UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

Machine Learning in Biometrics

Deep Learning in Biometrics

Ruggero Donida Labati

Academic year 2020/2021

1

Content

2

1. Research trends

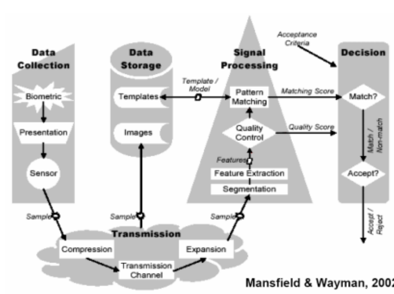
1.1) Research trends in biometrics

Research Trends

- Increase the accuracy
- Multimodal and multibiometric systems
- Reduce the sensor costs
- Less-cooperative acquisition techniques
- Increase the usability and user's acceptance
- Increase of the distances from the sensors
- Continuous authentication
- Physiological signals
- Adaptive biometrics
- Security and privacy
- New biometric traits
- Mobile and wearable biometrics

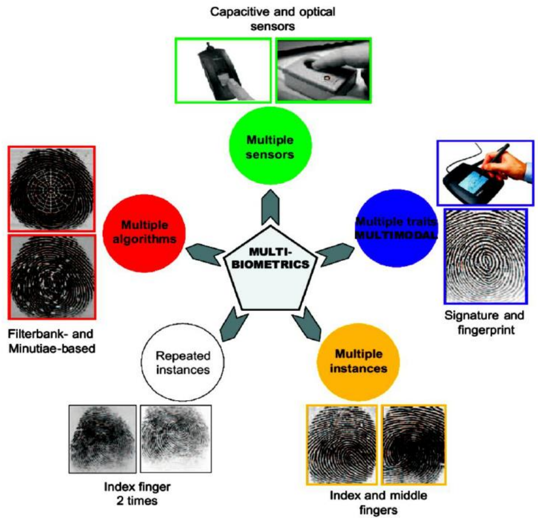
Increase the Accuracy

- Enhancement
- Segmentation
- Feature extraction
- Matching
- Classification methods
- Deep learning



Mansfield & Wayman, 2002

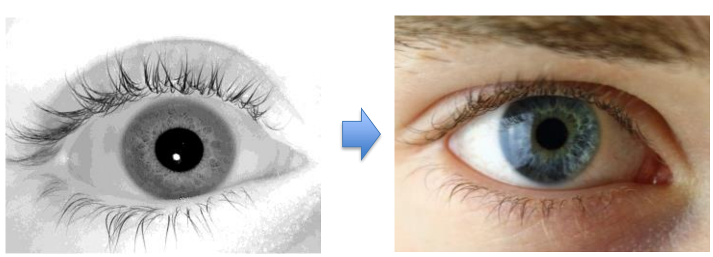
Multimodal Systems



Advantages
- Accuracy
  - FNMR, FMR
- Enrollment
  - FTE
  - Universal registration requirements
- Anti-Spoofing

Reduce the Sensor Costs

- New application scenarios
- Wider diffusion of the technology



Less-cooperative Acquisition Techniques

Increase the Usability and User's Acceptance

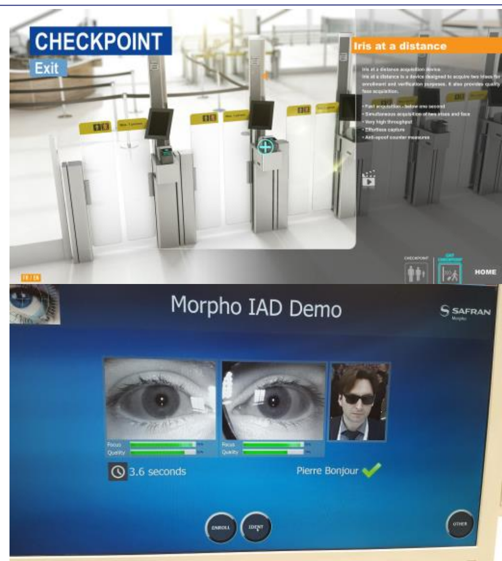
- ISO 9241-11: The usability is the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use
- User acceptance: subjective

Figure 17 Tall Participant Struggling at 99.1 cm (39 in.) and 30°

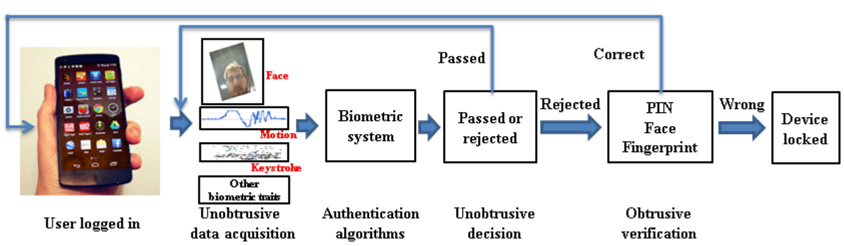
- https://www.nist.gov/sites/default/files/nistir-7504-height-angle.pdf

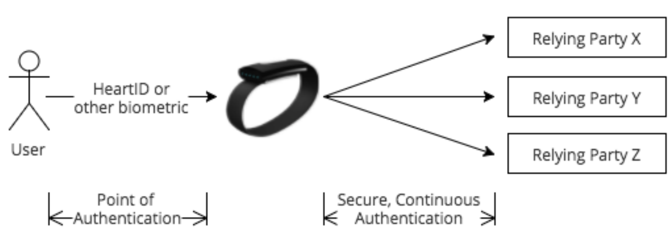
Ruggero Donida Labati - Deep Learning in Biometrics - 2020-201
Ph.D. degree in Computer Science, Università degli Studi di Milano
UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

Increase of the Distances from the Sensors

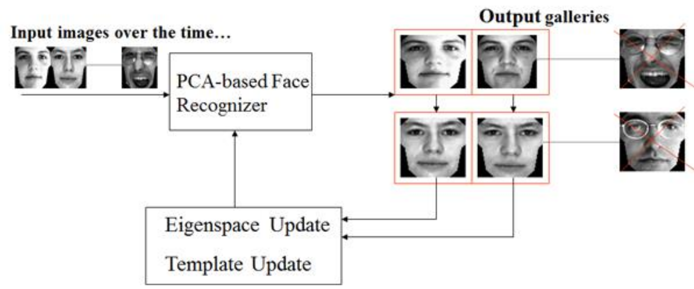


Continuous Authentication



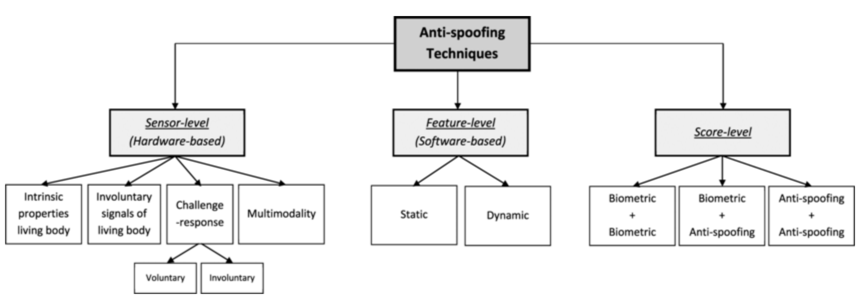


Adaptive Biometrics



13

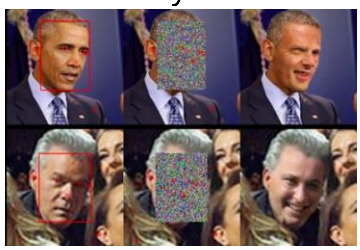
Security and Privacy: Anti-spoofing



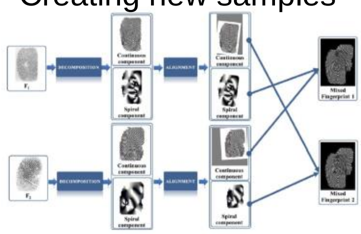
14

Security and Privacy: Sample Protection

Anonymization

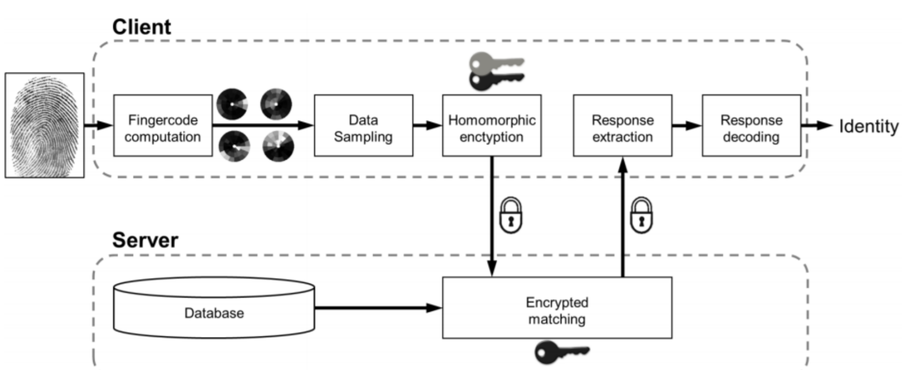


Creating new samples

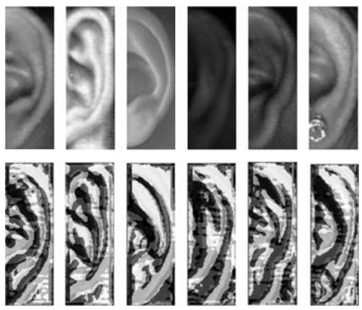


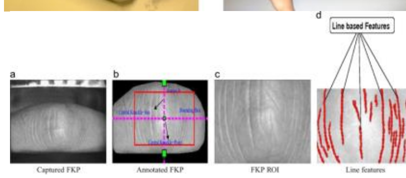
Encryption

Security and Privacy: Template Protection

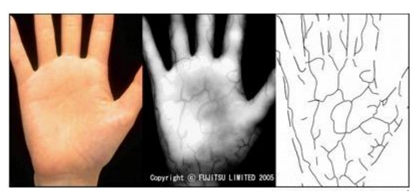


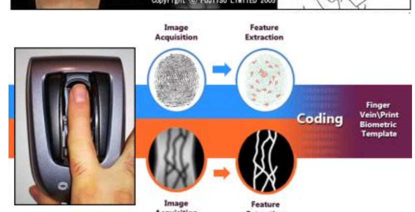
New Biometric Traits (1/5)





New Biometric Traits (2/5)

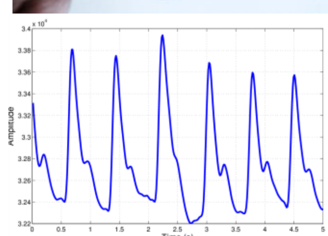




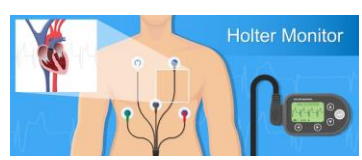
New Biometric Traits (3/5)

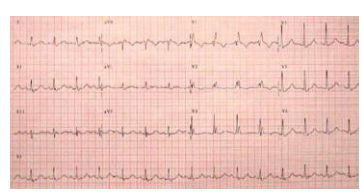
New Biometric Traits (4/5)

PPG

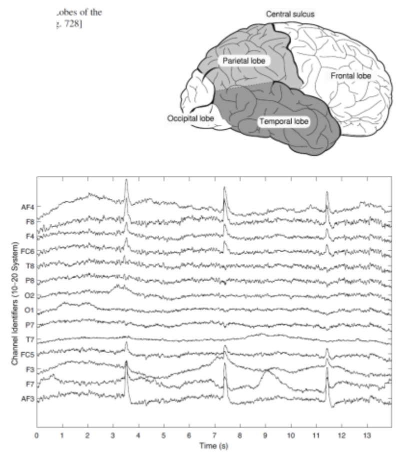


ECG





New Biometric Traits (5/5)



Ruggero Donida Labati - Deep Learning in Biometrics - 2020-201
Ph.D. degree in Computer Science, Università degli Studi di Milano

UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

21

Mobile and Wearable Biometrics

Ruggero Donida Labati - Deep Learning in Biometrics - 2020-201
Ph.D. degree in Computer Science, Università degli Studi di Milano

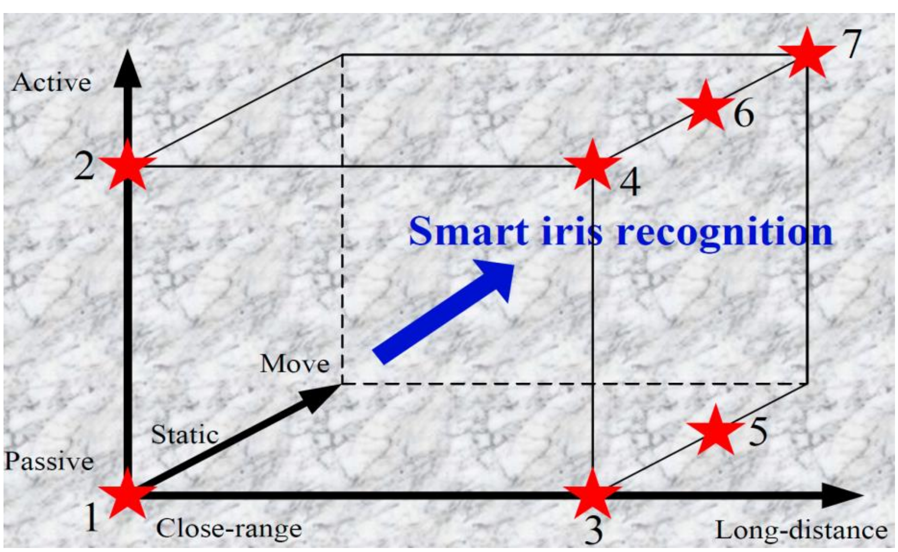
UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

22

1.2) Iris

23

Iris Recognition Roadmap (1/5)

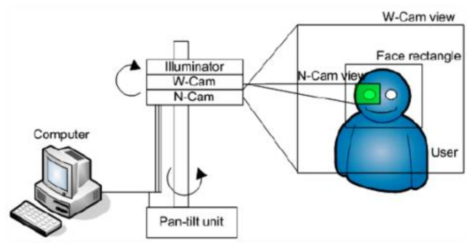


24

Iris Recognition Roadmap (2/5)

1: close-range iris recognition


2: active iris recognition



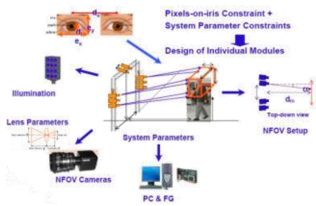
Iris Recognition Roadmap (3/5)

3: iris recognition at a distance

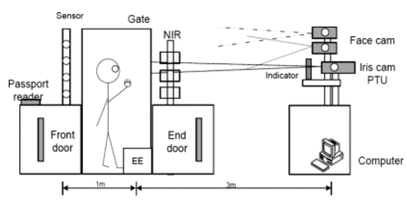
4: active iris recognition at a distance

Iris Recognition Roadmap (4/5)

5: passive iris recognition on the move

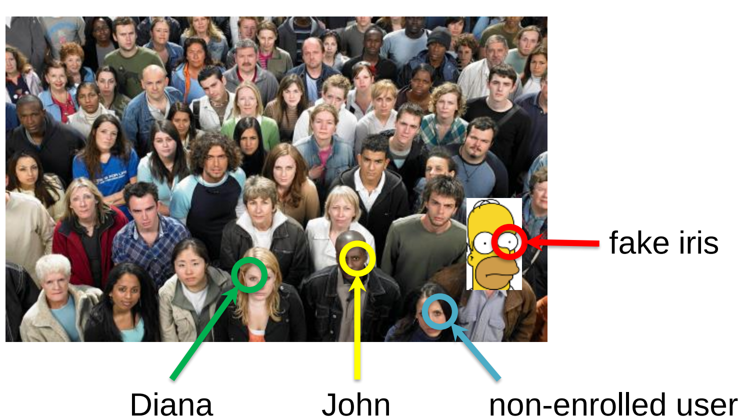


6: active iris recognition on the move



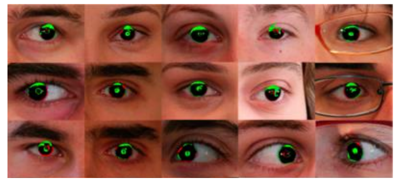
Iris Recognition Roadmap (5/5)

7: iris recognition for surveillance

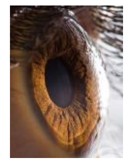


Open Problems

- Segmentation



- Gaze deviation



- Pupil dilation

1.3) Face

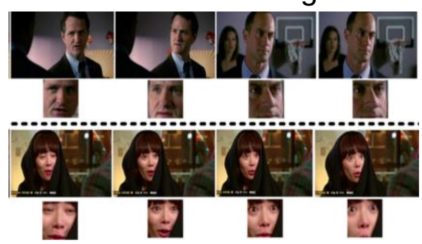
Recent Applications

Surveillance

Mobile

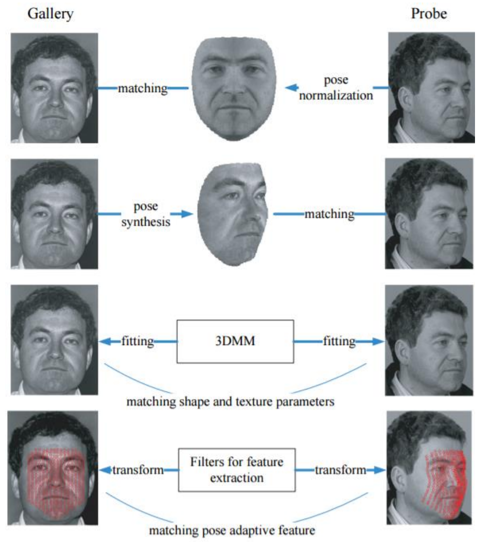
Gaming

Video indexing

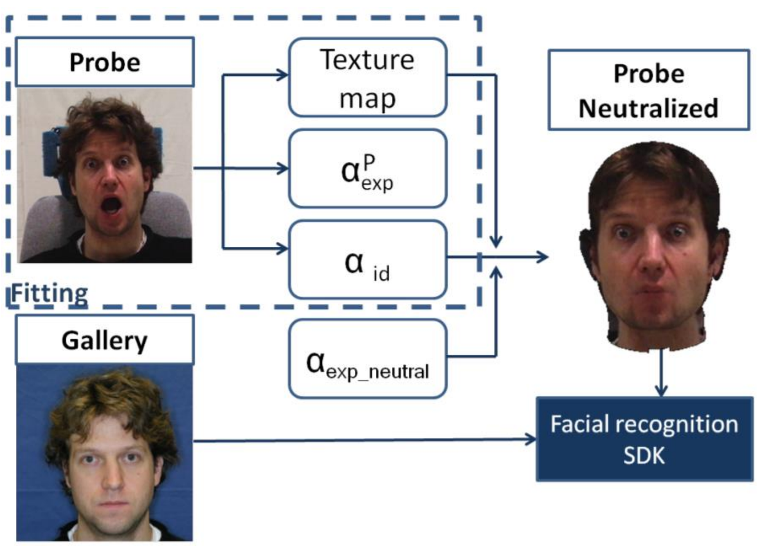


Open Problems: Occlusions

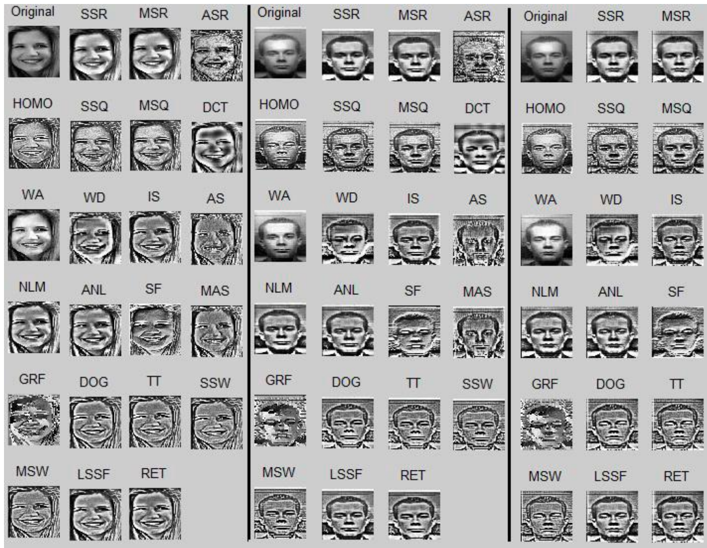
Open Problems: Pose Variations



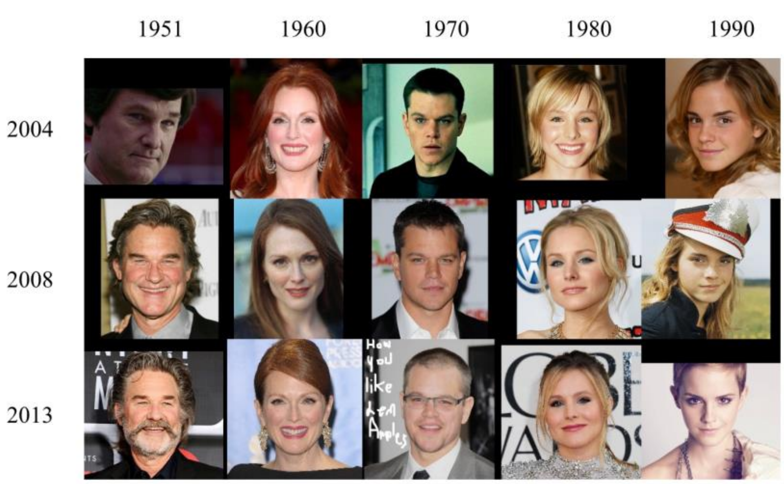
Open problems: expression variations



Open Problems: Illumination



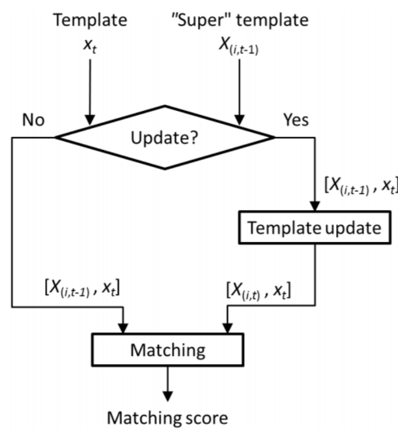
Open Problems: Aging (1/2)



Open Problems: Aging (2/2)

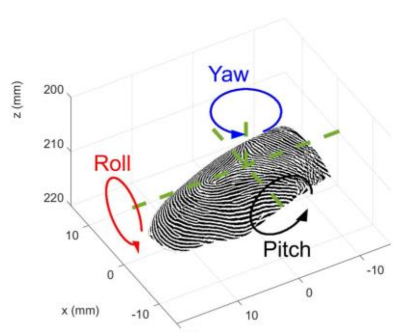
Simulation

Template update



37

1.4) Touchless fingerprint



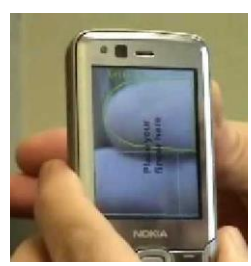
38

Touchless Fingerprint Images



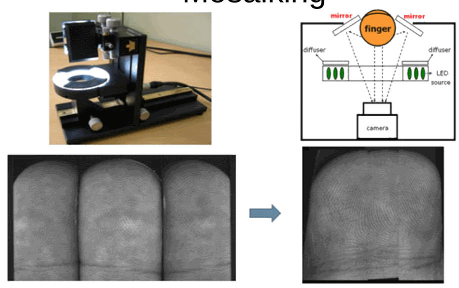

- R. Donida Labati, V. Piuri, F. Scotti, *Touchless Fingerprint Biometrics*, CRC Press, August, 2015.

Possible Applications of Touchless Fingerprint Biometrics

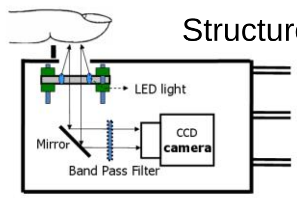



Touchless Fingerprint: Some Existing Systems (1/2)

Mosaiking

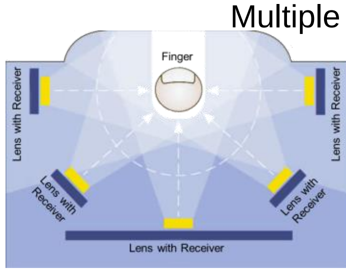


Structured light



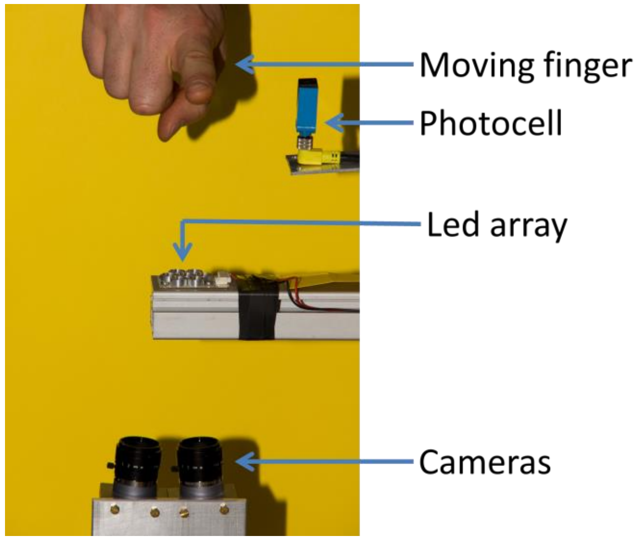
Touchless fingerprint: some existing systems (2/2)

Multiple views



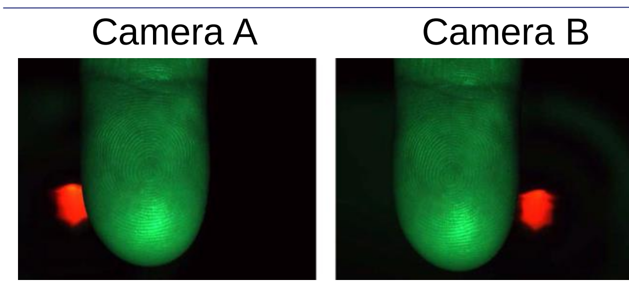
Absorbed light

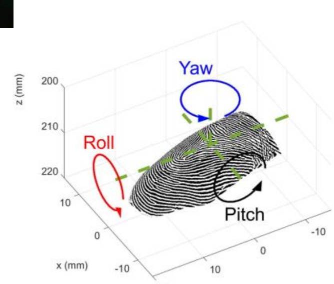
Fingerprint Recognition on the Move



43

Contactless Acquisition



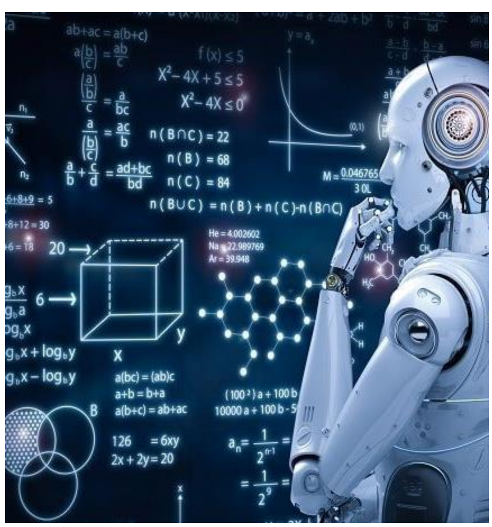


44

Overview of Different Technologies

Aspect	Touch-based	Touchless
Accuracy	EER = 0.03%	EER = 0.06%
Scalability	High	To be further investigated
Interoperability	High	To be improved
Security	Latent fingerprints	No latent fingerprints
Privacy	Data protection techniques	Data protection techniques
Cost	10$ to 5000$	0$ to 5000$
Usability	Medium	High
User acceptance	Medium	High
Speed	Template extraction + matching	3D reconstruction + template extraction + matching

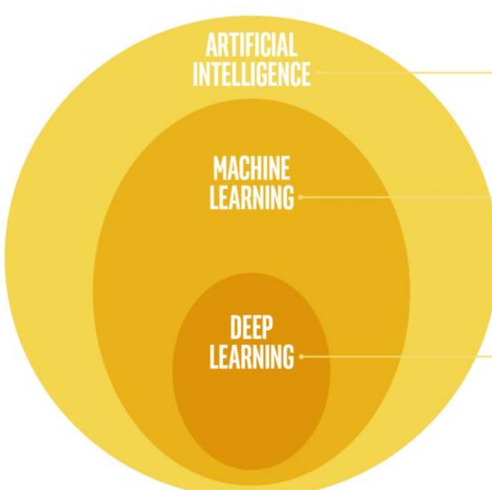
2. Machine learning



Ruggero Donida Labati - Deep Learning in Biometrics - 2020-201
Ph.D. degree in Computer Science, Università degli Studi di Milano
UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

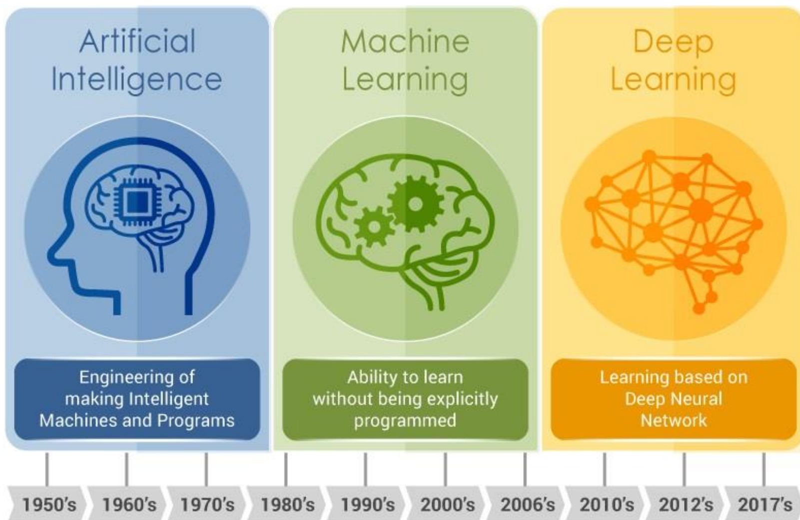
2.1 Introduction

Deep Learning, Machine Learning and AI

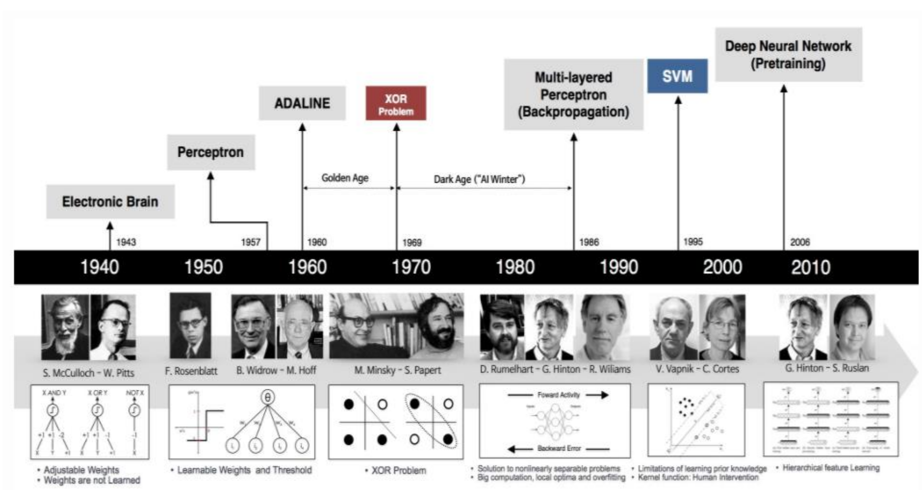


- **AI** is an umbrella term for machines capable of perception, logic, and learning.
- **Machine learning** employs algorithms that learn from data to make predictions or decisions, and whose performance improves when exposed to more data over time.
- **Deep learning** uses many-layered neural networks to build algorithms that find the best way to perform tasks on their own, based on vast sets of data.

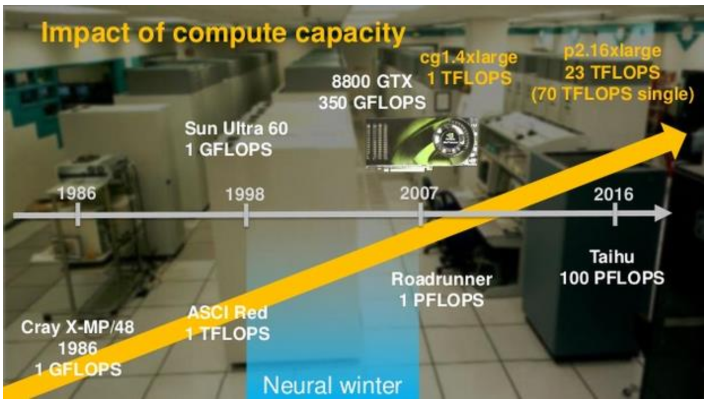
Timeline (1/2)



Timeline (2/2)



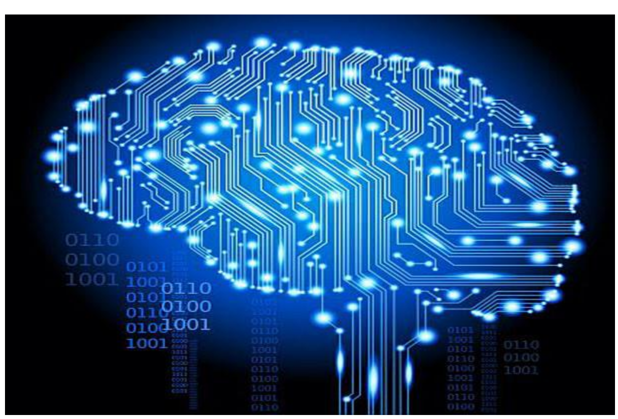
Computational Resources



In the history of artificial intelligence, an AI winter is a period of reduced funding and interest in artificial intelligence research

Machine Learning

- Supervised
- Unsupervised



Machine Learning in Biometrics

Biometric Applications

ML Tasks *Broad Categories*	*Supervised*	*Unsupervised*
Discrete	Classification Computer vision | Image Classification Speech, handwriting recognition Drug discovery	Clustering K-means, mean-shift Large-scale clustering problem Hierarchical clustering, GMM
Continuous	Regression Computer vision | Object Detection Linear, logistic regression	Reduction of Dimensionality PCA, LDA (Kernel) Density Estimation

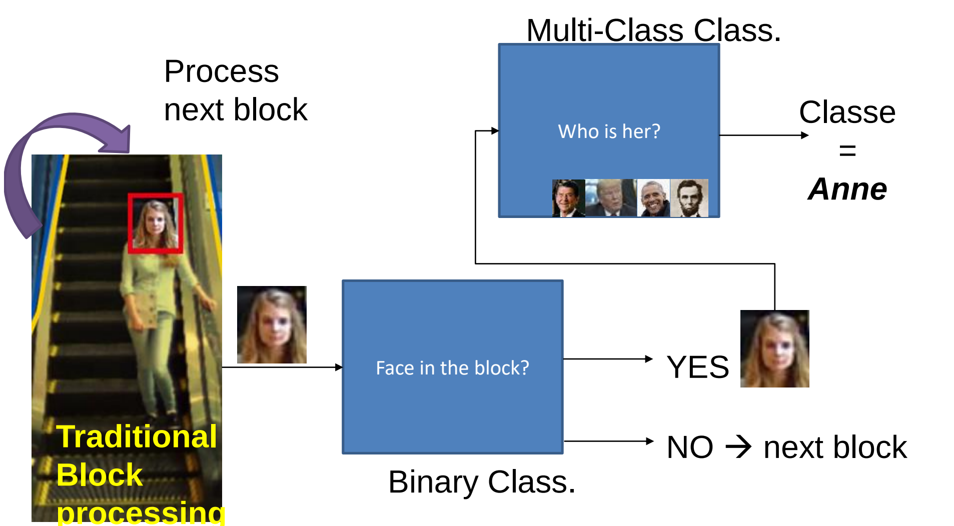
Almost everything…

User Partitions, Data Understanding

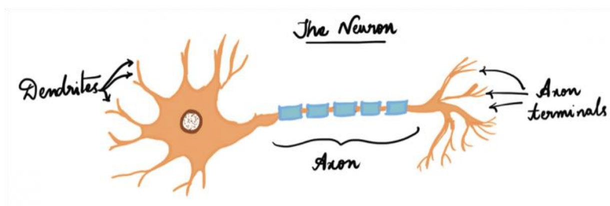
Age Estimation, Soft Biometrics, Quality Assessment

General… Data Understanding, Input processing…

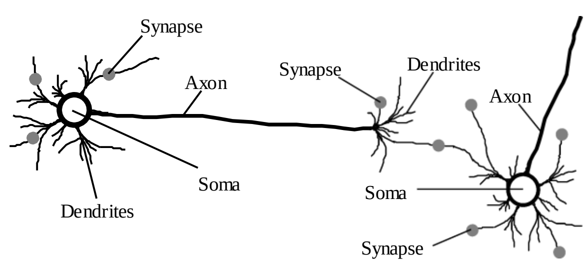
Biometric Primitives are classifiers: face detection and identification



2.2 Feedforward neural networks



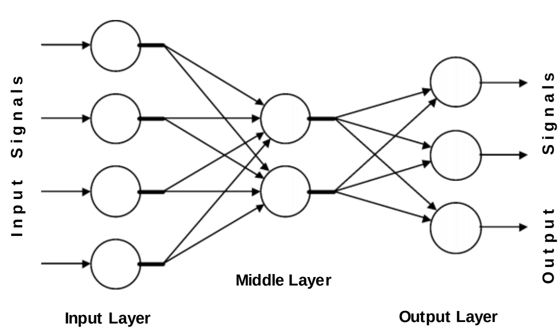
Biological Neural Network



Very Simplified Description of Neural Information Processing

- Axon terminal releases chemicals, called neurotransmitters.
- These act on the membrane of the receptor dendrite to change its polarization.
- (The inside is usually 70mV more negative than the outside.)
- Decrease in potential dierence: excitatory synapse
- Increase in potential dierence: inhibitory synapse
- If there is enough net excitatory input, the axon is depolarized.
- The resulting action potential travels along the axon.
- (Speed depends on the degree to which the axon is covered with myelin.)
- When the action potential reaches the terminal buttons, it triggers the release of neurotransmitters

Artificial Neural Network (1/2)

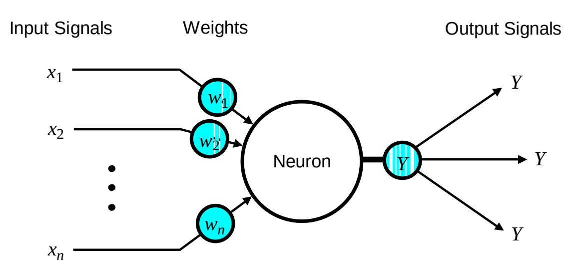


Artificial Neural Network (2/2)

Biological Neural Network	Artificial Neural Network
Soma	Neuron
Dendrite	Input
Axon	Output
Synapse	Weight

The Neuron as a Simple Computing Element

Diagram of a neuron



Basic Neuron Model: Perceptron

- The neuron computes the weighted sum of the input signals and compares the result with a **threshold value**, ϑ. If the net input is less than the threshold, the neuron output is –1. But if the net input is greater than or equal to the threshold, the neuron becomes activated and its output attains a value +1.
- The neuron uses the following transfer or **activation function**:

$$X = \sum_{i=1}^{n} x_i w_i \qquad Y = \begin{cases} +1, & \text{if } X \geq \theta \\ -1, & \text{if } X < \theta \end{cases}$$

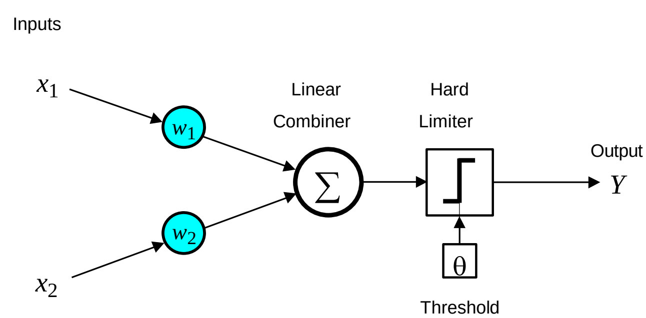
- This type of activation function is called a **sign function**.

The Perceptron

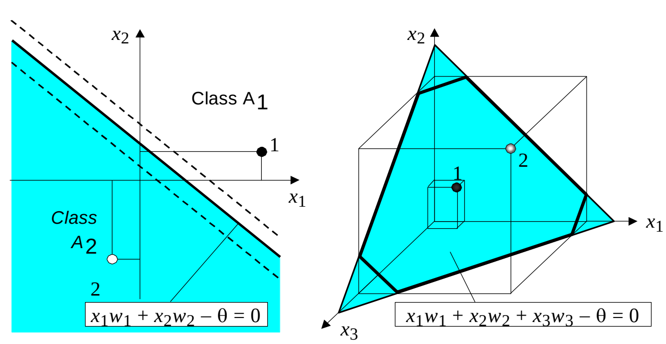
- The aim of the perceptron is to classify inputs, $x_1, x_2, \ldots, x_n$, into one of two classes, say A_1 and A_2.
- In the case of an elementary perceptron, the n- dimensional space is divided by a **hyperplane** into two decision regions. The hyperplane is defined by the **linearly separable function**:

$$\sum_{i=1}^{n} x_i w_i - \theta = 0$$

Single-layer Two-input Perceptron



Linear Separability in the Perceptrons



(*a*) Two-input perceptron. (*b*) Three-input perceptron.

How Does the Perceptron Learn Its Classification Tasks? (1/2)

- This is done by making small adjustments in the weights to reduce the difference between the actual and desired outputs of the perceptron.
- The initial weights are randomly assigned, usually in the range [- 0.5, 0.5], and then updated to obtain the output consistent with the training examples.

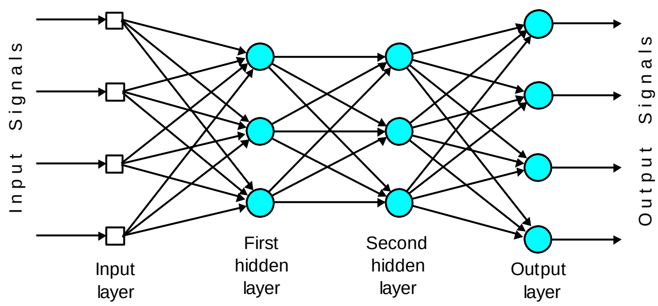
How Does the Perceptron Learn Its Classification Tasks? (2/2)

- If at iteration *p*, the actual output is $Y(p)$ and the desired output is $Y_d(p)$, then the error is given by:

  $e(p) = Y_d(p) - Y(p)$ where $p = 1, 2, 3, \ldots$

  Iteration *p* here refers to the *p*th training example presented to the perceptron.
- If the error, $e(p)$, is positive, we need to increase perceptron output $Y(p)$, but if it is negative, we need to decrease $Y(p)$.

Multilayer Perceptron with Two Hidden Layers



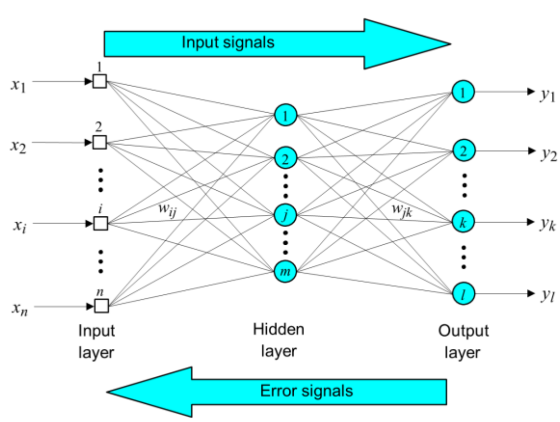
Back-propagation Neural Network (1/2)

- Learning in a multilayer network proceeds the same way as for a perceptron.
- A training set of input patterns is presented to the network.
- The network computes its output pattern, and if there is an error - or in other words a difference between actual and desired output patterns - the weights are adjusted to reduce this error.

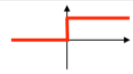
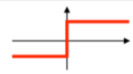
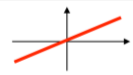
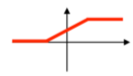
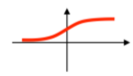
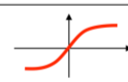
Back-propagation Neural Network (2/2)

- In a back-propagation neural network, the learning algorithm has two phases.
- First, a training input pattern is presented to the network input layer. The network propagates the input pattern from layer to layer until the output pattern is generated by the output layer.
- If this pattern is different from the desired output, an error is calculated and then propagated backwards through the network from the output layer to the input layer. The weights are modified as the error is propagated.

Three Layer Back-propagation Neural Network



Activation Functions

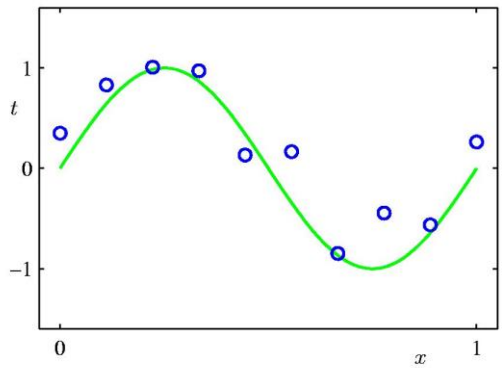
Activation function	Equation	1D Graph
Unit step (Heaviside)	$\phi(z)=\begin{cases}0, & z<0,\\ 0.5, & z=0,\\ 1, & z>0,\end{cases}$	
Sign (Signum)	$\phi(z)=\begin{cases}-1, & z<0,\\ 0, & z=0,\\ 1, & z>0,\end{cases}$	
Linear	$\phi(z)=z$	
Piece-wise linear	$\phi(z)=\begin{cases}1, & z\geq\frac{1}{2},\\ z+\frac{1}{2}, & -\frac{1}{2}<z<\frac{1}{2},\\ 0, & z\leq-\frac{1}{2},\end{cases}$	
Logistic (sigmoid)	$\phi(z)=\frac{1}{1+e^{-z}}$	
Hyperbolic tangent	$\phi(z)=\frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	

71

2.3 Overfitting

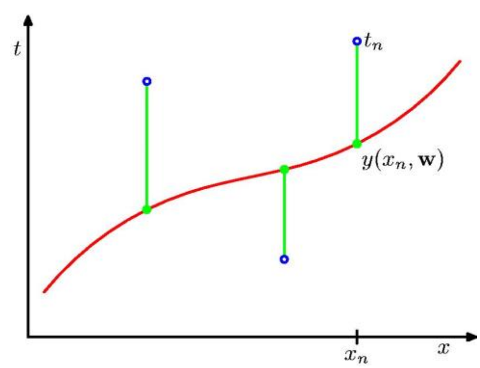
72

Polynomial Curve Fitting



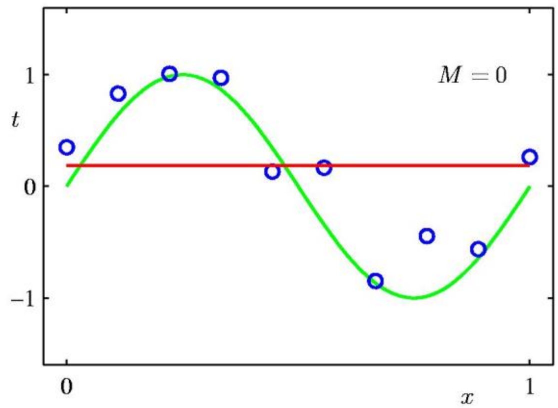
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Sum-of-Squares Error Function

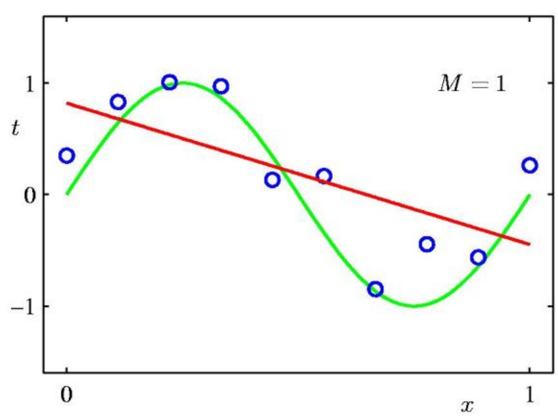


$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

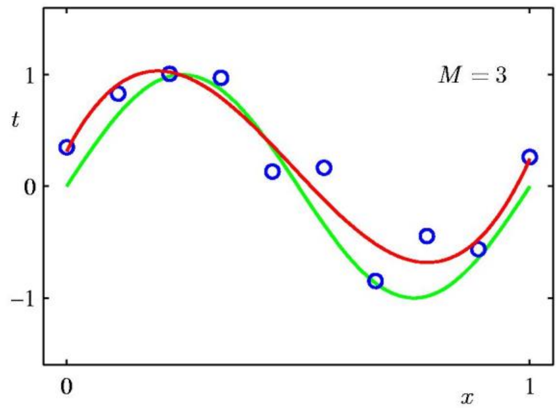
0th Order Polynomial



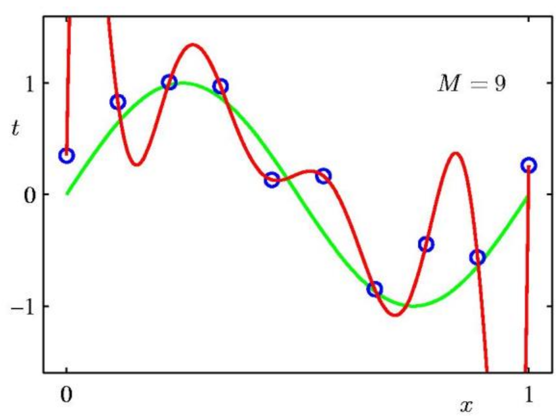
1st Order Polynomial



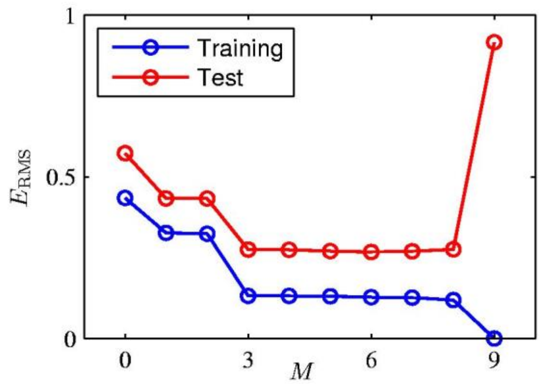
3st Order Polynomial



9st Order Polynomial



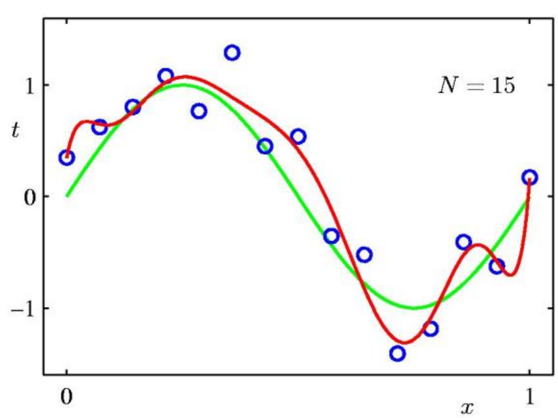
Overfitting



Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$

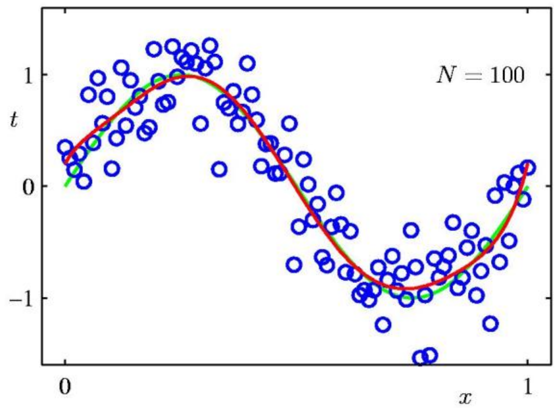
Dataset Size: *N* = 15

9th Order Polynomial



Dataset Size: *N* = 100

9th Order Polynomial



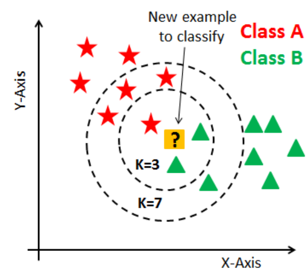
Rule of Thumb for Training Feedforward Neural Networks

1. Set and analyze a configuration
   - Activation functions
   - Learning rate
   - Learning goal
   - Maximum number of epochs
2. Increase the number of neurons in the hidden layer until the model overfits
3. Create a new configuration, maybe increasing the number of hidden layers

- Different results for each training

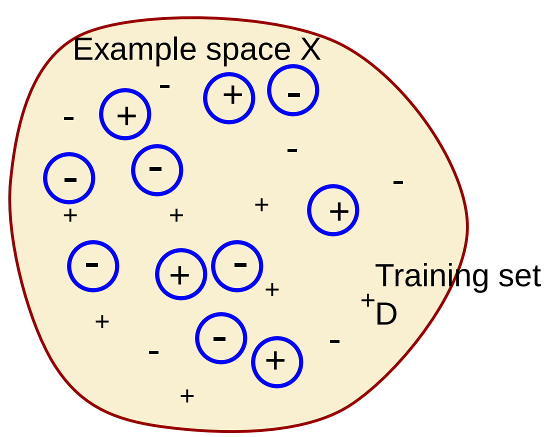
⇨ Compute the mean of multiple trainings

2.4 k Nearest Neighbor



k Nearest Neighbor (kNN)

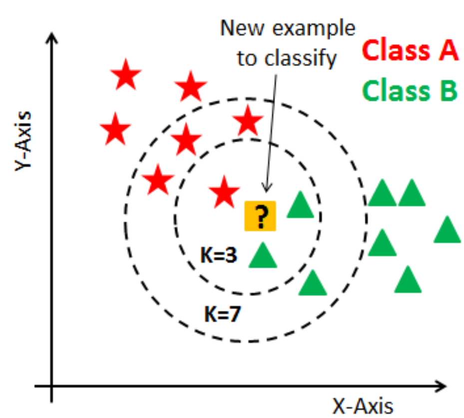
- Values of concept f(x) given on training set
  D = {(x_i,f(x_i)) for i=1,…,N}



Distance Metrices

- d(x,x') measures how "far" two examples are from one another, and must satisfy:
  - $d(x,x) = 0$
  - $d(x,x') \geq 0$
  - $d(x,x') = d(x',x)$
- Common metrics
  - Euclidean distance (if dimensions are in same units)
  - Manhattan distance (different units)
- Axes should be weighted to account for spread
  - $d(x,x') = \alpha_h|\text{height-height'}| + \alpha_w|\text{weight-weight'}|$
- Some metrics also account for correlation between axes (e.g., Mahalanobis distance)

Voting

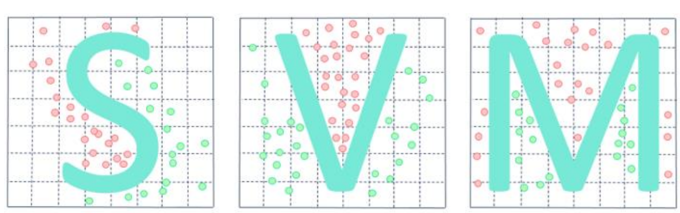


Computational Properties of K-NN

- Training time is nil
- Naïve k-NN: O(N) time to make a prediction
- Special data structures can make this faster
  - k-d trees
  - Locality sensitive hashing

  See R&N
- … but are ultimately worthwhile only when d is small, N is very large, or we are willing to approximate

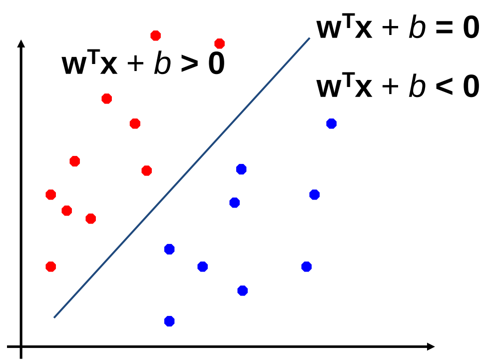
2.5 Support vector machine (SVM)



Ruggero Donida Labati - Deep Learning in Biometrics - 2020-201
Ph.D. degree in Computer Science, Università degli Studi di Milano
UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

Perceptron Revisited: Linear Separators

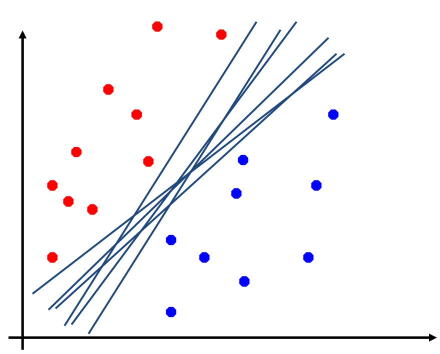
- Binary classification can be viewed as the task of separating classes in feature space



$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$

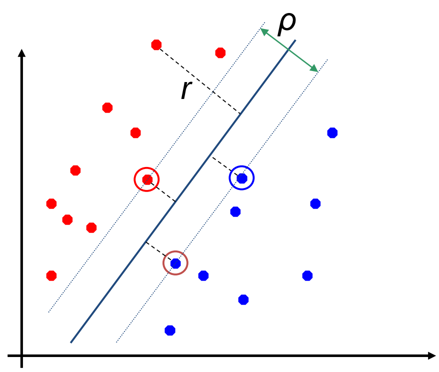
Linear Separators

- Which of the linear separators is optimal?



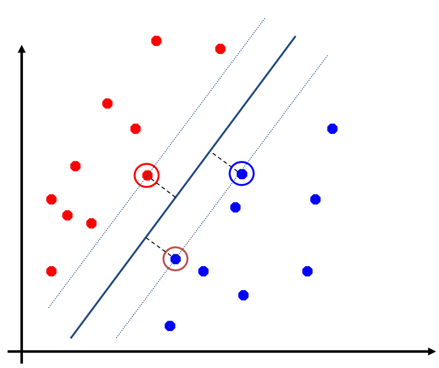
Classification Margin

- Distance from example $\mathbf{x}_i$ to the separator is $r = \frac{\mathbf{w}^T\mathbf{x}_i + b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***
- ***Margin*** ρ of the separator is the distance between support vectors



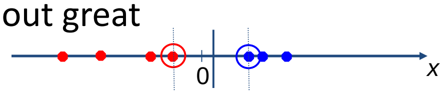
Maximum Margin Separation

- Maximizing the margin is good according to intuition
- Implies that only support vectors matter; other training examples are ignorable

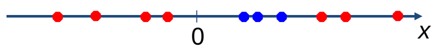


Non-linear SVMs

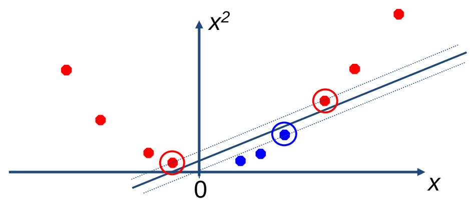
- Datasets that are linearly separable with some noise work out great



- But what are we going to do if the dataset is just too hard?

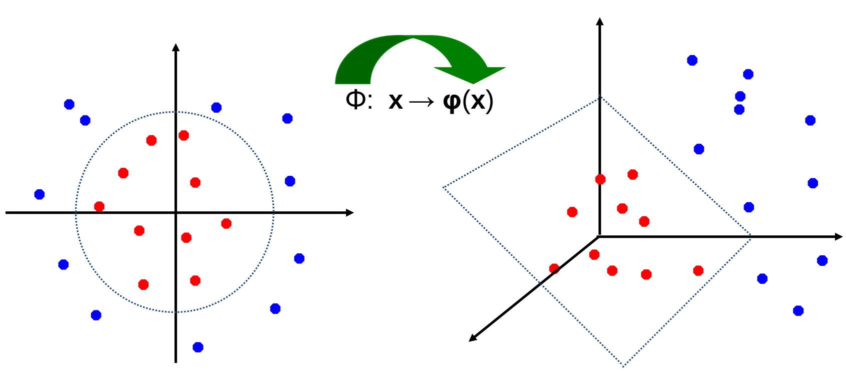


- How about… mapping data to a higher-dimensional space

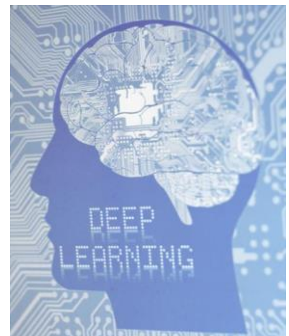


Non-linear SVMs: Feature Spaces

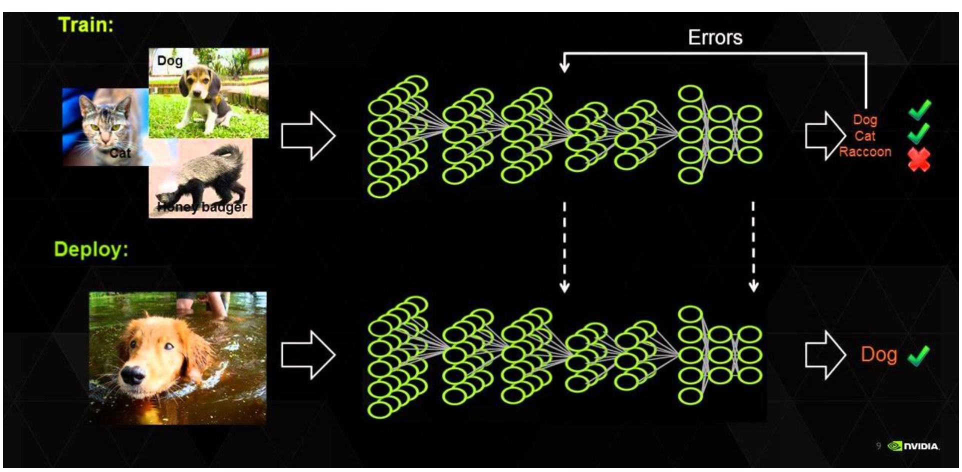
- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable



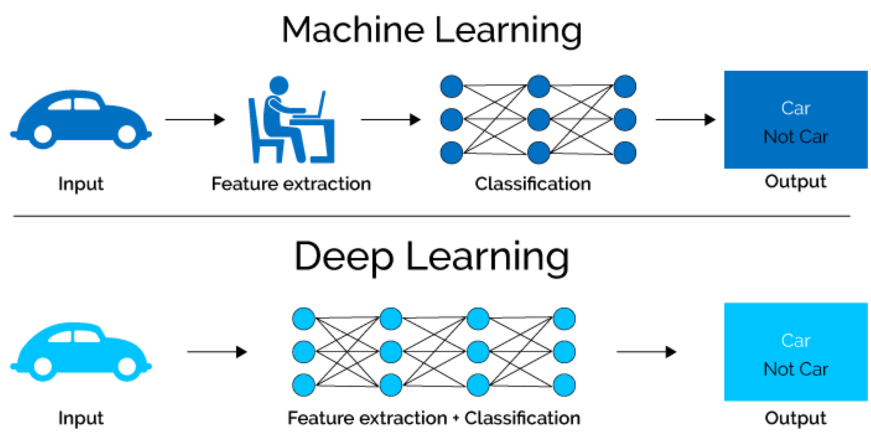
2.6 Introduction to deep learning



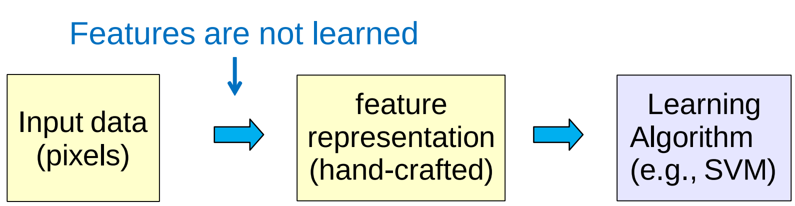
Deep Learning Appraoch



Machine Learning vs Deep Learning



Traditional Pattern Recognition Approach

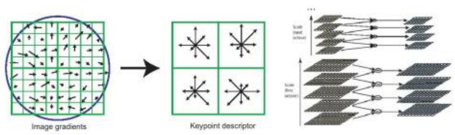


Image

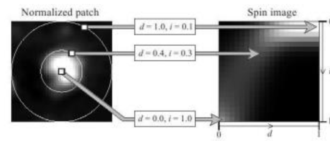
Low-level vision features (edges, SIFT, HOG, etc.)

Object detection / classification

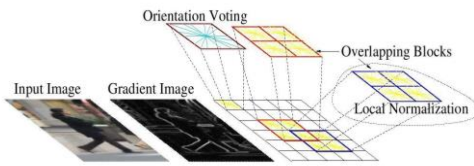
Feature Extraction



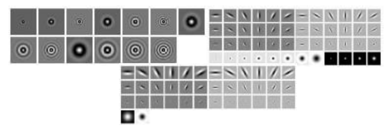
SIFT



Spin image



HoG



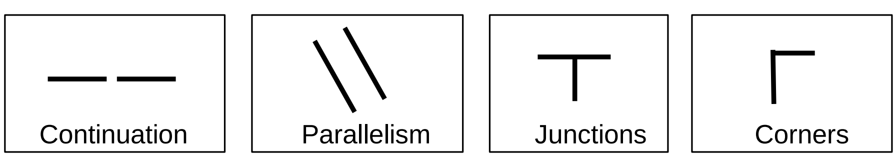
Textons

and many others:

SURF, MSER, LBP, Color-SIFT, Color histogram, GLOH, …..

Mid Level Representations

- Mid-level cues



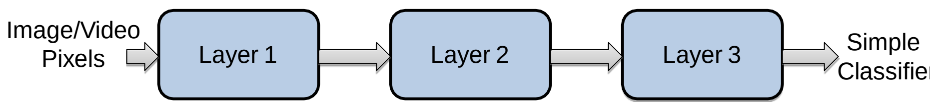
“Tokens” from Vision by D.Marr: 

- Object parts:

- Difficult to hand-engineer→ What about learning them?

Learning Feature Hierarchy (1/2)

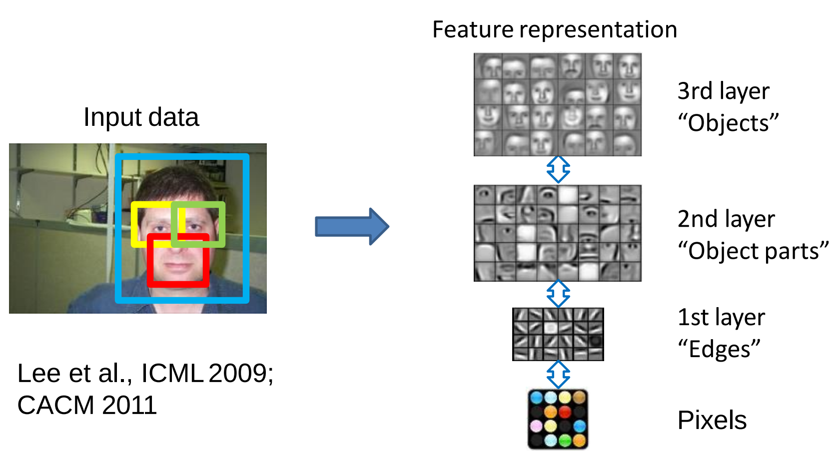
- Learn hierarchy
- All the way from pixels → classifier
- One layer extracts features from output of previous layer



- Train all layers jointly

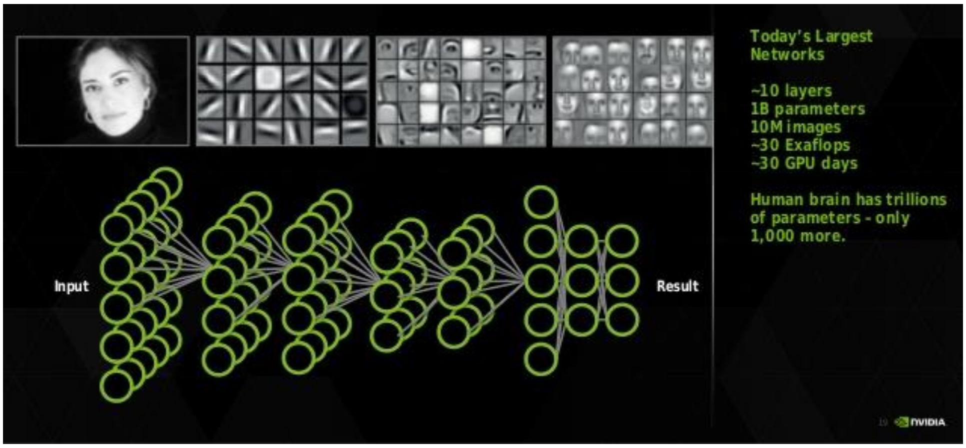
Learning Feature Hierarchy (2/2)

Learn **useful higher-level features** from images



Lee et al., ICML 2009;
CACM 2011

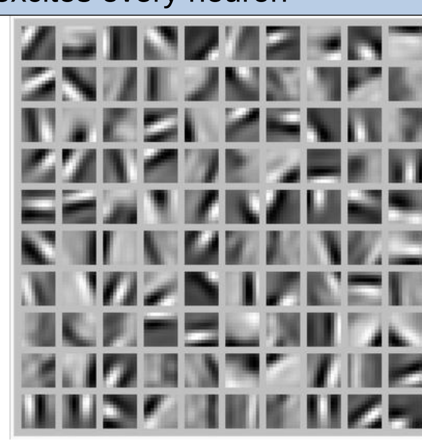
What makes Deep Learning deep?



Deep Learning Tasks

- Usually best when **input space is locally structured** – spatial or temporal: images, language, etc. vs arbitrary input features
- Example:
  - "view" of a learned vision feature layer
  - each square in the figure shows the input image that maximally activates one of the 100 unit

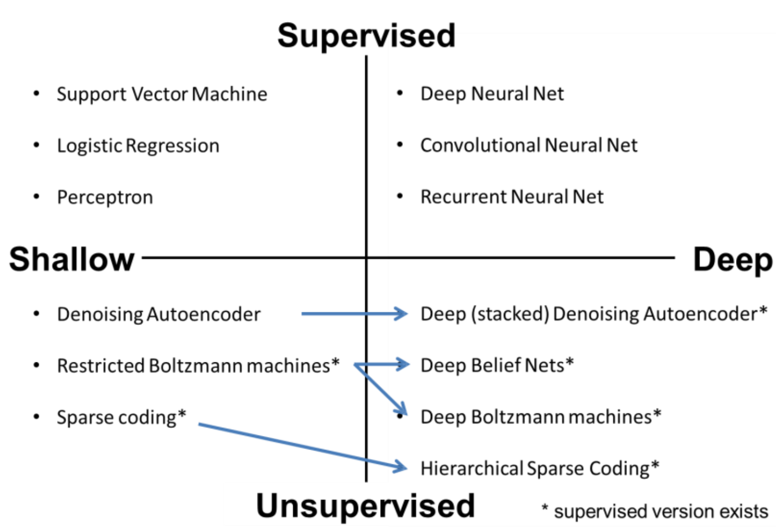
To understand how the network works, evaluate the inputs that excites every neuron



Why Deep Learning

- Biological Plausibility – e.g. Visual Cortex
- Hastad proof - Problems which can be represented with a polynomial number of nodes with *k* layers, may require an exponential number of nodes with *k*-1 layers (e.g. parity)
- Highly varying functions can be **efficiently** represented with deep architectures
  - Less weights/parameters to update than a less efficient shallow representation
- Sub-features created in deep architecture can potentially be shared between multiple tasks
  - Type of Transfer/Multi-task learning

Taxonomy



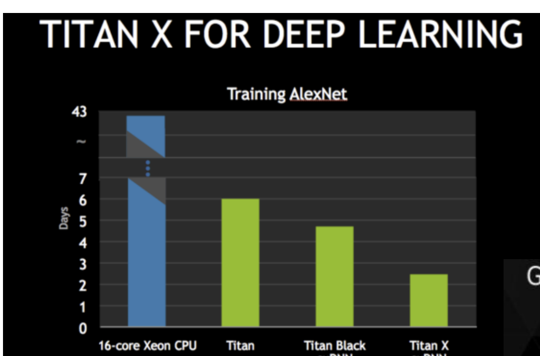
Why is Deep Learning Possible?

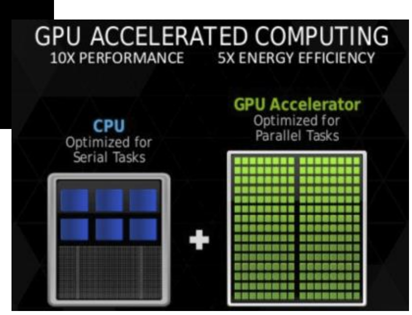
- Mass Storage → need of data!
  - More data available (TB HD, data centers,..)
- Higher Performance of Computer
  - Larger memory in handling the data
  - Greater computational power for calculating and even online learning

Nvidia GXT Titan

Tensor Processing Unit (TPU)

CPU vs GPU

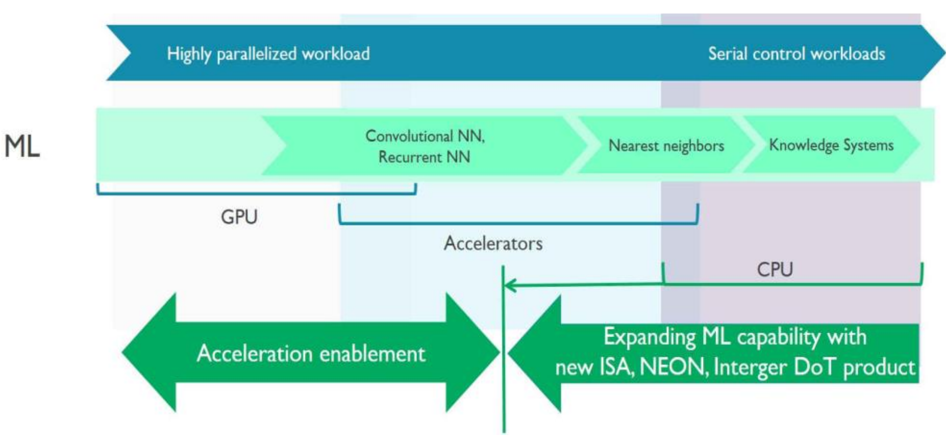




Server Farms

- Amazon (AWS)
- Google
- Facebook
- IBM (AzureML)
- …

GPU is Not Always Needed



Moving ML to the “Edge”

Example: ML on an ARM Processors

- Compute Library (ARM): set of functions for
  - ML frameworks like Google’s TensorFlow
  - Imaging and vision projects
  - Providing **portable code** that can run across various Arm system configurations
- Example:
  - the neural networks training considering examples often don’t require very high accuracy data, meaning that math calculations can usually be **performed on 16-bit or even 8-bit data**, rather than large 32 or 64-bit entries
- The majority of neural network processing uses **8-bit fixed-point matrix multiplication**, the Armv8.2-A architecture introduced support for half-precision (FP16) and integer dot products (INT8) floating point SIMD (single instruction multiple data) NEON instructions to accelerate ML NN processing

TPU

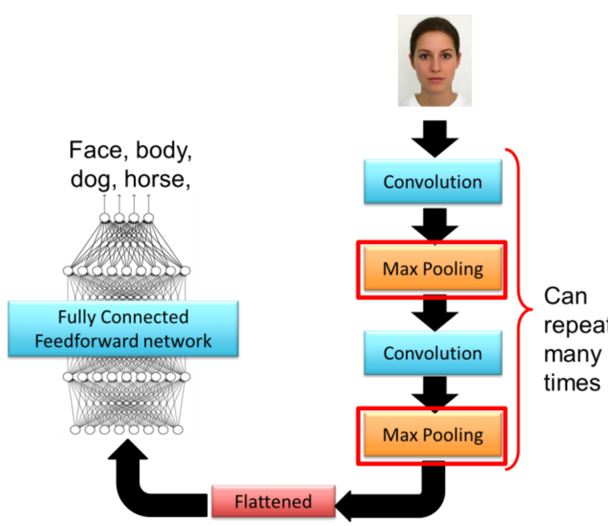
- A tensor processing unit (TPU) is an AI accelerator application-specific integrated circuit (ASIC) developed by Google specifically for neural network machine learning

Intel Movidius

- What can you do with a Raspberry PI and a Movidius?
- https://developer.movidius.com

3. Preview of the next lecture

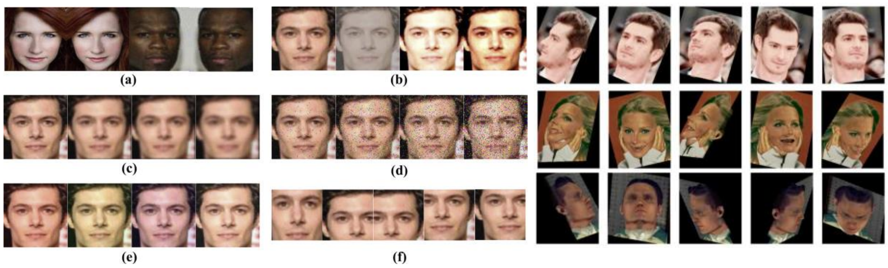
CNN and other networks



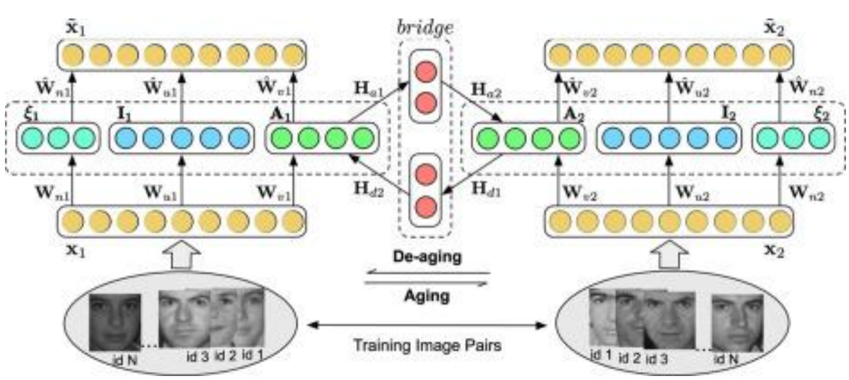
Design of biometric applications based on deep learning techniques

Landmark perturbation for face alignment.

Flipping
patches (clipping)
color casting
blurring



Biometric systems based on deep learning



4. Summary

Summary

1. Research trends
   1) Research trends in biometrics
   2) Iris
   3) Face
   4) Touchless fingerprint
2. Machine learning
   1) Introduction
   2) Feedforward neural networks
   3) k nearest neighbor
   4) Support vector machines
   5) Introduction to deep learning

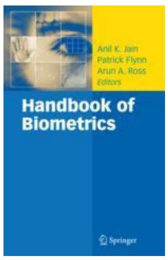
Research Trends and Open Problems

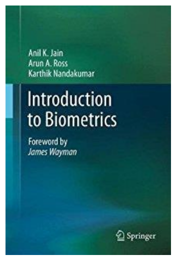
- Many recent research trends
- There still open problems
- The research community is very active

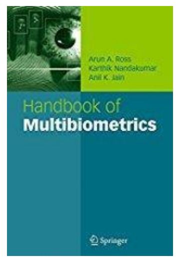
Machine Learning

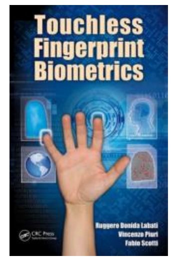
- Introduction
- Feedforward neural networks
- Overfitting
- k nearest neighbor
- Support vector machines
- Introduction to deep learning

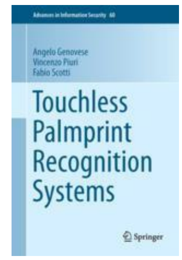
Thank you!


Anil K. Jain
Patrick Flynn
Arun A. Ross
Editors
Handbook of Biometrics
Springer


Anil K. Jain
Arun A. Ross
Karthik Nandakumar
Introduction to Biometrics
Foreword by
James Wayman
Springer


Arun A. Ross
Karthik Nandakumar
Anil K. Jain
Handbook of Multibiometrics
Springer


Touchless Fingerprint Biometrics
Ruggero Donida Labati
Vincenzo Piuri
Fabio Scotti


Angelo Genovese
Vincenzo Piuri
Fabio Scotti
Touchless Palmprint Recognition Systems
Springer