A gentle introduction to Deep Learning

14.07.2020 | PD Dr. Juan J. Durillo

Agenda

Introduction

Introduction to (Deep) Neural Networks for Machine Learning

Computer Vision as working example

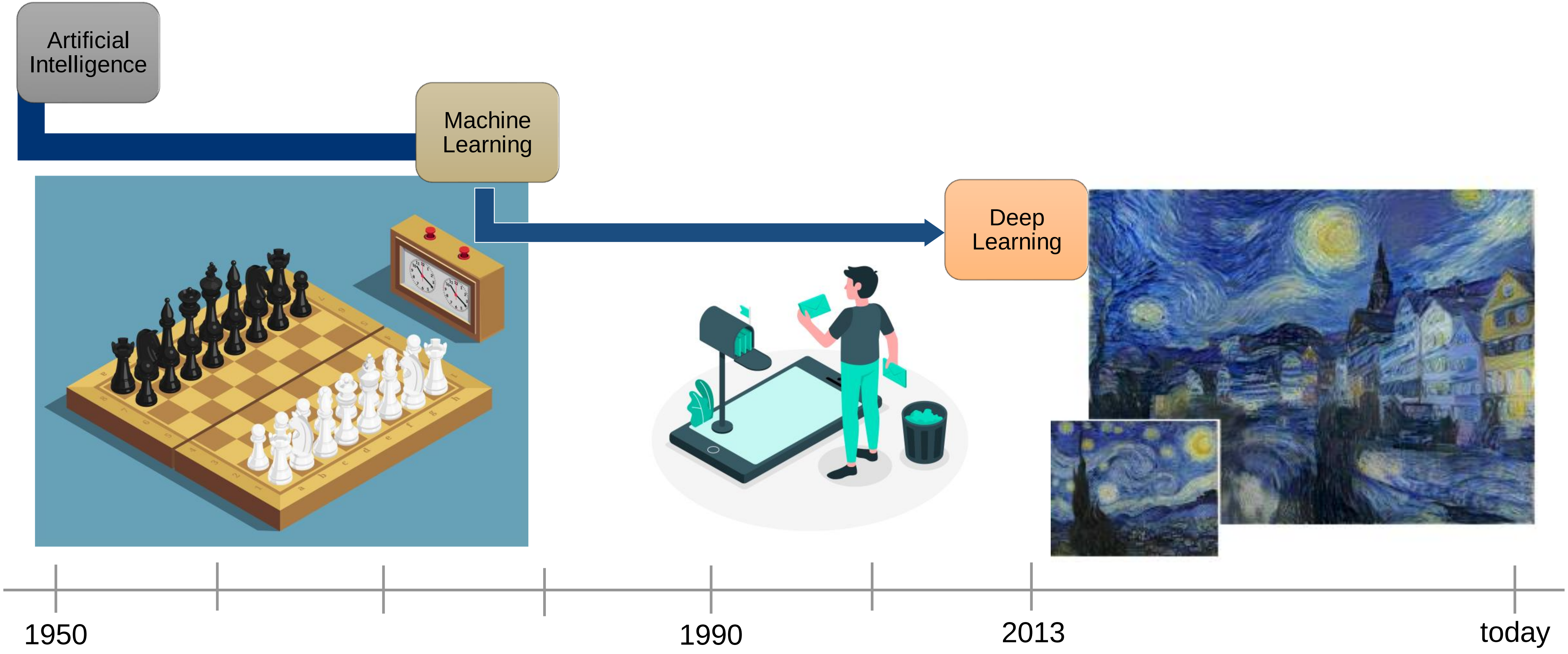
Introduction to Convolutional Neural Networks

Deep Neural Network Architecture

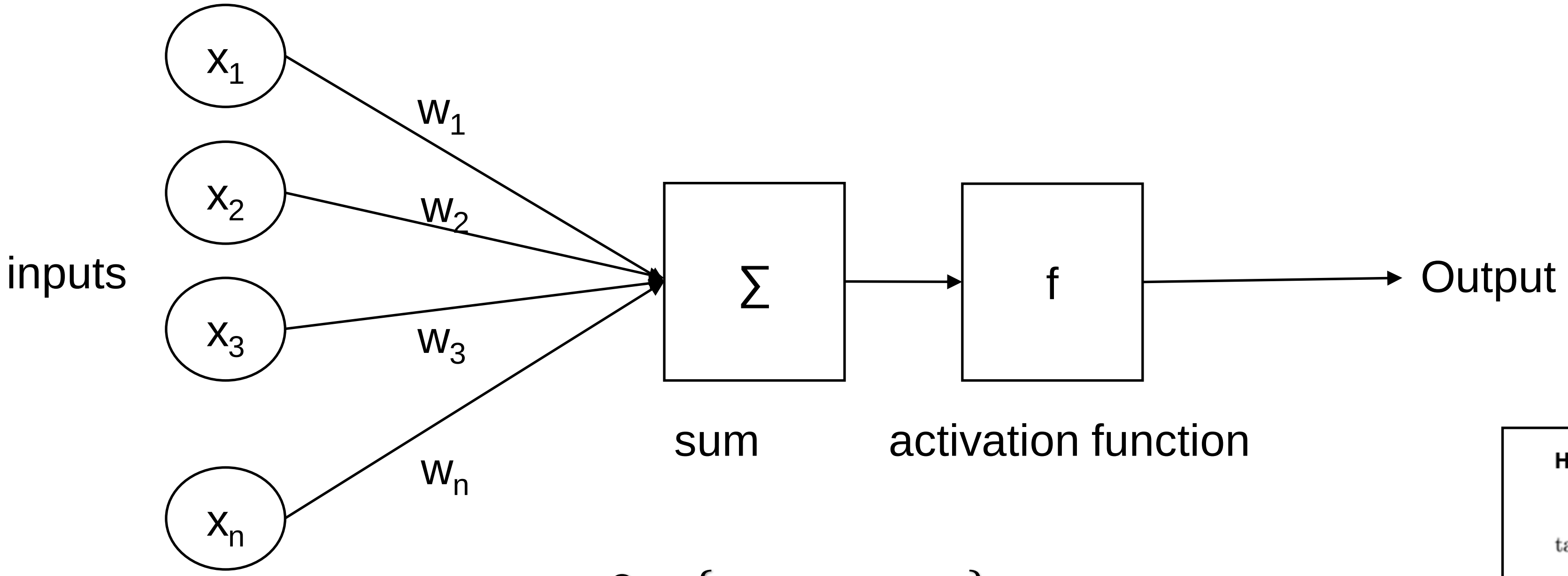
Data Augmentation

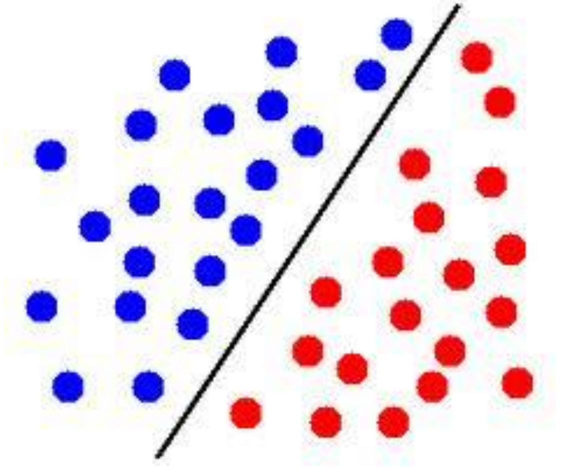
Hands on session: Implementing a Convolutional Neural Network from scratch

Artificial + Intelligence



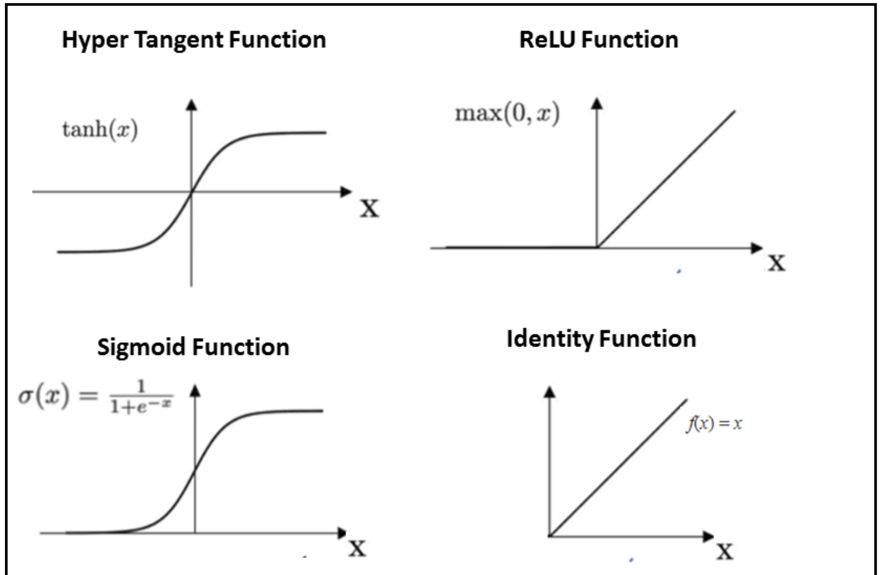
Perceptron - Artificial Neuron





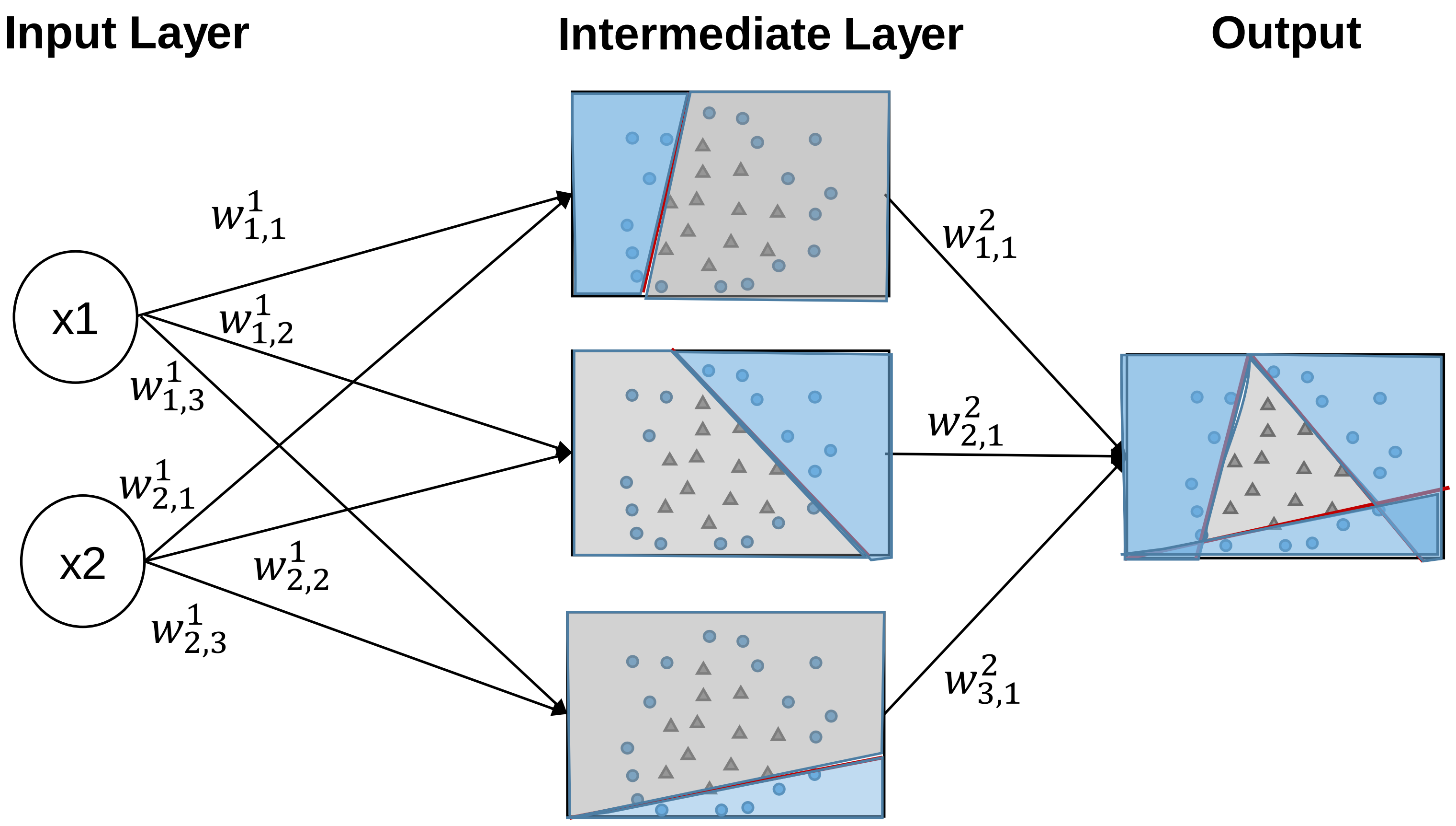
$$\Theta = \{w_1\,, w_2 \ldots, w_n\,\}$$

Single artificial neurons work well for linearly separable datasets (indeed output is the activation effect on a linear combination of the input)



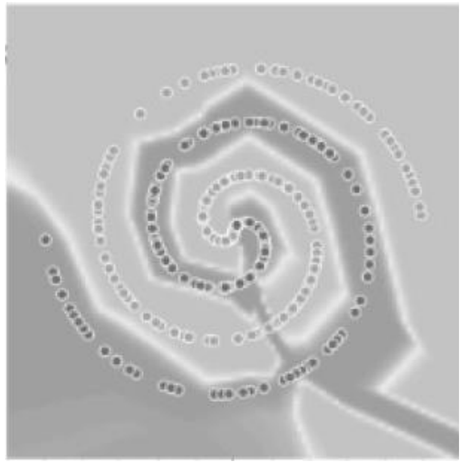
most popular activation functions

Neural Network



$$\Theta = \{w^1_{1,1}, w^1_{1,2}, w^1_{1,3}, w^1_{2,1}, w^1_{2,2}, w^1_{2,3}, w^2_{1,1}, w^2_{2,1}, w^2_{2,3}\}$$

- Works well even when the data is not linearly separable



(Supervised) Learning

- Data domain Z: $X \times \Upsilon$

  X → domain of the input data

  Υ → set of labels (knowledge)

X: 32 x 32 color images

Υ : labels

$\{$truck, car, horse, bird, boat$\}$

Example (CIFAR10 dataset)

- Data Distribution is a probability distribution over a data domain
- Training set $z_1, \ldots, z_n$ from Z assumed to be drawn from the Data Distribution D
- Validation set $v_1, \ldots, v_m$ from Z also assumed to be drawn from D
- A machine learning model is a function that given a set of parameters Θ and z from Z produces a prediction
- The prediction quality is measured by a differentiable non-negative scalar-valued loss function, that we denote $\ell(\Theta; z)$

(Supervised) Learning

- Given Θ we can define the expected loss as: $L(\Theta) = \mathbb{E}_{z \sim D}[\ell(\Theta; z)]$

- Given D, ℓ, and a model with parameter set Θ, we can define learning as:

  "The task of finding parameters Θ that achieve low values of the expected loss, while we are given access to only n training examples"

- The mentioned task before is commonly referred to as *training*

- Empirical average loss given a subset of the training data set $S(z_1, \ldots, z_n)$ as:

$$\hat{L}(\Theta) = \frac{1}{n} \sum_{t=1}^{n} [\ell(\Theta; z_t)]$$

— Usually a proxy function, easier to understand by humans, is used for describing how well the training is performed (e.g., accuracy)

(Supervised) Learning

- The dominant algorithms for training neural networks are based on mini-batch stochastic gradient descent (SGD)

- Given an initial point Θ_0 SGD attempt to decrease $\hat{L}$ via the sequence of iterates

$$\Theta_t \leftarrow \Theta_{t-1} - n_t g(\Theta_{t-1}; B_t)$$

$$g(\Theta; B) = \frac{1}{|B|} \sum\nolimits_{z \epsilon B} \nabla \ell(\Theta; z)$$

Definitions

B_t: random subset of training examples

n_t: positive scalar (learning rate)

epoch: update the weights after going over all training set

Computer Vision

Why? Focus on a kind of Deep Neural Network called Convolutional Neural Network (CNN)

CNNs ability to extract multi-scale localized spatial features and compose them to construct highly expressive representations led to breakthroughs in almost all machine learning areas

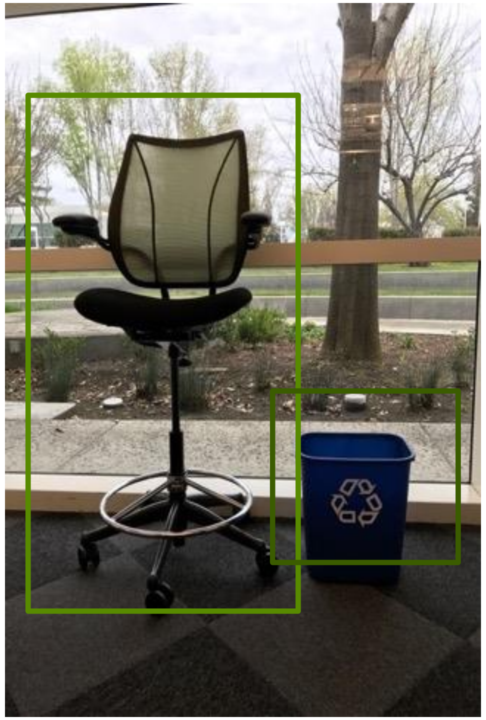
COMPUTER VISION TASKS

predicting the type or class of an object in an image

Image Classification

predicting the type or class on an object in an image and draw a bounding box around it

Image Classification + Localization



predicting the location of objects in an image via bounding boxes and the classes of the located objects

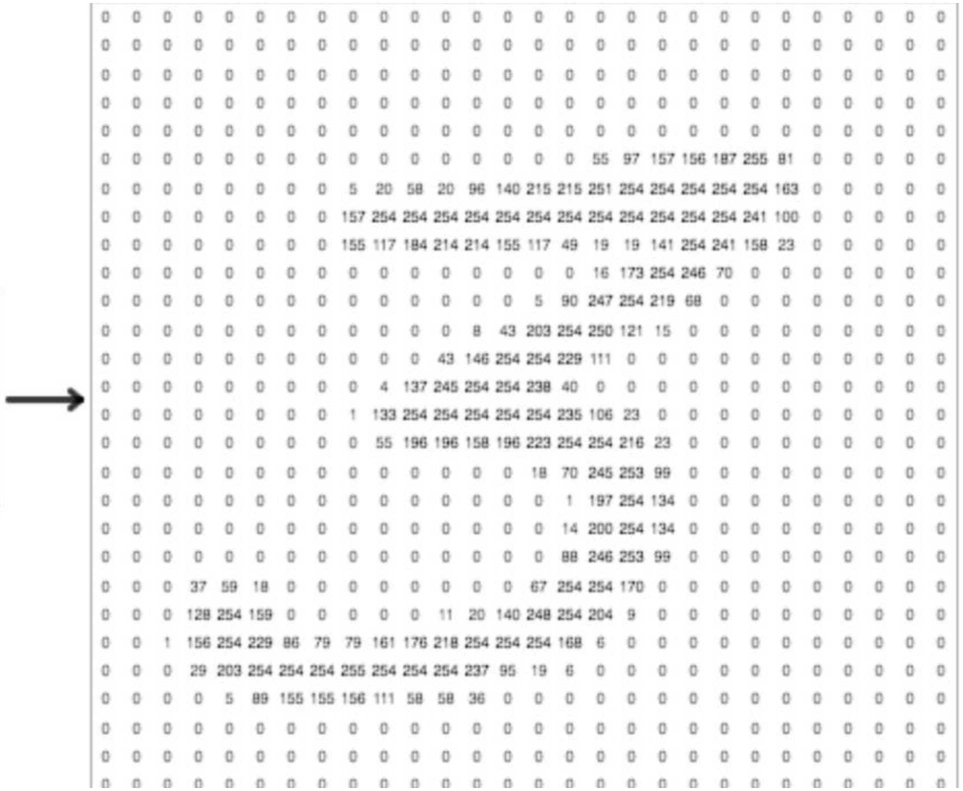
Object Detection

predicting the class to which each pixel in the image belongs to

Image Segmentation

On Input Representation

28 x 28
= 784 pixels



image

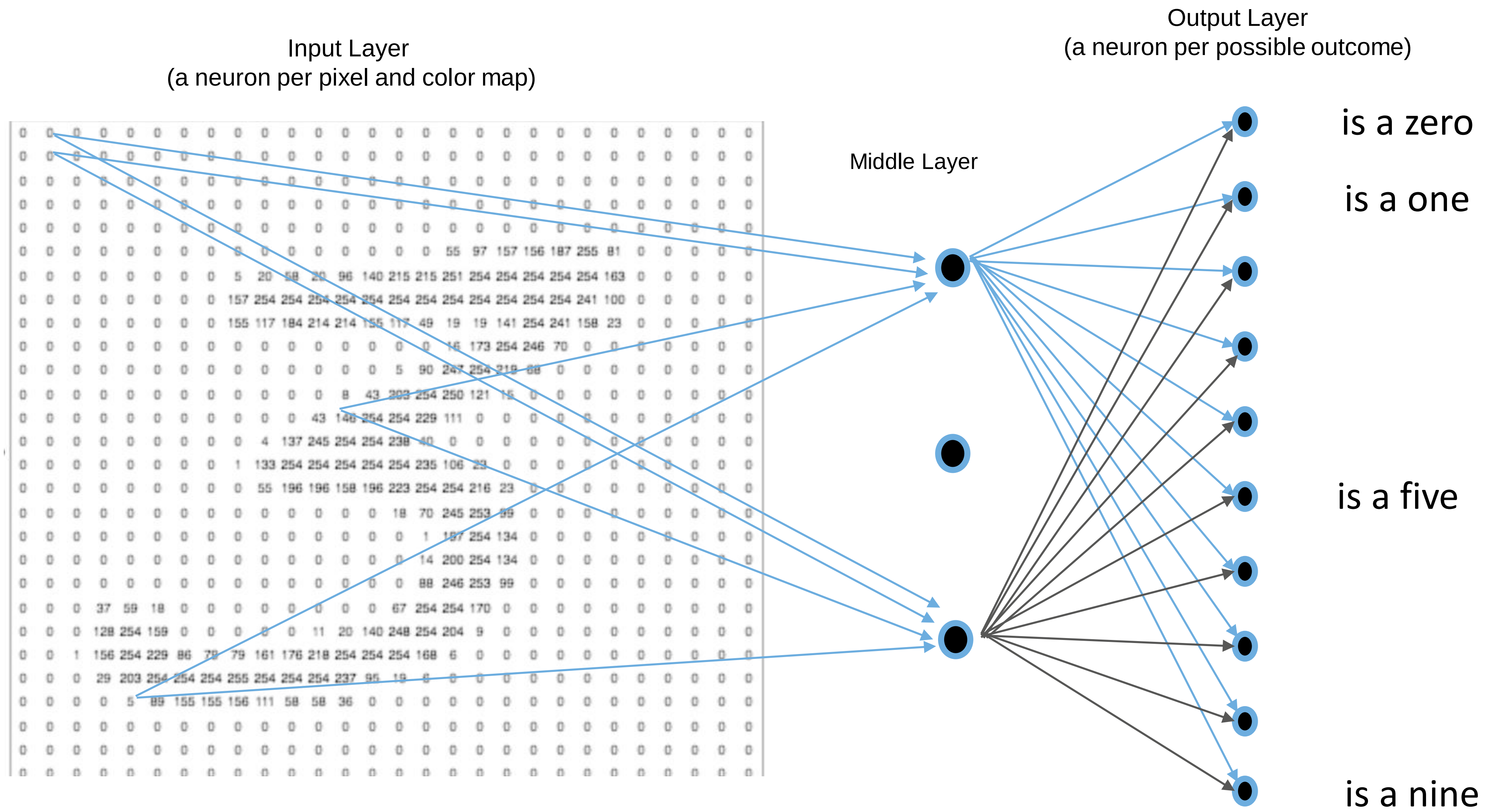
```
dict=['EOS','a','my','sleeps','on','dog','cat','the','bed','floor']
```

sentence = ['a', 'dog', 'sleeps', 'on', 'the', 'floor', 'EOS']

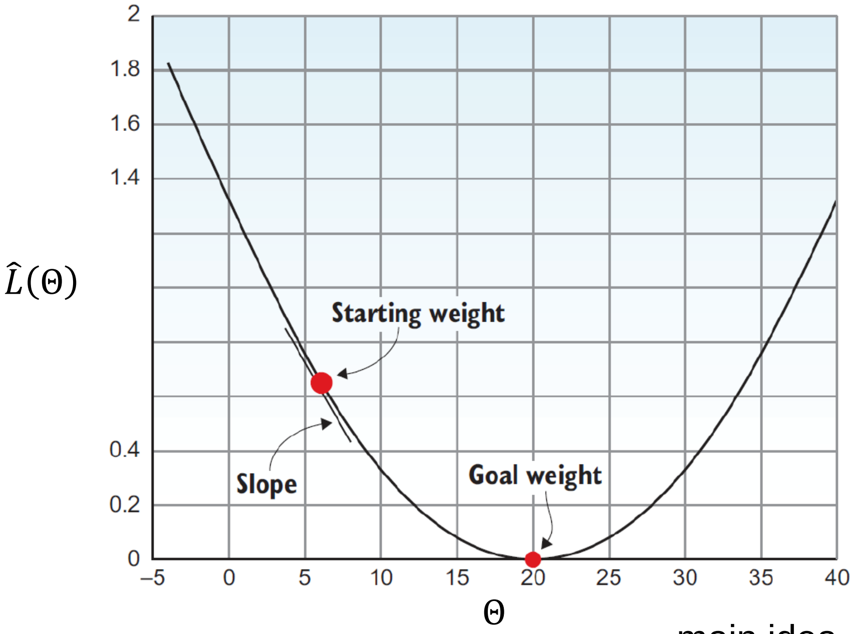
```
[[ 0.  1.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  1.  0.  0.  0.  0.]
 [ 0.  0.  0.  1.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  1.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  1.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  1.]
 [ 1.  0.  0.  0.  0.  0.  0.  0.  0.  0.]]
```

language

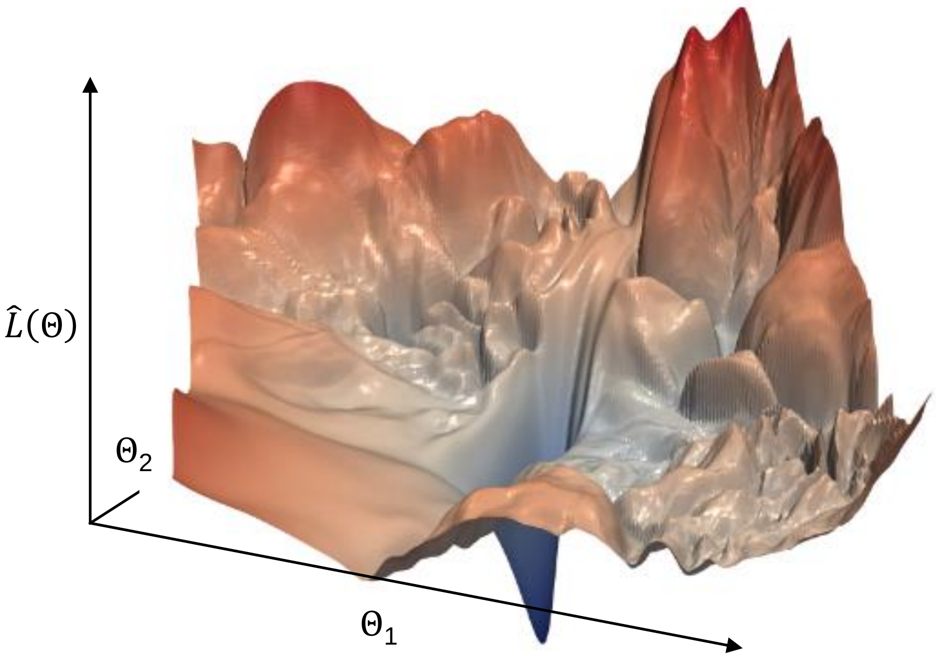
Neural Networks for Image Classification



Training Neural Networks



main idea



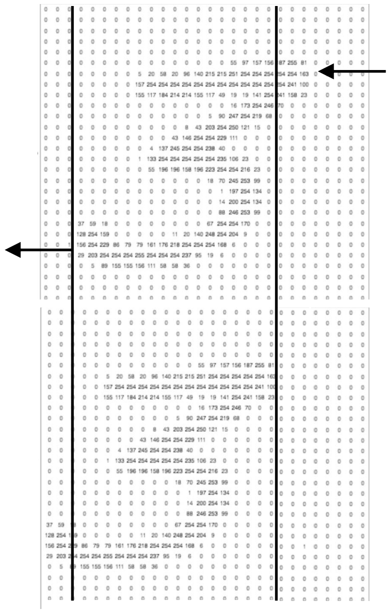
how the surface looks like in reality

<u>Stochastic Gradient Descent</u>

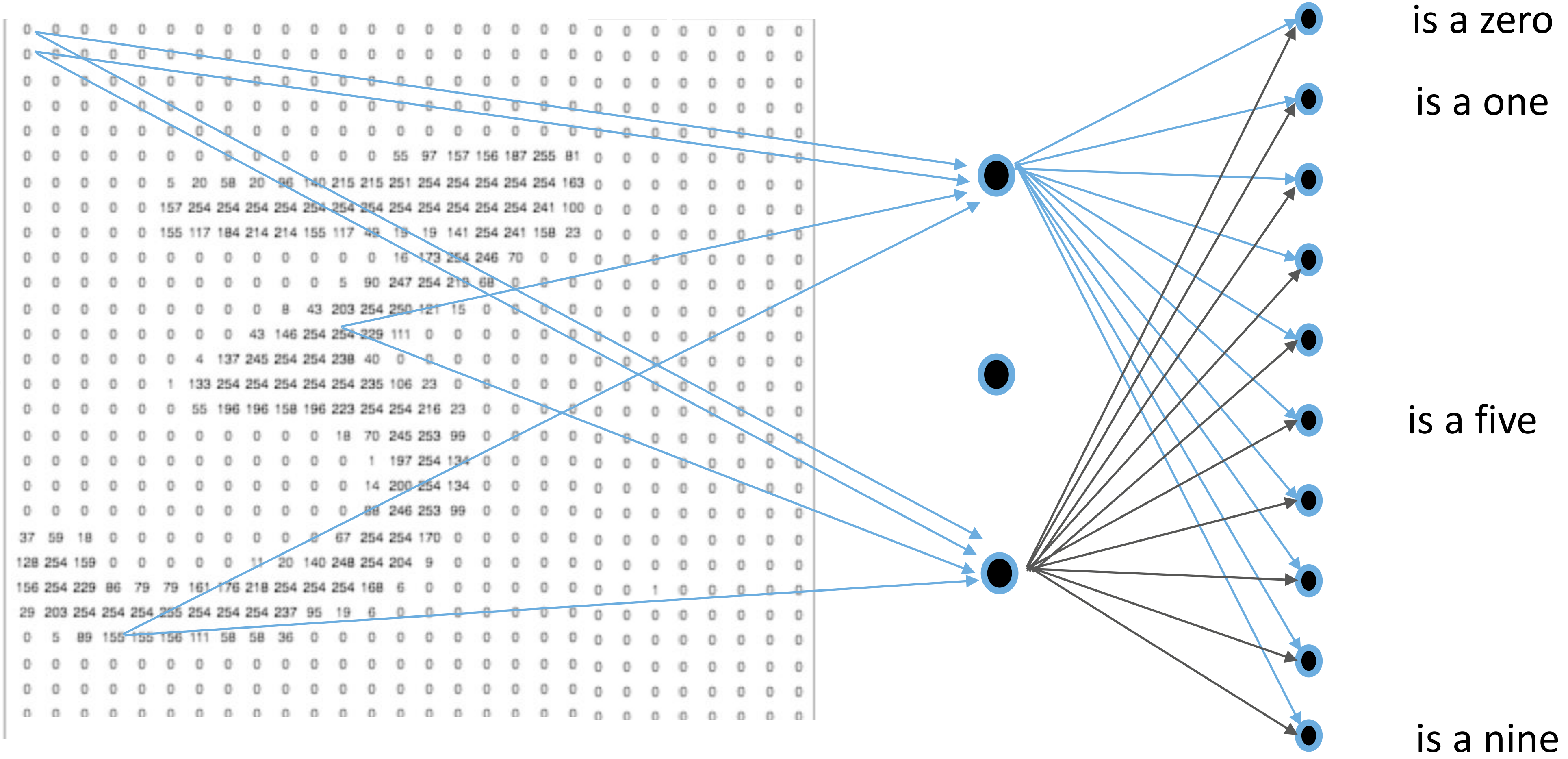
$$\Theta_t \leftarrow \Theta_{t-1} - \eta_t g(\Theta_{t-1}; B_t)$$

$$g(\Theta; B) = \frac{1}{|B|} \sum_{z \epsilon B} \nabla \ell(\Theta; z)$$

Neural Networks for Image Classification

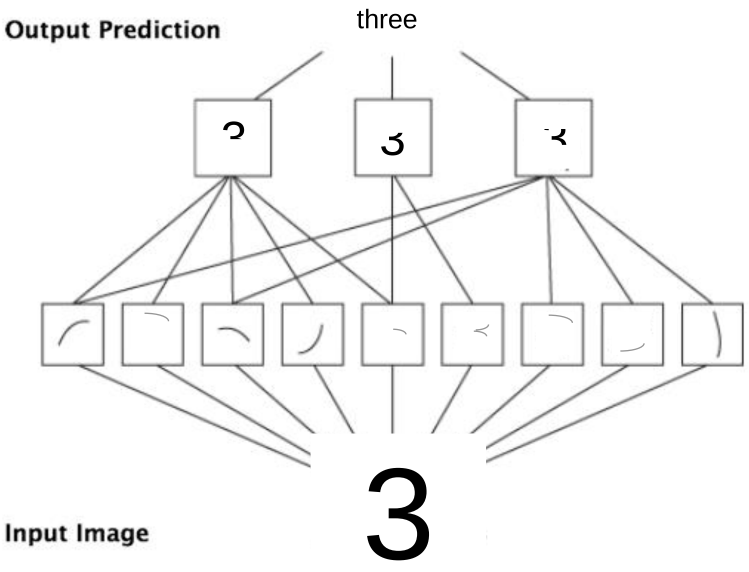


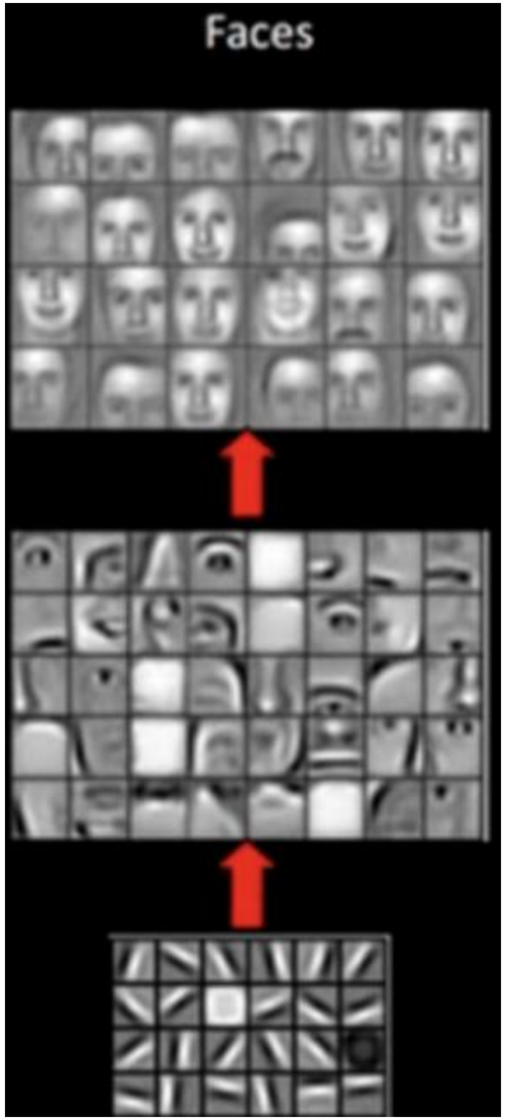
shift to the left



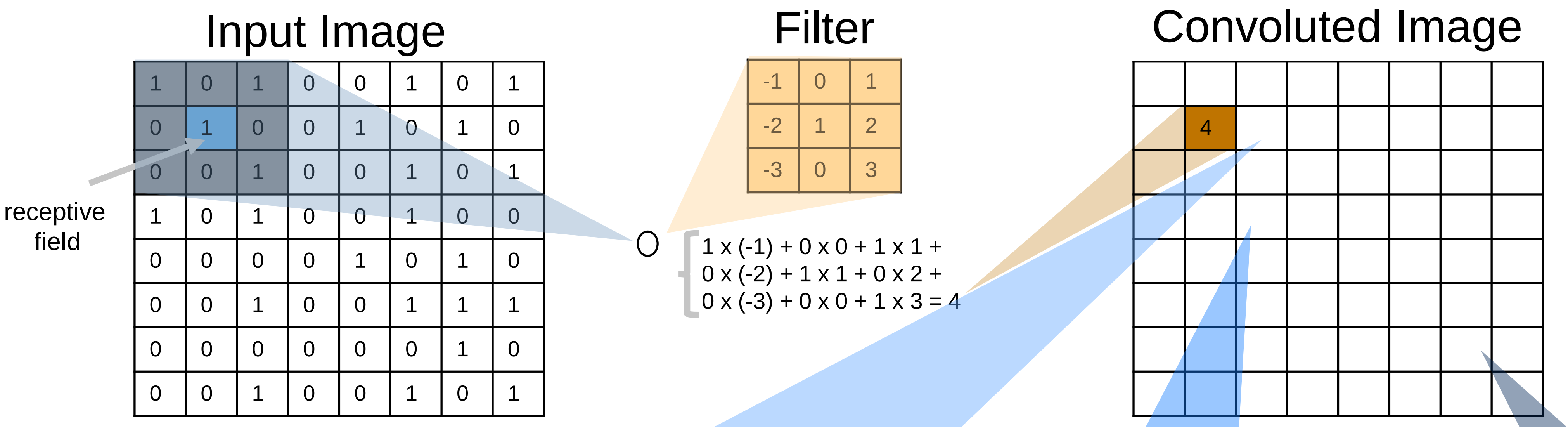
No More Feature Engineering

Co





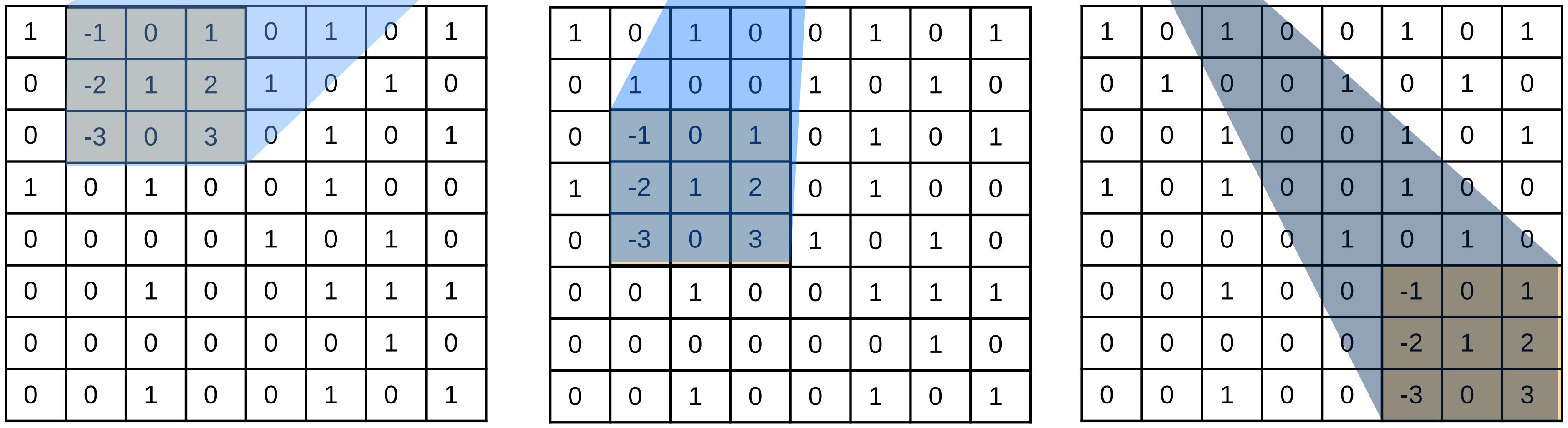
Learning features from data: Convolutions



Filter is convoluted with all the pixels of the image

How many units the filter moves horizontally or vertically is called **stride** and can be different in both dimensions

The stride defines the size of the convoluted image



Filters

Input Image:

Can we get only vertical lines out of this picture?

1	0	-1

filter 1

1	0	-1
1	0	-1
1	0	-1

filter 2

1	0	0	0	-1
1	0	0	0	-1
1	0	0	0	-1
1	0	0	0	-1
1	0	0	0	-1

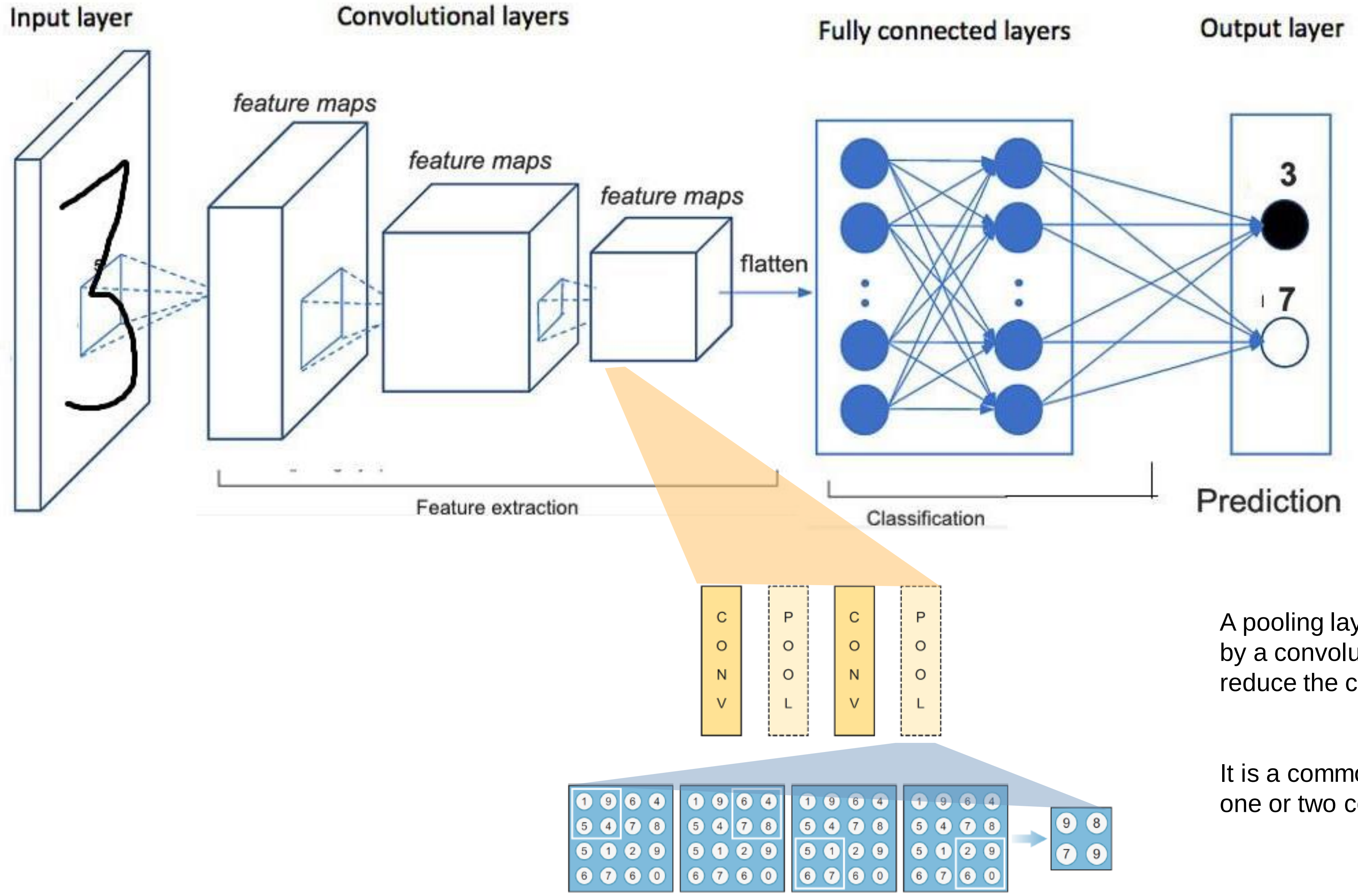
filter 3

try the code yourself (in octave)!

```
 I=imread(<path-to-image>);
GRAY=rgb2gray(I)
FILTER=[ 1 0 -1; 1 0 -1; 1 0 -1]; % filter 2
CONVOLUTED=conv2(GREY,FILTER);
Imwrite(CONVOLUTED, <path-to-result>);
```

*The London skyline image is designed by Freepik

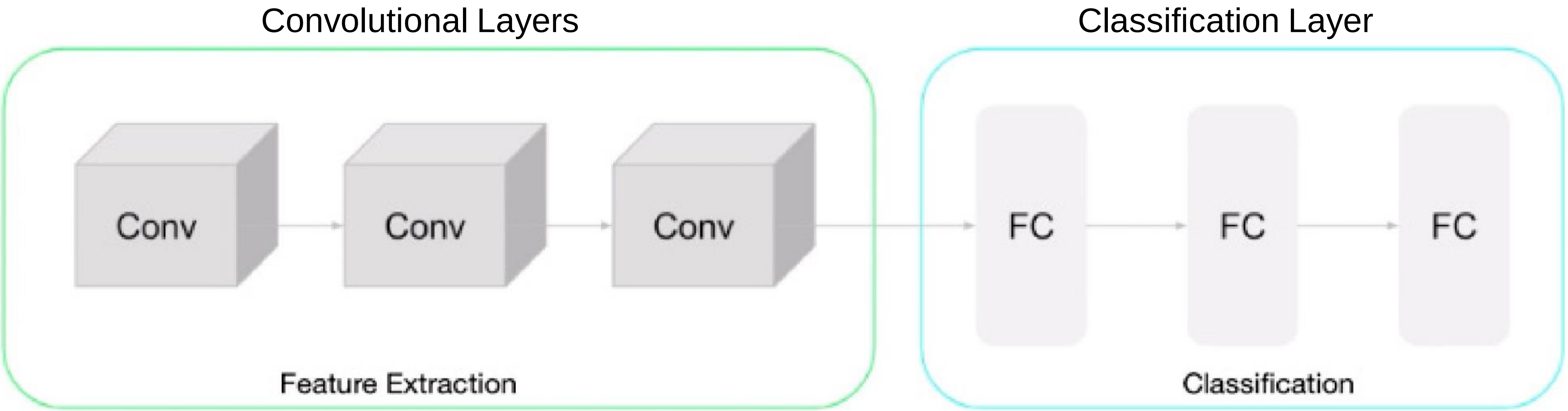
Convolutional Neural Networks (CNN)



A pooling layer down sample the feature maps produced by a convolution into smaller number of parameters to reduce the computational complexity.

It is a common practice to add pooling layers after each one or two convolutions layers in the CNN architecture.

CNN Architecture: A Common Pattern and its Influence

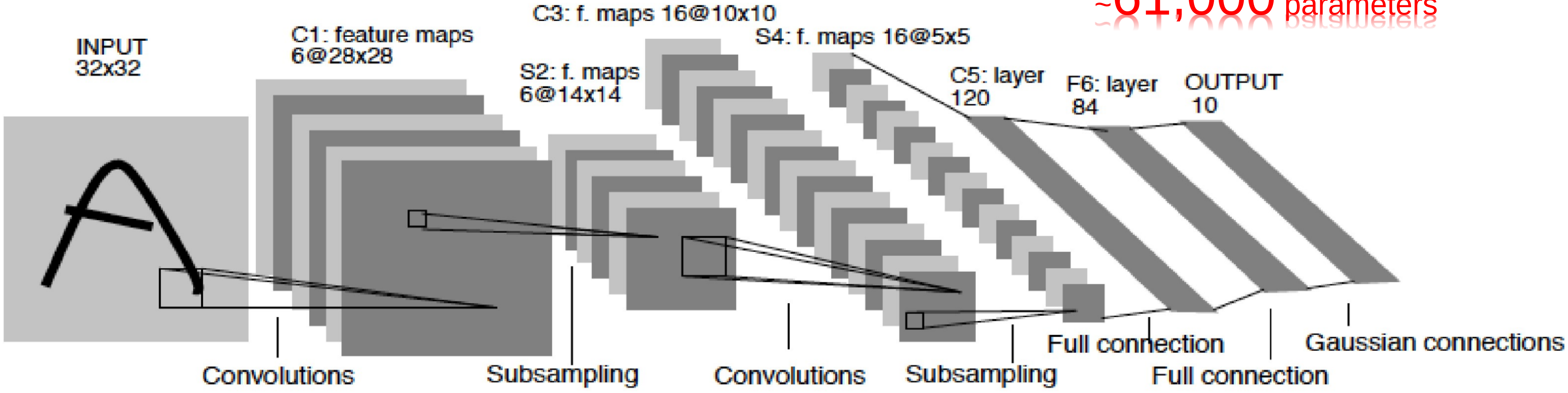


The execution time required during a forward pass through a neural network is bounded from below by the number of floating point operations (FLOPs).

This FLOP count depends on the deep neural network architecture and the amount of data.

LeNet Architecture

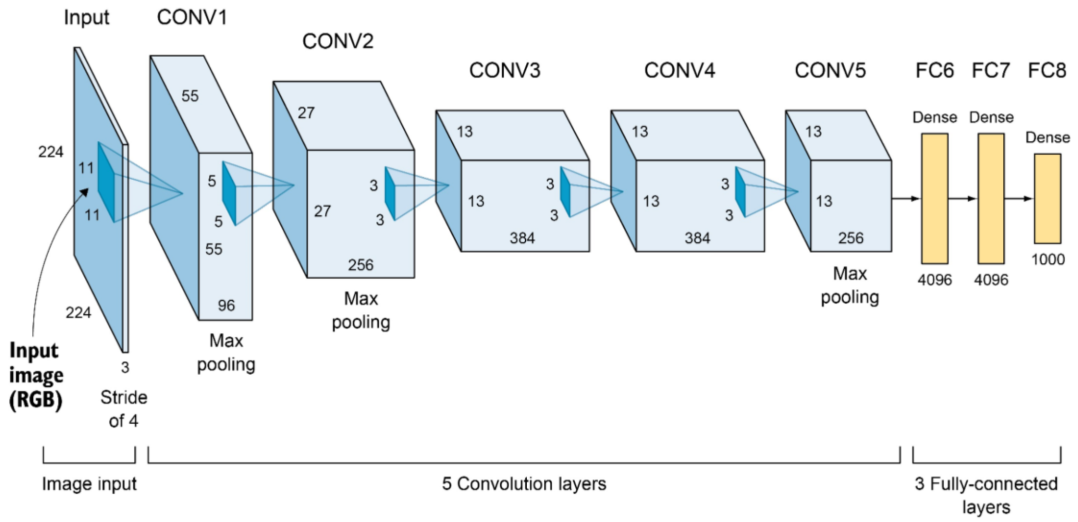
~61,000 parameters



Architecture summary :

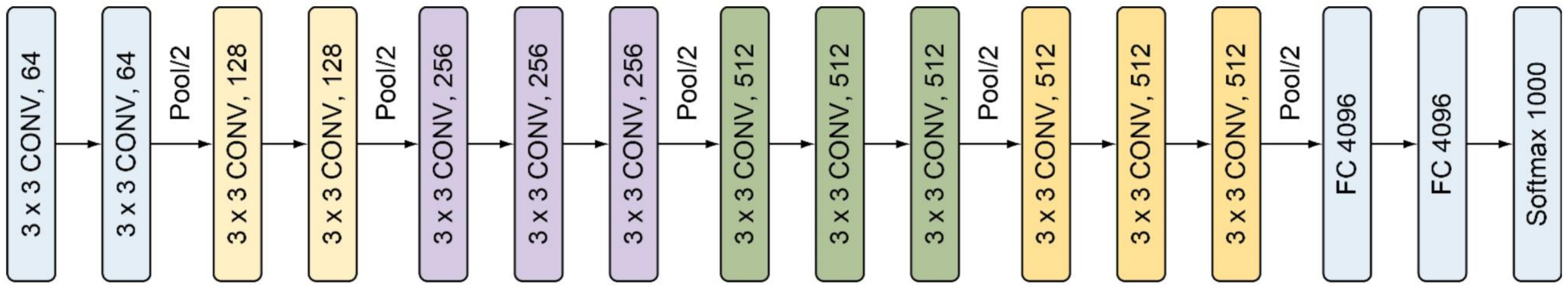
- 3 convolutional layers filters in all the layers equal to 5x5
  (layer 1 depth = 6, layer 2 depth = 16, layer 3 depth = 120)
- As activation function the tanh function is used

AlexNet and VGG Architectures



AlexNet

~60,000,000 parameters

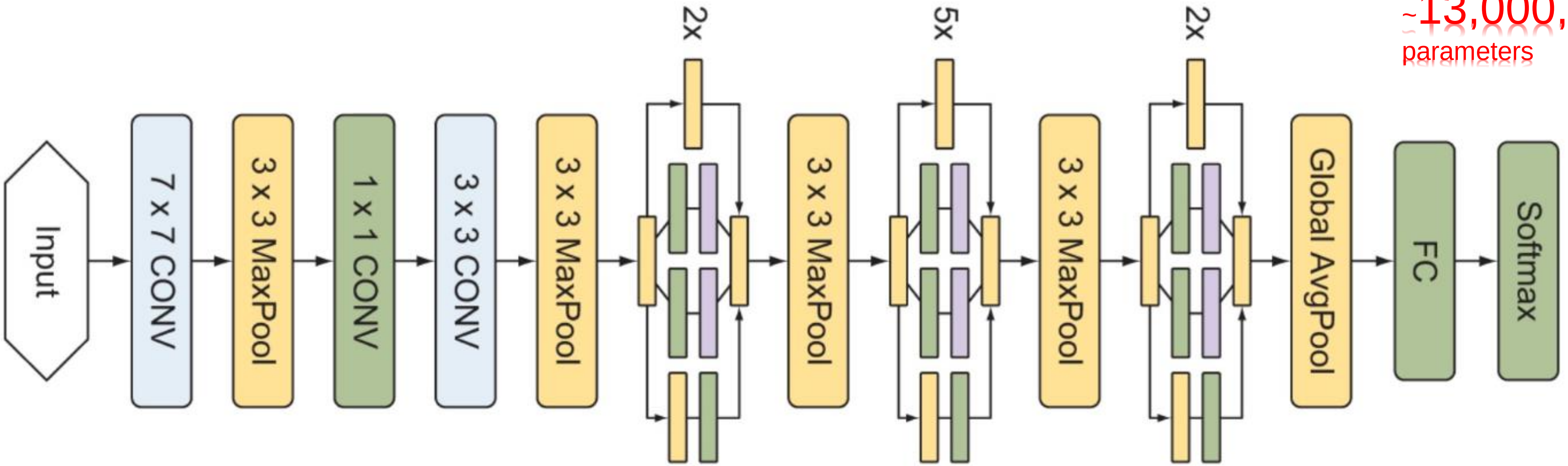


VGG16

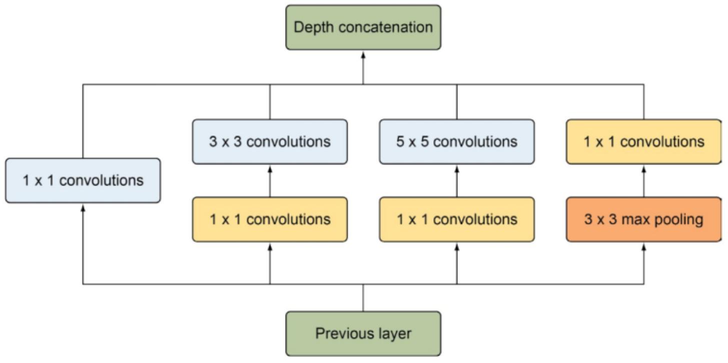
~138,000,000 parameters

GoogleNet

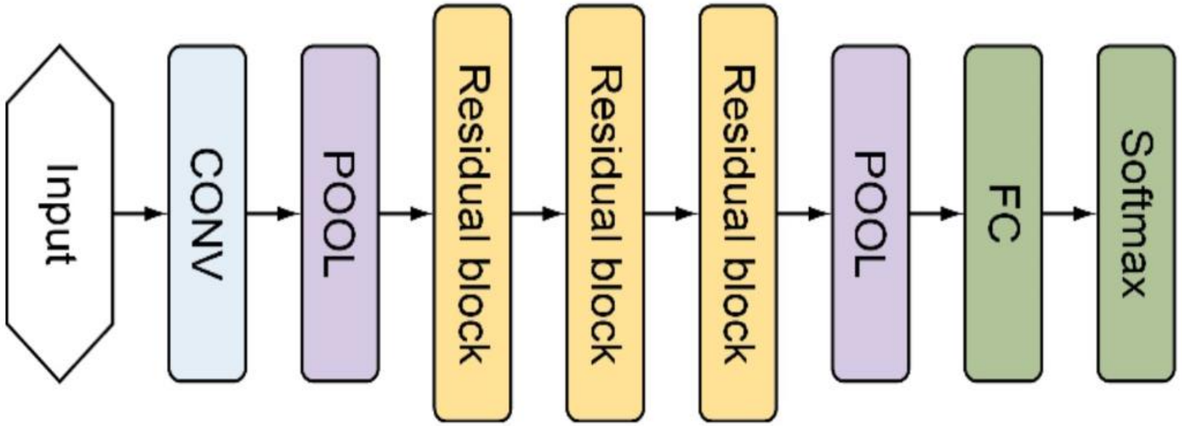
~13,000,000 parameters

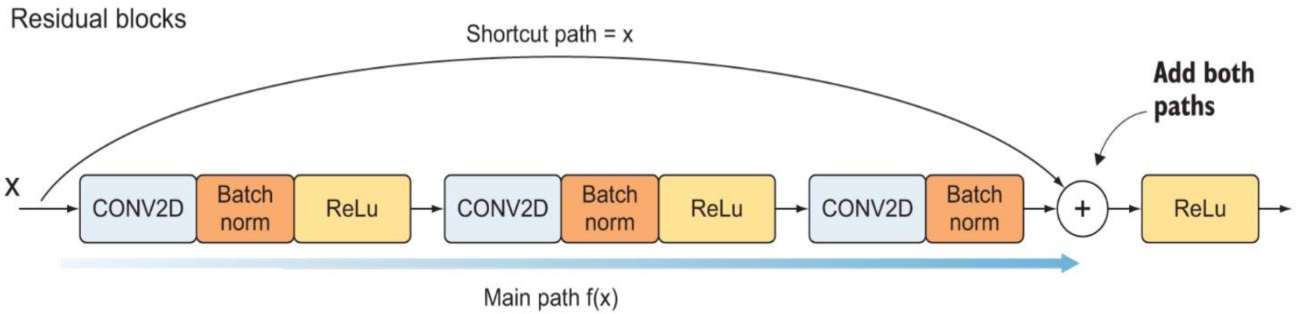


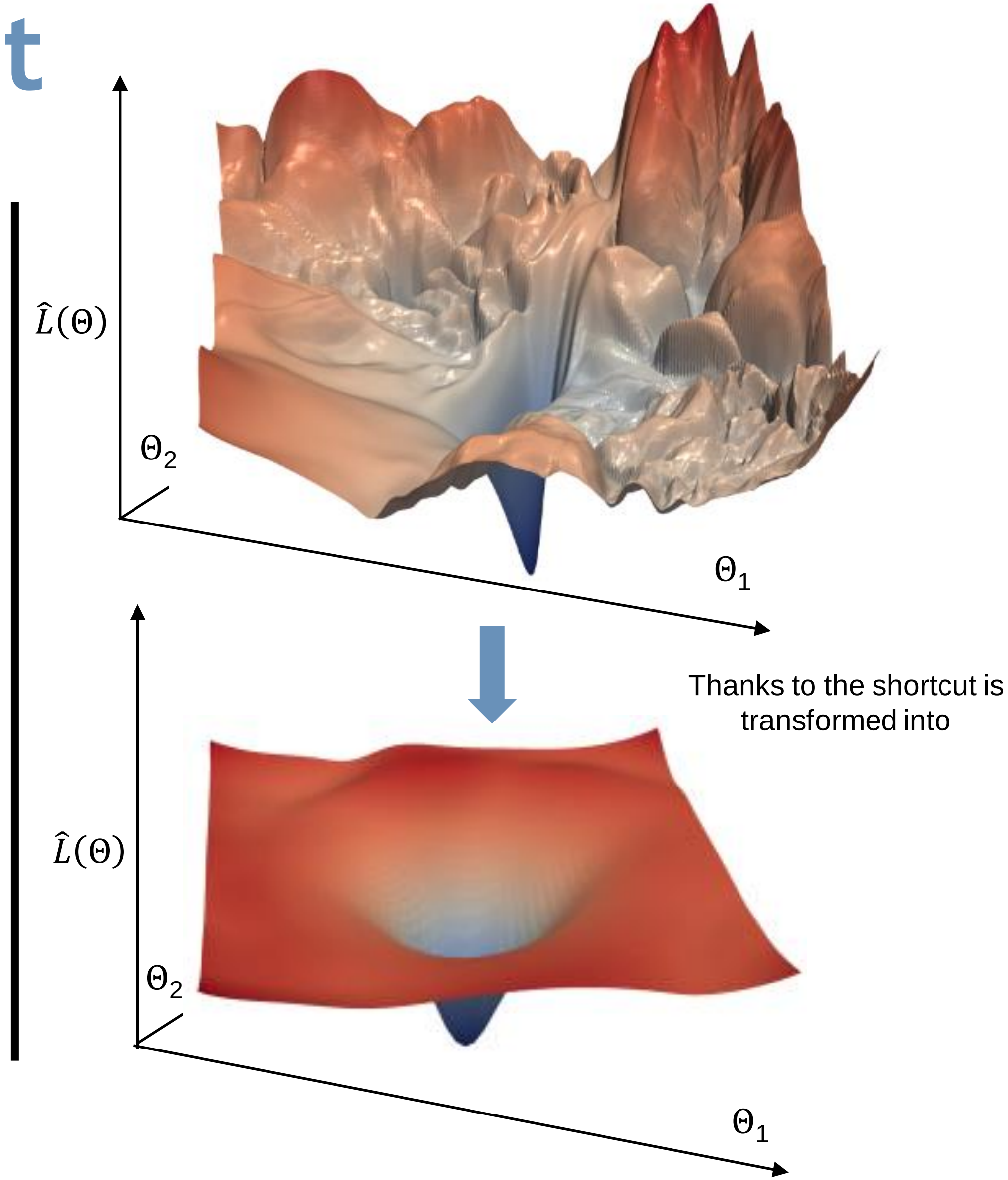
- What is the best kernel size for each layer?
- Concatenating filters instead of stacking them for reducing computational expenses



RestNet

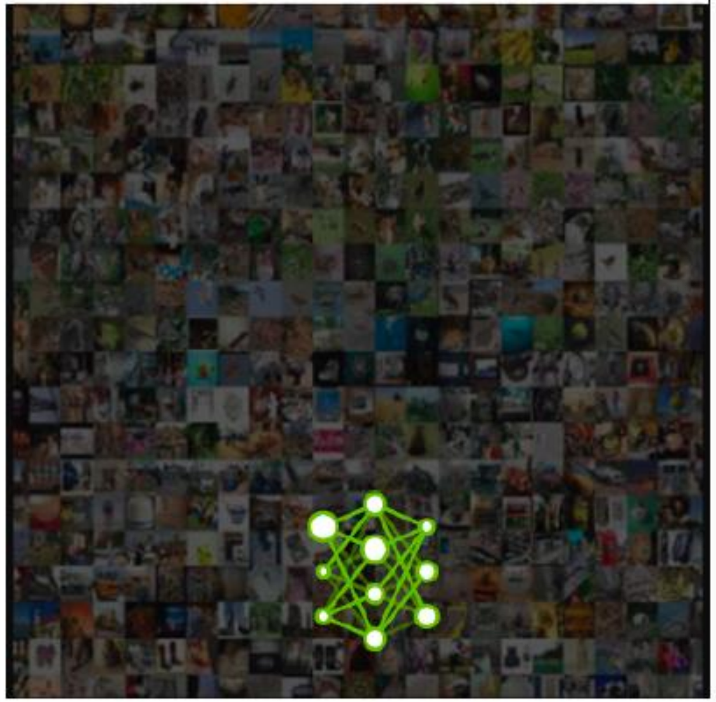

Input
CONV
POOL
Residual block
Residual block
Residual block
POOL
FC
Softmax


Residual blocks
Shortcut path = x
Add both paths
x
CONV2D
Batch norm
ReLu
CONV2D
Batch norm
ReLu
CONV2D
Batch norm
+
ReLu
Main path f(x)


$\hat{L}(\Theta)$
Θ_2
Θ_1
Thanks to the shortcut is transformed into
$\hat{L}(\Theta)$
Θ_2
Θ_1

Increasing complexity

7 Exaflops
60 Million Parameters

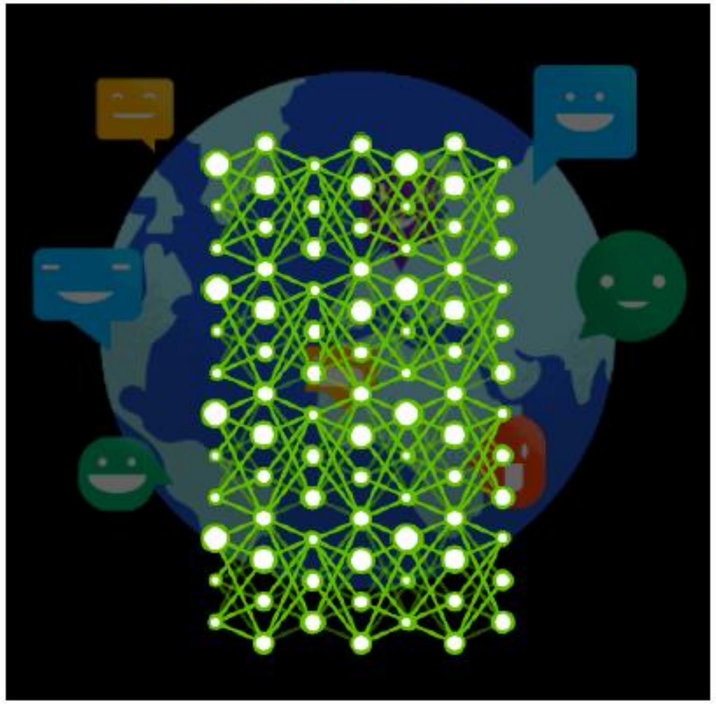


2015 - Microsoft ResNet
Superhuman Image Recognition

20 Exaflops
300 Million Parameters

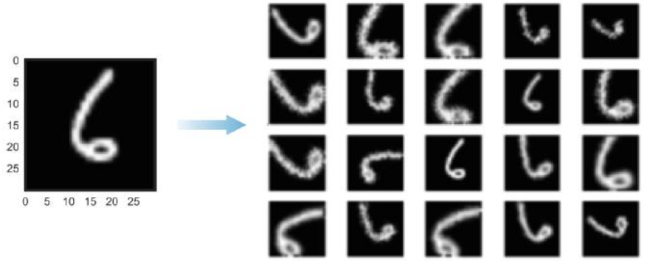
2016 - Baidu Deep Speech 2
Superhuman Voice Recognition

100 Exaflops
8700 Million Parameters



2017 - Google Neural Machine Translation
Near Human Language Translation

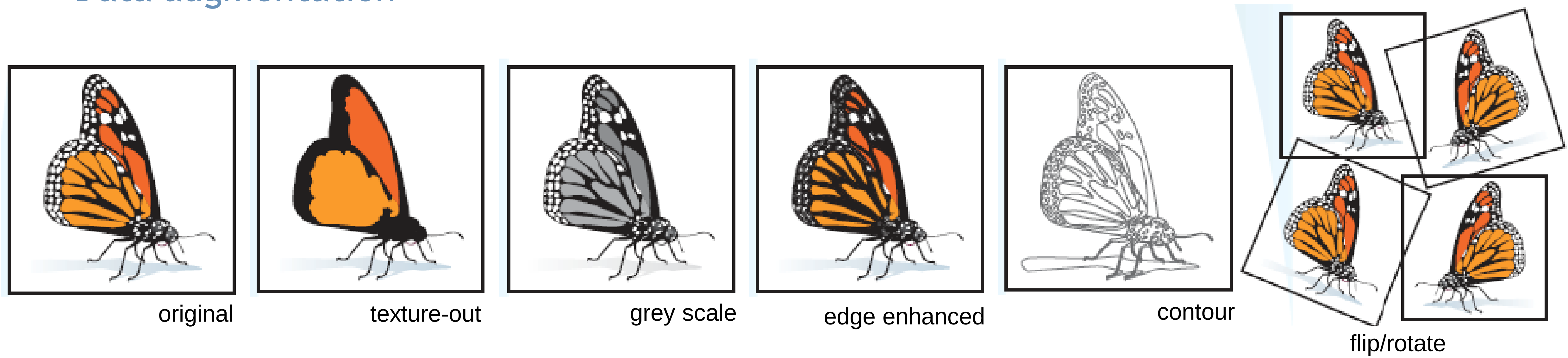
How much data?



A non desired issue with NN is known as overfitting

Getting more data not always viable

Data augmentation



Summary

Brief introduction to Deep Learning with emphasis in Deep Convolutional Neural Networks

Review of basic concepts: from perceptron to the learning task

Debrief of most important concepts of neural network architectures

Introduction to data augmentation

Code review