FUNDAMENTALS OF DEEP LEARNING

PD. Dr. Juan J. Durillo

lrz

VIENNA SCIENTIFIC CLUSTER

NVIDIA DEEP LEARNING INSTITUTE

THE GOALS OF THIS COURSE

- Get you up and on your feet quickly
- Build a foundation to tackle a deep learning project right away
- We won't cover the whole field, but we'll get a great head start
- Foundation from which to read articles, follow tutorials, take further classes

https://courses.nvidia.com/dli-event

VSC_FDL_AMBASSADOR_AP21

NVIDIA | DEEP LEARNING INSTITUTE

www.nvidia.com/dli

NVIDIA | DEEP LEARNING INSTITUTE

AGENDA

- Part 1: An Introduction to Deep Learning
- Part 2: How a Neural Network Trains
- Part 3: Convolutional Neural Networks
- Part 4: Data Augmentation and Deployment
- Part 5: Pre-trained Models
- Part 6: Advanced Architectures

HAVE FUN!

HISTORY OF AI

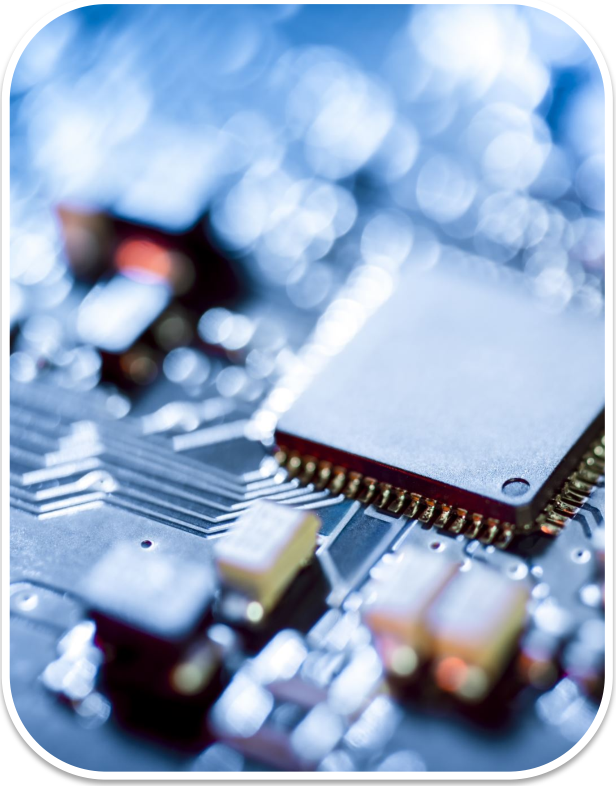
BEGINNING OF ARTIFICIAL INTELLIGENCE

COMPUTERS ARE MADE IN PART TO COMPLETE HUMAN TASKS

EARLY ON, GENERALIZED INTELLIGENCE LOOKED POSSIBLE

TURNED OUT TO BE HARDER THAN EXPECTED

EARLY NEURAL NETWORKS



Inspired by biology

Created in the 1950's

Outclassed by Von Neumann Architecture

EXPERT SYSTEMS

Highly complex

Programmed by hundreds of engineers

Rigorous programming of many rules

EXPERT SYSTEMS - LIMITATIONS

What are these three images?

THE DEEP LEARNING REVOLUTION

DATA

- Networks need a lot of information to learn from
- The digital era and the internet has supplied that data

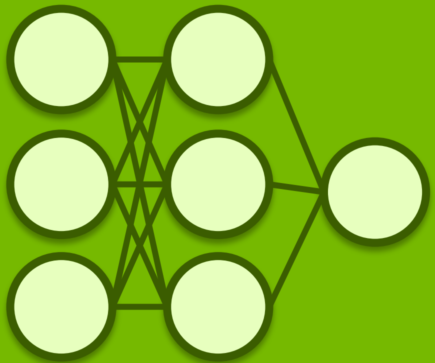
COMPUTING POWER

Need a way for our artificial “brain” to observe lots of data within a practical amount of time.

THE IMPORTANCE OF THE GPU

A Rendered Image

A Neural Network

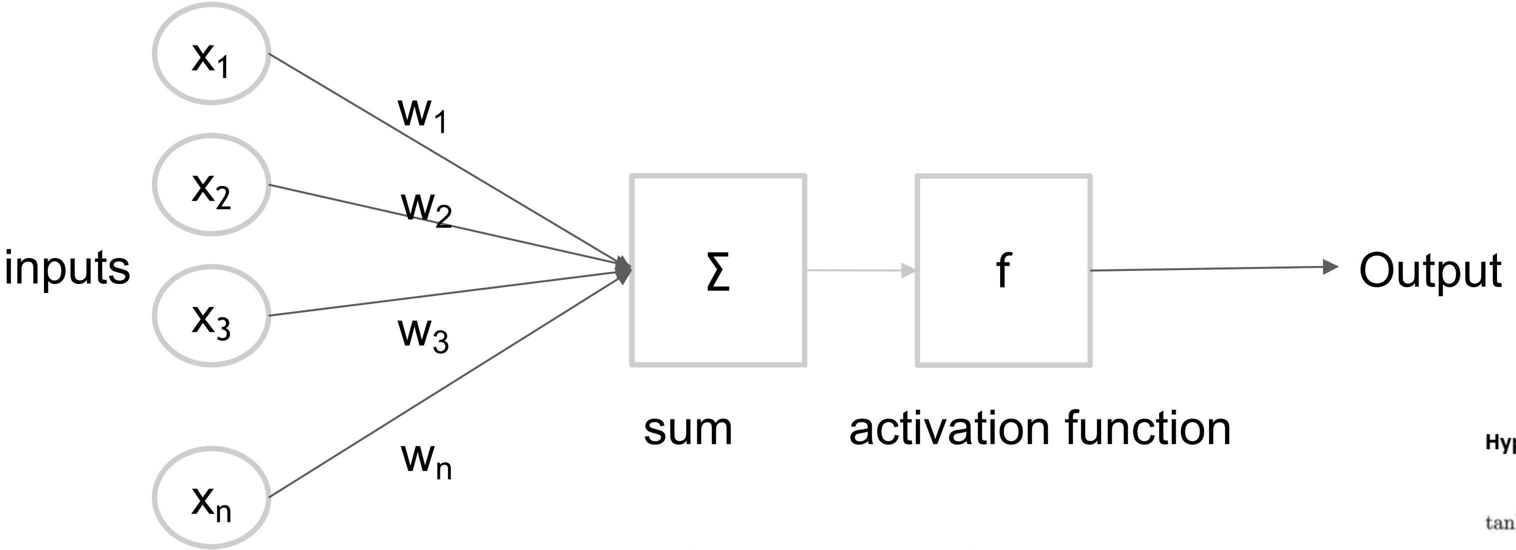


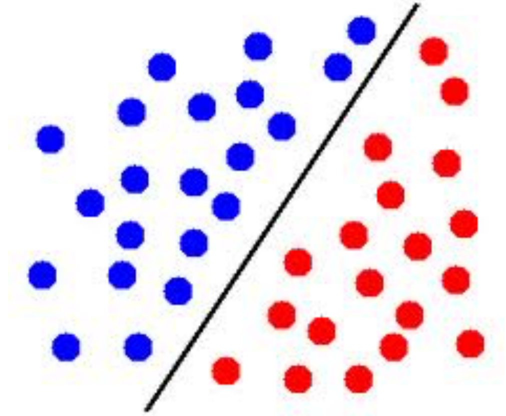
WHAT IS DEEP LEARNING?

A (brief) introduction to Machine Learning

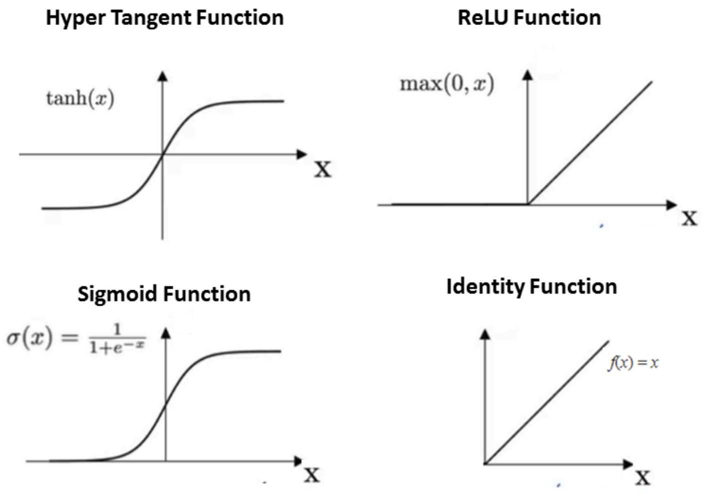
28.04.2021 | PD Dr. Juan J. Durillo

Perceptron - Artificial Neuron





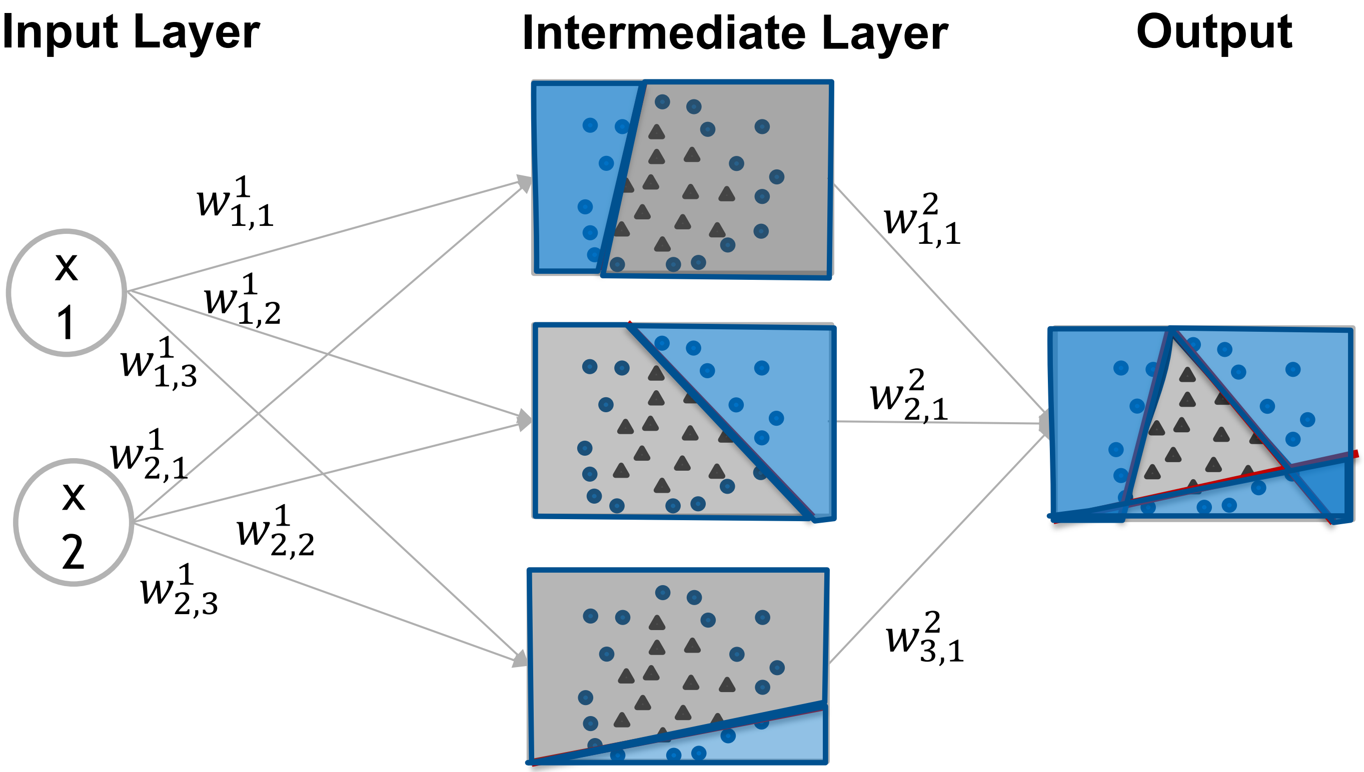
$$\Theta = \{w_1, w_2 \dots, w_n\}$$



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

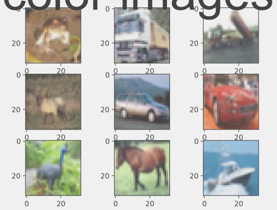


$$\Theta = \{w^1_{1,1}, w^1_{1,2}, w^1_{1,3}, w^1_{2,1}, w^1_{2,2}, w^1_{2,3}, w^2_{1,1}, w^2_{2,1}, w^2_{2,3}\}$$

- Works well even when the data is not linearly separable

(SUPERVISED) LEARNING

X: 32 x 32 color images



Υ : labels

{truck, car, horse, bird, boat}

Example (CIFAR10 dataset)

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

  X → domain of the input data

  Υ → set of labels (knowledge)

- 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, …, 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

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

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

B_t: random subset of training examples

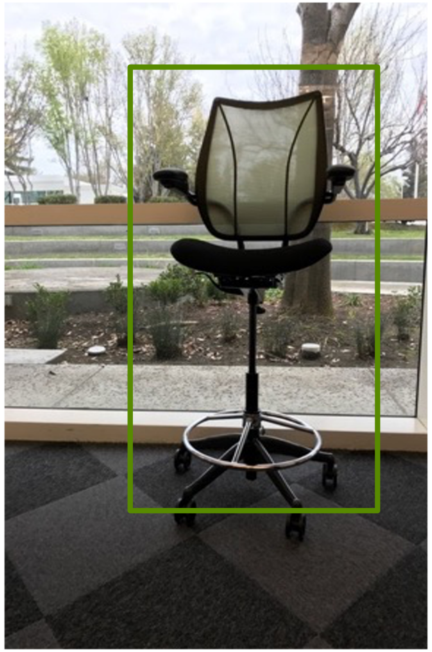
n_t: positive scalar (learning rate)

epoch: update the weights after going over all training set

COMPUTER VISION TASKS

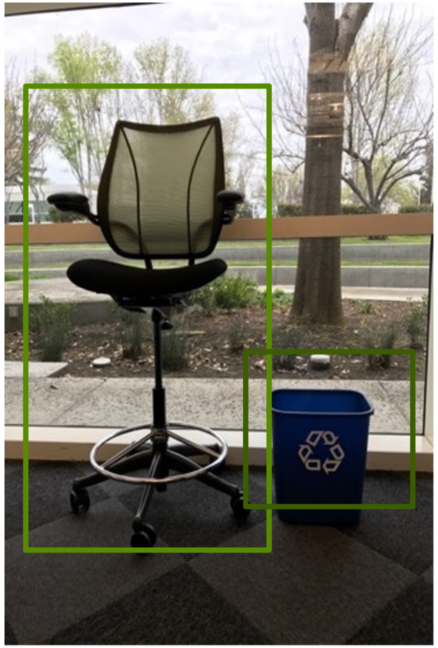
predicting the type or class of an object in an image

Image Classification



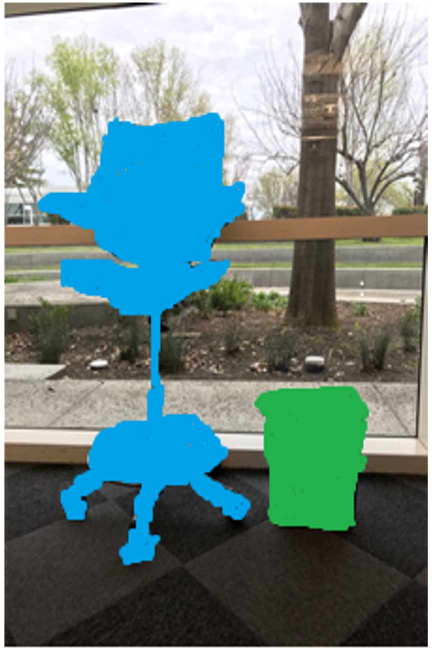
predicting the type or class on an object in an image and draw a bounding box around

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

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	55	97	157	156	187	255	81	0	0	0	0	0
0	0	0	0	0	0	0	0	5	20	58	20	96	140	215	215	251	254	254	254	254	254	163	0	0	0	0	0
0	0	0	0	0	0	0	0	157	254	254	254	254	254	254	254	254	254	254	254	254	241	100	0	0	0	0	0
0	0	0	0	0	0	0	0	155	117	184	214	214	155	117	49	19	19	141	254	241	158	23	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	173	254	246	70	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	90	247	254	219	68	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	8	43	203	254	250	121	15	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	43	146	254	254	229	111	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	4	137	245	254	254	238	40	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	133	254	254	254	254	254	235	106	23	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	55	196	196	158	196	223	254	254	216	23	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	70	245	253	99	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	197	254	134	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14	200	254	134	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	88	246	253	99	0	0	0	0	0	0	0	0	0
0	0	0	37	59	18	0	0	0	0	0	0	0	0	67	254	254	170	0	0	0	0	0	0	0	0	0	0
0	0	0	128	254	159	0	0	0	0	0	11	20	140	248	254	204	9	0	0	0	0	0	0	0	0	0	0
0	0	1	156	254	229	86	79	79	161	176	218	254	254	254	168	6	0	0	0	0	0	0	0	0	0	0	0
0	0	0	29	203	254	254	254	255	254	254	254	237	95	19	6	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	5	89	155	155	156	111	58	58	36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

image

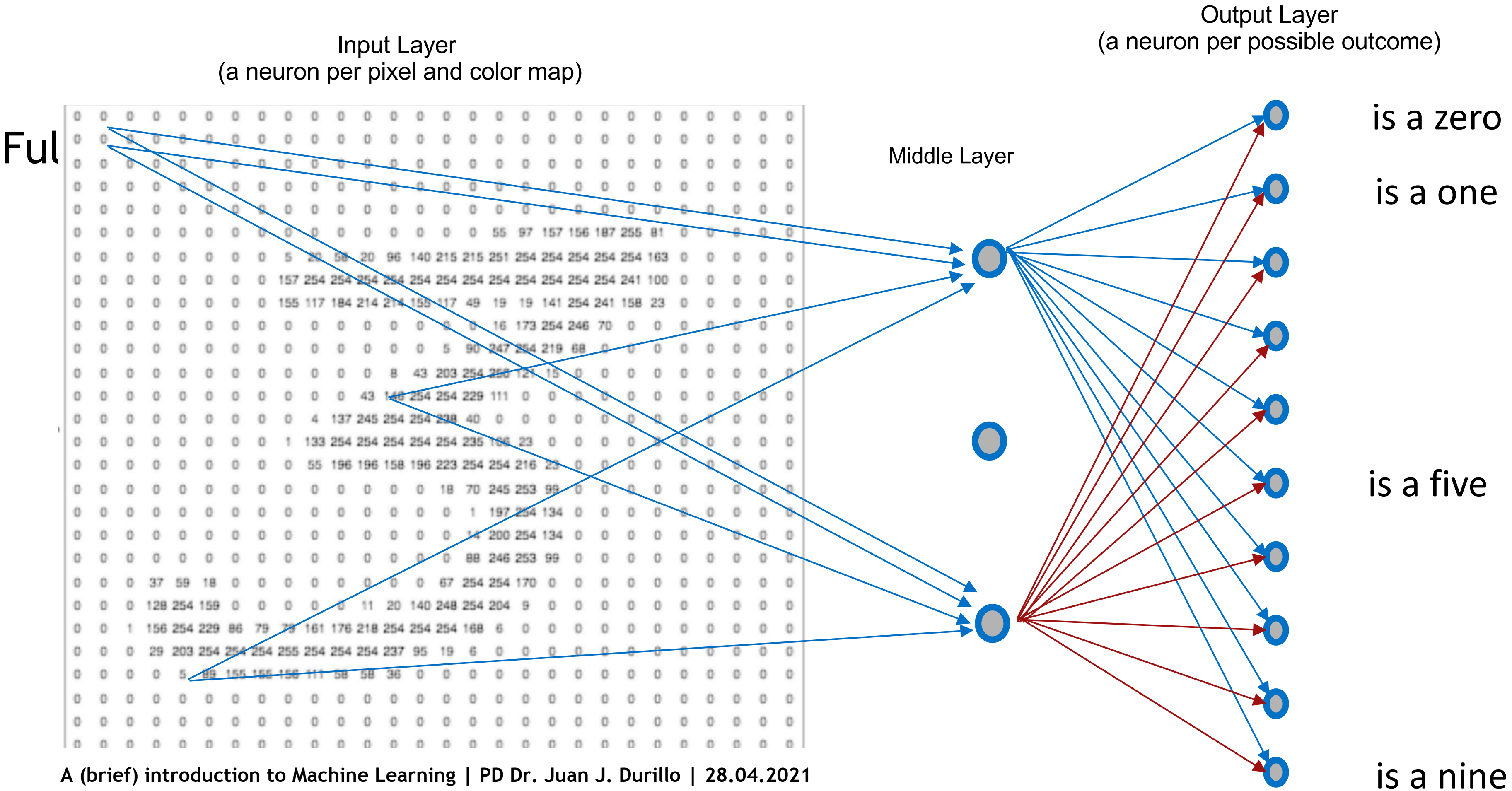
```
dict=['EOS','a','my','sleeps','on','dog','cat','the','bed','floor']
```

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

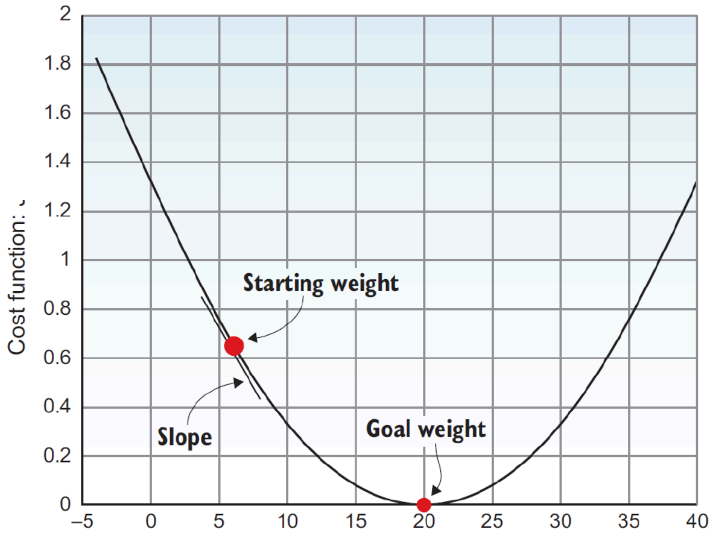
```
[[ 0.  1.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  1.  0.  0.  0.  0.]
 [ 0.  0.  0.  1.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  1.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  1.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  1.]
 [ 1.  0.  0.  0.  0.  0.  0.  0.  0. 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

language

NEURAL NETWORKS FOR IMAGE CLASSIFICATION



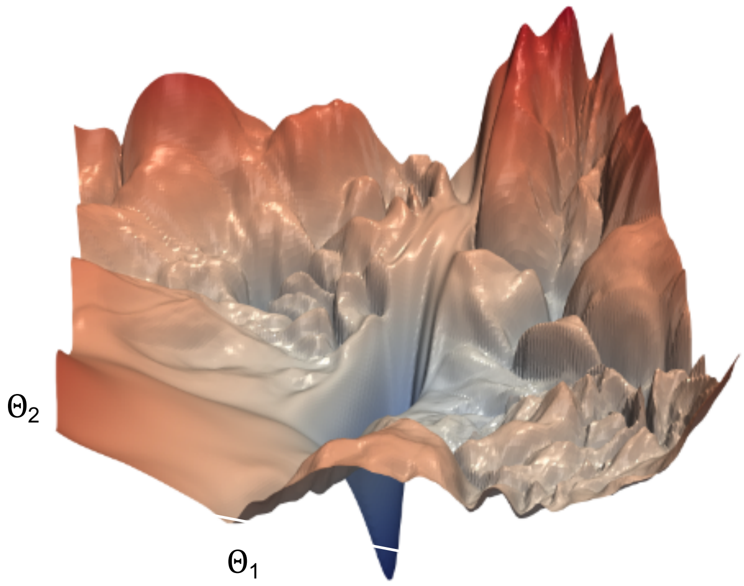
TRAINING NEURAL NETWORKS



Θ

main idea

$\hat{L}(\Theta)$



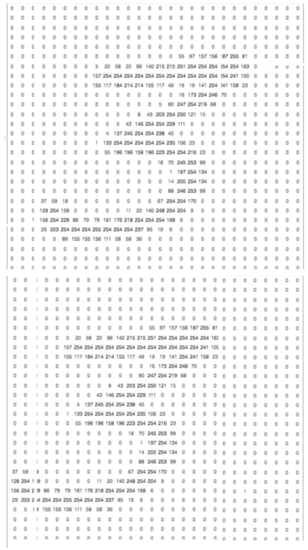
how the surface looks like in reality

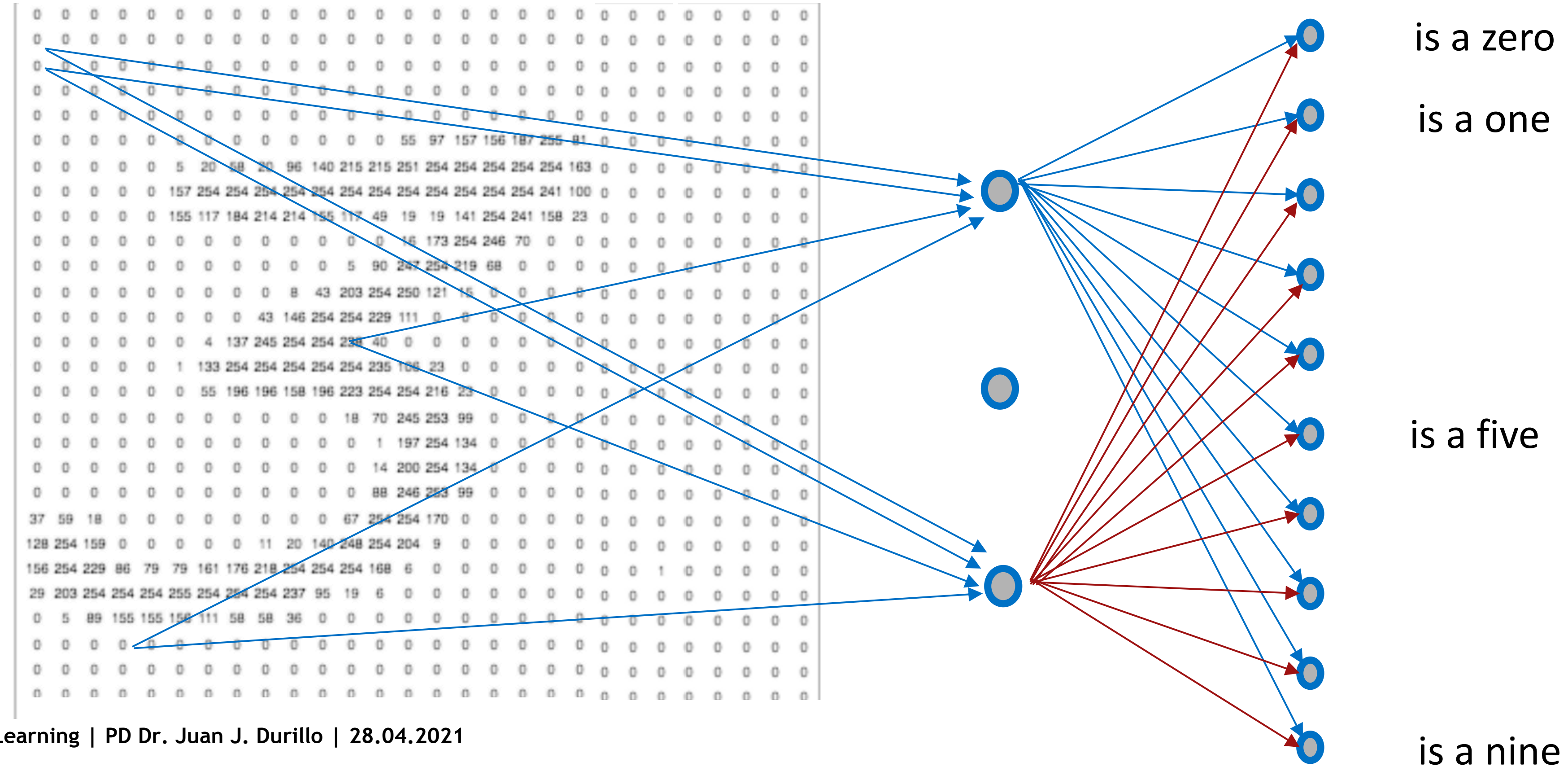
Stochastic Gradient Descent

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

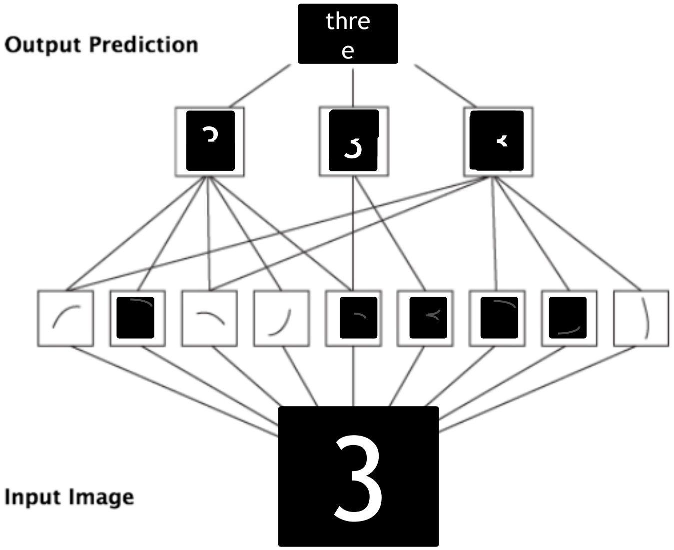
$$g(\Theta; B) = \frac{1}{|B|} \sum_{z \epsilon B} \nabla \ell(\Theta; z)$$

NEURAL NETWORKS FOR IMAGE CLASSIFICATION

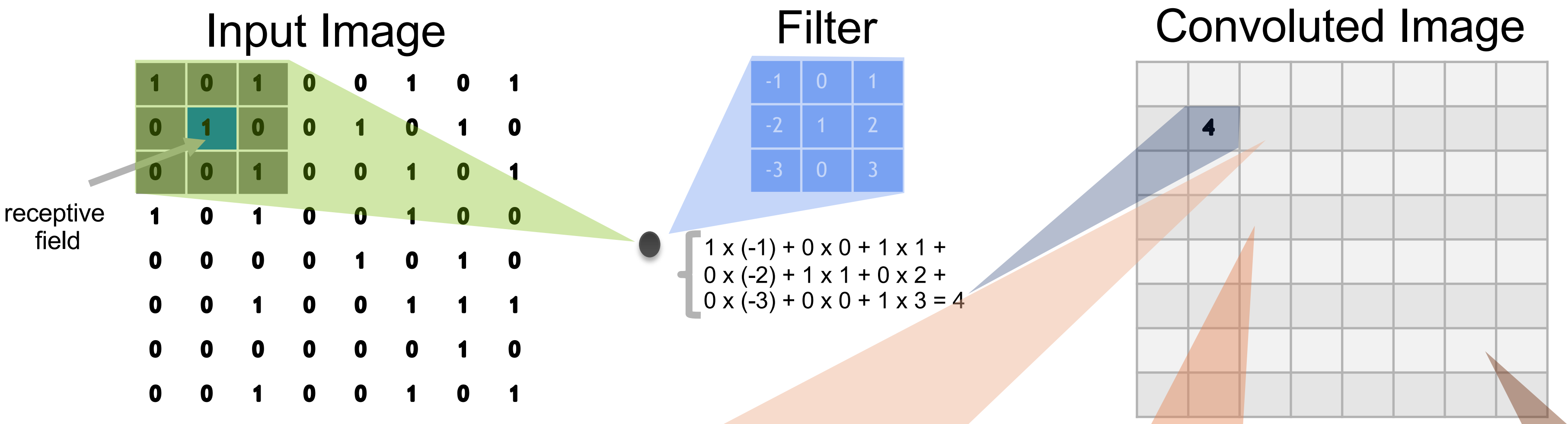




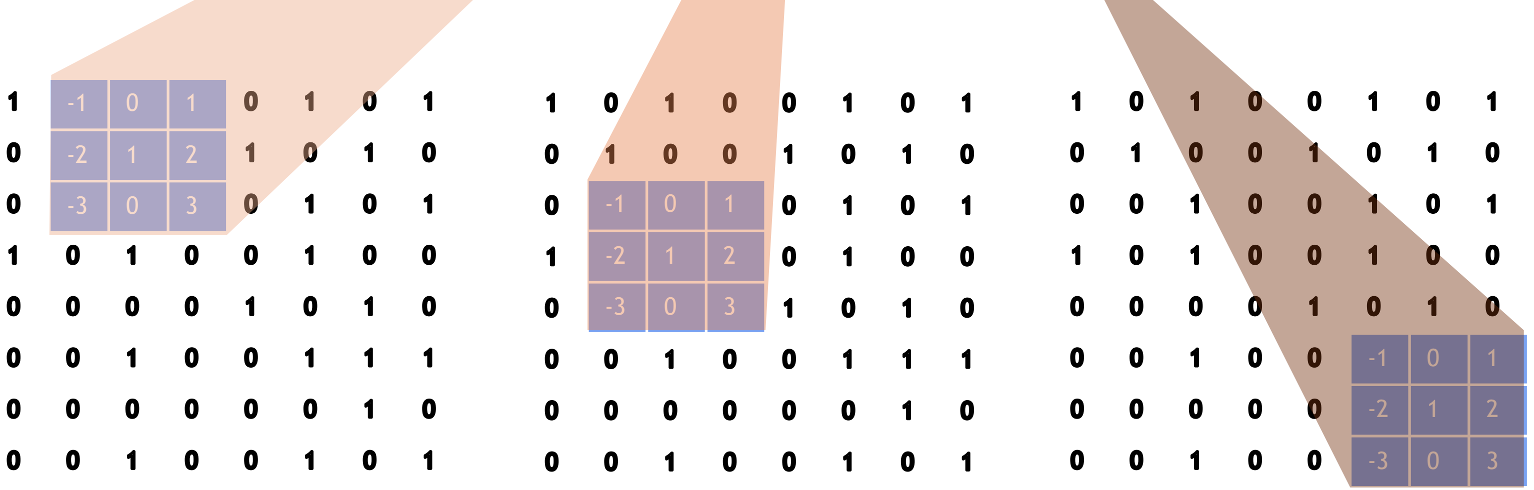
NO MORE FEATURE ENGINEERING



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

*The London skyline image is designed by Freepik

FILTERS

Input Image:

Can we get only vertical lines out of this picture?

$$\begin{matrix} 1 & 0 & -1 \end{matrix}$$

filter 1

$$\begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}$$

filter 2

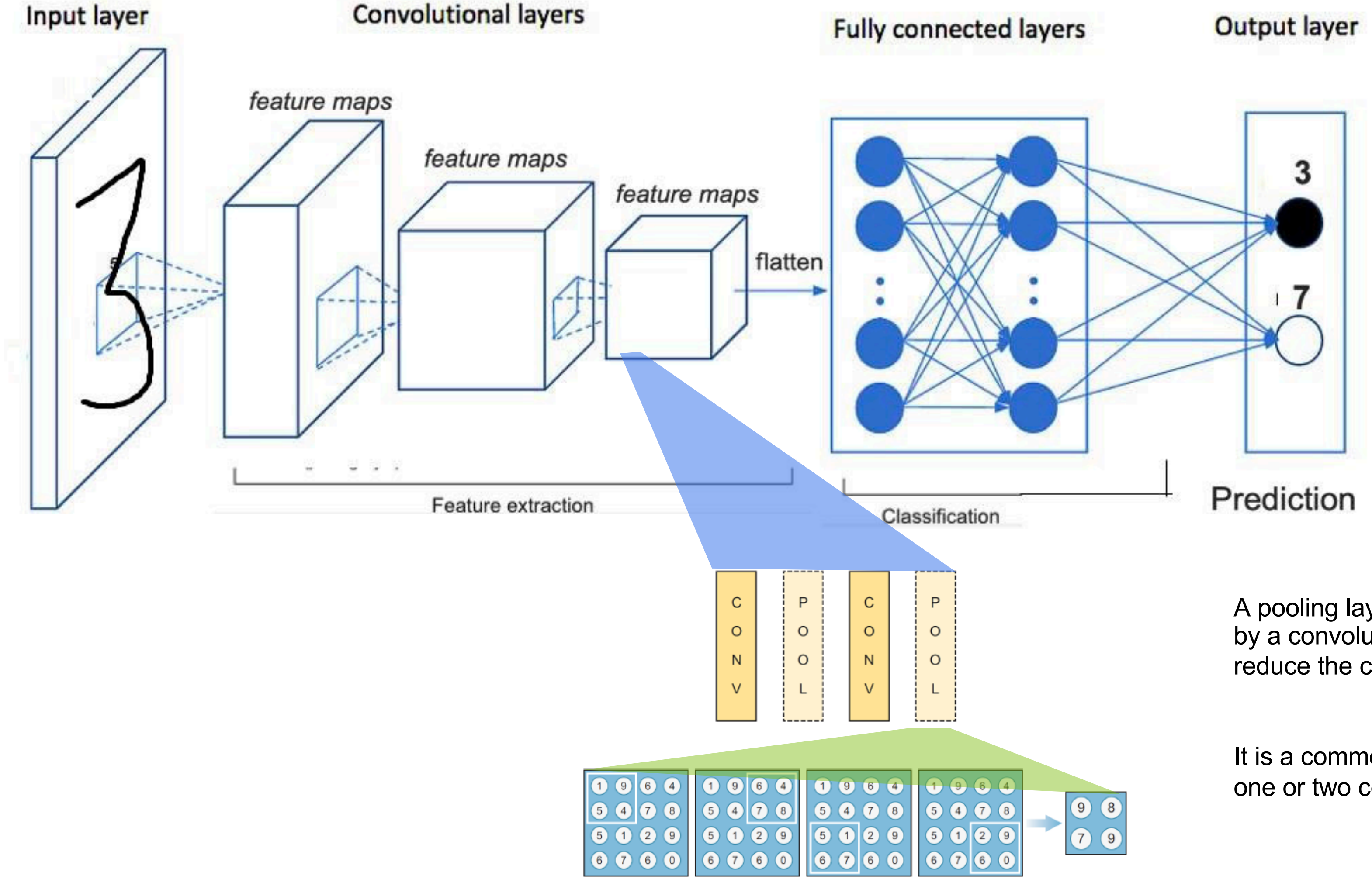
$$\begin{matrix} 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & -1 \end{matrix}$$

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>);
```

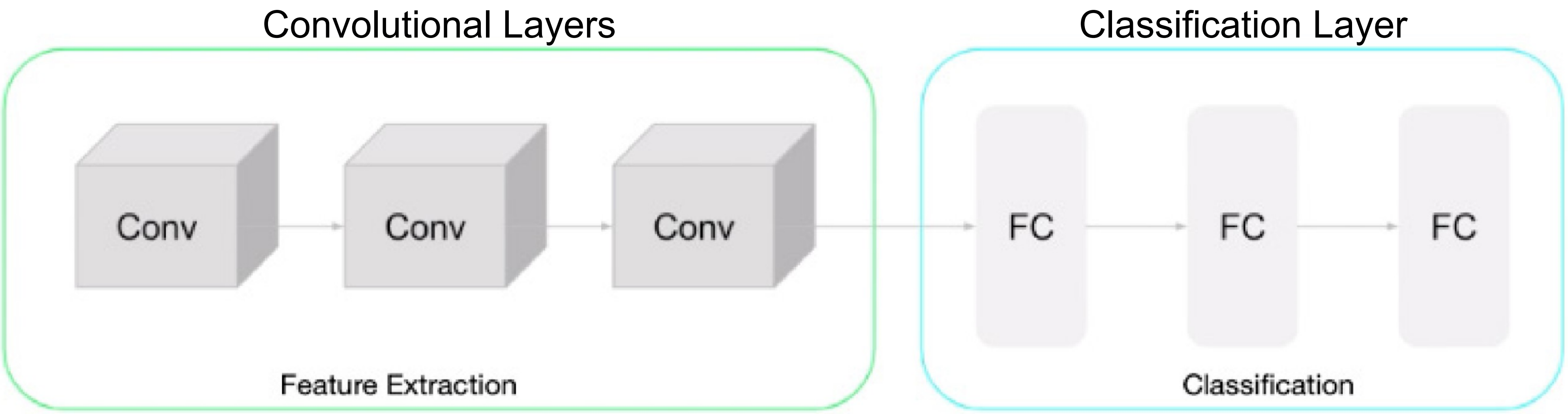
CONVOLUTIONAL NEURAL NETWORKS (CNN)



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

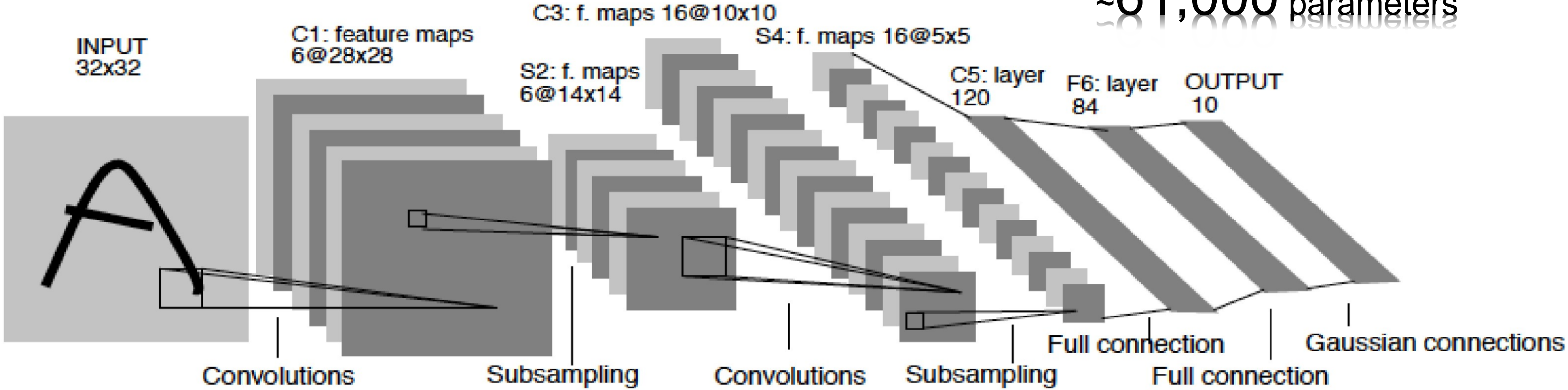


The execution time required during a forward pass through a neural network is bounded from below by the number of floating point operations (FLOPs).

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

LENET ARCHITECTURE

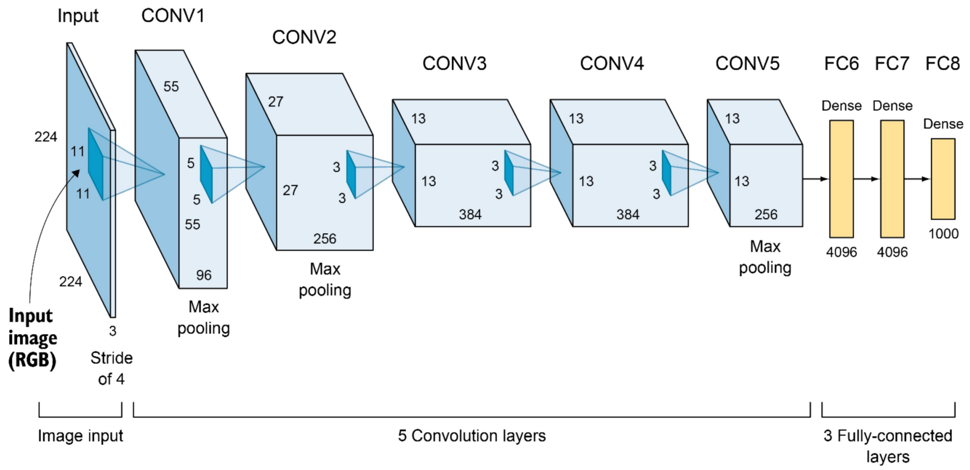
~61,000 parameters



Architecture summary :

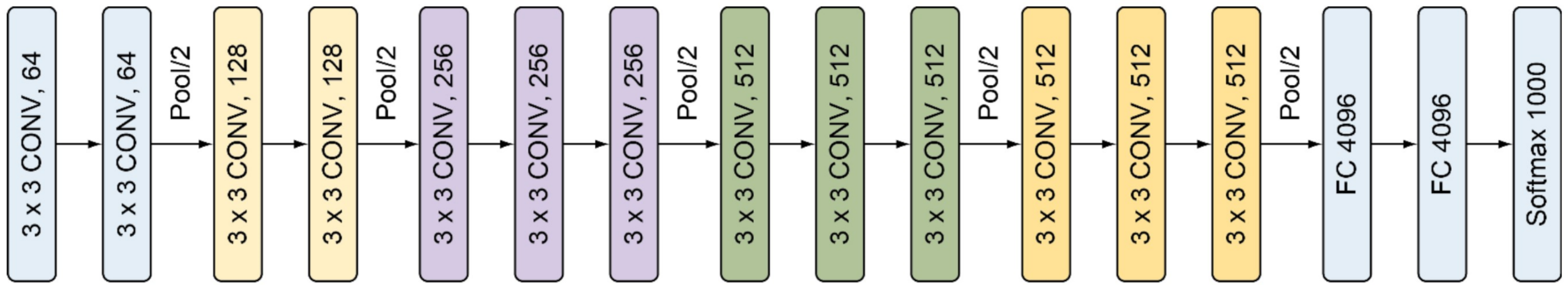
- 3 convolutional layers filters in all the layers equal to 5x5
  (layer 1 depth = 6, layer 2 depth = 16, layer 3 depth = 120)
- As activation function the tanh function is used

ALEXNET AND VGG ARCHITECTURES



AlexNet

~60,000,000 parameters

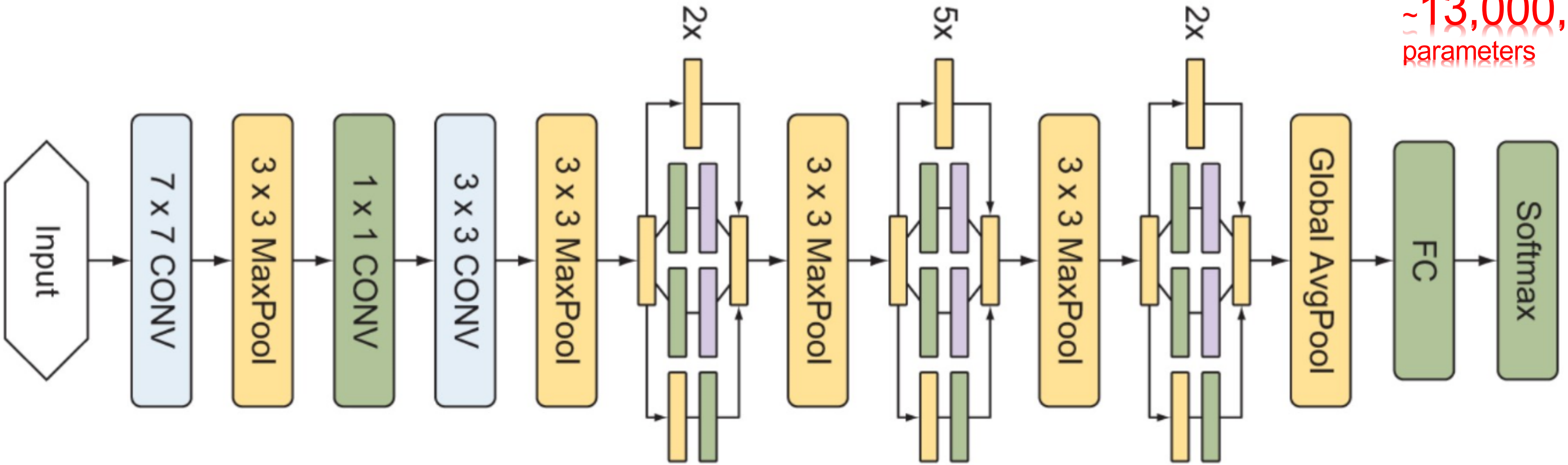


VGG16

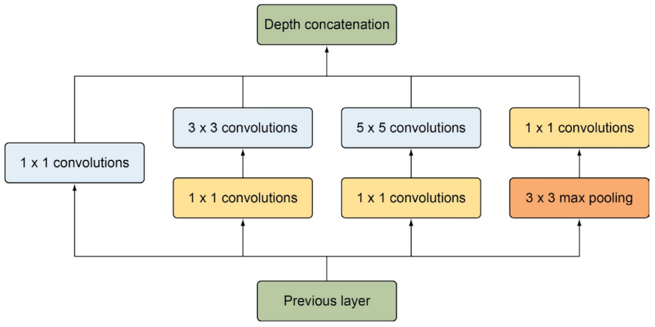
~138,000,000 parameters

GOOGLENET

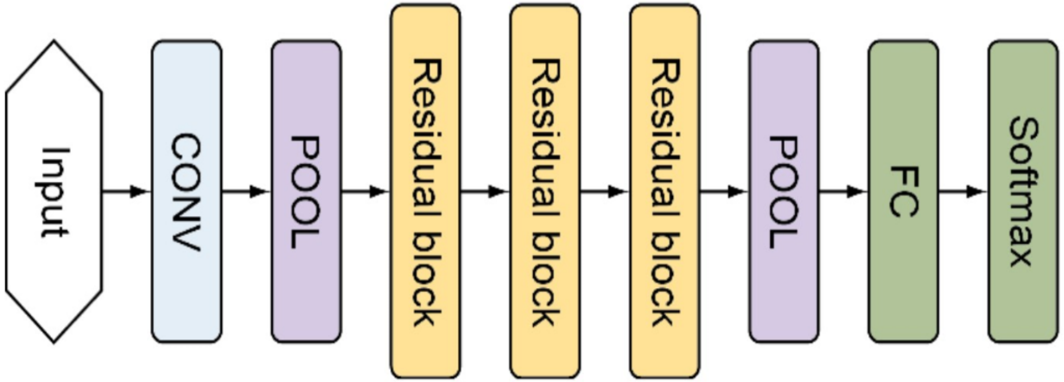
~13,000,000 parameters

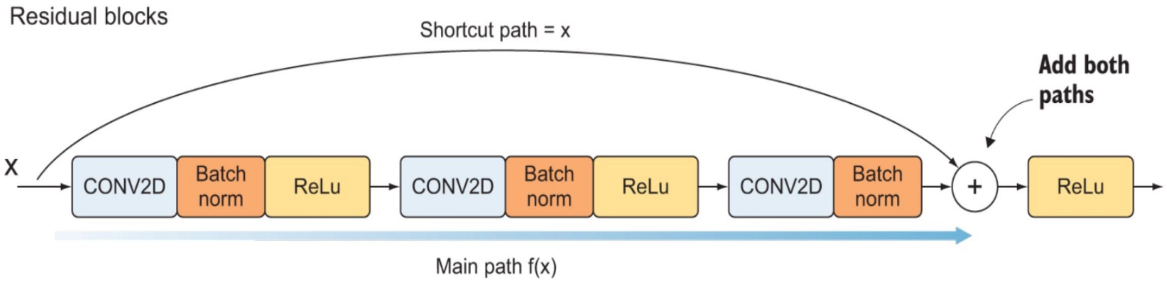


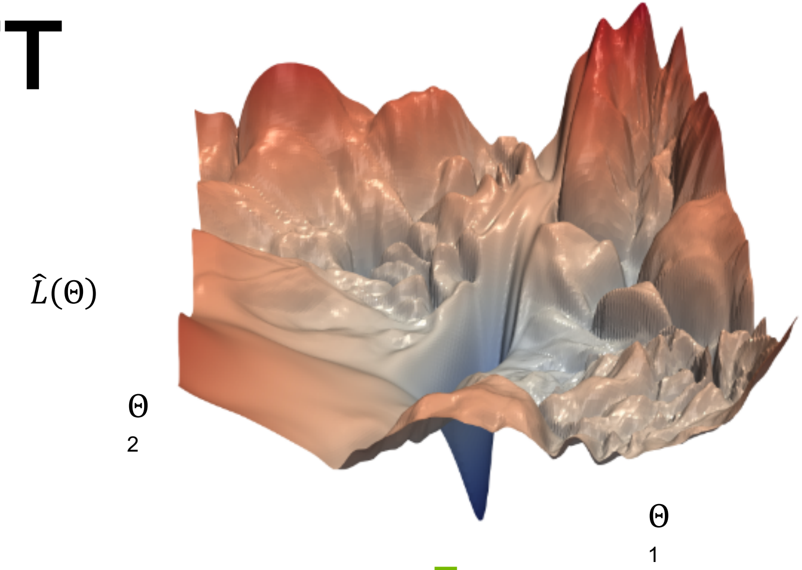
- What is the best kernel size for each layer?
- Concatenating filters instead of stacking them for reducing computational expenses



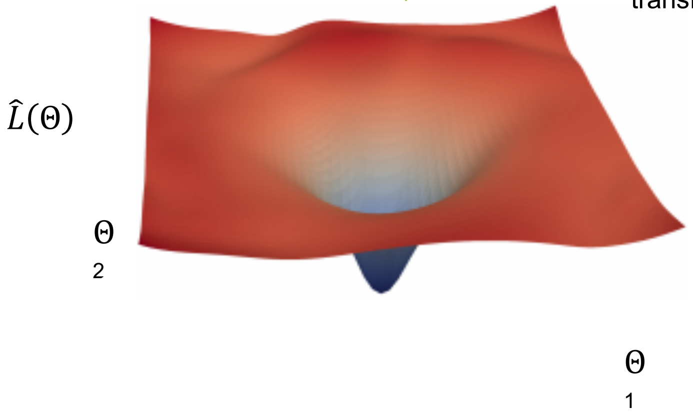
RESTNET





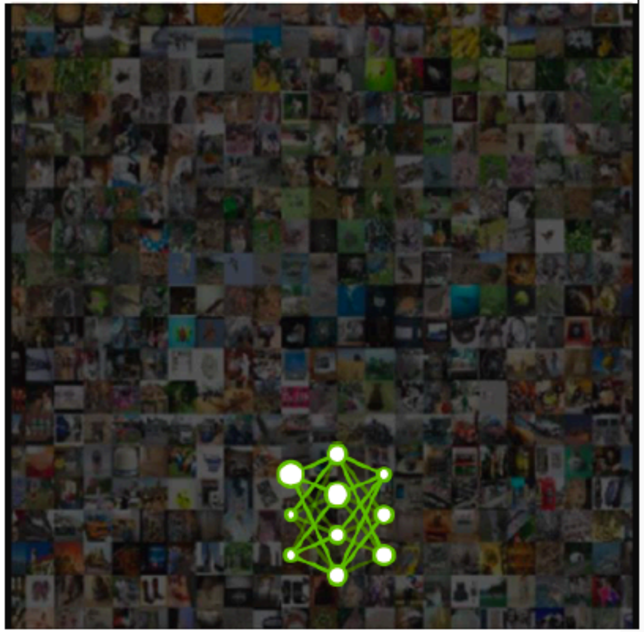


Thanks to the shortcut is transformed into



INCREASING COMPLEXITY

7 Exaflops
60 Million Parameters

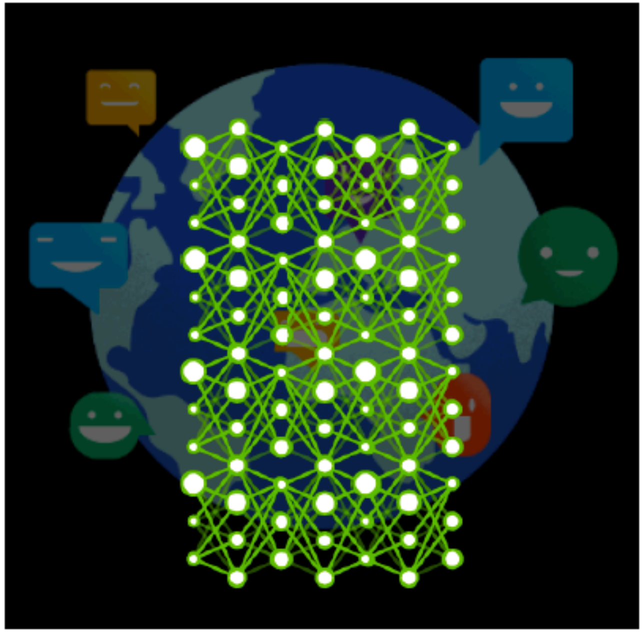


2015 - Microsoft ResNet
Superhuman Image Recognition

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

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

DEEP LEARNING FLIPS TRADITIONAL PROGRAMMING ON ITS HEAD

TRADITIONAL PROGRAMMING

Building a Classifier

1

Define a set of rules for classification

2

Program those rules into the computer

3

Feed it examples, and the program uses the rules to classify

MACHINE LEARNING

Building a Classifier

1

Show model the examples with the answer of how to classify

2

Model takes guesses, we tell it if it's right or not

3

Model learns to correctly categorize as it's training. The system learns the rules on its own

THIS IS A FUNDAMENTAL SHIFT

WHEN TO CHOOSE DEEP LEARNING

Classic Programming

If rules are clear and straightforward, often better to just program it

Deep Learning

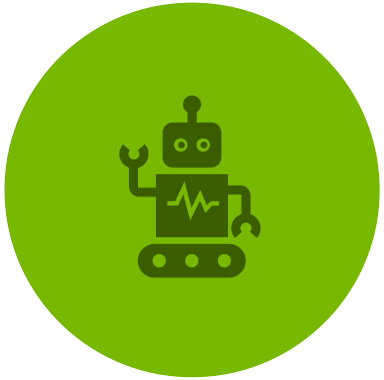
If rules are nuanced, complex, difficult to discern, use deep learning

DEEP LEARNING COMPARED TO OTHER AI

- Depth and complexity of networks
- Up to billions of parameters (and growing)
- Many layers in a model
- Important for learning complex rules

HOW DEEP LEARNING IS TRANSFORMING THE WORLD

COMPUTER VISION



ROBOTICS AND MANUFACTURING

OBJECT DETECTION

SELF DRIVING CARS

NATURAL LANGUAGE PROCESSING

REAL TIME TRANSLATION

VOICE RECOGNITION

VIRTUAL ASSISTANTS

RECOMMENDER SYSTEMS

CONTENT CURATION

TARGETED ADVERTISING

SHOPPING RECOMMENDATIONS

REINFORCEMENT LEARNING

ALPHAGO BEATS WORLD CHAMPION IN GO

AI BOTS BEAT PROFESSIONAL VIDEOGAMERS

STOCK TRADING ROBOTS

OVERVIEW OF THE COURSE

HANDS ON EXERCISES

- Get comfortable with the process of deep learning
- Exposure to different models and datatypes
- Get a jump-start to tackle your own projects

STRUCTURE OF THE COURSE

“Hello World” of Deep Learning

Train a more complicated model

New architectures and techniques to improve performance

Pre-trained models

Transfer learning

PLATFORM OF THE COURSE

GPU powered cloud server

JupyterLab platform

Jupyter notebooks for interactive coding

SOFTWARE OF THE COURSE

- Major deep learning platforms:
  - TensorFlow + Keras (Google)
  - Pytorch (Facebook)
  - MXNet (Apache)
- We'll be using TensorFlow and Keras
- Good idea to gain exposure to others moving forward

FIRST EXERCISE: CLASSIFY HANDWRITTEN DIGITS

HELLO NEURAL NETWORKS

Train a network to correctly classify handwritten digits

- Historically important and difficult task for computers

Try learning like a Neural Network

- Get exposed to the example, and try to figure out the rules to how it works

LET’S GO!

NVIDIA

DEEP
LEARNING
INSTITUTE