# Introduction to embbeded AI

B. Miramond / UCA

# Labs sessions

- Lab0 – Installation of the training framework
- Lab1 – First training on specific dataset
  - Learn a model and optmize/reduce the size of the networks
- Lab2 – First / manual deployment on MCU
- Lab3 – Automatic deployment on MCU
- Lab4 – Collect data and build your own dataset
- Lab5 – Processing of audio data / project
- Lab6 - Project

# Organization of the lectures

1. Introduction to embedded AI
2. Machine learning and artificial neural networks
3. Supervised vs unsupervised learning
4. Convolutional neural networks
5. MicroAI a software framework for neural compression
6. Challenges about Edge AI

# Some references

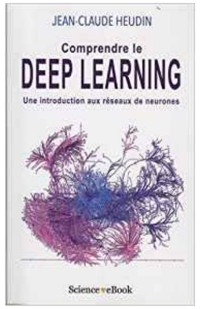
- **Comprendre le Deep Learning** - Une introduction aux réseaux de neurones
  JC Heudin - Ed. Science eBook

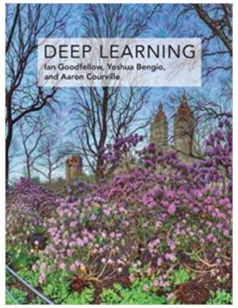
- **Deep Learning**
  Ian Goodfellow, Joshua Bengio et Aaron Courville - Ed. MIT Press

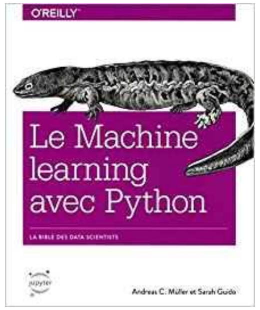
- **Le Machine Learning avec Python**
  Andreas C.Mueller et Sarah GUIDO, Collection O'Reilly

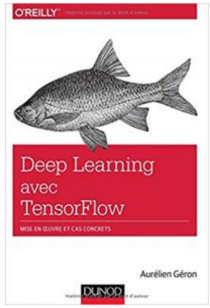
- **Deep Learning avec TensorFlow**
  Aurélien Géron – Ed. Dunod

- **Machine Learning avec Scikit-Learn**
  Aurélien Géron – Ed. Dunod



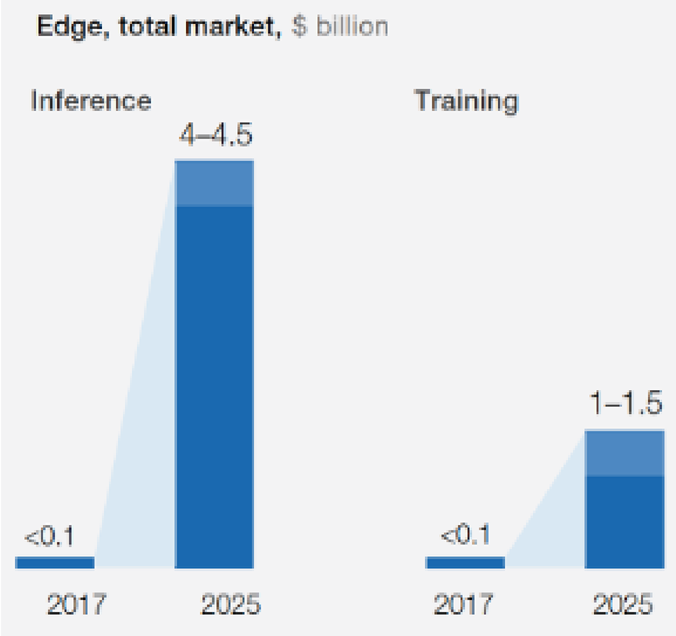



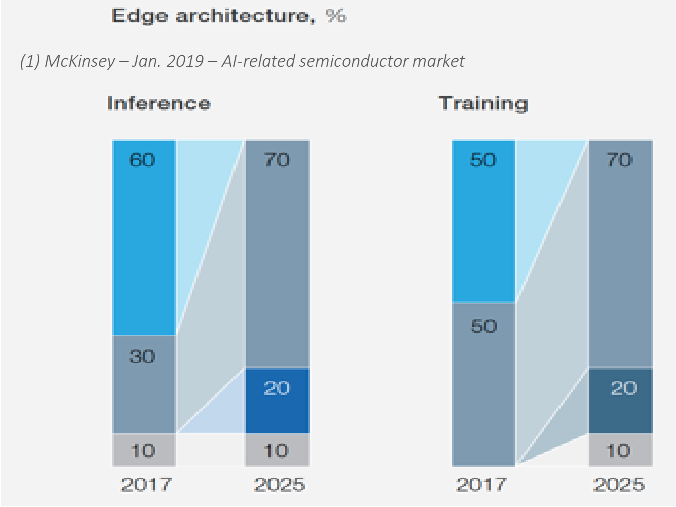




# Introduction to embbeded AI

B. Miramond / UCA

# From embedded systems to Edge Intelligence




(1) McKinsey – Jan. 2019 – AI-related semiconductor market

Data volume explodes with AI, 5G, IoT

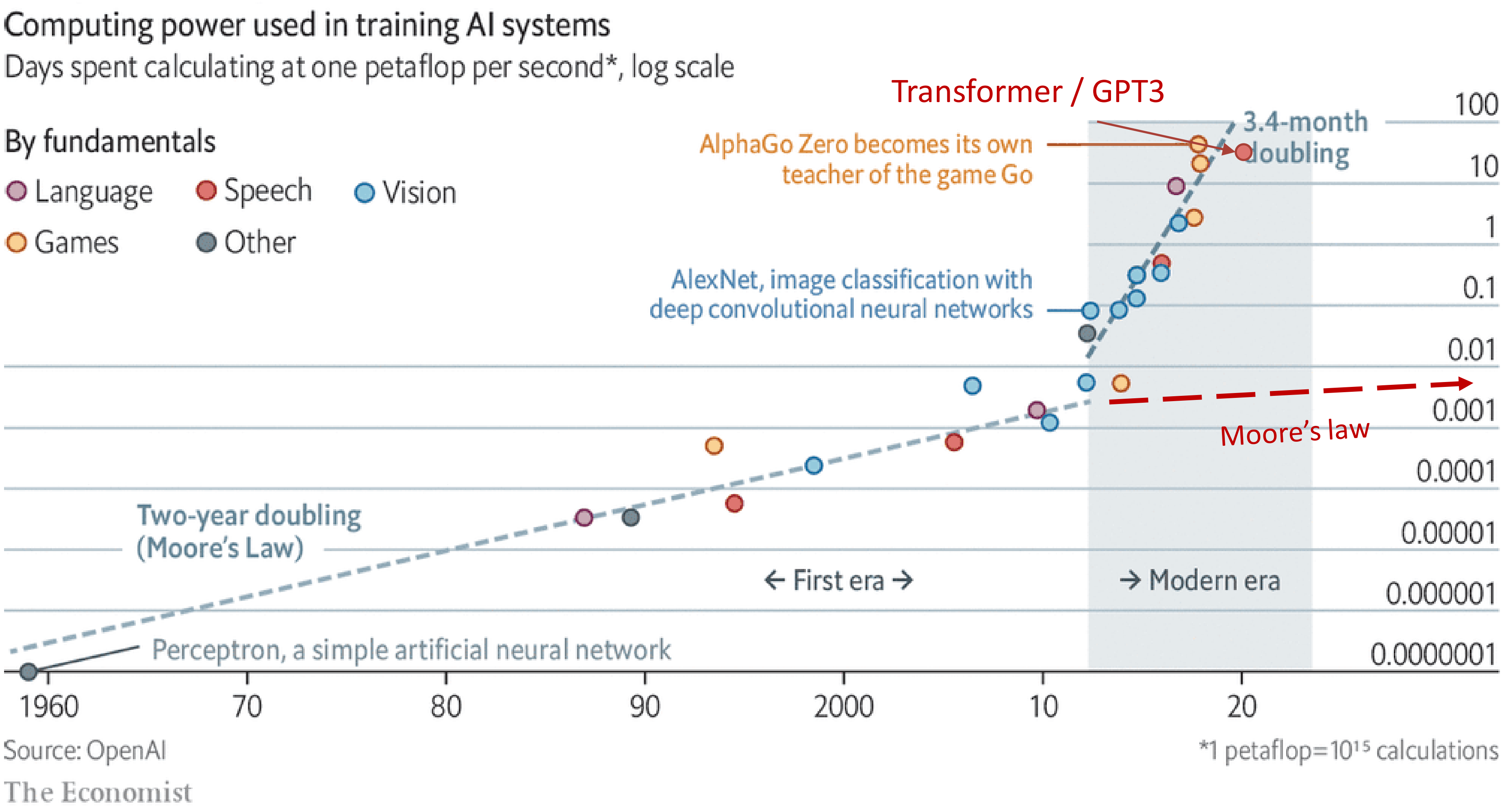
ONLY 25% of usable data reaches a datacenter

75% of data must be analyzed on site immediately

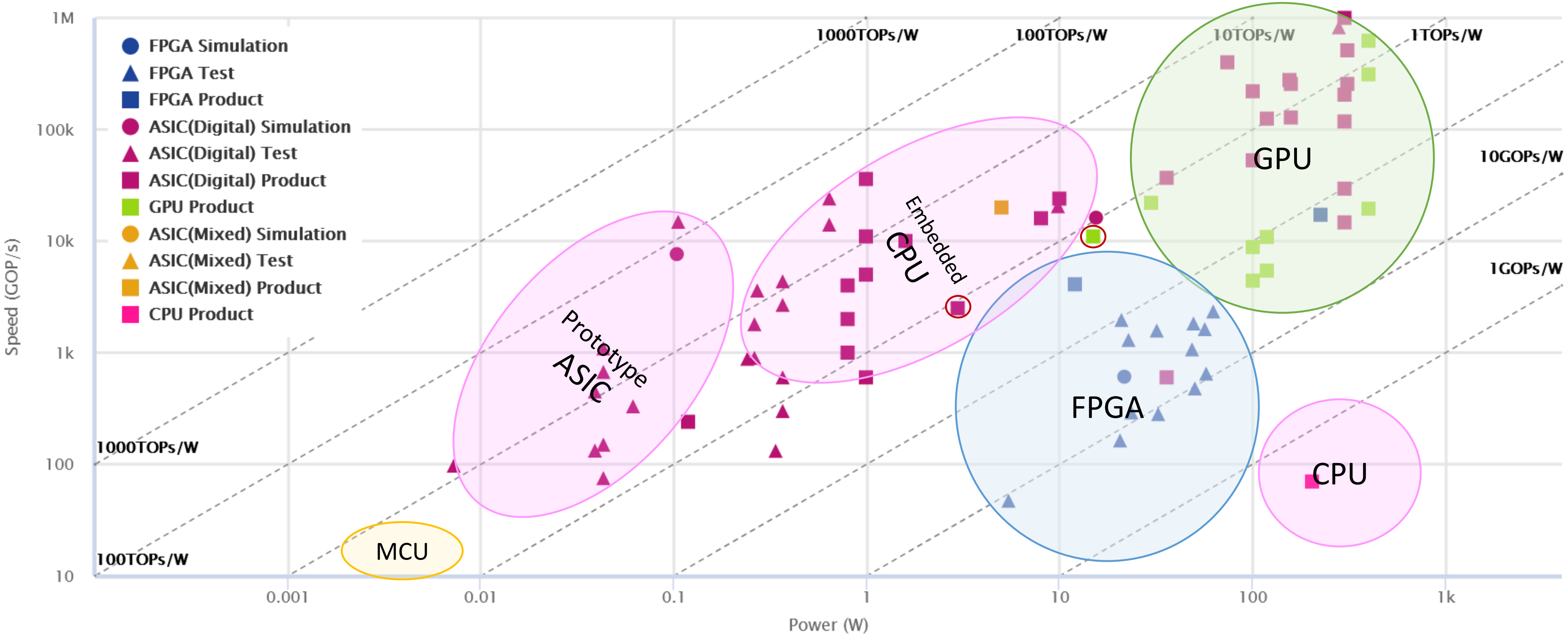
AI / Edge processors market has important growth
GPUs and FPGAs should not dominate this market.
The impact on traditionally dominant companies in France and Europe will be immense in Aerospace, Automotive, Defense, Telecom,...

1 Application-specific integrated circuit.
2 Central processing unit.
3 Field programmable gate array.
4 Graphics-processing unit.

# A contrasted picture on Cloud Artificial Intelligence

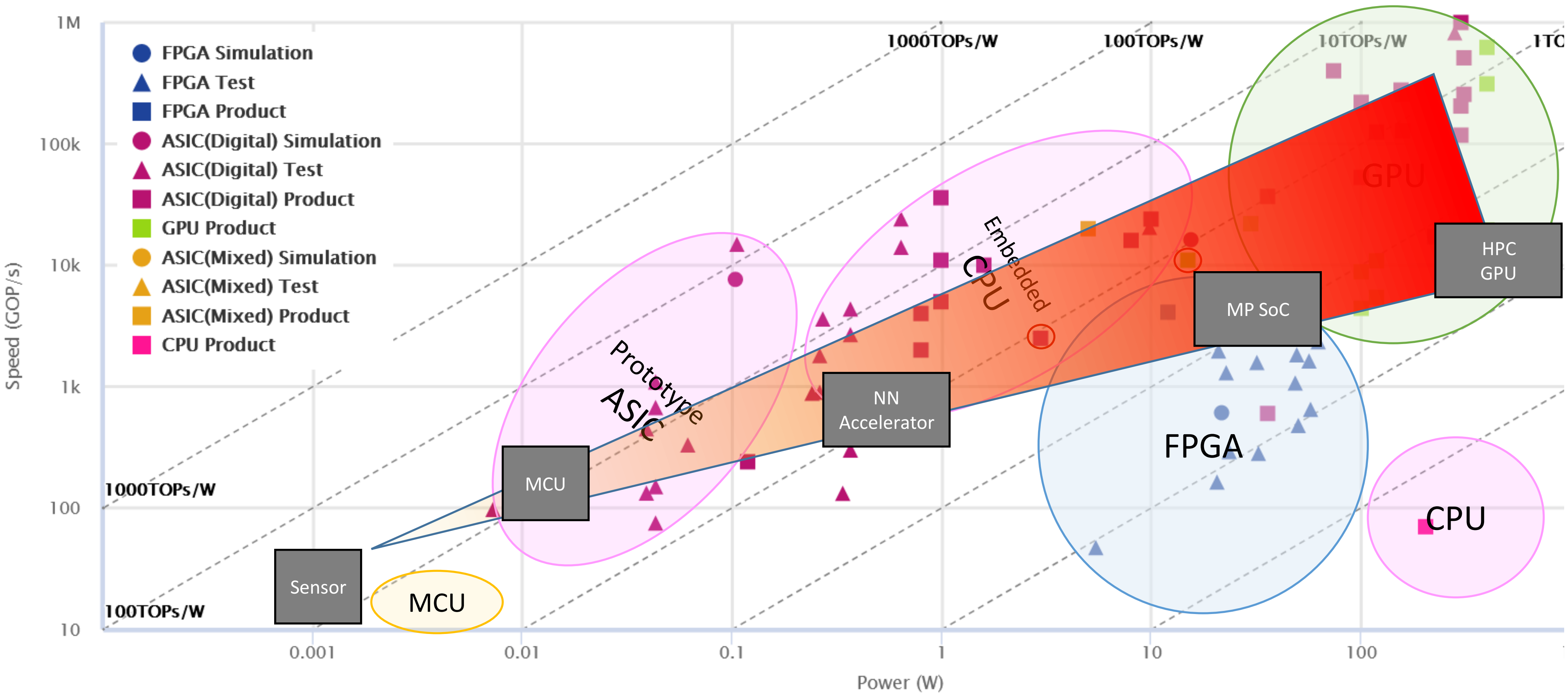


# Digital Neural Network Accelerators




https://nicsefc.ee.tsinghua.edu.cn/projects/neural-network-accelerator/

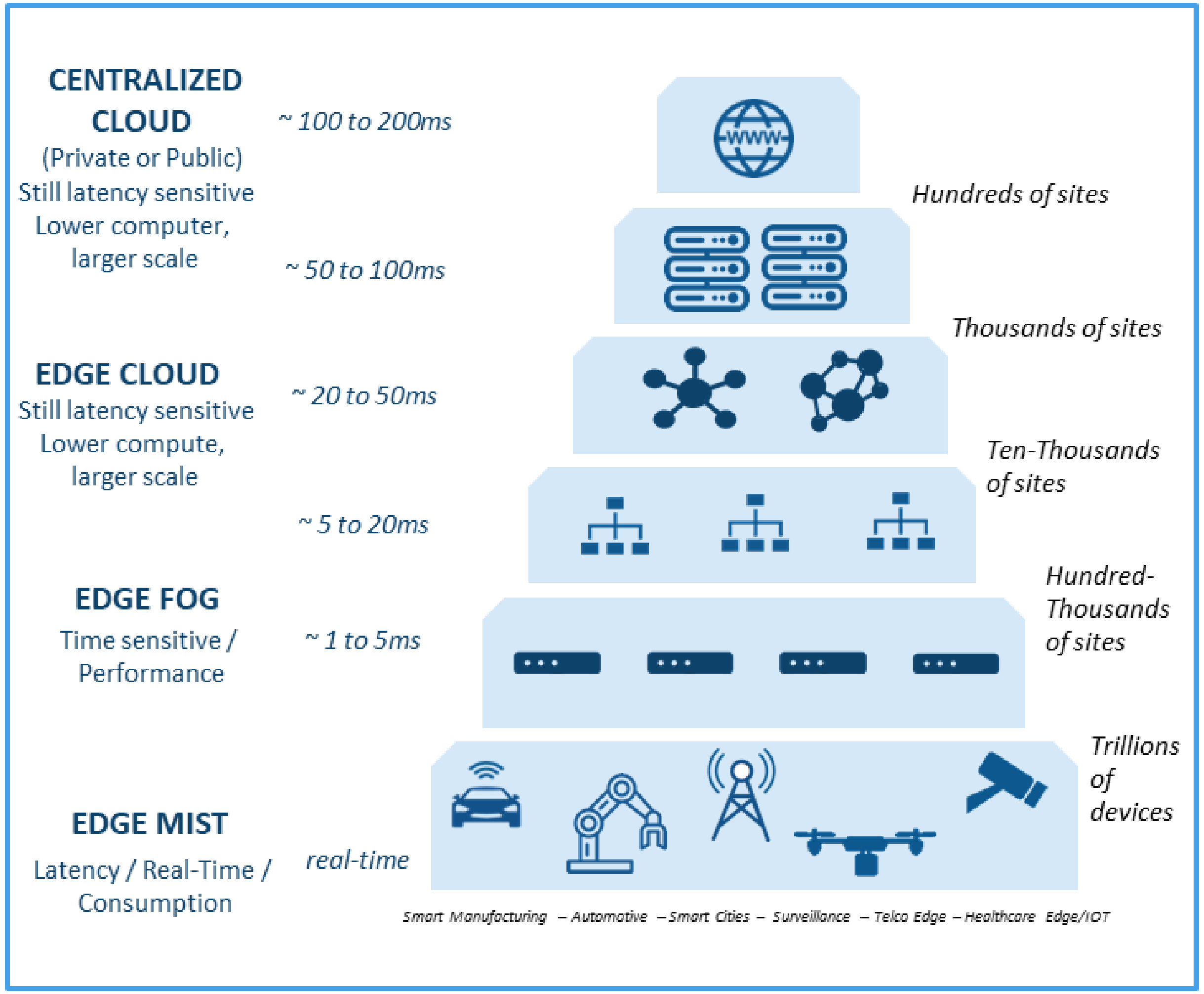
# Digital Neural Network Accelerators




https://nicsefc.ee.tsinghua.edu.cn/projects/neural-network-accelerator/

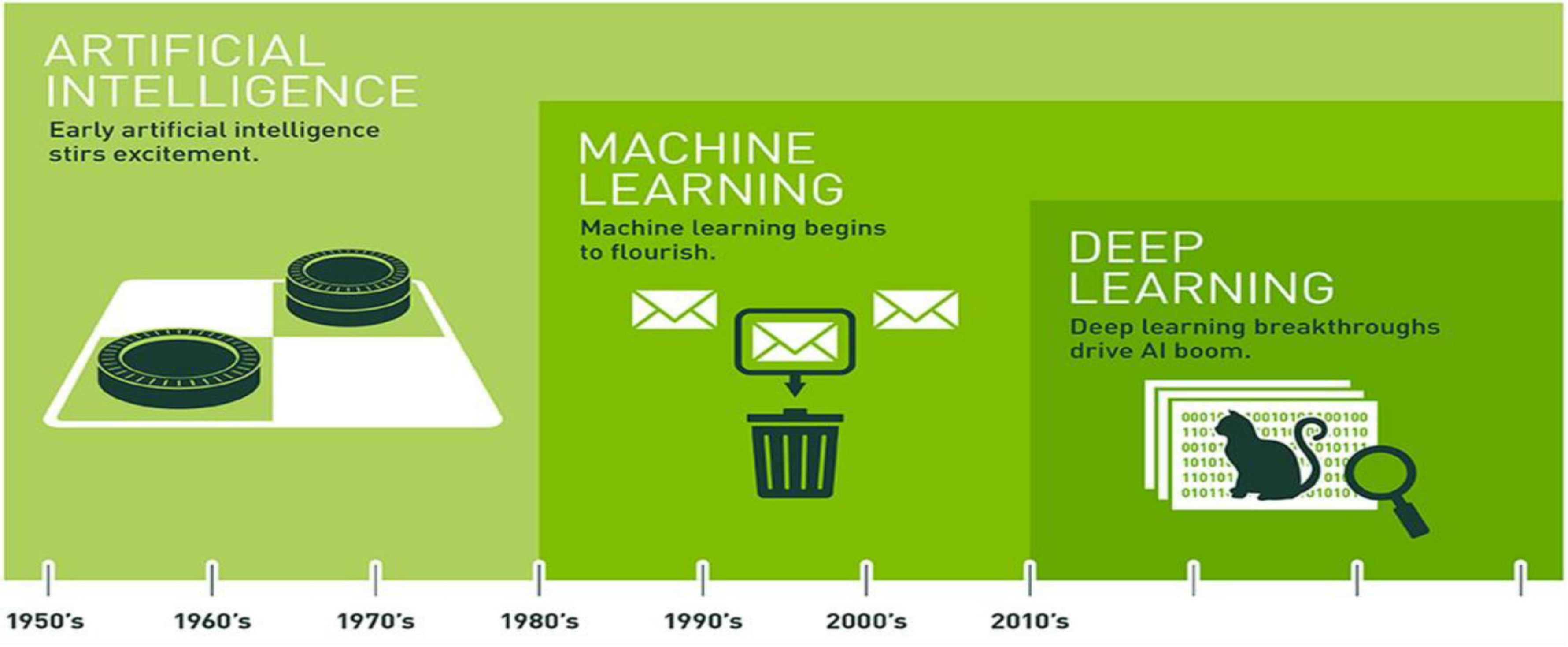
- Specialized chips for AI calculation in the cloud
  - Nvidia GPU, US
  - Google TPU, US
  - Baidu Kunlun, CH
  - GraphCore, EN
  - Intel Movidius, US
  - Cerebras, US => 300.000 cores per wafer, 15kW
- At the Edge
  - NVIDIA Jetson can provide 11 T FLOPs, dissipating up to 15 W
  - Myriad X 4TOPS dissipating up to 1,5 W
  - Google Coral = 4 TOPS for 2W
  - …

# Edge Lines and their specific constraints




| | Memory | Computation | Power | Efficiency |
|---|---|---|---|---|
| Edge Servers | GB | 1 Tops | 100 W | 10 Gops/W |
| Gateway | MB | 100 Gops | 1 W | 100 Gops/W |
| IoT Nodes | Hundreds of kB | 1 Gops | 1 mW | 1 000 Gops/W |

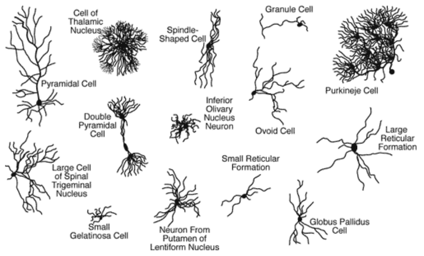
# Deep Learning, the last trend of AI

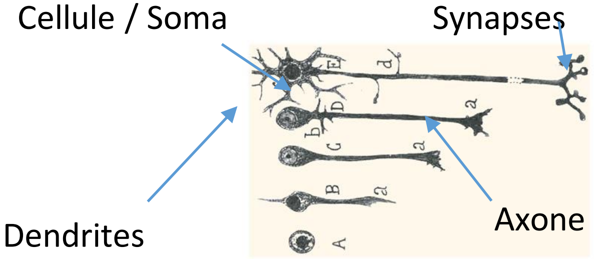


# The short story of neural networks

## 1890, Ramon Y Cajal

Diversity of morphology and behaviour



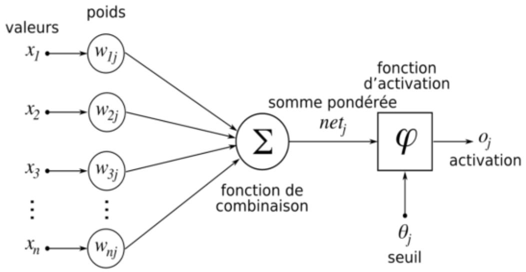




Chemical Communication

Electrical Transmission

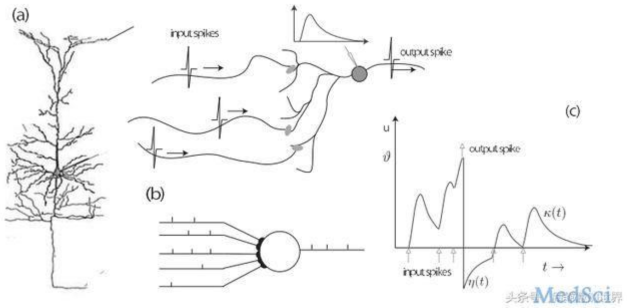
## 1950, Warren McCulloch et Walter Pitts

Binary inputs/outputs



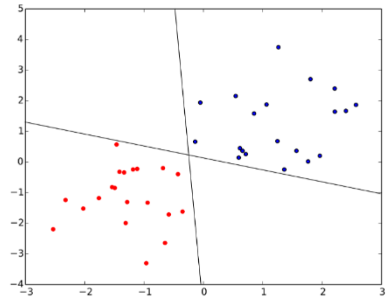

## 1952, Hodgkin Huxley

Spike dynamics




## 1957, Frank Rosebblatt

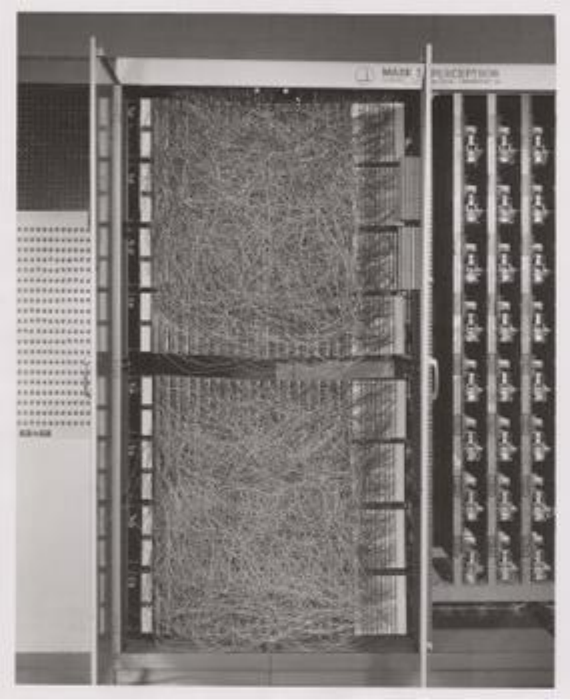
real input, linearly separable data



Cajal, R. S. Recollections of My Life (translated by E. H. Craigie with the assistance of . Cano) (Am. Phil. Soc., Philadelphia, 1937; reprinted by MIT Press, Cambridge, Massachusetts, 1989)

# Three generations of neural networks

- **First generation of neural networks - Perceptron**
  - Formal neuron (Mc Culloh & Pitts)
    - Binary input / output
    - threshold gates
    - Linearly separable data
  - Perceptron (Rosenblatt)
    - Real input/output
    - Single-layer perceptrons
    - Linearly separable data
    - Learning algorithm
- **Second generation – Back Propagation**
  - Multi-layer perceptron
  - the output layer would give a probability value for a givenoutcome
  - Rumelhart backpropagation (BP) trainin galgorithm (Gradient descent), 1986
  - Requires computational units with derivable activation function
- **Third generation - Spiking neuron**
  - Bio-inspired modeling
  - Discrete representation of inputs/outputs
  - Time representation
  - Internal state

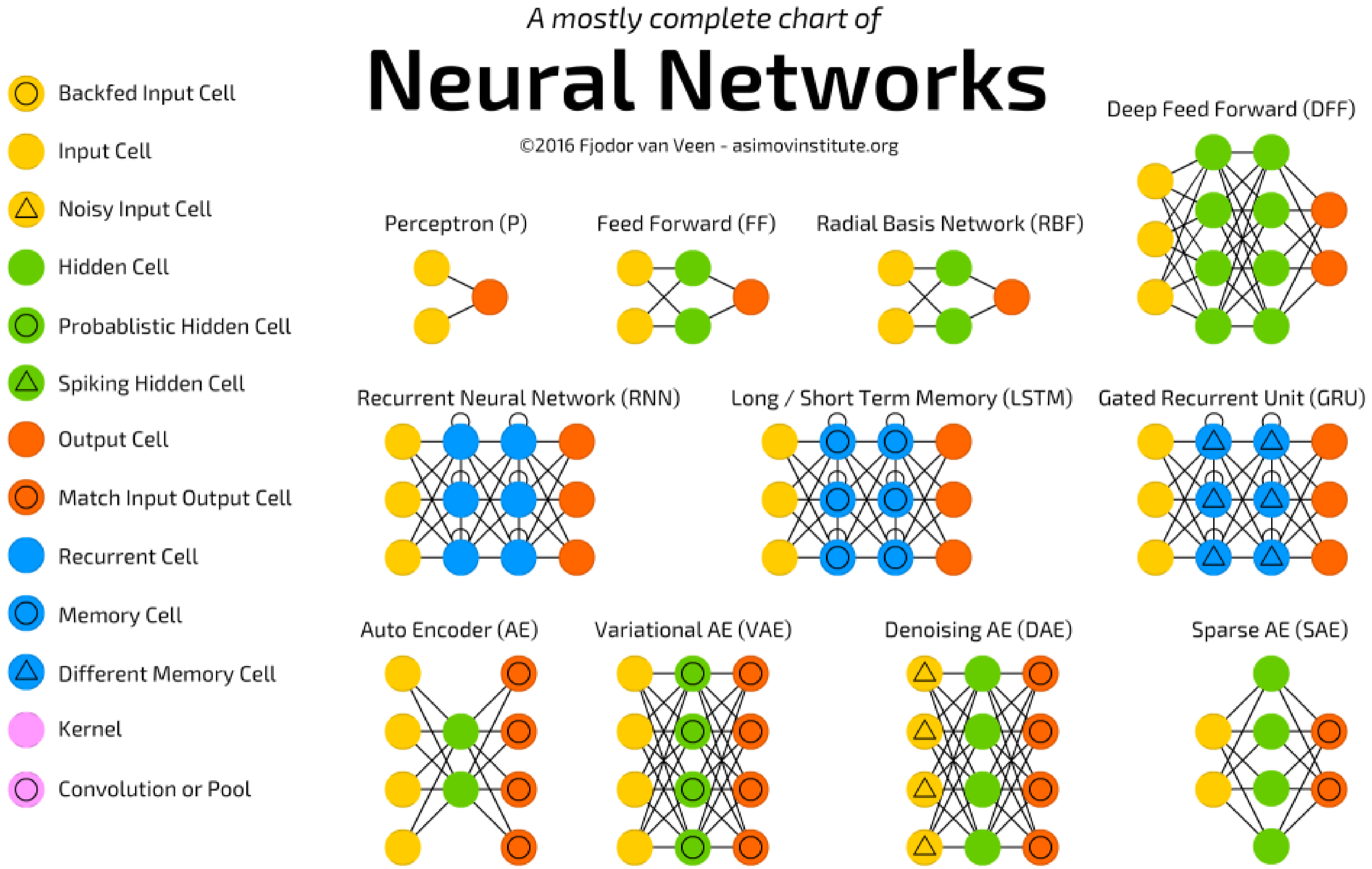


*1958, Mark I Perceptron machine, Cornell*



*W. Mass, Networks of Spiking Neurons: The Third Generation of Neural Network Models, 1996*

# ANN, one method of Machine Learning

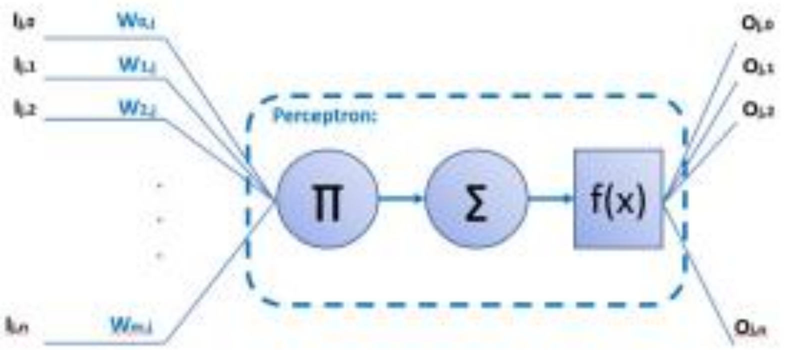


# Perceptron algorithm

- Each change of **w** decreases the error on a specific point. However, changes for several points are correlated, that is different points could change the weights in opposite directions. Thus, this iterative algorithm requires several loops to converge.
- Guarantee to find a separating hyperplane if one exists–if data is linearly separable
- If data are not linearly separable, then this algorithm loops indefinitely

$$s_j^l(t) = \sum_{i=0}^{N_{l-1}} w_{ij}^l(t) \times y_i^{l-1}(t)$$

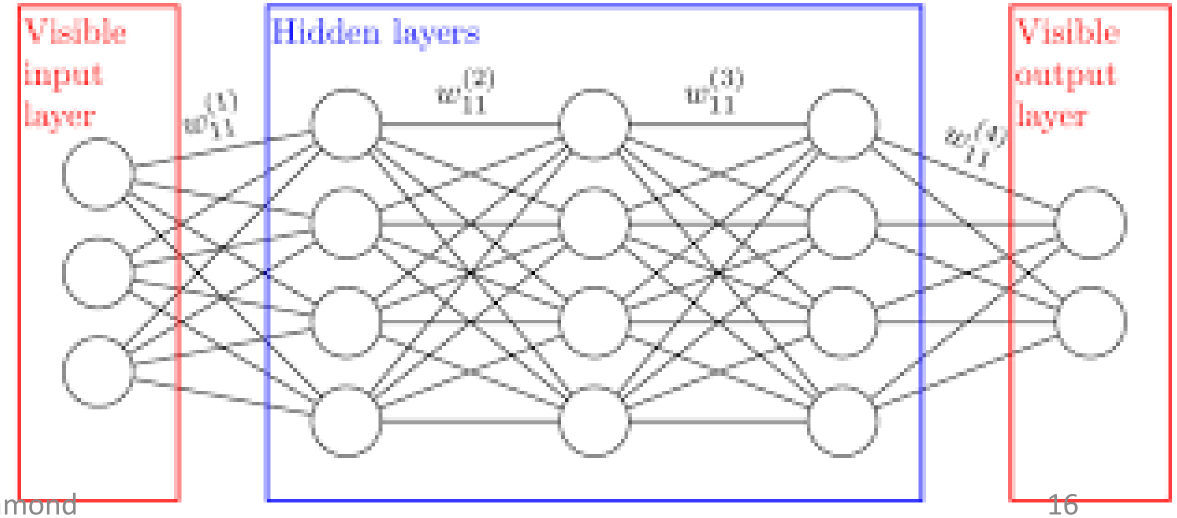
$$y_j^l(t) = f(s_j^l(t)),$$




*Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386.*

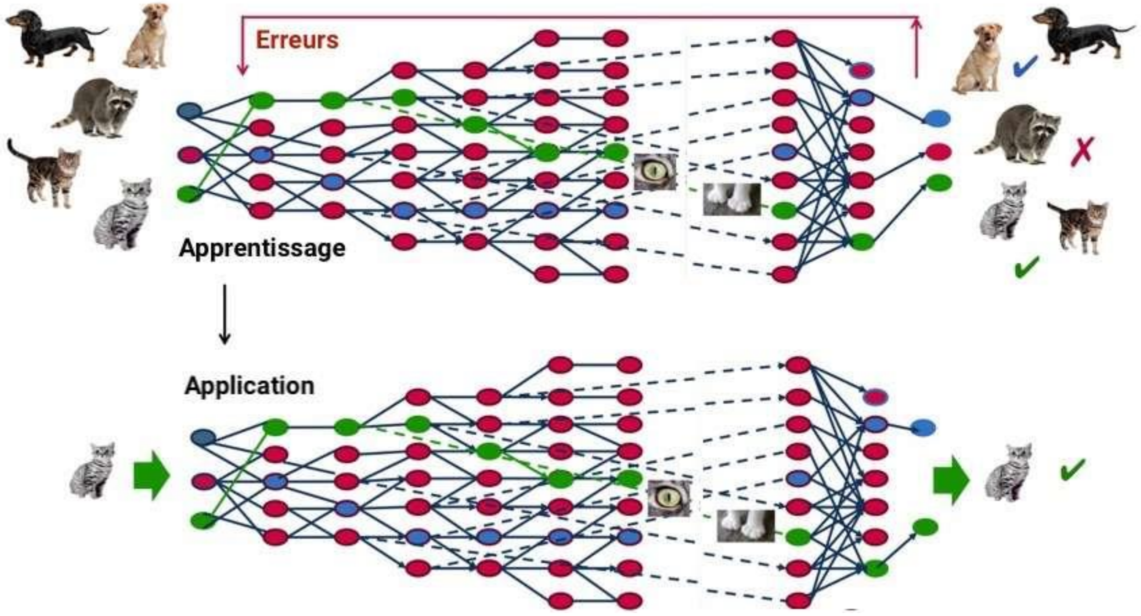
# MLP: Multi-layer perceptron

- **Solution:** Combine multiple linear separators.
- Introduction of "hidden" units into NN make them much more powerful: they are no longer limited to linearly separable problems.
- Earlier layers transform the problem into more tractable problems for the latter layers.
- Learning takes place by adjusting the weights in the network, so that the desired output is produced whenever a training instance is presented.

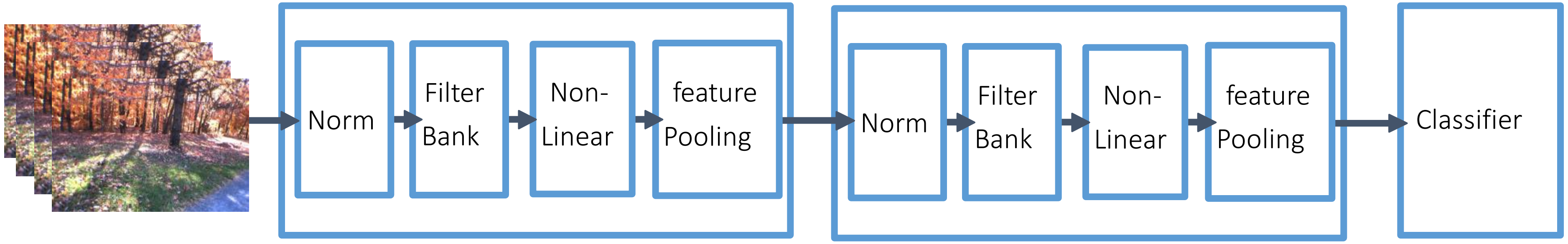


# Deep learning

- Often: inputs **x** are **raw signals** or **feature vectors**,
- Often: outputs **y** are vectors which **highest value** indicate the **category of the input**.
- Instead of directly mapping **x** to **y**, constructs a graph of intermediate representations, associated through very simple mathematical functions called **layers**,
- Training: Backpropagate the gradient of the loss throughout the architecture.

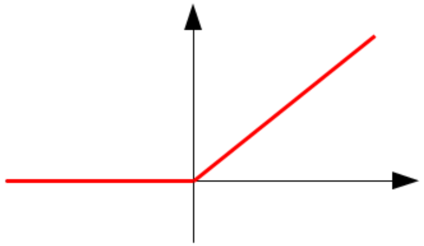


# Overall architecture: multiple stages of Normalization → filter bank → non-linearity → pooling




- Normalization: variations on whitening
  - Subtractive: average removal, high pass filtering
  - Divisive: local contrast normalization, variance normalization
- Filter bank: dimension expansion, projection on overcomplete basis
- Non-linearity: sparsification, saturation, lateral inhibition....
  - Rectification (relu), component-wise shrinkage, tanh,..

    ***ReLU (x )=max (x , 0)***

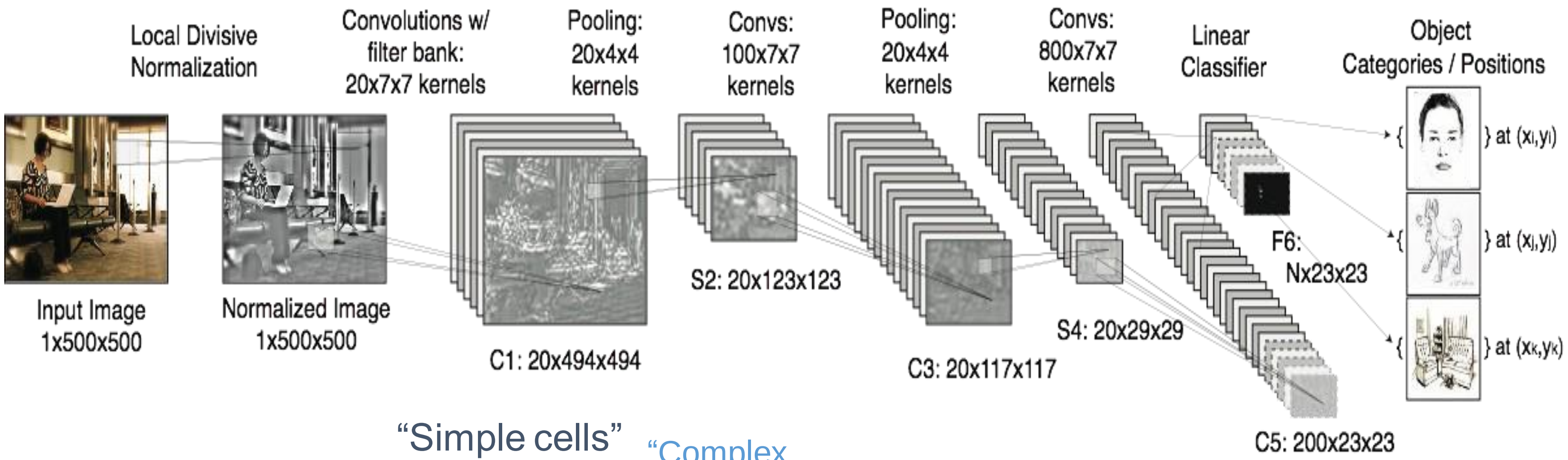


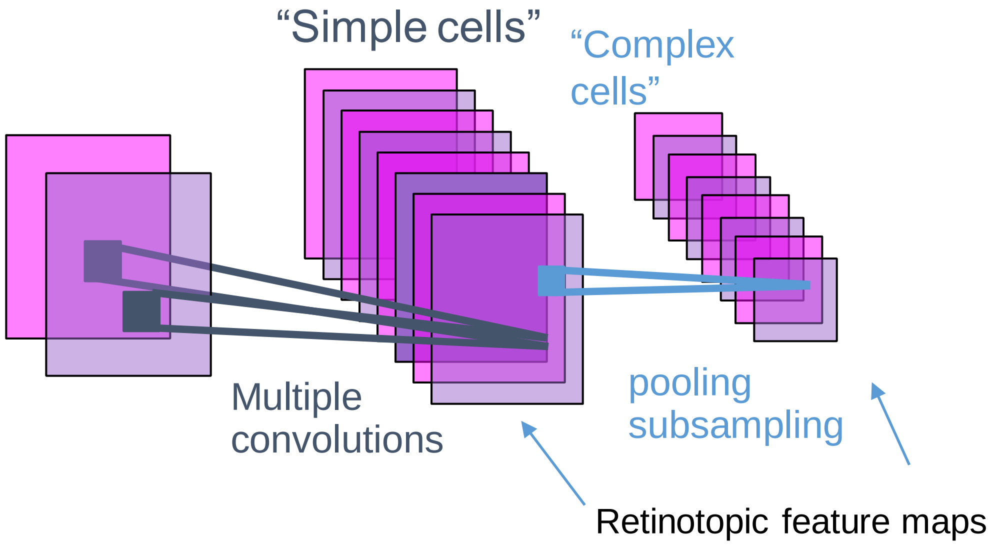
- Pooling: aggregation over space or feature type
  - Max, Lp norm, log prob.

$$X_i; \quad L_p: \sqrt[p]{X_i^p}; \quad PROB: \frac{1}{b}\log\left(\sum_i e^{bX_i}\right)$$

# The convolutional net model

## (Multistage Hubel-Wiesel system)



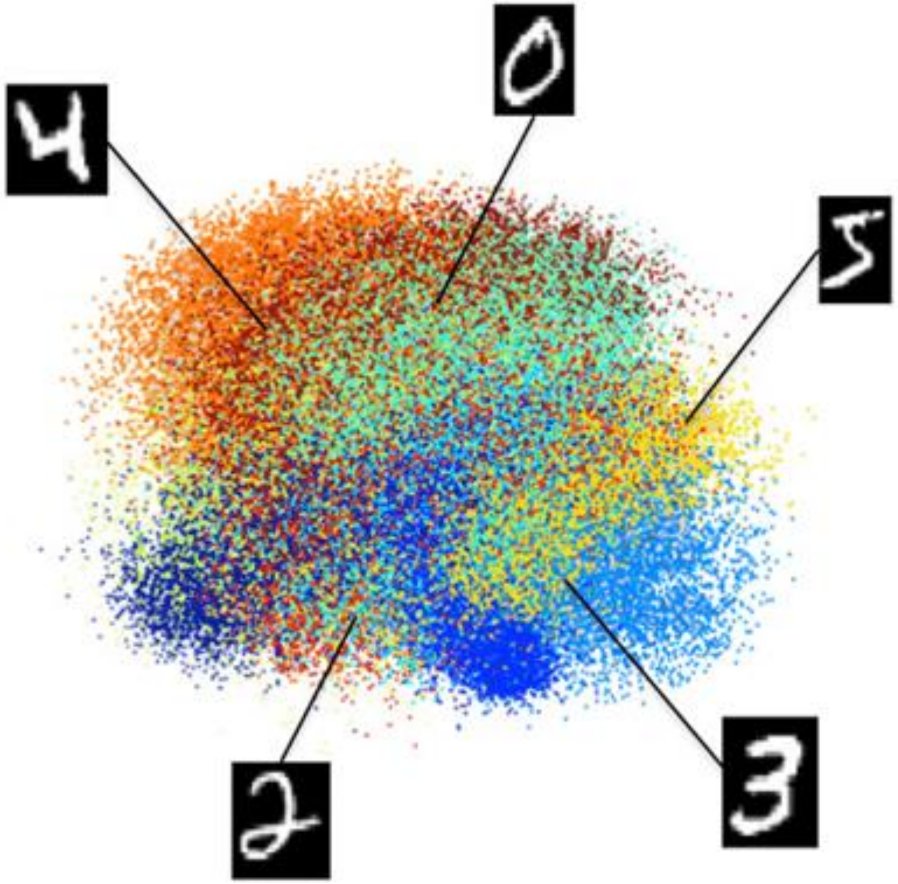




- Training is supervised
- With stochastic gradient descent

[LeCun et al. 89]

[LeCun et al. 98]

# Example of data, the MNIST dataset non-linearly separable data



# EdgeAI, let’s play !

The field of possibilities is only limited by your imagination

**Plug**

**Train**

**Embed**

**Play**

**Repeat …**


IDEX **<u>Sith</u>** project, F. Ferrero, L. Rodriguez, B. Miramond