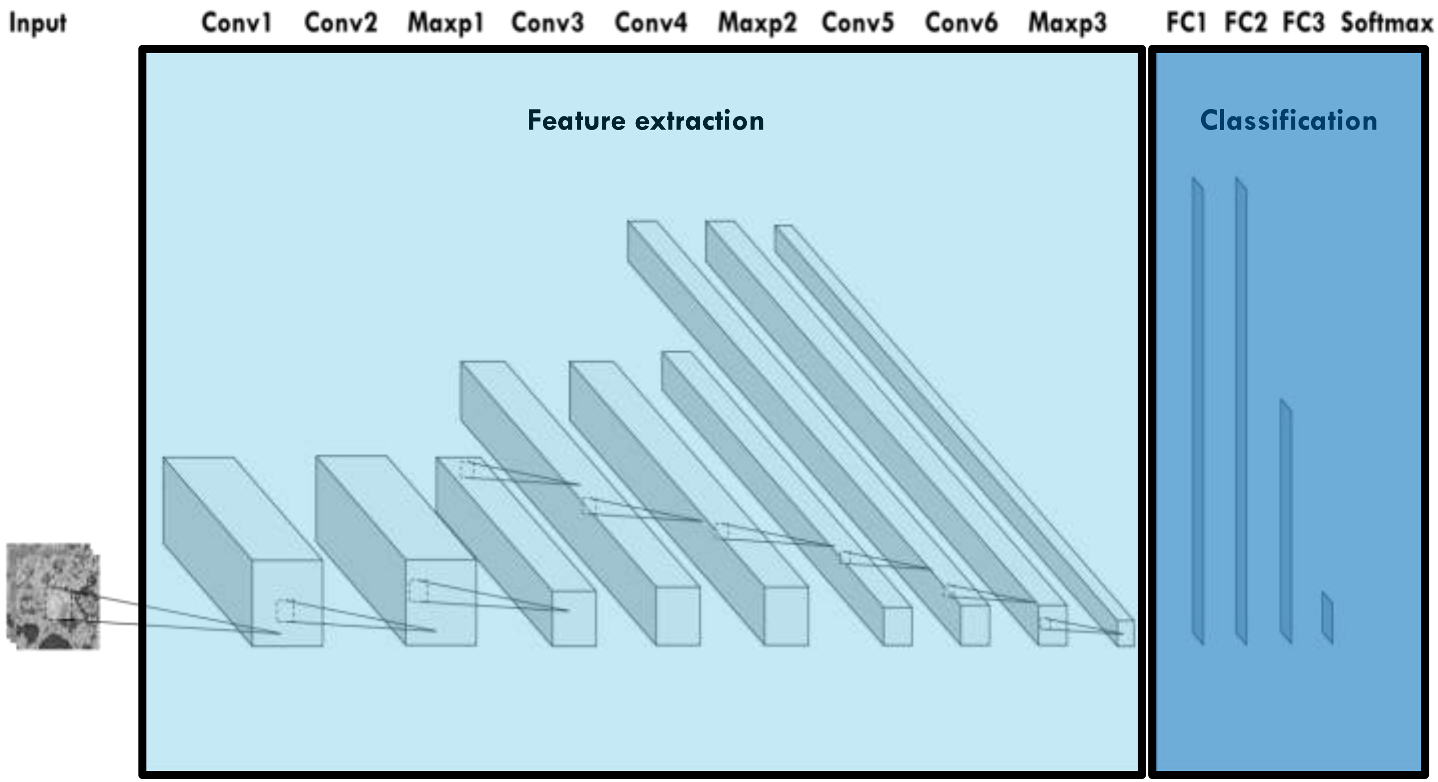
-0.7	0.2	0.1
0.3	0.5	0.4
-0.2	-0.4	0.2

3x3 filter

$$z = b + \sum_i x_i w_i$$

Convolutional neural networks

Typical architecture of a CNN

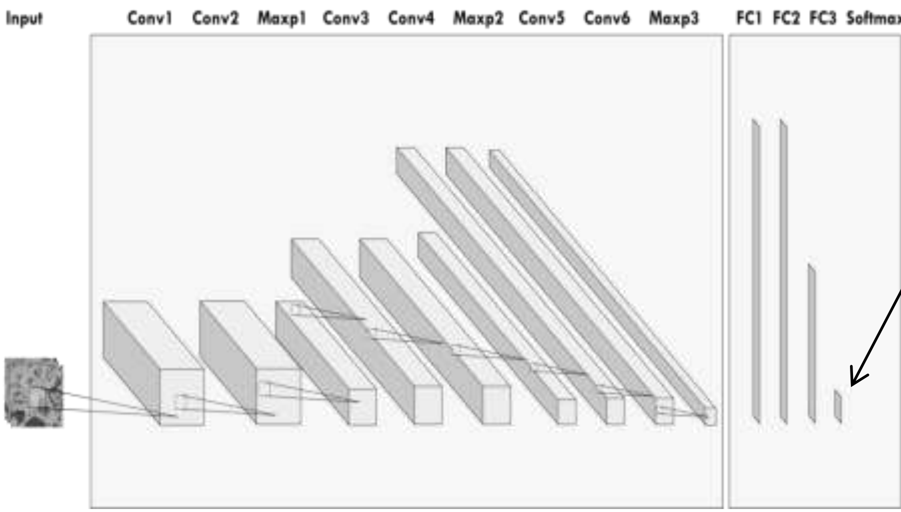


Spatial resolution is decreased e.g. via max-pooling while more abstract image features are detected in deeper layers.

Convolutional neural networks

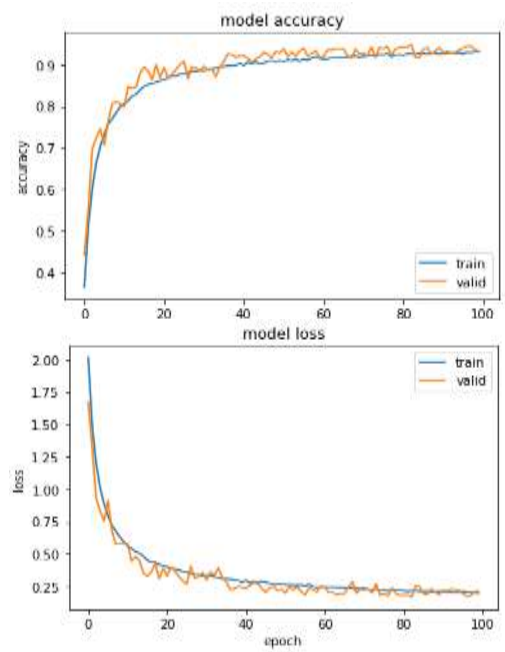
Forward pass

Calculate the output based on the current parameters



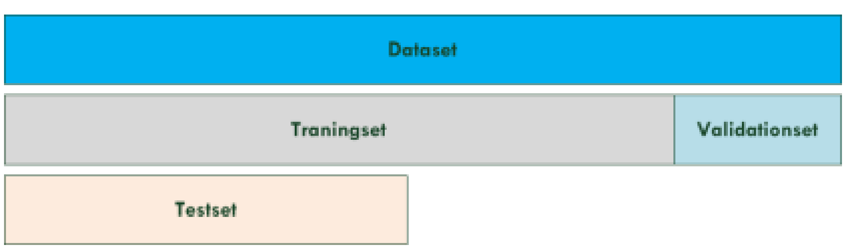
$$Loss = -\frac{1}{N}\sum_{i}^{N} y_i\left(\log(\hat{y}_i)\right)$$

Categorical crossentropy

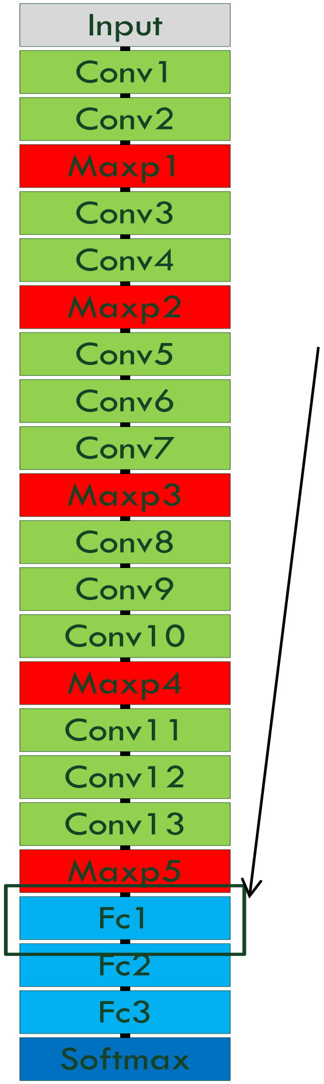


Backward pass

Update the parameters to minimize the loss



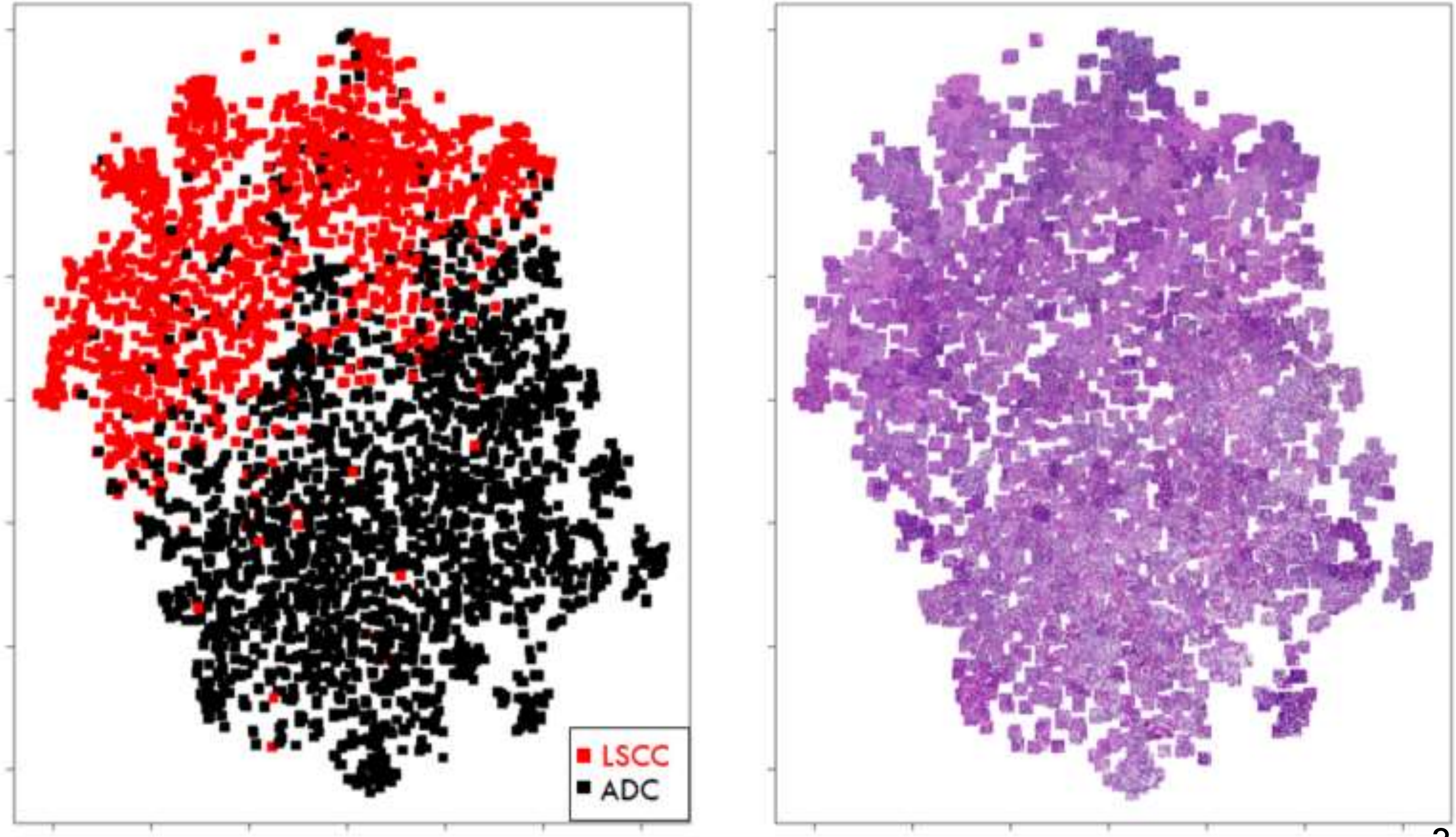
Deep Learning for feature extraction



- Pretrained VGG16 network, trained on ImageNet data as a feature extractor
- 4096 features of the third last layer ("Fc1") before the activation function were extracted for each image
- 2D tsne representation of all images in the training set.
- Very good separation and if you take a close look, you see that similar images are close to each other

Deep Learning for feature extraction

2D tsne representation of ImageNet features



Thank you, Questions?