Master in Internet of Things for eHealth

M5. Smart Data Knowledge / Analytics

Deep Learning
(Introduction)

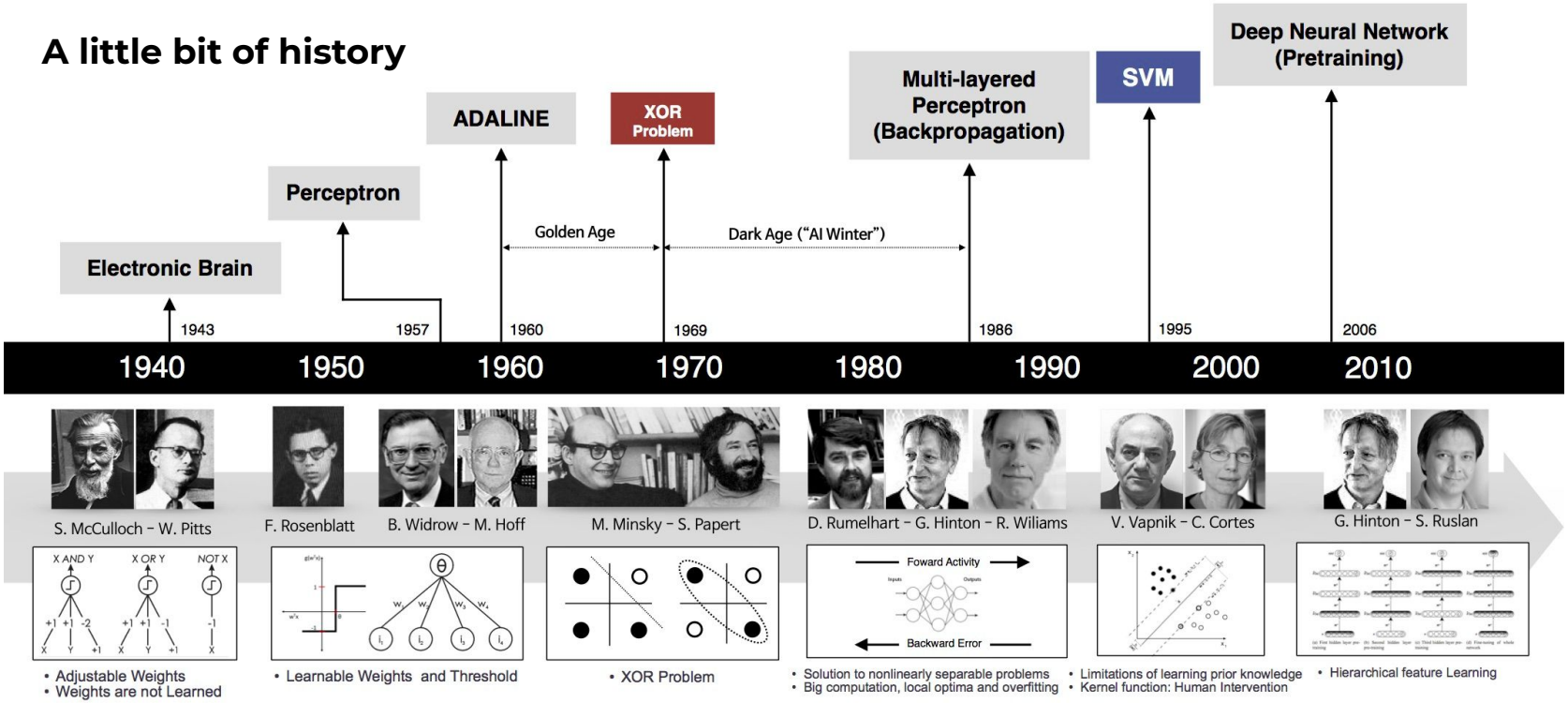
Instructor **David Gerónimo**
research@davidgeronimo.com

UAB Universitat Autònoma de Barcelona

January 21st 2019

Introduction

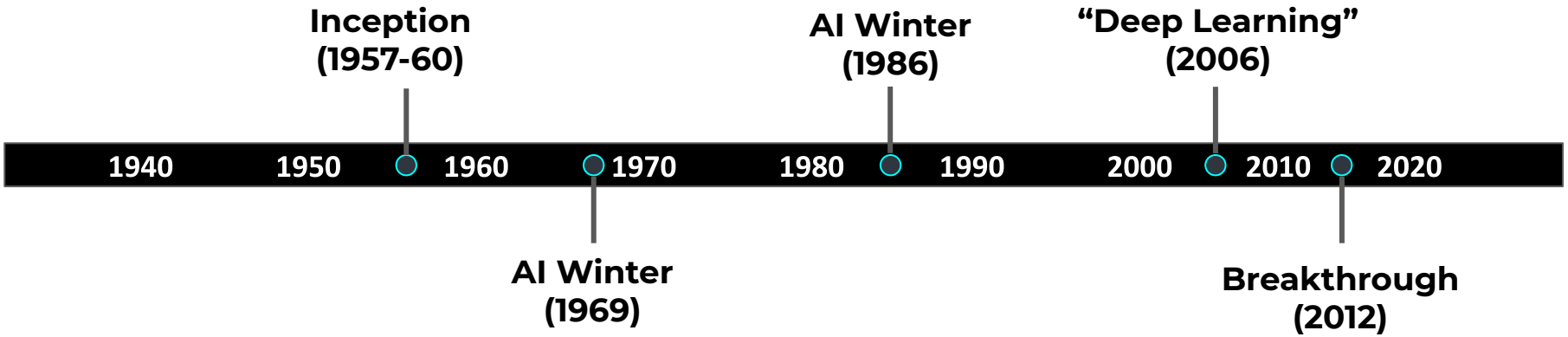
- **A little bit of history**



(Unknown author, taken from Andrew L. Beam slides)

Introduction

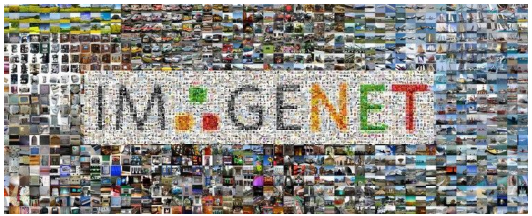
- A little bit of history

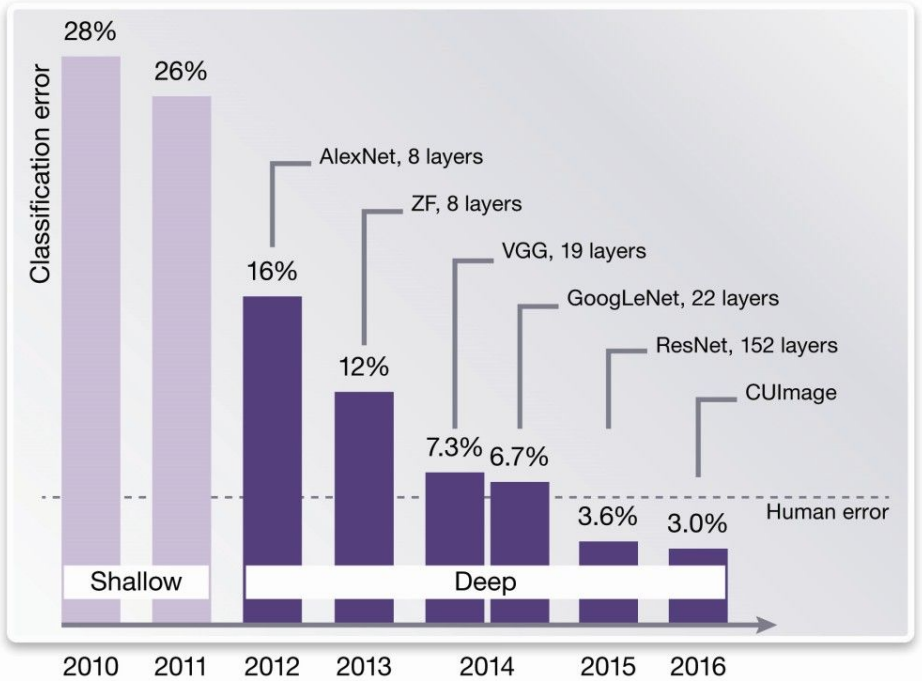


Introduction

- **A little bit of history**

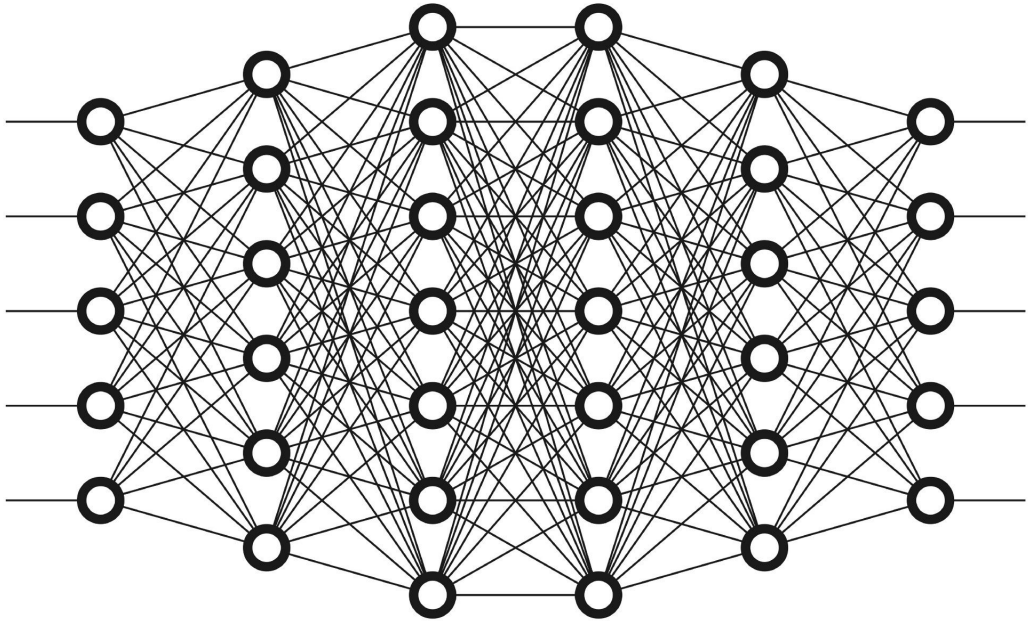
(ILSVRC)
Internet Large Scale Visual Recognition Competition





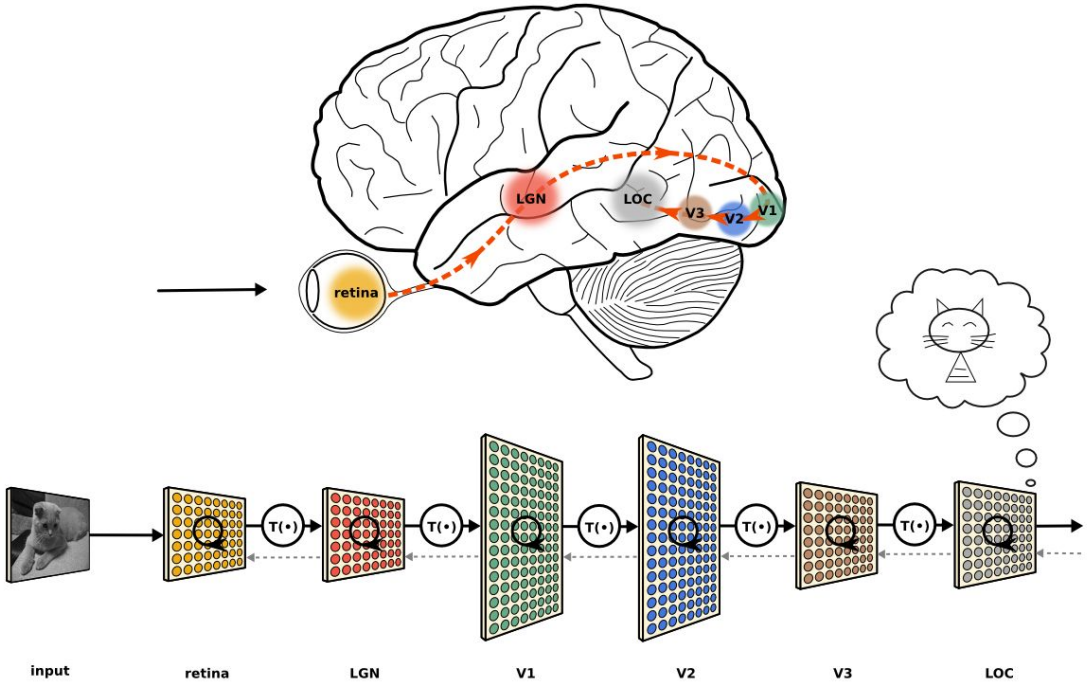
Introduction

- **Deep Neural Network: Hierarchy of multiple layers of artificial neurons that processes information using non-linear transformations.**



Introduction

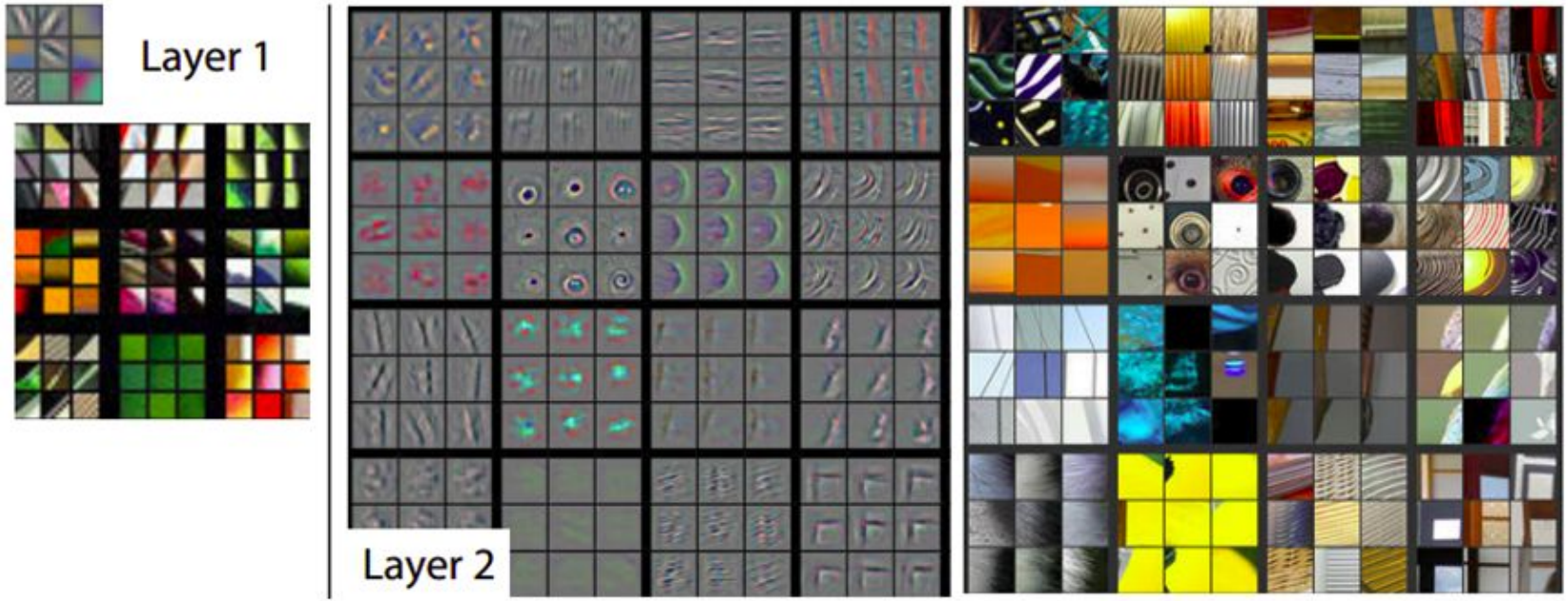
- **Hierarchy of neuron layers that mimic the brain**



(c) Jonas Kubilius

Introduction

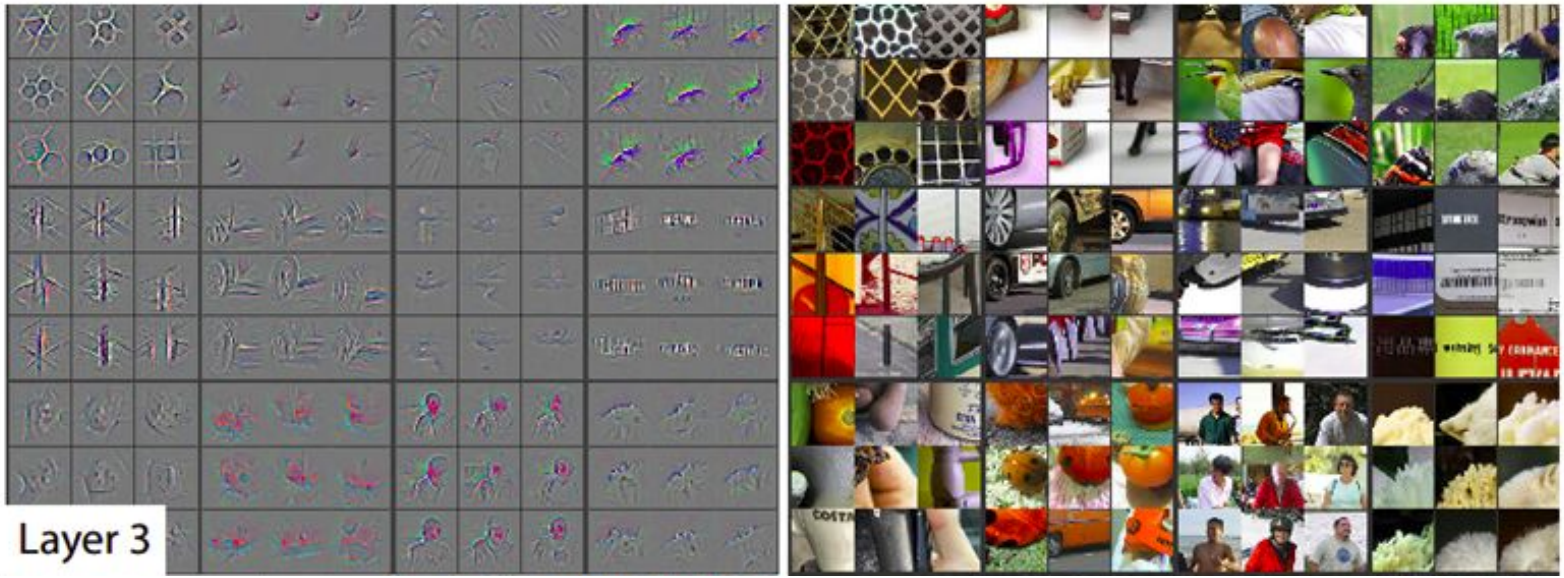
- **Visualization of the different layers**



(c) Rob Fergus

Introduction

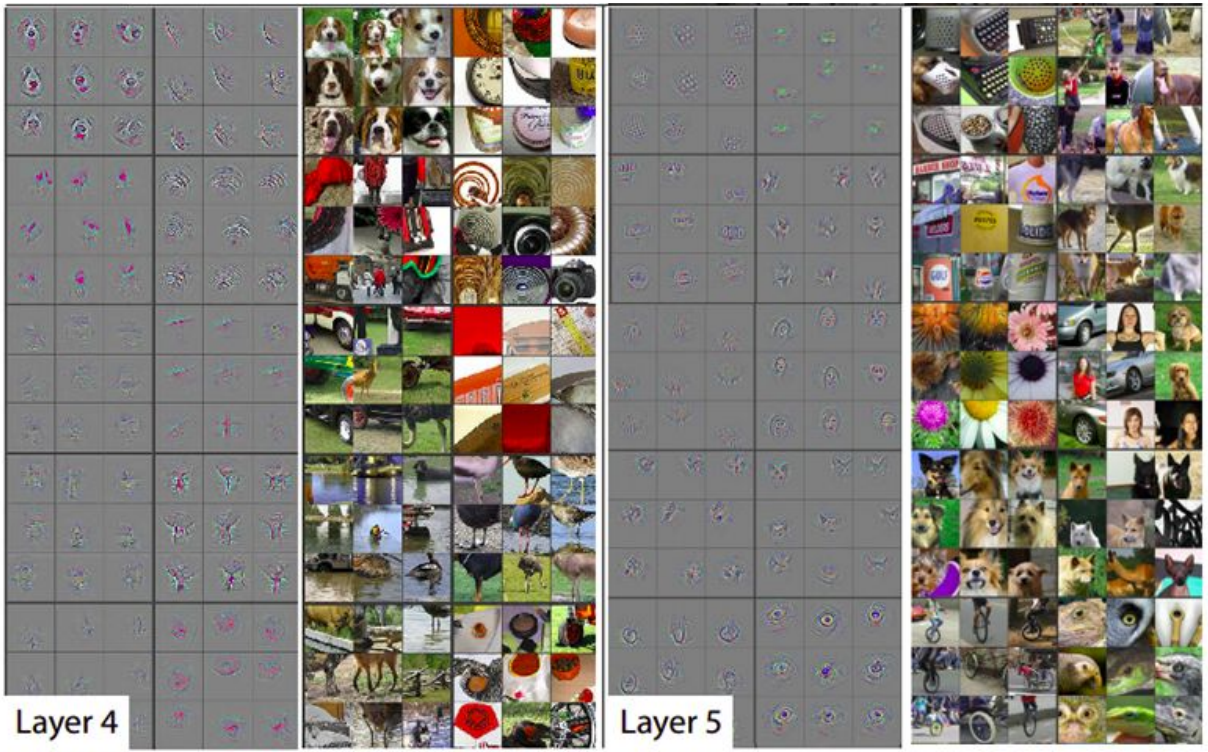
- **Visualization of the different layers**



(c) Rob Fergus

Introduction

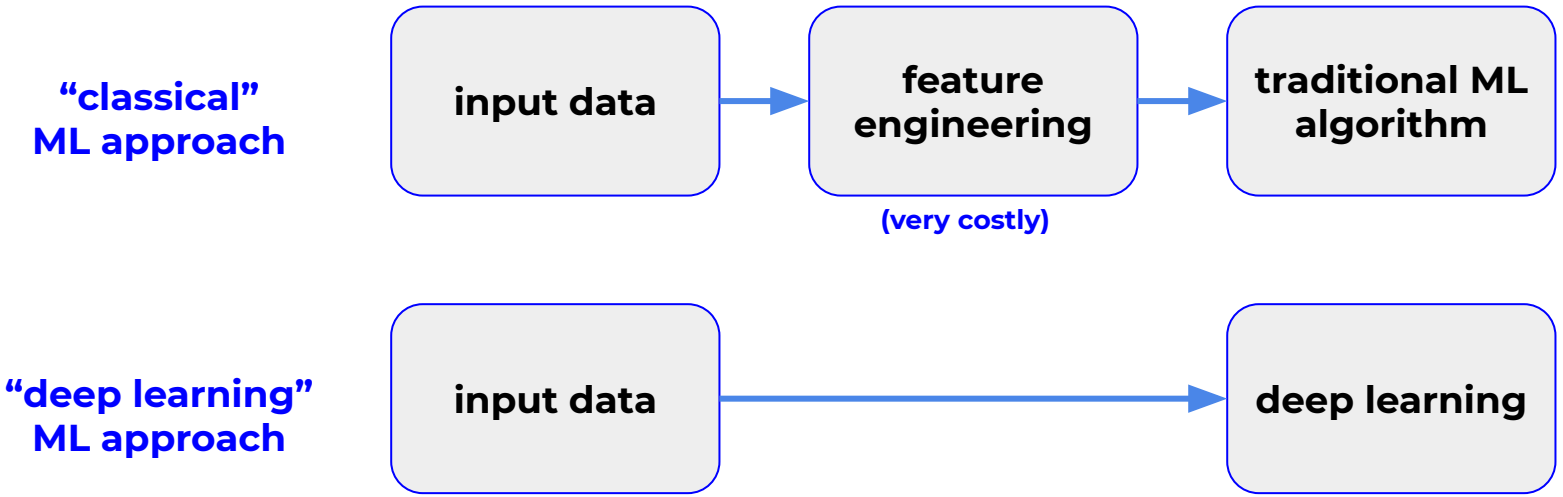
- **Visualization of the different layers**



(c) Rob Fergus

Introduction

- Avoids “feature engineering”

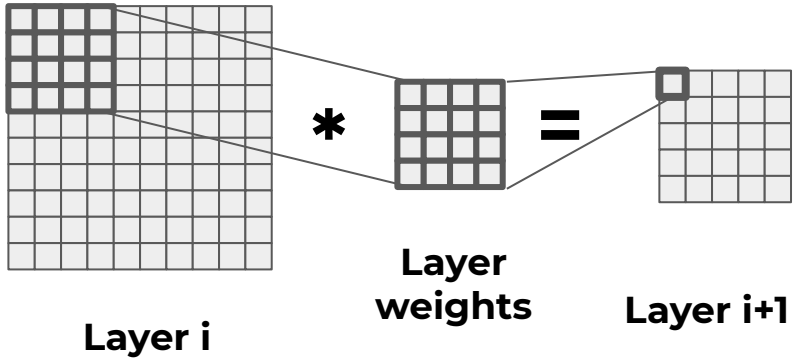


Some concepts

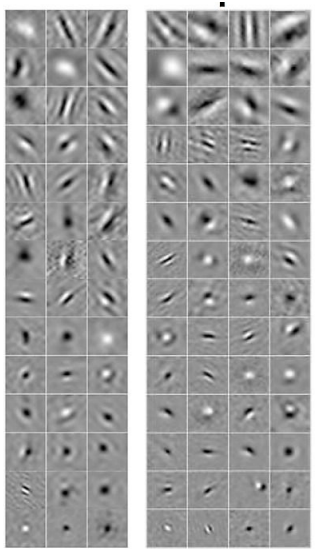
- **Mini-batch:** Set of few samples for which the gradient is averaged in a forward-backward pass.
- **Iteration:** One forward-backward pass using a batch of samples.
- **Epoch:** Once all the model has seen all the samples of the training set.
- **Capacity:** How much information can the network store.
- **Learning Rate:** Amount of optimization per iteration.
- **Cost/Loss function:** Function that maps predictions vs groundtruth (reality) to a number.
- **Gradient Descent:** Optimization algorithm for minimizing the loss function.

Convolutional Neural Networks (CNN)

- **Concept: instead of working on “single” neurons, it works on neurons in a region**



LeCun (1990s)

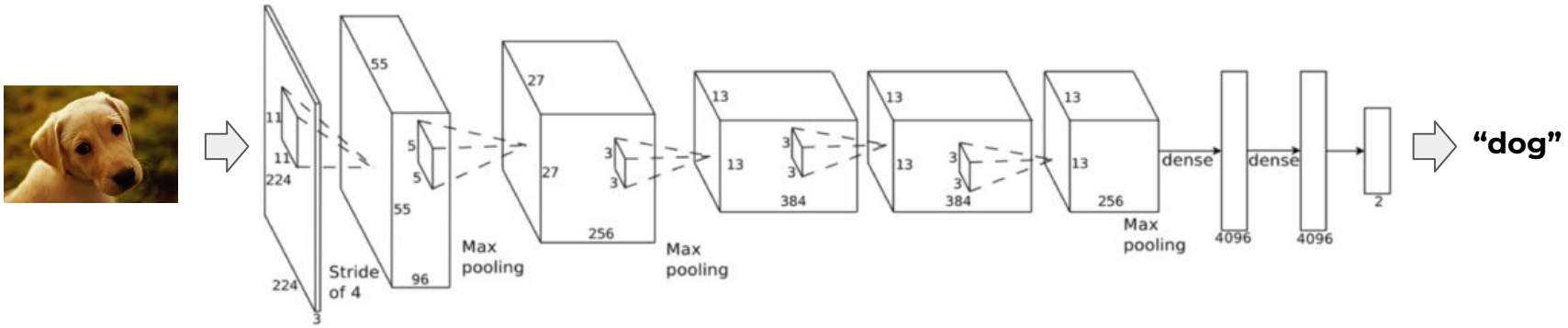


Biologically inspired (Receptive Fields of macaque V1 neurons)

(c) Joel Zylberberg et al.

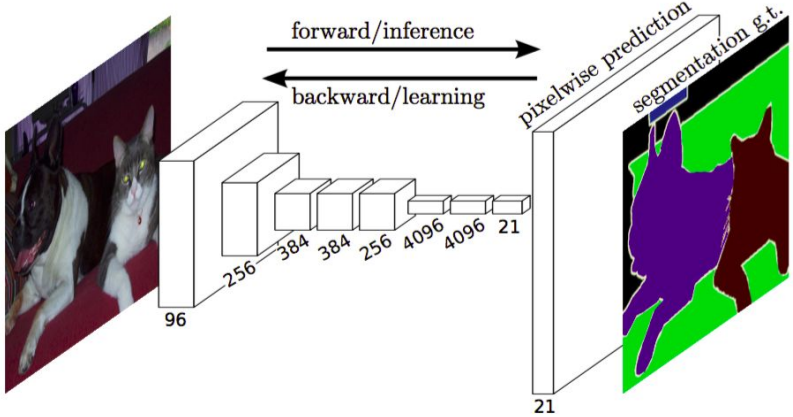
CNN Applications

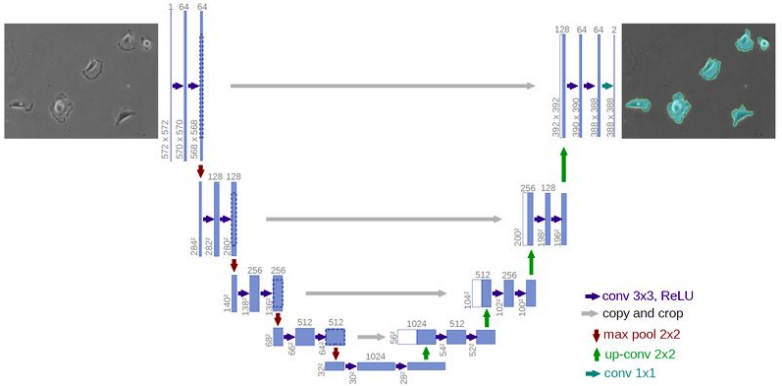
- **Classification**



CNN Applications

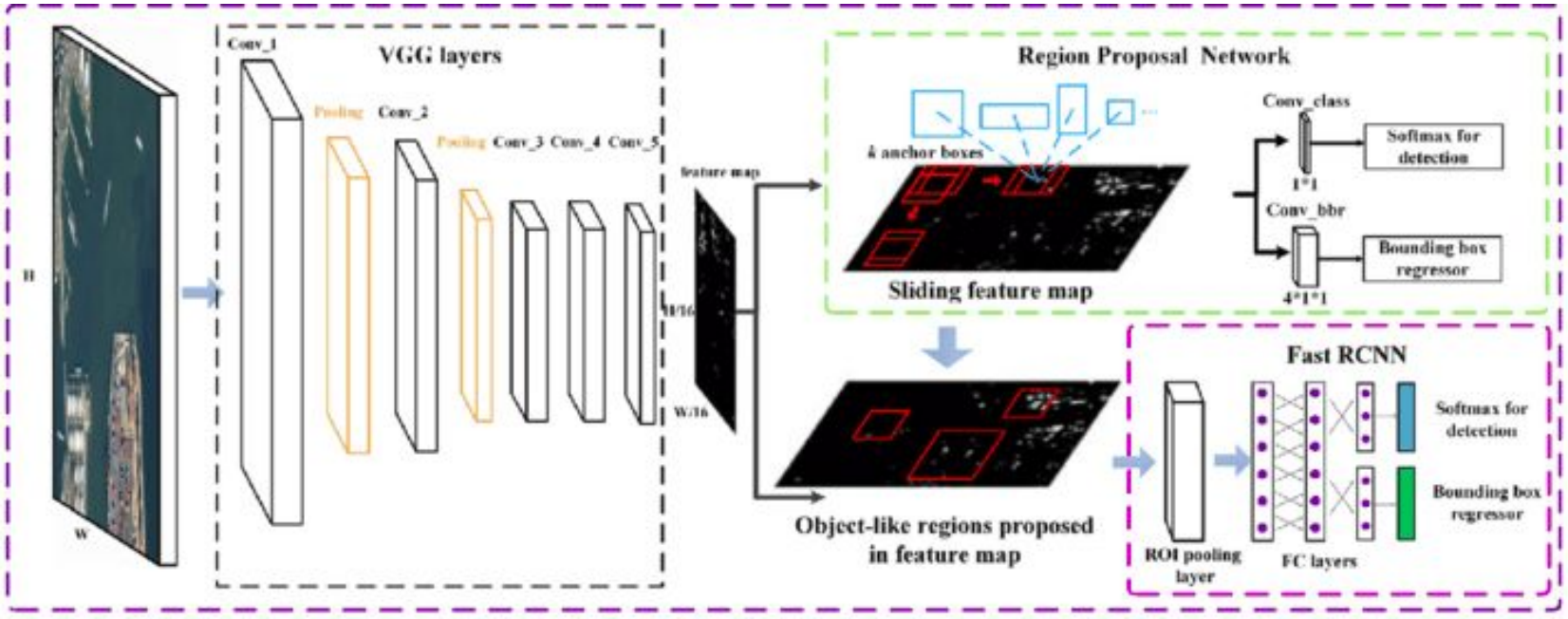
- **Segmentation**





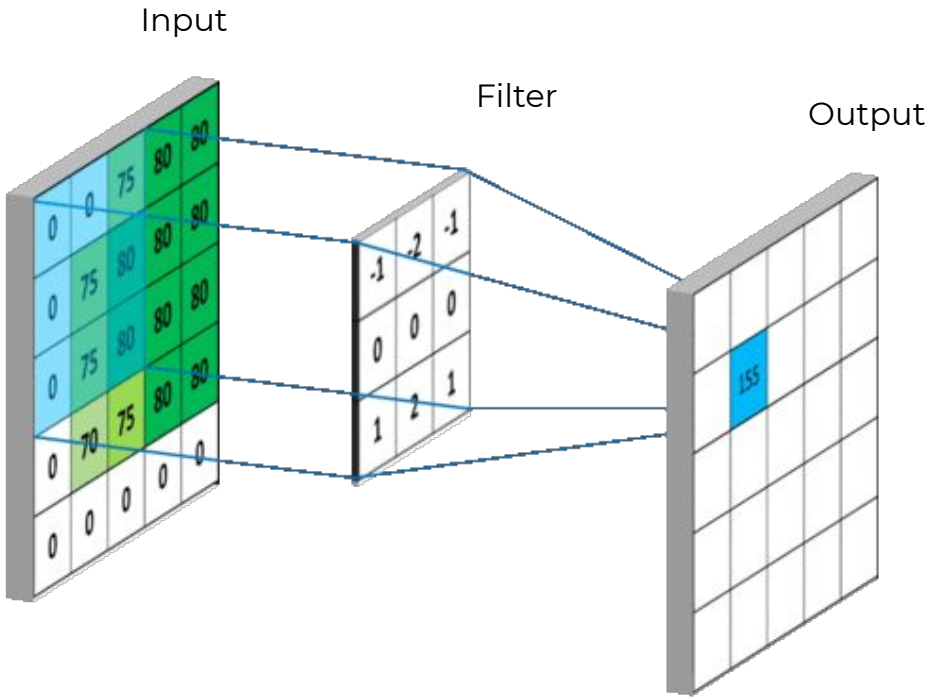
CNN Applications

- **Detection**



Convolutional Neural Networks (CNN)

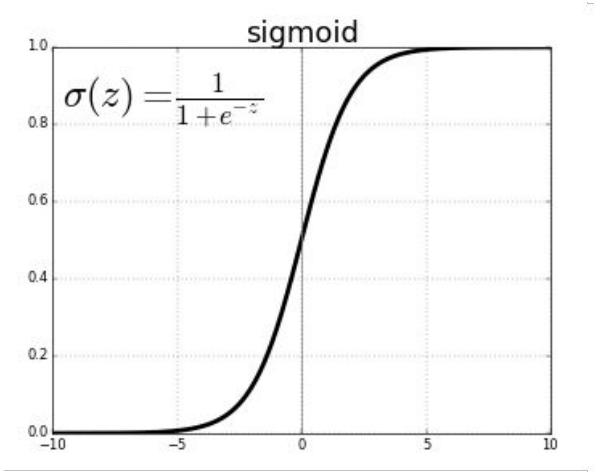
- **Typical modules (layers)**
  - **SpatialConvolution**
  - ReLu
  - Pooling
  - Fully Connected

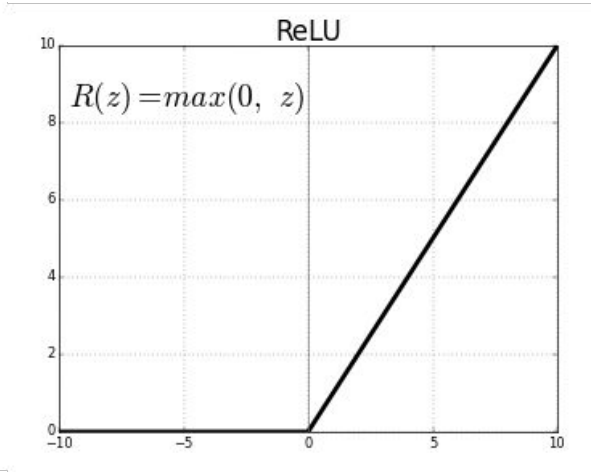


Convolutional Neural Networks (CNN)

- **Typical modules (layers)**
  - SpatialConvolution
  - **ReLu**
  - Pooling
  - Fully Connected

- pro: Better gradient propagation (fewer vanishing gradients)
- con: Non-differentiable at 0





(c) towardsdatascience.com

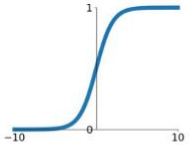
Convolutional Neural Networks (CNN)

- **Typical modules (layers)**
  - SpatialConvolution
  - **ReLu**
  - Pooling
  - Fully Connected

Other activation functions

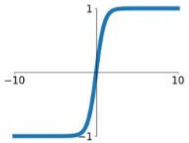
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



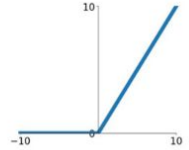
tanh

$$\tanh(x)$$



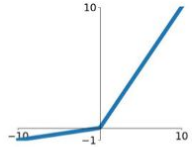
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

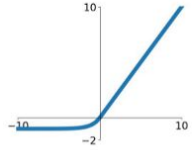


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

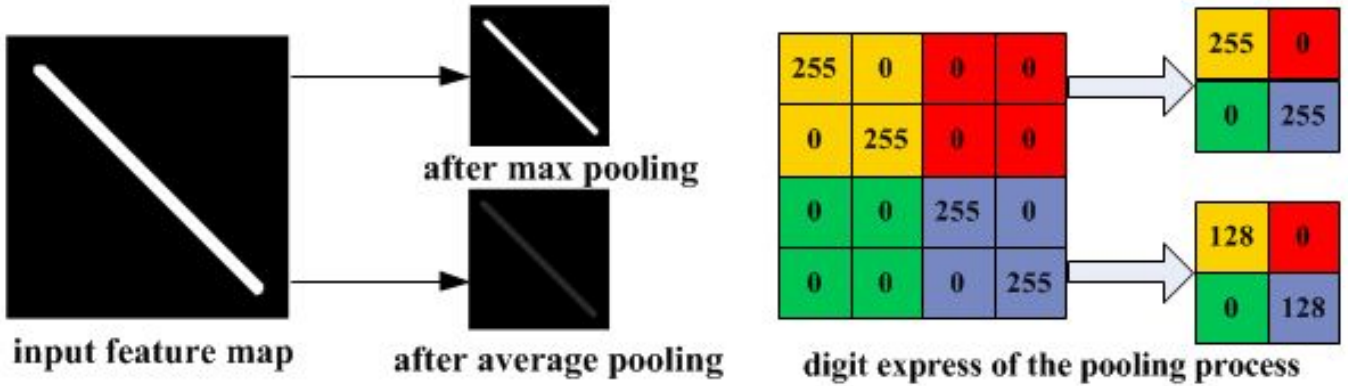
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



(c) Shruti Jadon @ Medium

Convolutional Neural Networks (CNN)

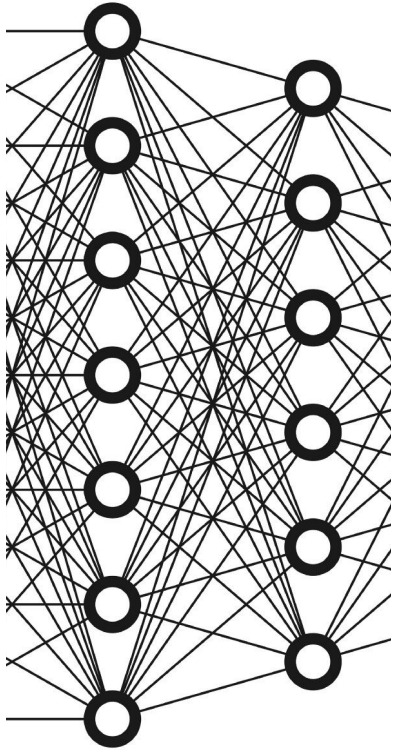
- **Typical modules (layers)**
  - SpatialConvolution
  - ReLu
  - **Pooling**
  - Fully Connected



(c) Hanly Wang

Convolutional Neural Networks (CNN)

- **Typical modules (layers)**
  - SpatialConvolution
  - ReLu
  - Pooling
  - **Fully Connected**



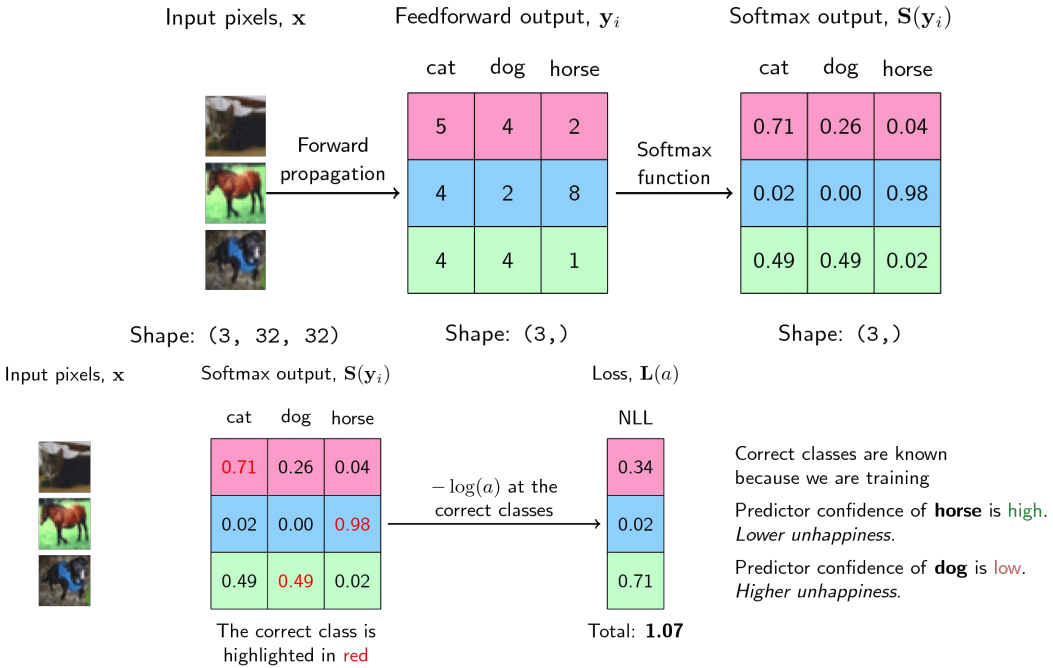
Also known as “Linear”

Convolutional Neural Networks (CNN)

- **Typical criteria**
  - **Negative Log Likelihood (NLL)**
  - Binary Cross Entropy (BCE)
  - Mean Square Error (MSE)

SoftMax

$$S(f_{y_i}) = \frac{e^{f_{y_i}}}{\sum_j e^{f_j}}$$



Convolutional Neural Networks (CNN)

- **Typical criteria**
  - **Negative Log Likelihood (NLL)**
  - Binary Cross Entropy (BCE)
  - Mean Square Error (MSE)

$$loss(o, t) = -x_t$$

Convolutional Neural Networks (CNN)

- **Typical criteria**
  - Negative Log Likelihood (NLL)
  - **Binary Cross Entropy (BCE)**
  - Mean Square Error (MSE)

$$loss(o,t) = -\frac{1}{n}\sum_{i}(t_i \log(o_i)))+(1-t_i)(log)(1-o_i)$$

Convolutional Neural Networks (CNN)

- **Typical criteria**
  - Negative Log Likelihood (NLL)
  - Binary Cross Entropy (BCE)
  - **Mean Square Error (MSE)**

$$loss(o,t) = \frac{1}{n}\sum_{i} |o_i - t_i|^2$$