UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

Machine Learning in Biometrics

Deep Learning in Biometrics

Ruggero Donida Labati

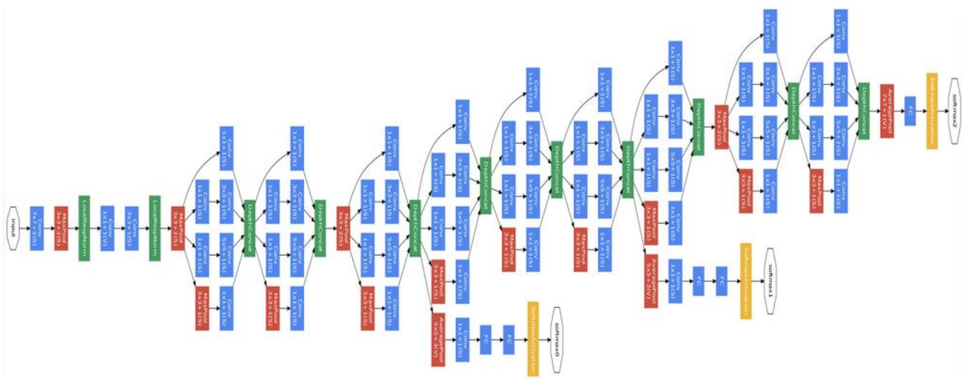
Academic year 2020/2021

1

Content

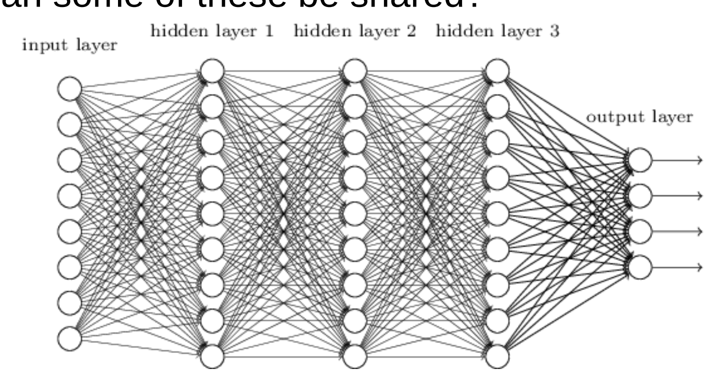
2

1. Convolutional Neural Networks



Starting from Feedforward Neural Networks

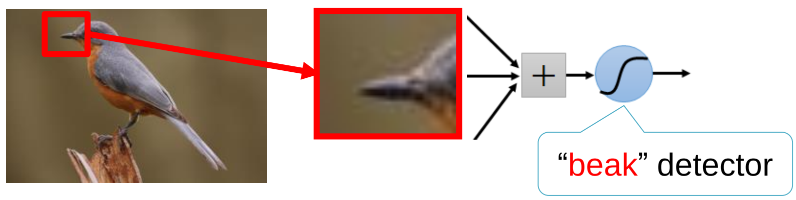
- We know it is good to learn a small model
- From this fully connected model, do we really need all the edges?
- Can some of these be shared?



Learning from an Image

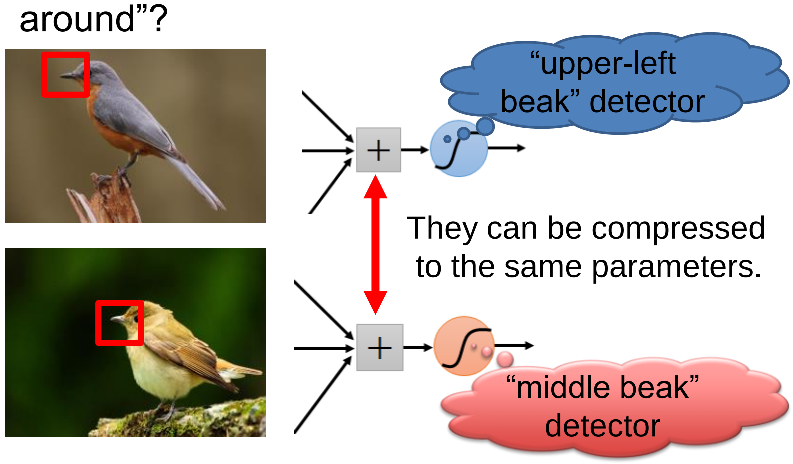
- Some patterns are much smaller than the whole image

Can represent a small region with fewer parameters?



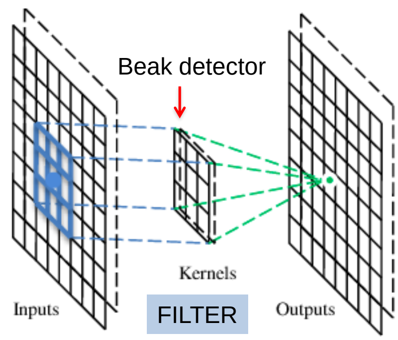
Same Pattern Appears in Different Places

- What about training a lot of such “small” detectors and each detector must “move around”?



A Convolutional Layer

- A CNN is a neural network with some convolutional layers (and some other layers)
- A convolutional layer has a number of filters that does convolutional operation



The convolution output is high if the input image is similar to the filter

Convolutional Kernels

These are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 input image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

⋮ ⋮

Each filter detects a small pattern (3 x 3).

Convolution (1/3)

Filter 1

1	-1	-1
-1	1	-1
-1	-1	1

stride=1

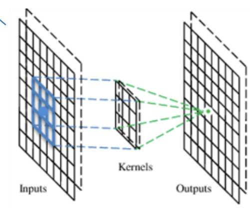
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

Dot product

3 -1

6 x 6 input image

Analyze this task



Convolution (2/3)

Filter 1

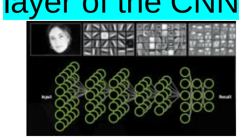
1	-1	-1
-1	1	-1
-1	-1	1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

3 -3

6 x 6 input image

This one can be one of the convolutioanl layer of the CNN



Convolution (3/3)

Filter 1

1	-1	-1
-1	1	-1
-1	-1	1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

Convolutional Layer

Filter **2**

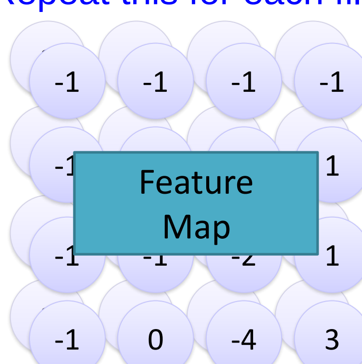
-1	1	-1
-1	1	-1
-1	1	-1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Repeat this for each filter



Two 4 x 4 images, forming 2 x 4 x 4 matrix

Color Images: 3 RGB Channels

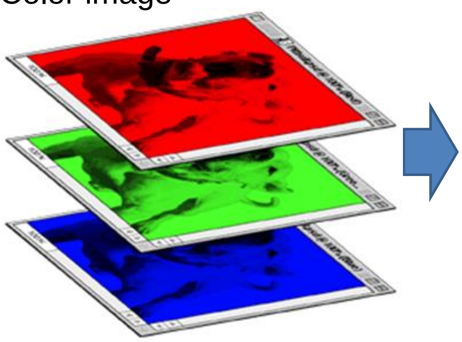
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Color image



1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

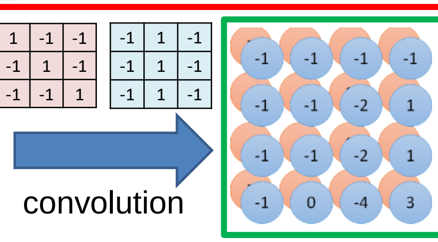
Convolution v.s. Fully Connected (1st Layer)

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

image

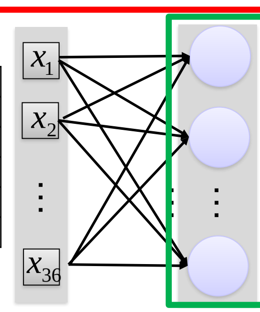
1	-1	-1
-1	1	-1
-1	-1	1

-1	1	-1
-1	1	-1
-1	1	-1

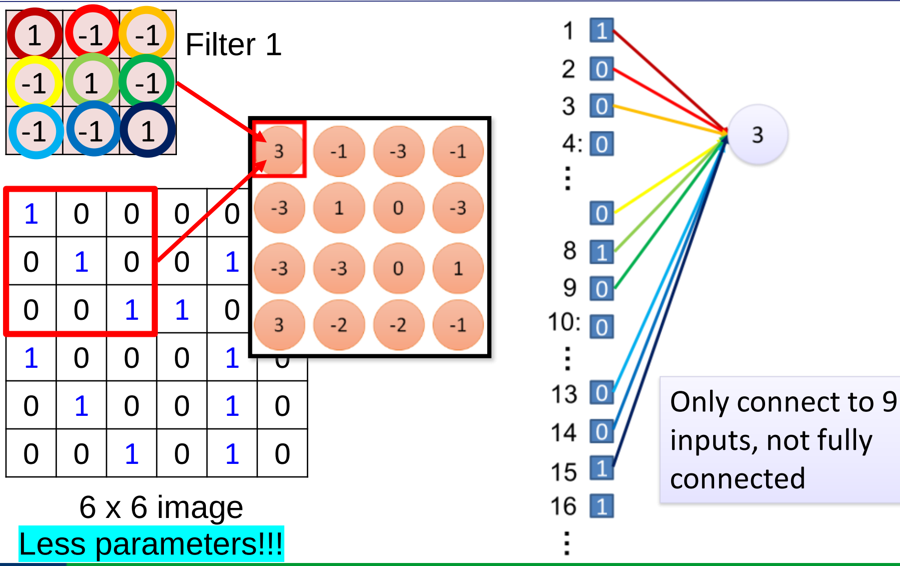


Fully-connected

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

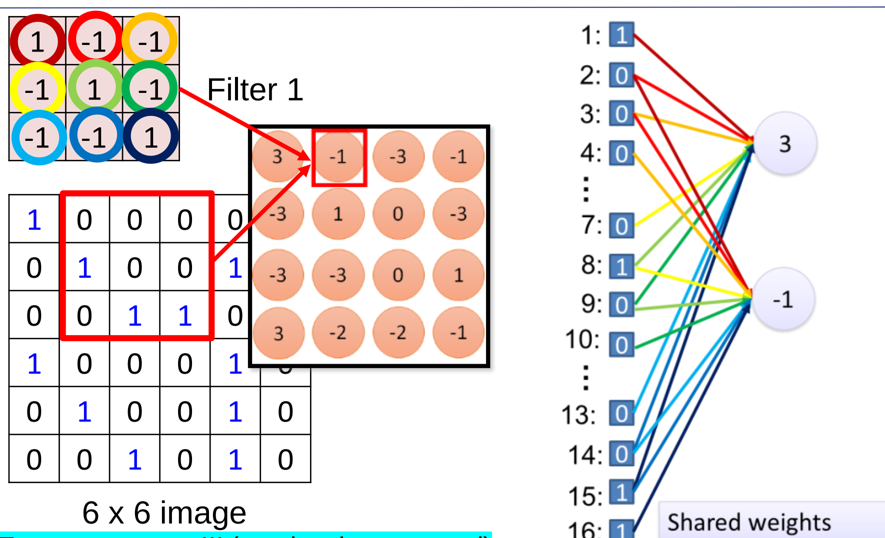


Convolution v.s. Fully Connected (1/3)



Less parameters!!!

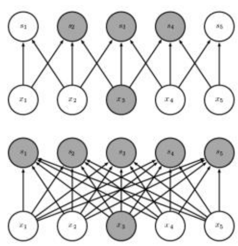
Convolution v.s. Fully Connected (2/3)

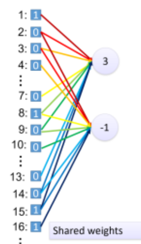


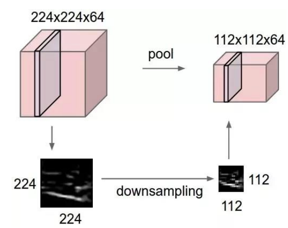
Even parameters!!! (previously computed)

Convolution v.s. Fully Connected (3/3)

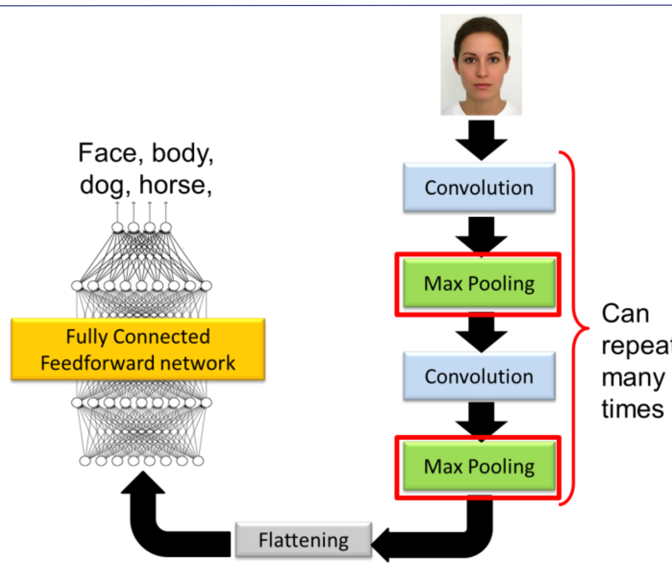
- CNNs and fully connected networks can be used to solve the same problems
- CNNs reduce the number of connections
- CNNs can share the previous computations between the neurons
- CNNs can use pooling operations to reduce the computational complexity





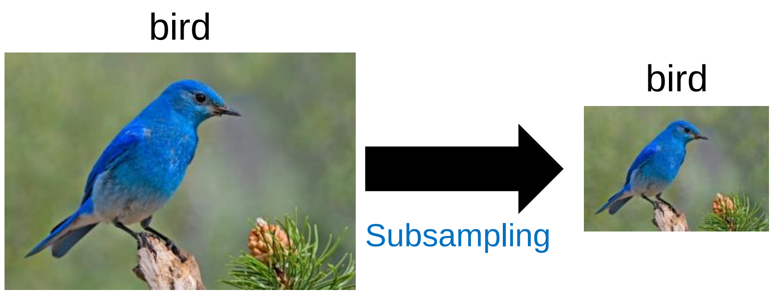


CNN Architecture



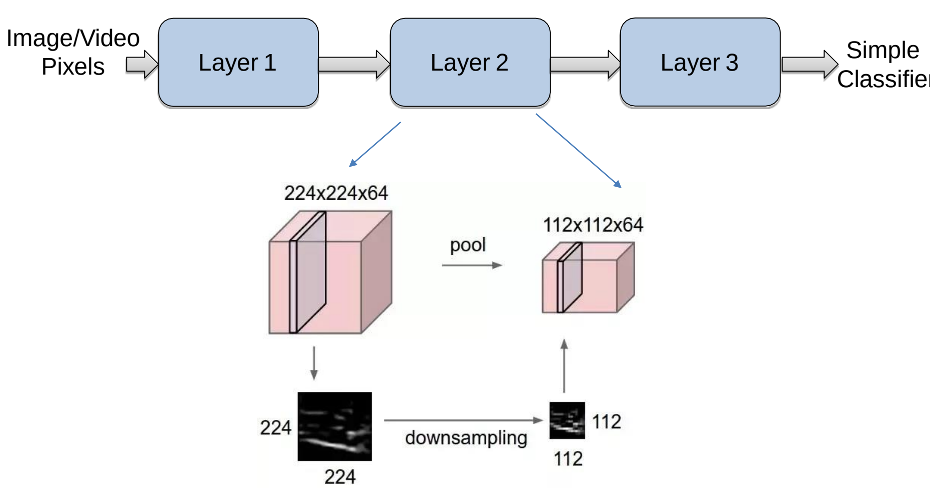
Why Pooling?

- Subsampling pixels will not change the object



- We can subsample the pixels to make image smaller
  - ➢fewer parameters to characterize the image

Pooling

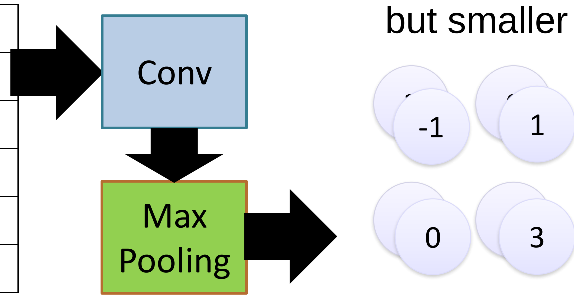


Max Pooling

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

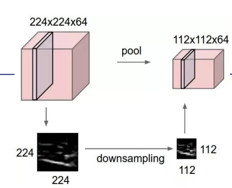
New image
but smaller



2 x 2 image

Each filter
is a channel

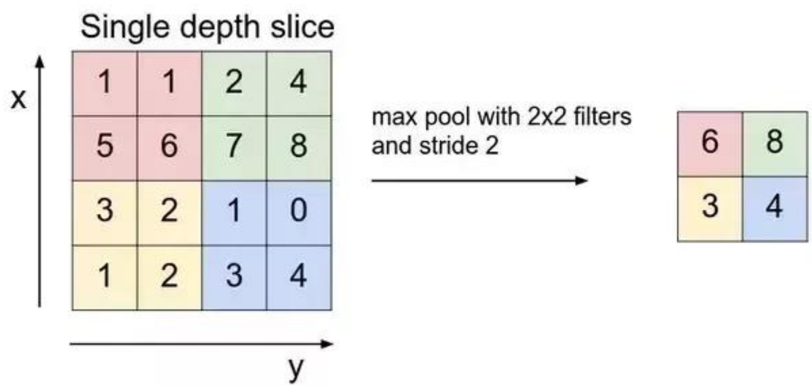
Sub-sampling (Pooling)



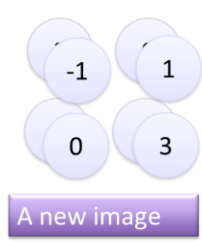
Sub-sampling (Pooling) allows number of features to be diminished, non-overlapped

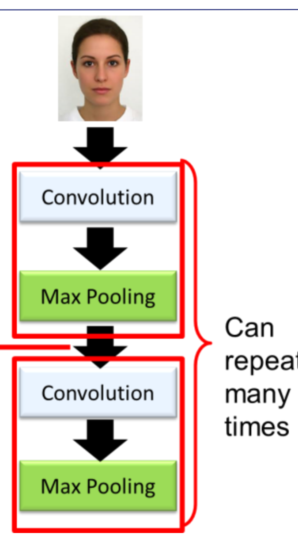
- –Reduces spatial resolution and thus naturally decreases importance of exactly where a feature was found, **just keeping the rough location**
- –**Averaging** or **Max-Pooling**
  - 2x2 pooling would do 4:1 compression, 3x3 9:1, etc.
- –Pooling smooths the data and **makes the data invariant to small translational changes**
- –Since after first layer, there are always multiple feature maps to connect to the next layer, it is a **pre-made human decision** as to which previous maps the current map receives inputs from

Max Pooling: Example



The Whole CNN (1/2)

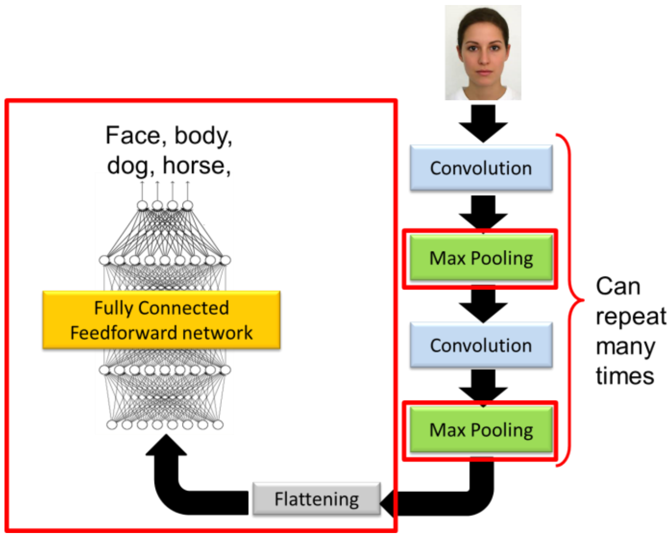




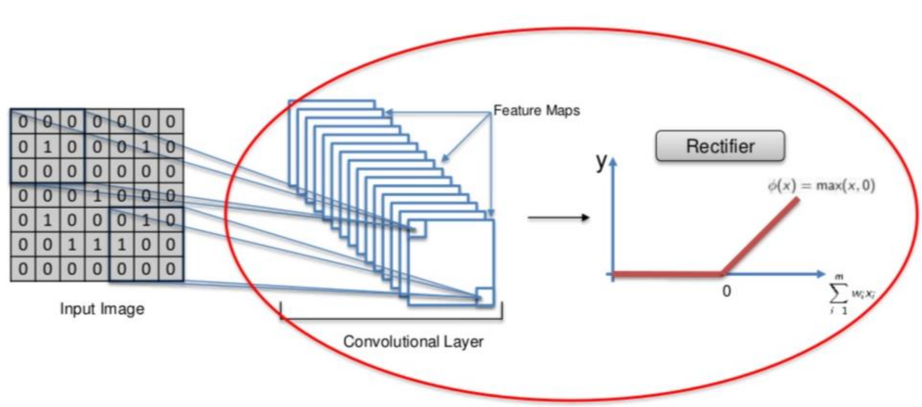
Smaller than the original image

The number of channels is the number of filters

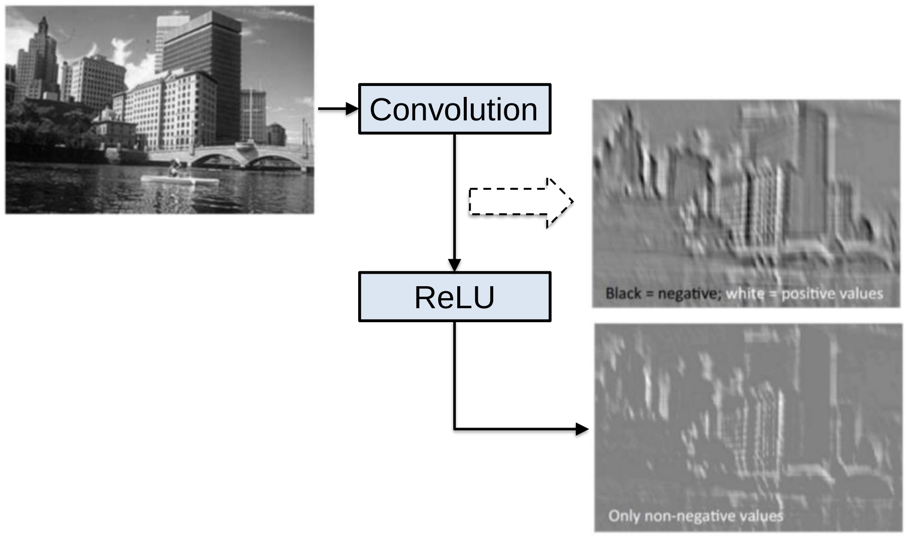
The Whole CNN (2/2)



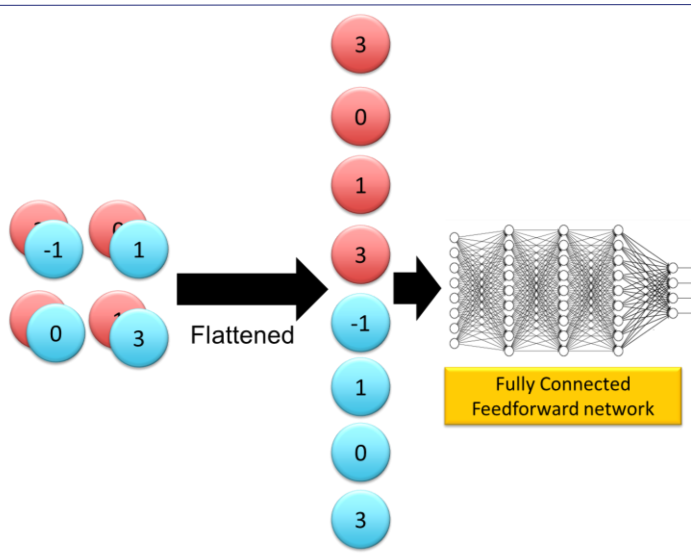
Optional Step: Rectified Linear Units (ReLU)



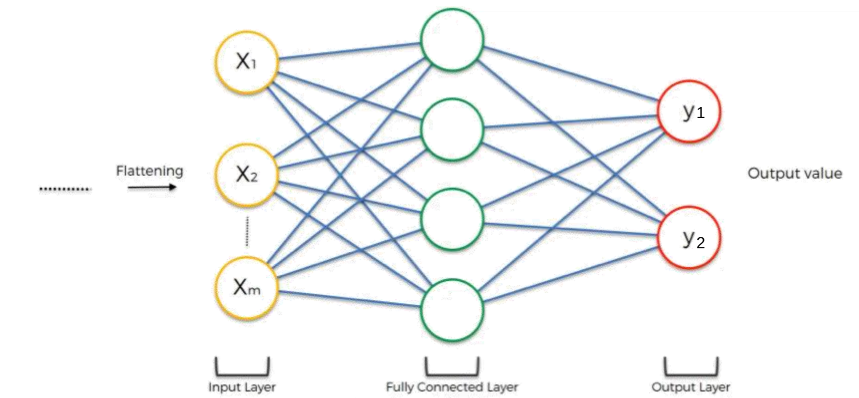
ReLU: Example



Flattening

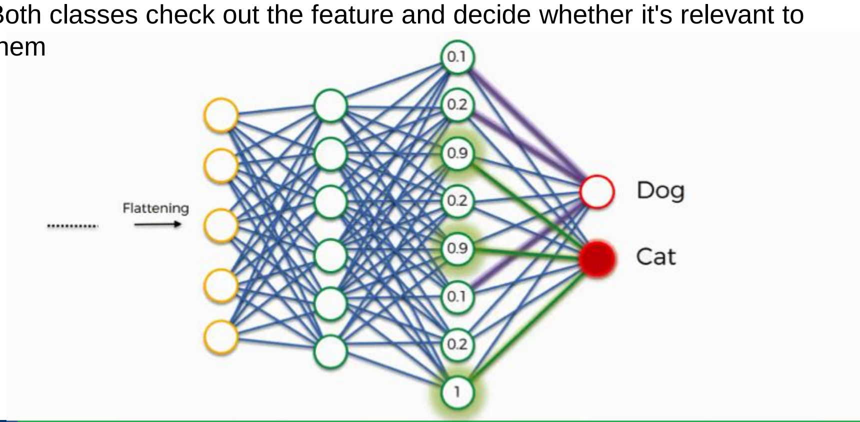


Fully Connected Layers (1/2)



Fully Connected Layers (2/2)

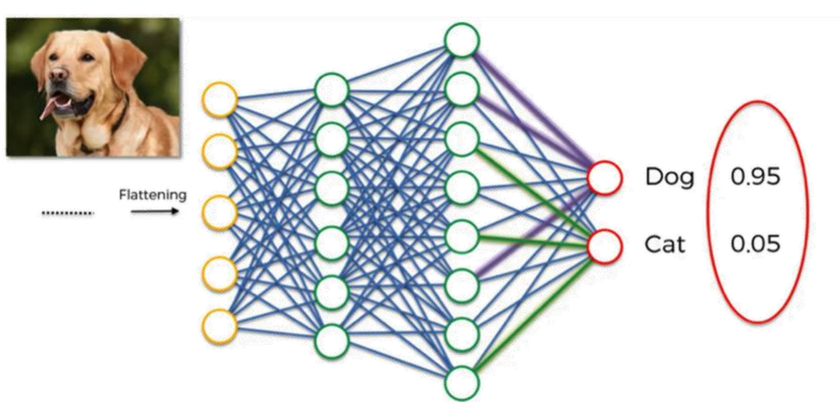
- The neuron in the fully-connected layer detects a certain feature
- It preserves its value
- It communicates this value to both the “dog” and the “cat” classes
- Both classes check out the feature and decide whether it's relevant to them



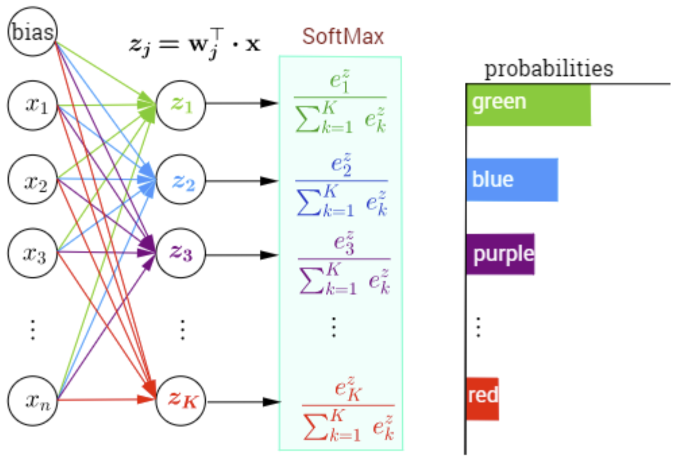
Loss Function

- The loss function informs us of how accurate our network is, which we then use in optimizing our network in order to increase its effectiveness
- In the context of artificial neural networks, we call this calculation a "cost function" or a mean squared error
- Some frequently used loss functions:
  - mean squared error
  - cross-entropy function

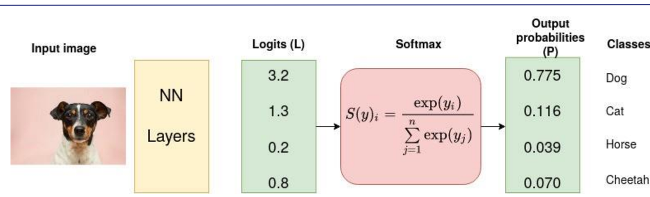
Softmax and Cross-Entropy



Softmax



Cross-Entropy

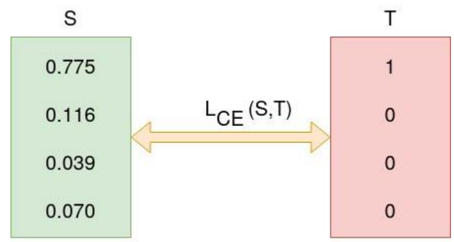


- The purpose of the Cross-Entropy is to take the output probabilities (P) and measure the distance from the truth values
- Cross-Entropy is defined as

$$L_{\mathrm{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \quad \text{for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

Cross-Entropy: Example



The categorical cross-entropy is computed as follows

$$
\begin{aligned}
L_{CE} &= -\sum_{i=1} T_i \log(S_i) \\
&= -\left[1\log_2(0.775) + 0\log_2(0.126) + 0\log_2(0.039) + 0\log_2(0.070)\right] \\
&= -\log_2(0.775) \\
&= 0.3677
\end{aligned}
$$

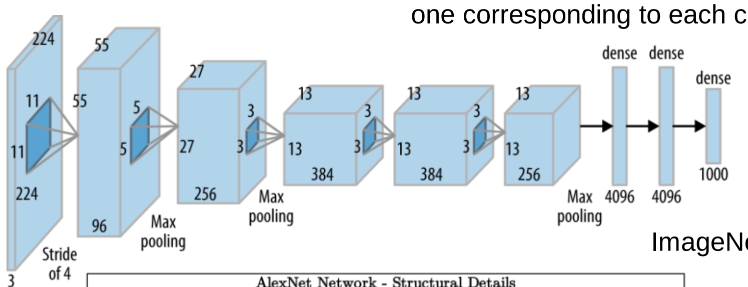
Why use the Cross-Entropy Function Rather Than the Mean Squared Error?

- At the beginning of the backpropagation process, the output value is usually minimal and gradient is also usually very low, making it difficult for the neural network to actually utilize the data it has in adjusting the weights and optimizing itself
- The cross-entropy function, through its logarithm, allows the network to better asses such small errors and work to eliminate them

- The cross-entropy function is only that useful with convolutional neural networks, most particularly for purposes of **classification**
- For **regression** problems, the mean squared error becomes more preferable

Alexnet

Output: 1000x1 probability vector
one corresponding to each class

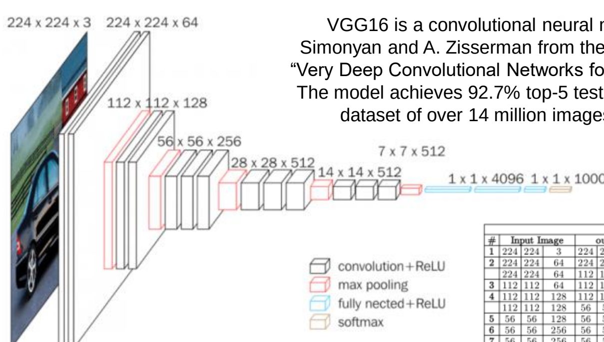


ImageNet DB

AlexNet Network - Structural Details													
Input			Output			Layer	Stride	Pad	Kernel size		in	out	# of Param
227	227	3	55	55	96	conv1	4	0	11	11	3	96	34944
55	55	96	27	27	96	maxpool1	2	0	3	3	96	96	0
27	27	96	27	27	256	conv2	1	2	5	5	96	256	614656
27	27	256	13	13	256	maxpool2	2	0	3	3	256	256	0
13	13	256	13	13	384	conv3	1	1	3	3	256	384	885120
13	13	384	13	13	384	conv4	1	1	3	3	384	384	1327488
13	13	384	13	13	256	conv5	1	1	3	3	384	256	884992
13	13	256	6	6	256	maxpool5	2	0	3	3	256	256	0
						fc6			1	1	9216	4096	37752832
						fc7			1	1	4096	4096	16781312
						fc8			1	1	4096	1000	4097000
						Total							62,378,344

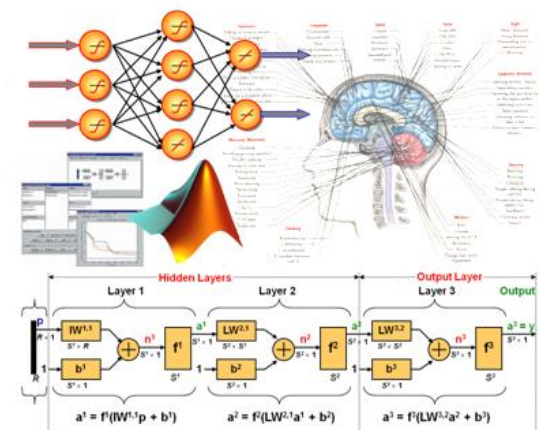
VGG-16 (Classification and Detection)

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.



VGG16 - Structural Details													
#	Input Image			output			Layer	Stride	Kernel		in	out	Param
1	224	224	3	224	224	64	conv3-64	1	3	3	3	64	1792
2	224	224	64	224	224	64	conv3064	1	3	3	64	64	36928
	224	224	64	112	112	64	maxpool	2	2	2	64	64	0
3	112	112	64	112	112	128	conv3-128	1	3	3	64	128	73856
4	112	112	128	112	112	128	conv3-128	1	3	3	128	128	147584
	112	112	128	56	56	128	maxpool	2	2	2	128	128	65664
5	56	56	128	56	56	256	conv3-256	1	3	3	128	256	295168
6	56	56	256	56	56	256	conv3-256	1	3	3	256	256	590080
7	56	56	256	56	56	256	conv3-256	1	3	3	256	256	590080
	56	56	256	28	28	256	maxpool	2	2	2	256	256	0
8	28	28	256	28	28	512	conv3-512	1	3	3	256	512	1180160
9	28	28	512	28	28	512	conv3-512	1	3	3	512	512	2359808
10	28	28	512	28	28	512	conv3-512	1	3	3	512	512	2359808
	28	28	512	14	14	512	maxpool	2	2	2	512	512	0
11	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
12	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
13	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
	14	14	512	7	7	512	maxpool	2	2	2	512	512	0
14	1	1	25088	1	1	4096	fc		1	1	25088	4096	102764544
15	1	1	4096	1	1	4096	fc		1	1	4096	4096	16781312
16	1	1	4096	1	1	1000	fc		1	1	4096	1000	4097000
							Total						138,423,208

2. Training CNNs



Training

- Back-propagation
  - Sparse Connections of CNNs decrease the complexity of Back-Propagation
  - ReLU activation function relieves the vanishing gradient problem
- Stochastic Gradient Descent

Slide credit from Nadav Cohen, “Adam: A Method for Stochastic Optimization”

Loss Minimization Problem

Loss minimization problem:

$$\min_W \left\{ L(W) := \frac{1}{m} \sum_{i=1}^{m} \ell(W; x_i, y_i) + \lambda r(W) \right\}$$

- $\{(x_i, y_i)\}_{i=1}^m$ – training instances (x_i) and corresponding labels (y_i)
- W – network parameters to learn
- $\ell(W; x_i, y_i)$ – loss of network parameterized by W w.r.t. (x_i, y_i)
- $r(W)$ – regularization function (e.g. $\|W\|_2^2$)
- $\lambda > 0$ – regularization weight

Slide credit from Nadav Cohen, “Adam: A Method for Stochastic Optimization”

Large-Scale Setting

- Many network parameters (e.g. $dim(W) \sim 10^8$)
  $\Longrightarrow$ computing Hessian (second order derivatives) is expensive
- Many training examples (e.g. $m \sim 10^6$)
  $\Longrightarrow$ computing full objective at every iteration is expensive

Slide credit from Nadav Cohen, “Adam: A Method for Stochastic Optimization”

Optimization Methods Requirements

- **First-order** – update based on objective value and gradient only
- **Stochastic** – update based on subset of training examples:

$$L_t(W) := \frac{1}{b}\sum_{j=1}^{b} \ell(W; x_{i_j}, y_{i_j}) + \lambda r(W)$$

$\{(x_{i_j}, y_{i_j})\}_{j=1}^{b}$ – random *mini-batch* chosen at iteration t

Slide credit from Nadav Cohen, "Adam: A Method for Stochastic Optimization"

Stochastic Gradient Descent (SGD)

Update rule:

$$\begin{aligned} V_t &= \mu V_{t-1} - \alpha \nabla L_t(W_{t-1}) \\ W_t &= W_{t-1} + V_t \end{aligned}$$

- $\alpha > 0$ – *learning rate* (typical choices: 0.01, 0.1)
- $\mu \in [0, 1)$ – *momentum* (typical choices: 0.9, 0.95, 0.99)

Momentum smooths updates, enhancing stability and speed.

Slide credit from Nadav Cohen, "Adam: A Method for Stochastic Optimization"

Variants to the Basic SGD

- Nestrov's Accelerated Gradient
- Adaptive Gradient (AdaGrad)
- Root Mean Square Propagation (RMSProp)
- Adaptive Moment Estimation (Adam)

Slide credit from Nadav Cohen, "Adam: A Method for Stochastic Optimization"

ADAM

Motivation

Combine the advantages of:

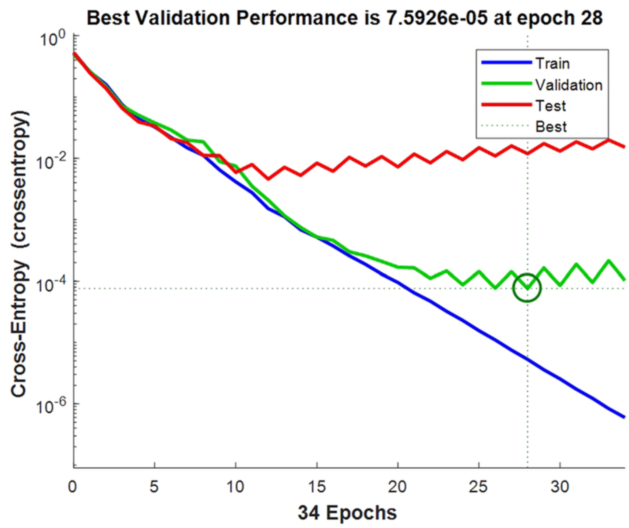
- AdaGrad – works well with sparse gradients
- RMSProp – works well in non-stationary settings

Idea

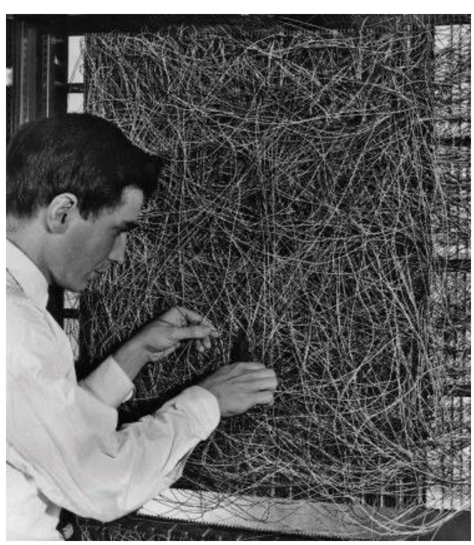
- Maintain exponential moving averages of gradient and its square
- Update proportional to $\frac{\text{average gradient}}{\sqrt{\text{average squared gradient}}}$

Slide credit from Nadav Cohen, "Adam: A Method for Stochastic Optimization"

How to Train a CNN?



3. Greedy Layer-wise Training



Problems in Training Deep Neural Networks (1/2)

- Difficulties of supervised training of deep networks
  - Early layers of MLP do not get trained well
  - Diffusion of Gradient – error attenuates as it propagates to earlier layers
  - Leads to very slow training
  - Exacerbated since top couple layers can usually learn any task "pretty well" and thus the error to earlier layers drops quickly as the top layers "mostly" solve the task– lower layers never get the opportunity to use their capacity to improve results, they just do a random feature map
  - Need a way for early layers to do effective work
  - Instability of gradient in deep networks: Vanishing or exploding gradient
    - Product of many terms, which unless “balanced” just right, is unstable
    - Either early or late layers stuck while “opposite” layers are learning

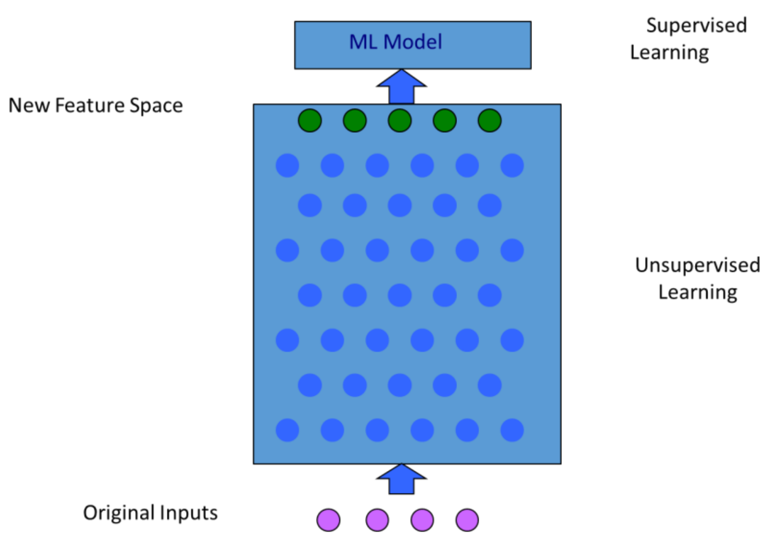
Problems in Training Deep Neural Networks (2/2)

- Often not enough labeled data available while there may be lots of unlabeled data
  - Can we use unsupervised/semi-supervised approaches to take advantage of the unlabeled data
- Deep networks tend to have more sensitive training issues problems than shallow networks during supervised training

Greedy Layer-wise Training

- One answer is greedy layer-wise training
  1. Train first layer using your data without the labels (unsupervised)
  2. Then freeze the first layer parameters and start training the second layer using the output of the first layer as the unsupervised input to the second layer
  3. Repeat this for as many layers as desired
  4. Use the outputs of the final layer as inputs to a supervised layer/model and train the last supervised layer(s) (leave early weights frozen)
  5. Unfreeze all weights and fine tune the full network by training with a supervised approach, given the *pre-training* weight settings

Deep Net with Greedy Layer-wise Training



Greedy Layer-wise Training

- Greedy layer-wise training avoids many of the problems of trying to train a deep net in a supervised fashion
  - –Each layer gets full learning focus in its turn since it is the only current "top" layer
  - –Can take advantage of unlabeled data
  - –When you finally tune the entire network with supervised training the network weights have already been adjusted so that you are in a good error basin and just need fine tuning. This helps with problems of
    - Ineffective early layer learning
    - Deep network local minima

Unsupervised Learning (1/2)

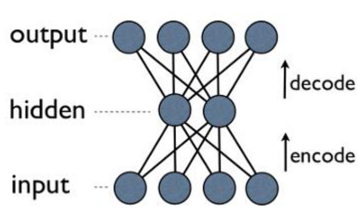
- Model distribution of input data
- Can use **unlabeled data** (unlimited)
- Can be refined with standard supervised techniques (e.g. backprop)
- Useful when the amount of labels is small

Unsupervised Learning (2/2)

- Main idea: model distribution of input data
  - Reconstruction error + regularizer (sparsity, denoising, etc.)
  - Log-likelihood of data
- Models
  - Basic: PCA, KMeans
  - Denoising autoencoders
  - Sparse autoencoders
  - Restricted Boltzmann machines
  - Sparse coding
  - Independent Component Analysis
  - …

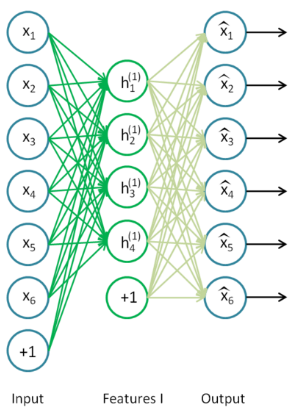
Autoencoders

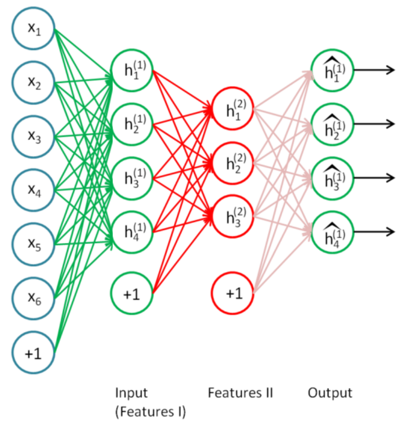
- Try to discover generic features of the data
  - Learn identity function by learning important sub-features (not by just passing through data)
  - Compression, etc.
  - Can use just new features in the new training set or concatenate both



Stacked Autoencoders (1/2)

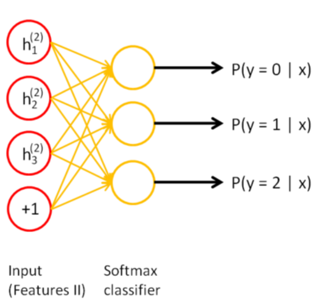
- Stack many (sparse) auto-encoders in succession and train them using greedy layer-wise training
- Drop the decode output layer each time

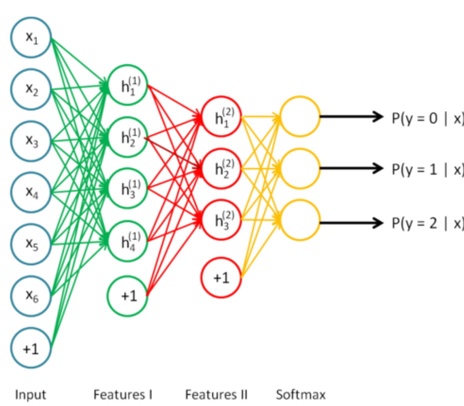




Stacked Autoencoders (2/2)

- Do supervised training on the last layer using final features
- Then do supervised training on the entire network to fine- tune all weights



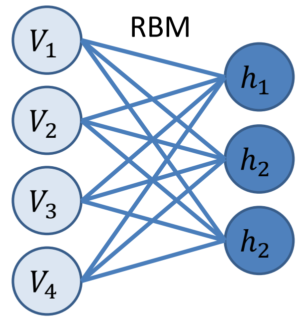


Boltzmann Machines (BM)

- RBMs are energy-based models, they associate a scalar energy to each configuration of the variables of interest
- Energy based probabilistic models define a probability distribution as:

$$p(x) = \frac{e^{-E(x)}}{Z} \text{ where } Z = \sum_x e^{-E(x)}$$

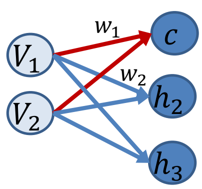
- An energy-based model can be learnt by performing (stochastic) gradient descent on the empirical negative log-likelihood of the training data, where the log-likelihood and the loss function are:



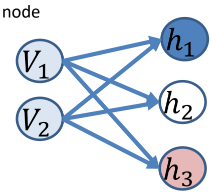
$$L(\theta, D) = \frac{1}{N}\sum_{x^i \epsilon D} log\, p(x^i) \quad \text{and} \quad l(\theta, D) = -L(\theta, D)$$

RBM Training

1. Forward Pass: Inputs are combined with an individual weights and a bias. Some hidden nodes are activated.

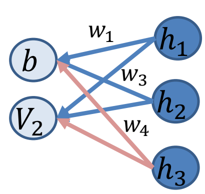


Input being passed to first hidden node



h_1 activates in this example

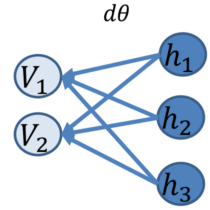
2. Backward Pass: Activations are combined with an individual weight and a bias. Results are passed to the visible layer.



Activations are passed to visible layer for reconstruction

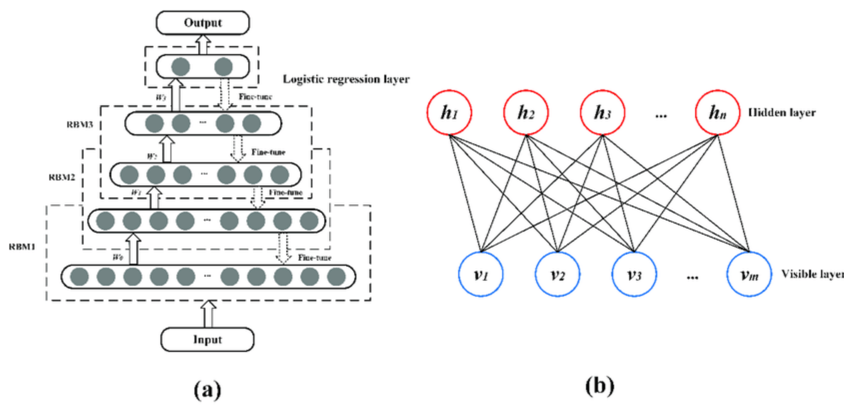
3. Divergence calculation: Input x and samples $\tilde{x}$ are compared in visible layer. Parameters are updated and steps are repeated

$$-\frac{dlogp(x)}{d\theta}$$



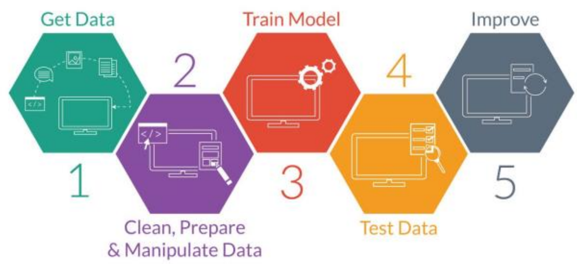
$Update\ W, b, c$

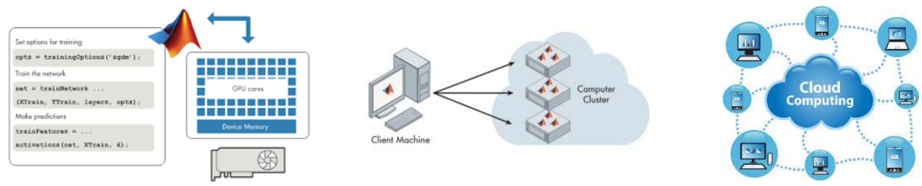
Deep Belief Networks (DBN)



4. Software for Deep Learning

How and Where?





SW Libraries and Toolboxes

- Hunderds of ML toolboxes are now avaiblaes
- **Deeplearn** tools
  - Apache Singa
  - Amazon Machine Learning
  - Azure ML Studio
  - Caffe
  - H2O
  - Massive Online Analysis (MOA)
  - MLlib (Spark)
  - mlpack,
  - Matlab toolboxes
  - Pattern
  - Scikit-Learn
  - Shogun
  - low
  - Theano
  - Torch
  - Veles

Matlab

- https://it.mathworks.com/campaigns/products/offer/deep-learning-with-matlab.html
- https://it.mathworks.com/videos/series/introduction-to-deep-learning.html

Caffe

- Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by Berkeley AI Research (BAIR) and by community contributors.
- http://caffe.berkeleyvision.org/

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

Theano

- Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

- http://deeplearning.net/software/theano/

theano

TensorFlow

- TensorFlow™ is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.
- Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains.
- https://www.tensorflow.org/

PyTorch

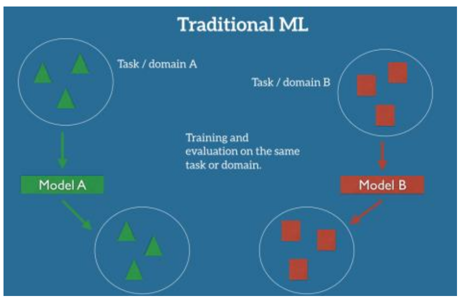
- PyTorch is a Python package that provides two high-level features
  - Tensor computation (like NumPy) with strong GPU acceleration
  - Deep neural networks built on a tape-based autograd system
- https://pytorch.org/

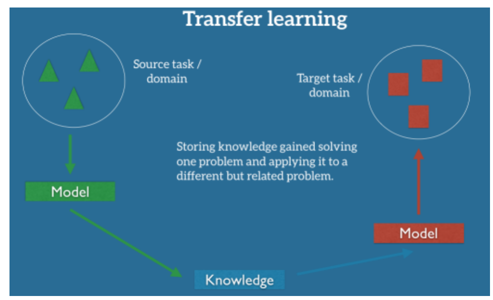
PYTORCH

5. Design of Biometric Systems

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

Transfer Learning to Avoid Overfitting





http://ruder.io/transfer-learning/

Pretrained Models

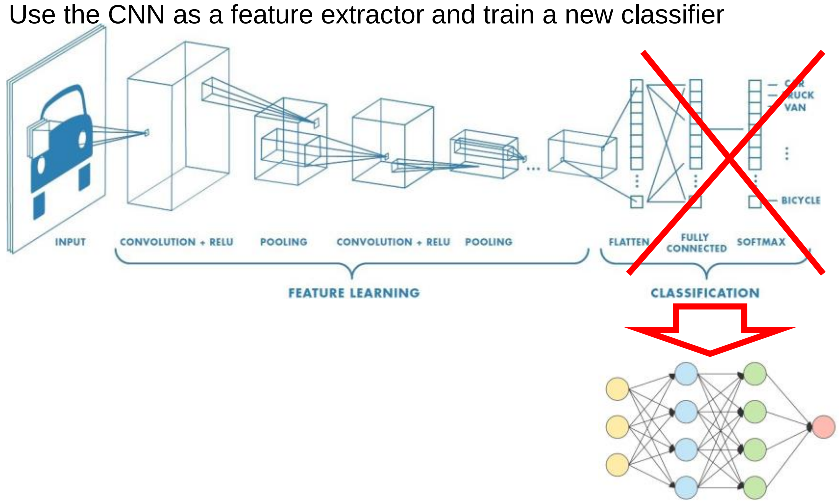
Caffe MODELS

GitHub

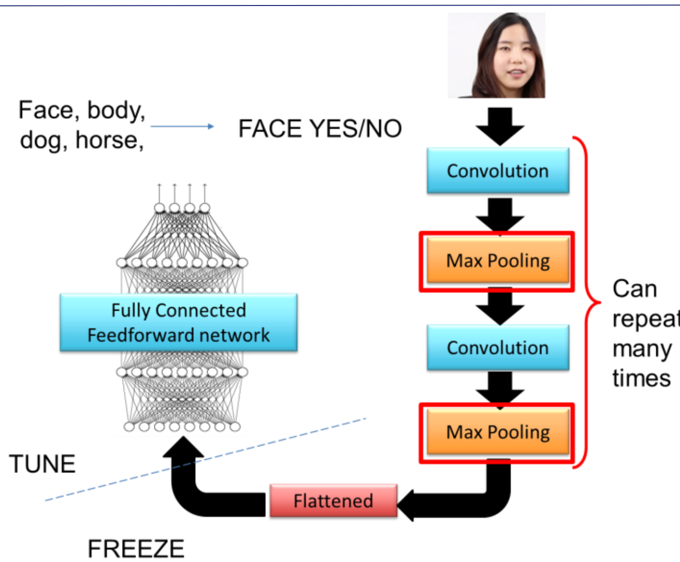
https://github.com/BVLC/caffe/wiki/Model-Zoo

CNNs as Feature Extractors

- Remove the fully connected layers
- Use the CNN as a feature extractor and train a new classifier



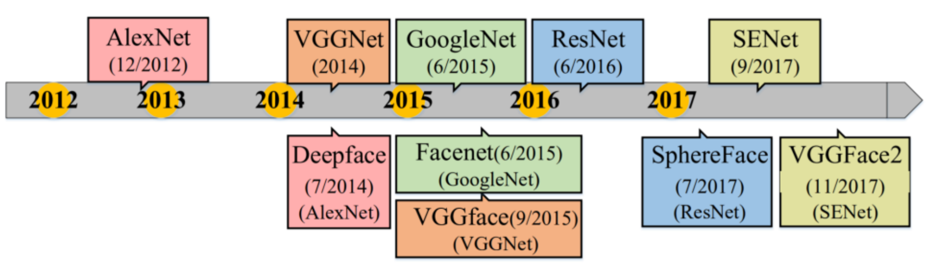
Fine-tuning (1/2)



Fine-tuning (2/2)

- This strategy is to not only replace and retrain the classifier on top of the ConvNet on the new dataset, but to also fine-tune the weights of the pretrained network by continuing the backpropagation
- It is possible to fine-tune all the layers of the ConvNet, or it's possible to keep some of the earlier layers fixed (due to overfitting concerns) and only fine-tune some higher-level portion of the network
- This is motivated by the observation that the earlier features of a ConvNet contain more generic features (e.g. edge detectors or color blob detectors) that should be useful to many tasks, but later layers of the ConvNet becomes progressively more specific to the details of the classes contained in the original dataset

Fine-tuning in Face Recognition



Classical Image Classifiers

- Multiclass

«What object is it?»

Class = ***Tree***

- Binary

«Is it a tree?»

Class = ***YES***

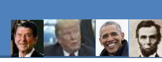
«same class?»

Class = ***YES***

Biometric Classifiers

- Multiclass = Identifier

«who is this person?»



Class = ***Donald***

- Binary = Verification

«are the same person?»

Class = ***NO***

Identification Using Deep Learning

Deep Learning

Sample

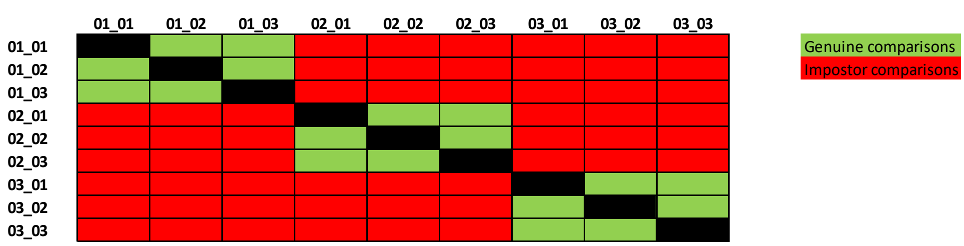
→ Trained Classifier → Integer Identifier

Use samples of every individual during the training step

Algoritmic

```
For each template i in Gallery
        M(i)  = identiry_verification(Fresh, Gallery(i))
end
ID = argmax(M)
```


Identification Verification: Imbalanced Classes



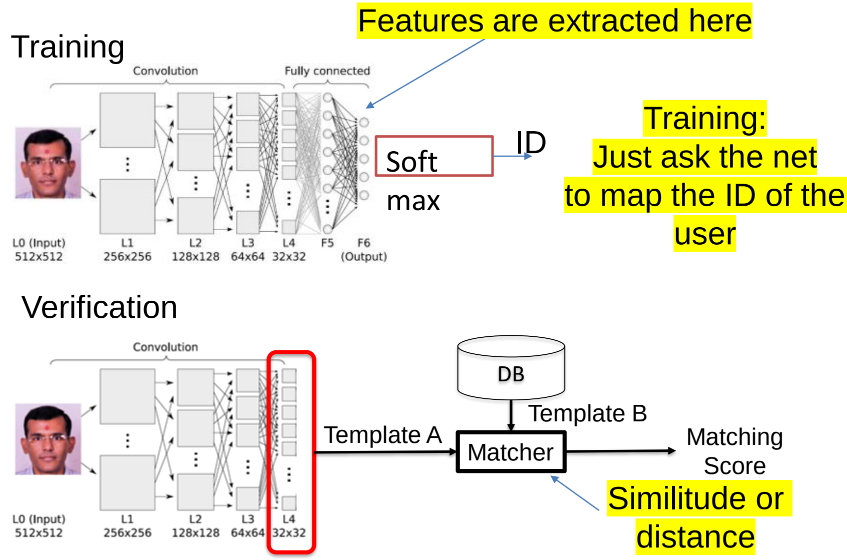
genuine comparisons:

$$N_{subjects} \times N_{sampleSubject} \times (N_{sampleSubject} - 1)$$

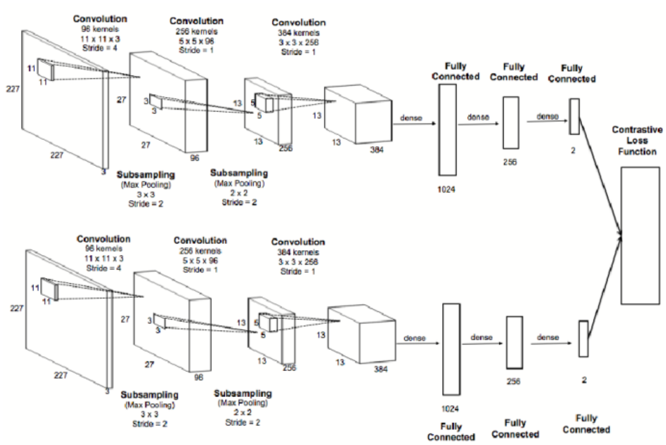
impostor comparisons:

$$(N_{subjects} \times N_{sampleSubject})^2 - (N_{subjects} \times N_{sampleSubject} \times (N_{sampleSubject} - 1))$$

Identity Verification Using Algorithmic Matchers

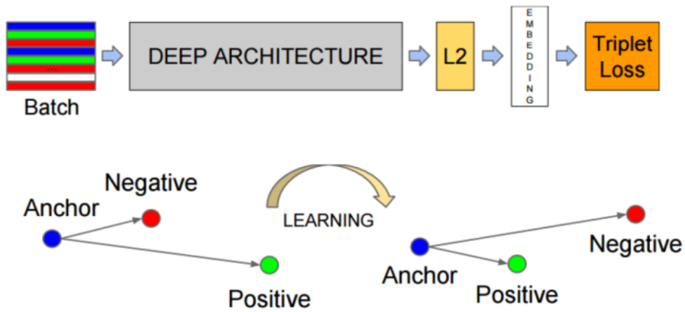


Identity Verification Using Siamese Networks



Y. Taigman, M. Yang, M. Ranzato and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 1701-1708

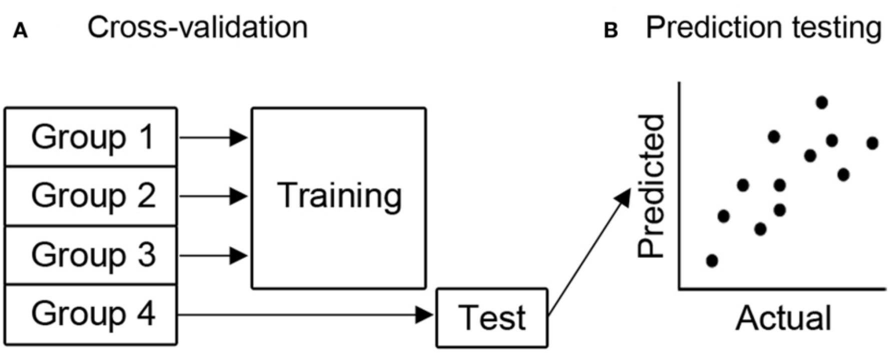
Triplet Loss



$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815-823.

General Approach: Cross Validation (CV)

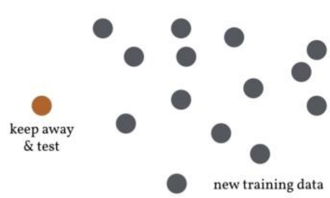


e_i = error for a specific partition

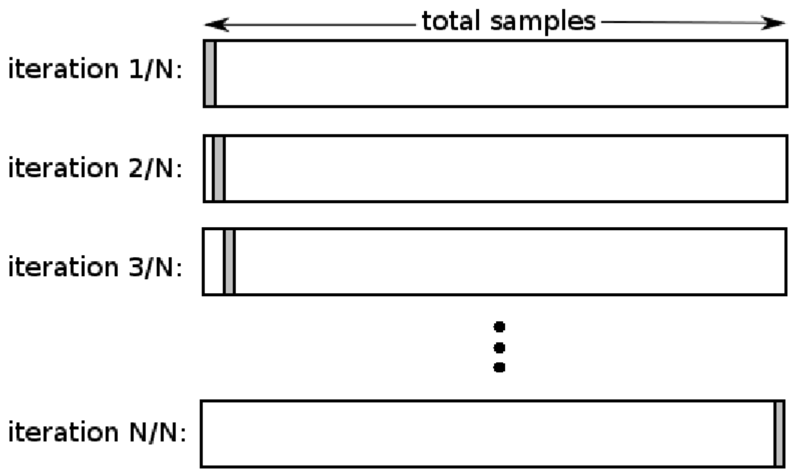
Leave One Out!
(LOO)

Data Set Partitioning: Leave One Out

- Is an extreme case of *k*-FCV → *k* equals the number of examples in the data set
- In each step only one instance is used to test the model whereas the rest of instances are used to learn it.



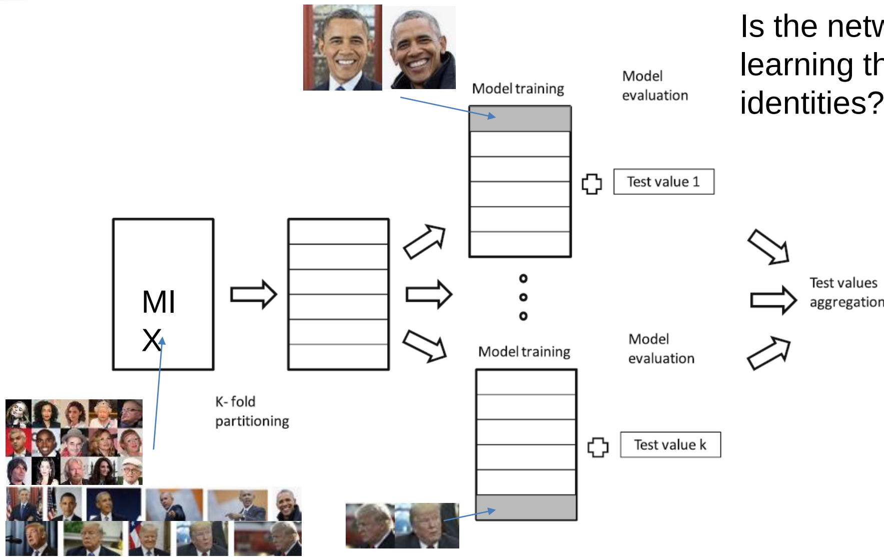
Data Set Partitioning: Leave One Out



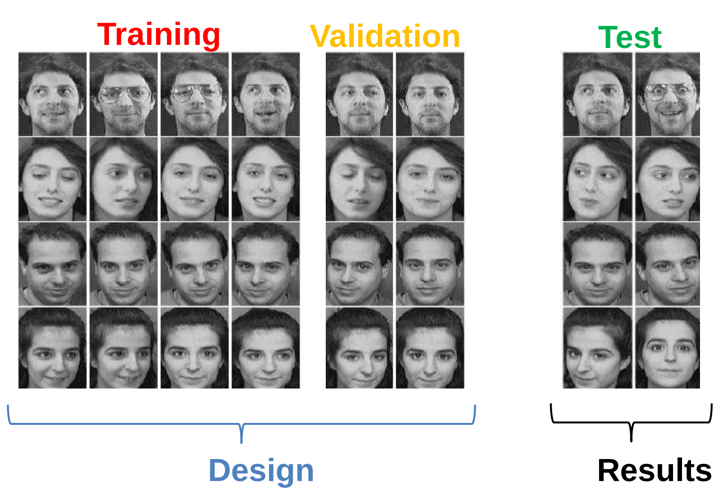
Leave One Person Out! (LOPO)

LOPO
L(NP)O

Is the network learning the identities?

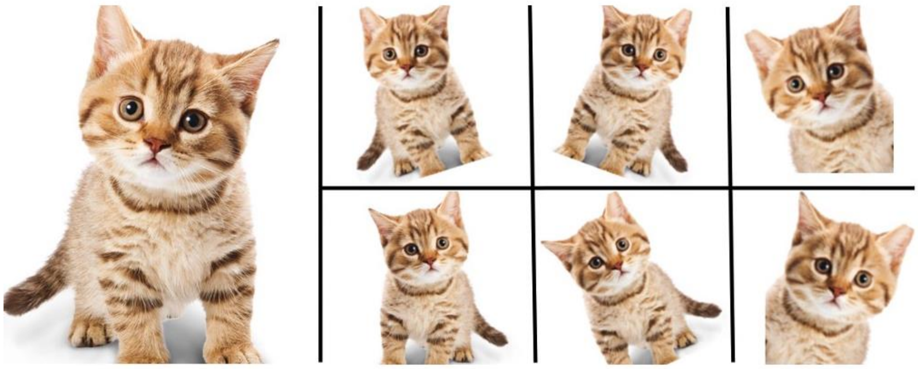


Training Deep Neural Networks for Identification



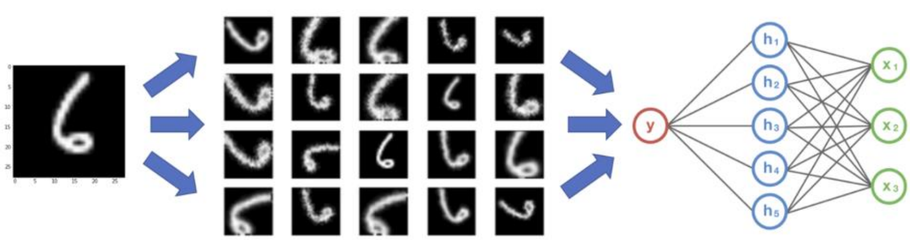
Data Augmentation

How to use Deep Learning with few images



Enlarge your Dataset

Data Augmentation: Add Noise

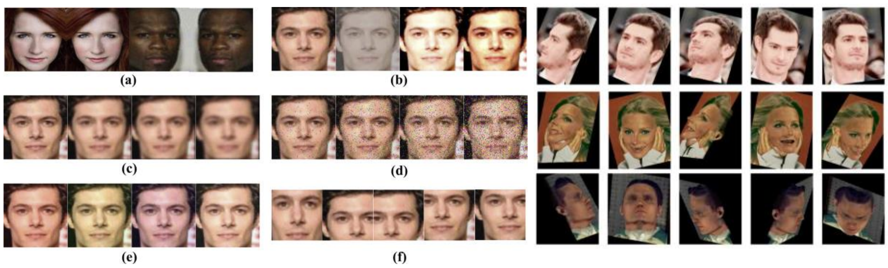


Same number,
but shifted/tilted, plus different types of noise

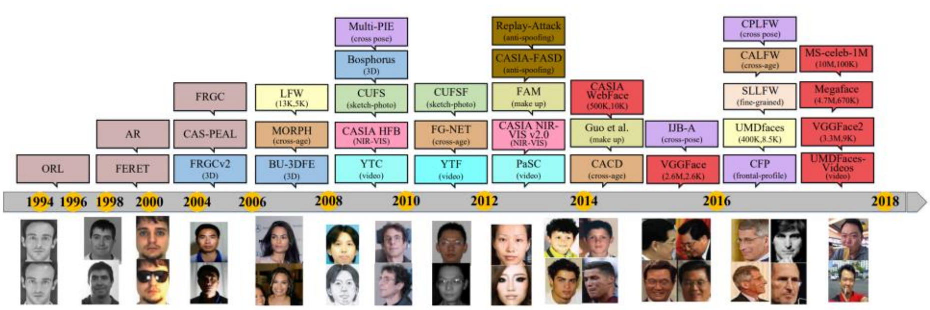
Data Augmentation: Also for Large Datasets

Landmark perturbation for face alignment.

Flipping
patches (clipping)
color casting
blurring

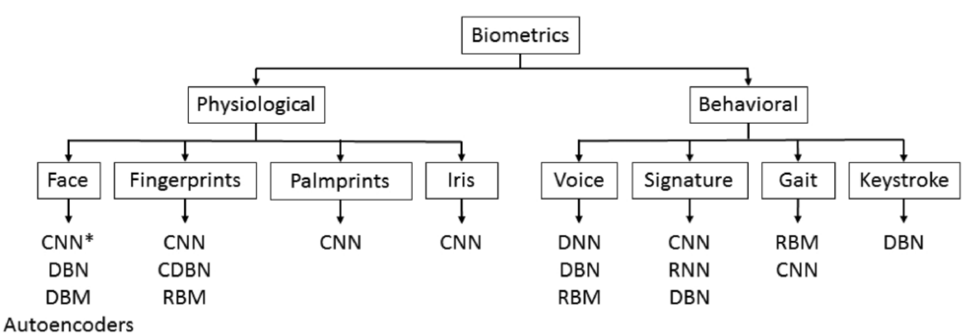


Evolution of Face Datasets



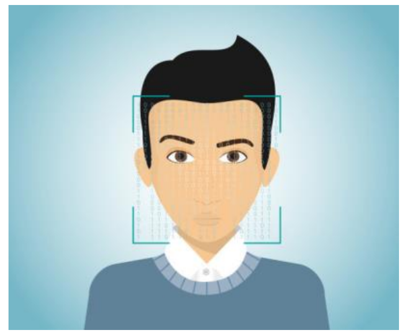
6. Biometric Applications

Different Deep Learning Techniques for Different Biometric Traits



Kalaivani Sundararajan,D. L. Woodard, Deep Learning for Biometrics: A Survey, ACM Comput. Surv. 51, 3, May 2018.

6.1 Face



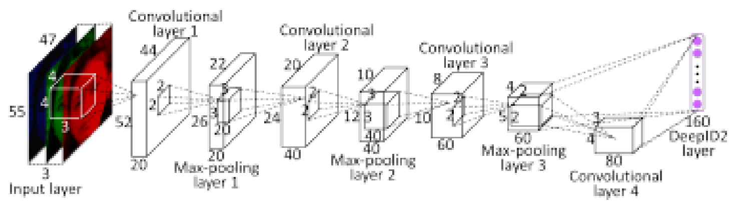
Some Deep Learning Approaches for Face Recognition

Method	Recognition mode	Input	#Convolution layers	#Pooling layers	#Fully connected layers	Classification	#nets
Sun et al. [136]	Verification	60 face patches (39 × 31 and 31 × 31 patches)	4 layers (20 × 4 × 4 SW, 40 × 3 × 3 SW, 60 × 3 × 3 LS, 80 × 2 × 2 US)	3 (max pooling with 2 × 2 filters)	1 (160-dim DeepID features)	Joint Bayesian	60
Sun et al. [135]	Verification	25 face patches	4 layers (20 × 4 × 4 SW, 40 × 3 × 3 SW, 60 × 3 × 3 LS, 80 × 2 × 2 LS)	3 (max pooling with 2 × 2 filters)	1 (160-dim DeepID2 features)	Joint Bayesian	25
Sun et al. [137]	Verification	25 face patches	4 layers (128 × 4 × 4 SW, 128 × 3 × 3 SW, 128 × 3 × 3 LS, 128 × 2 × 2 LS)	3 (max pooling with 2 × 2 filters)	4 (512-dim DeepID2+ features)	Joint Bayesian	25
Hu et al. [59]	Verification - Medium	58 × 58 faces	3 layers (16 × 5 × 5, 32 × 4 × 4, 48 × 3 × 3)	3 layers (2 × 2 filters)	1 (160-dim)	Joint Bayesian	1
Taigman et al. [142]	Verification	152 × 152 RGB faces	5 layers (32 × 11 × 11 SW, 16 × 9 × 9 SW, 16 × 9 × 9 LS, 16 × 7 × 7 LS, 16 × 5 × 5 LS)	1 layer (3 × 3 filters)	1 (4096-dim)	Weighted χ^2 similarity	1
Zhu et al. [176]	Identification	96 × 96 grayscale faces	3 layers (32 × 5 × 5 LS, 32 × 5 × 5 LS, 32 × 5 × 5 LS)	2 layers (2 × 2 filters)	1 (96 × 96 reconstruction layer)	LDA	1
Pattabhi et al. [108]	Identification	28 × 32 faces	2 layers (6 × 5 × 5, 12 × 5 × 5)	2 layers (2 × 2 filters)		Neural network	1
Liu et al. [84]	Verification & Identification	7 face patches	9 layers	Yes	1 layer	softmax layer & triplet loss	7
Zhou et al. [174]	Verification	4 face patches	10 layers	Yes	1 layer	softmax layer	4
Chen et al. [18]	Verification & Identification	100 × 100 faces	10 layers (32 × 3 × 3, 64 × 3 × 3 (2), 128 × 3 × 3, 96 × 3 × 3, 192 × 3 × 3, 128 × 3 × 3, 256 × 3 × 3, 160 × 3 × 3, 320 × 3 × 3)	5 layers (2 × 2 (4), 7 × 7 mean)	1 (10548-dim)	Joint Bayesian	1
Parkhi et al. [107]	Verification	224 × 224 face patches	13 layers (64 × 3 × 3 (2), 128 × 3 × 3 (2), 256 × 3 × 3 (3), 512 × 3 × 3 (3))	5 layers (2 × 2 max-pooling)	3 (4096, 4096, 2622)	triplet loss metric learning	1
Schroff et al. [128]	Verification, identification & clustering	224 × 224 face patches	11 layers (64 × 7 × 7, 64 × 1 × 1, 64 × 3 × 3, 192 × 1 × 1, 192 × 3 × 3, 384 × 1 × 1, 384 × 3 × 3, 256 × 1 × 1, 256 × 3 × 3, 256 × 1 × 1, 256 × 3 × 3)	4 layers (3 × 3 filters)	3 (32 × 128, 32 × 128, 128)	triplet loss embedding	1
Wen et al. [153]	Verification	-	6 layers (128 × 3 × 3, 128 × 3 × 3 (2), 128 × 3 × 3, 256 × 3 × 3LS, 256 × 3 × 3LS, 256 × 3 × 3LS)	4 layers (2 × 2)	1 (512-dim)	softmax & center loss	1
Sun et al. [138]	Verification	25 face patches	10 layers (64 × 3 × 3, 64 × 3 × 3, 96 × 3 × 3, 96 × 3 × 3, 192 × 3 × 3, 192 × 3 × 3, 256 × 3 × 3, 256 × 3 × 3, 256 × 3 × 3LS, 256 × 3 × 3LS)	4 (max pooling with 2 × 2 filters)	1 (512-dim)	Joint Bayesian	25

Kalaivani Sundararajan,D. L. Woodard, Deep Learning for Biometrics: A Survey, ACM Comput. Surv. 51, 3, May 2018.

Examples of Face Recognition Methods (1/2)

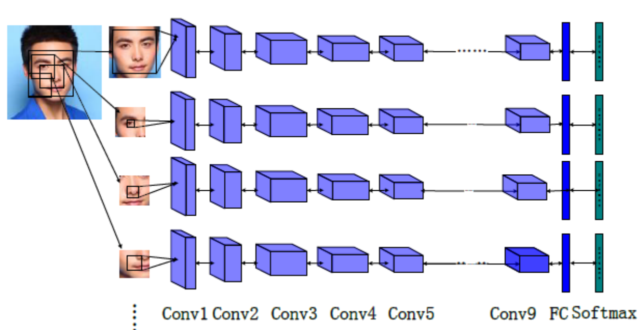
Feature extraction



Matching
– PCA for dimensionality reduction
– Log likelyhood ratio

Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang, Deep learning face representation by joint identification-verification, in Proc. of the 27th Int. Conf. on Neural Information Processing Systems, 2014.

Examples of Face Recognition Methods (1/2)



Jinguo Liu, Yafeng Deng, and Chang Huang, Targeting ultimate accuracy: Face recognition via deep embedding, 2015.

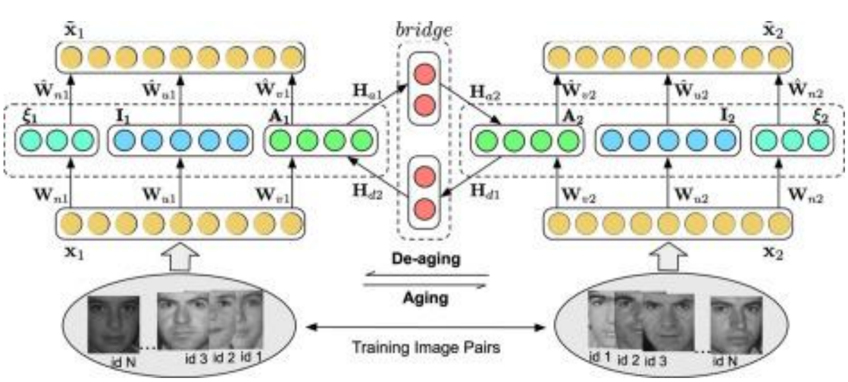
Some Results for Face Recognition

Table 2. Face Recognition Results: LFW (13,323 Images, 5,749 Celebrities)

Methods	Arch.	Training Dataset	Training #img	Training #subj	Protocol	Verification acc (%)	Open set acc (%)	Closed acc (%)
Cao et al. [14]	Joint Bayesian	-	-	-	unrestricted	96.33 ± 1.08	-	-
Chen et al. [17]	LBP	-	-	-	unrestricted	95.17 ± 1.13	-	-
Lu et al. [89]	Gaussian Face	-	-	-	unrestricted	98.52 ± 0.66	-	-
Best et al. [11]	COTS-s1+s4	-	-	-	unrestricted	-	66.5	35
Sun et al. [136]	CNN*	CelebFaces+	202,599	10,177	unrestricted	97.45 ± 0.26	-	-
Sun et al. [135]	CNN*	CelebFaces+	202,599	10,177	unrestricted	96.39 ± 0.13	-	-
Sun et al. [137]	CNN*	CelebFaces+, WDRef	290,000	12,000	Jain	99.47 ± 0.12	80.7	95.0
Taigman et al. [142]	CNN*	SFC	4,000,000	4,000	unrestricted	97.35 ± 0.25	-	-
Liu et al. [84]	CNN*	Private	1,200,000	18,000	unrestricted	99.41	**95.80**	**98.03**
Chen et al. [18]	CNN*	CASIA-WebFace	490,356	10,548	unrestricted	97.45 ± 0.70	-	-
Parkhi et al. [107]	CNN*	Private	2,600,000	2,622	unrestricted	98.95	-	-
Schroff et al. [128]	CNN*	Private	100M-200M	8M	unrestricted	**99.63 ± 0.09**	-	-
Zhou et al. [174]	CNN*	Megvii	5,000,000	20,000	-	99.5	-	-
Taigman et al. [143]	CNN*	SFC	4,500,000	55,000	unrestricted	97.17	46.3	72.3
Wen et al. [153]	CNN*	private dataset	700,000	-	-	99.28	-	-
Sun et al. [138]	CNN*	CelebFaces+, WDRef	290,000	12,000	-	99.30	-	-
Peng et al. [109]	CNN*	CASIA WebFace	494,414	10,575	-	96.60	-	-

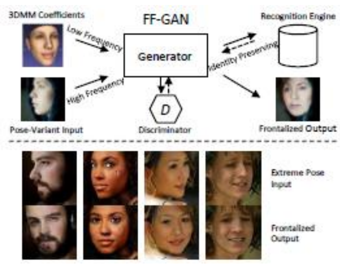
*Deep learning approach.

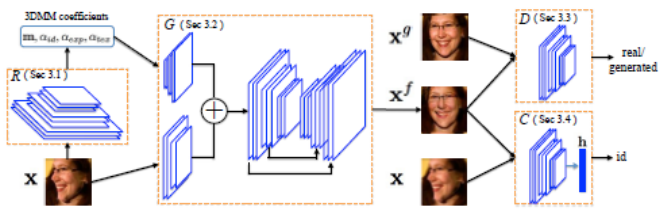
Age Invariant Face Recognition



Chenfei Xu, Qihe Liu, Mao Ye, Age invariant face recognition and retrieval by coupled auto-encoder networks, Neurocomputing, vol. 222, 2017, pp. 62-71.

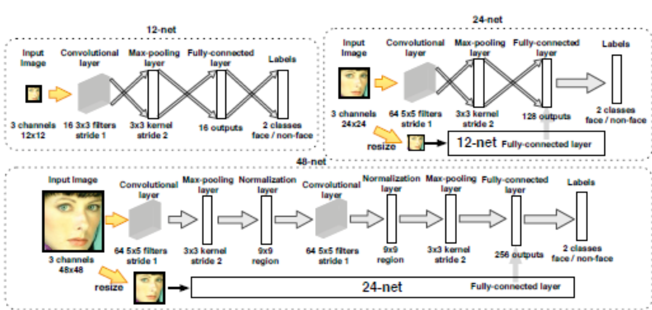
Rotation Invariant Face Recognition

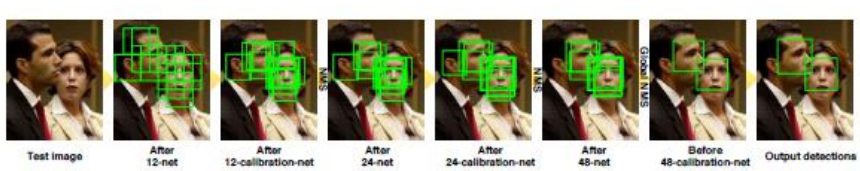




Chenfei Xu, Qihe Liu, Mao Ye, Age invariant face recognition and retrieval by coupled auto-encoder networks, Neurocomputing, vol. 222, 2017, pp. 62-71.

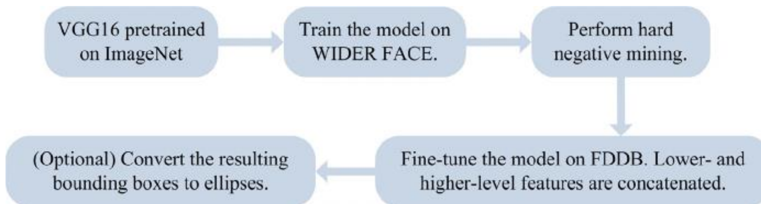
Face Detection (1/2)

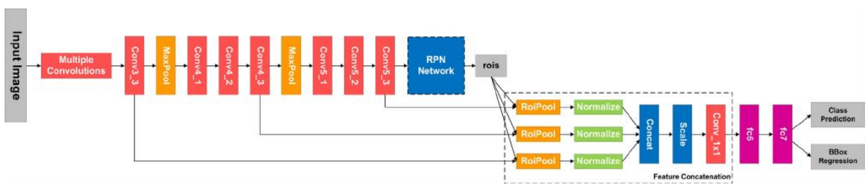




H. Li, Z. Lin, X. Shen, J. Brandt and G. Hua, A convolutional neural network cascade for face detection, in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 5325-5334.

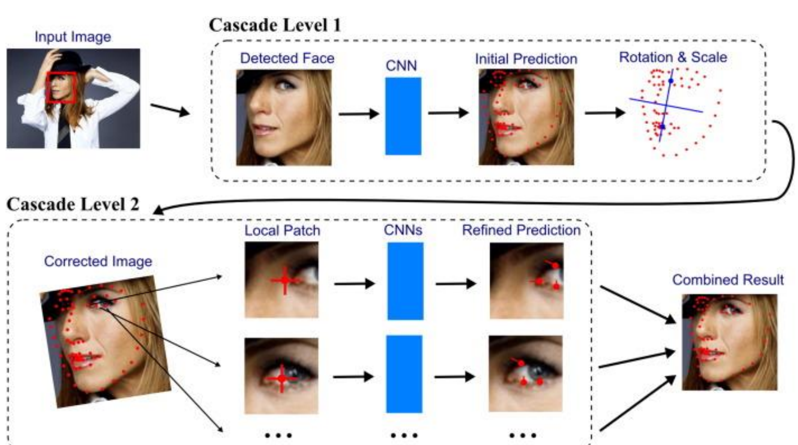
Face Detection (2/2)





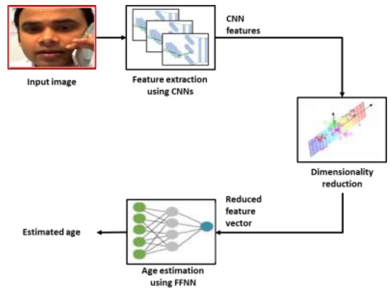
X. Sun, P. Wu, S.C.H. Hoi, Face detection using deep learning: An improved faster RCNN approach, Neurocomputing, volume 299, 2018, pp. 42-50.

Estimation of Fiducial Points



H. Fan, E. Zhou, Approaching human level facial landmark localization by deep learning, Image and Vision Computing, vol. 47, 2016, pp. 27-35.

Age Estimation



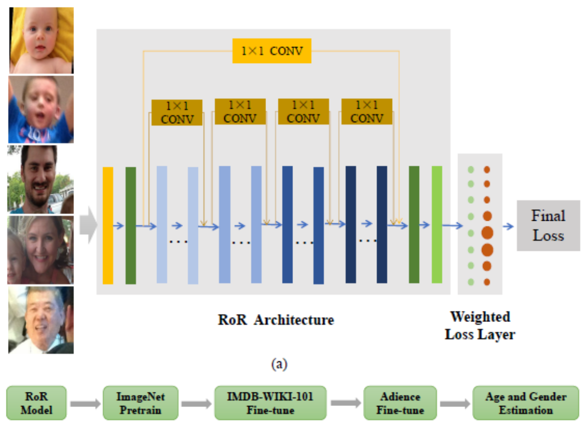
A. Anand, R. Donida Labati, A. Genovese, E. Muñoz, V. Piuri and F. Scotti, Age estimation based on face images and pre-trained Convolutional Neural Networks, in Proc. of the IEEE Symp. on Computational Intelligence for Security and Defense, pp. 1-7, November 27-30, 2017.

Some Results for Age Estimation

Attribute	Testing		Methods	Arch.	Training		#classes	MAE	Acc (%)
	Dataset	#img			Dataset	#img			
Age	MORPH-II	-	Guo et al. [49]	BIF + KCCA	-	-	-	3.98	-
		-	Guo et al. [48]	BIF + KPLS	-	-	-	4.04	-
		5-fold CV	Huerta et al. [62]	CNN*	MORPH-II	55,134	Regr.	3.88	-
		42,635	Yi et al. [162]	CNN*	MORPH-II	10,634	Regr.	3.63	-
		5,670	Qiu et al. [111]	CNN*	MORPH-II	47,582	Regr.	3.41	-
		44,634	Li et al. [81]	CNN*	MORPH-II	10,500	Regr.	3.61	-
		1,095	Wang et al. [151]	CNN*	MORPH-II	4,380	Regr.	4.77	-
		5-fold CV	Liu et al. [83]	CNN*	MORPH-II	-	Regr.	2.89	-
			Rothe et al. [121]	CNN*	IMDB-Wiki	523,051	101	**2.68**	-
			Han et al. [52]	CNN*	IMDB-Wiki	523,051	Regr	3.0	85.3
	Adience	5-fold CV	Levi et al. [79]	CNN*	Adience	26,000	8	-	84.70 ± 2.2
			Liu et al. [83]	CNN*	Adience, MORPH-II, ChaLearn	-	-	-	**98.2 ± 0.7**
			Rothe et al. [121]	CNN*	IMDB-Wiki	523,051	20	-	96.6 ± 0.9
	ChaLearn	1136	Liu et al. [83]	CNN*	MORPH-II, FG-Net	-	-	0.315	-
			Ranjan et al. [116]	CNN*	ChaLearn, Adience, MORPH	7,000	Regr	0.359	-
			Rothe et al. [121]	CNN*	IMDB-Wiki	523,051	101	**0.282**	-
			Liu et al. [87]	CNN*	CASIA WebFace, MORPH-II	1,315,000	-	0.287	-
			Ranjan et al. [115]	CNN*	IMDB-Wiki, Adience, MORPH	299,818	Regr	0.293	-
			Han et al. [52]	CNN*	IMDB-Wiki	523,051	3	0.289	-

CV, cross-validation; LOPO, leave one person out.
*Deep learning approach.

Gender and Ethnicity Estimation



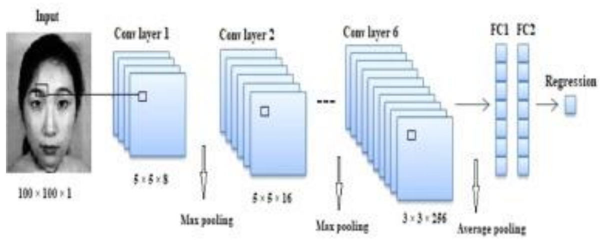
K .Zhang, C. Gao, L. Guo. M. Sun, X. Yuan, T.X. Han, Z. Zhao, B. Li, Age Group and Gender Estimation in the Wild With Deep RoR Architecture, IEEE Access, vol. 5 , 2017.

Some Results for Gender and Ethnicity Estimation

Attribute	Testing		Methods	Arch.	Training		#classes	MAE	Acc (%)
	Dataset	#img			Dataset	#img			
Gender	MORPH-II	-	Guo et al. [49]	BIF + KCCA	-	-	-	-	98.45
		-	Guo et al. [48]	BIF + KPLS	-	-	-	-	98.35
		42,635	Yi et al. [162]	CNN*	MORPH-II	10,634	2	-	97.90
		44,634	Li et al. [81]	CNN*	MORPH-II	10,500	2	-	**98.48**
			Han et al. [52]	CNN*	IMDB-Wiki	523,051	2	-	98.0
	AR	1,275	Jiang et al. [67]	CNN*	FERET, CAS-PEAL	10,800	2	-	70.50
		3,288	Juefei et al. [69]	CNN*	MugshotDB, Pinellas	89,003	2	-	**85.62**
	Adience	5-fold CV	Levi et al. [79]	CNN*	Adience	26,000	2	-	86.80 ± 1.4
Ethnicity	MORPH-II	-	Guo et al. [49]	BIF + KCCA	-	-	-	-	98.95
		-	Guo et al. [48]	BIF + KPLS	MORPH-II	10,634	2	-	**99.0**
		42,635	Yi et al. [162]	CNN*	MORPH-II	10,634	2	-	98.60
			Han et al. [52]	CNN*	IMDB-Wiki	523,051	3	-	98.6

CV, cross-validation; LOPO, leave one person out.
*Deep learning approach.

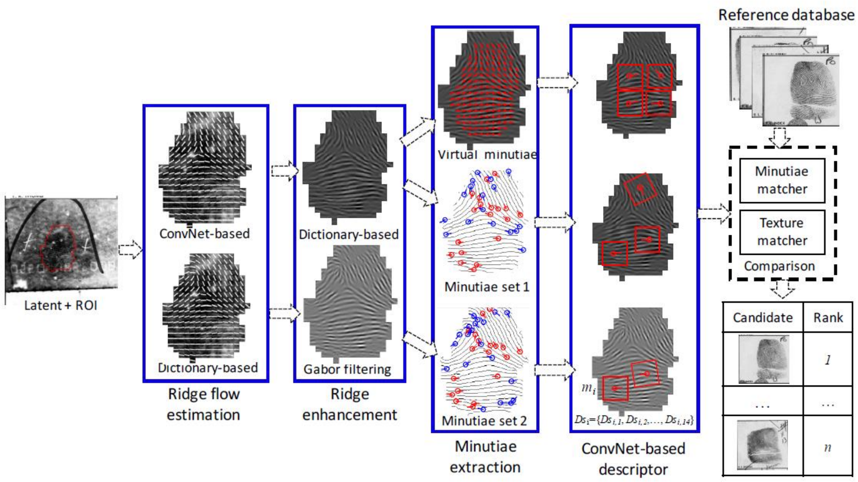
Emotion Estimation



N. Jain, S. Kumar, A. Kumar, P. Shamsolmoali, M. Zareapoor, Hybrid deep neural networks for face emotion recognition, Pattern Recognition Letters, 2018.

6.2 Fingerprint

Latent Fingerprint



K. Cao and A. K. Jain, Automated Latent Fingerprint Recognition, in IEEE Trans. on Pattern Analysis and Machine Intelligence, 2017.

Some Results for Fingerprint Recognition

Dataset	Methods	Representation	EER(%)	Rank1 (%)
FVC 2002	Hong et al. [57]	Gabor	24.34	-
	Chikkerur et al. [22]	STFT	**21.99**	-
	Sahasrabudhe et al. [122]	cRBM*	22.65	-
	Sahasrabudhe et al. [123]	cDBN*	23.95	-
NIST SD27	COTS latent AFIS	COTS	-	**67.0**
	CAO et al. [13]	CNN*	-	65.0
WVU DB	COTS latent AFIS	COTS	-	71.0
	CAO et al. [13]	CNN*	-	**75.0**

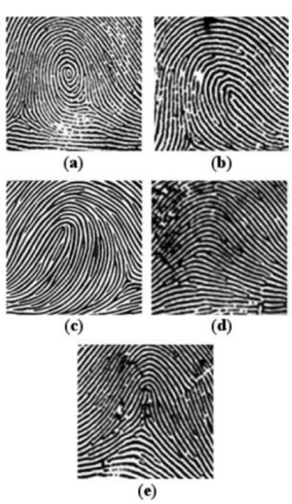
*Deep learning approach.

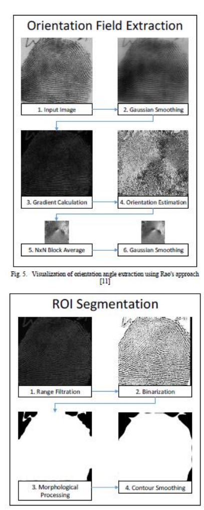
Live fingerprint acquisitions?

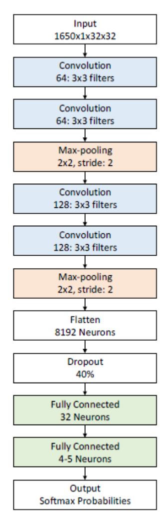
- Local features
- Repetitive pattern
- Rotations
- Non-linear distortions

Kalaivani Sundararajan,D. L. Woodard, Deep Learning for Biometrics: A Survey, ACM Comput. Surv. 51, 3, May 2018.

Fingerprint Classification

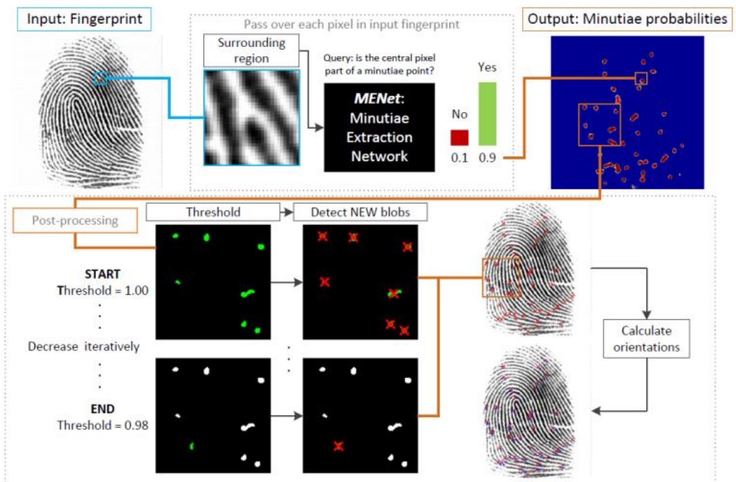






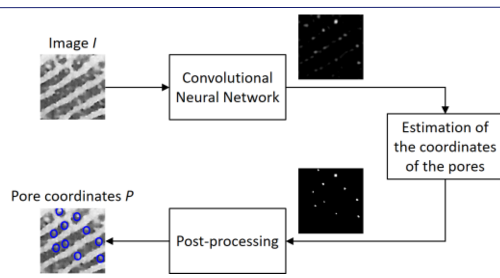
J. M. Shrein, Fingerprint classification using convolutional neural networks and ridge orientation images, 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017, pp. 1-8.

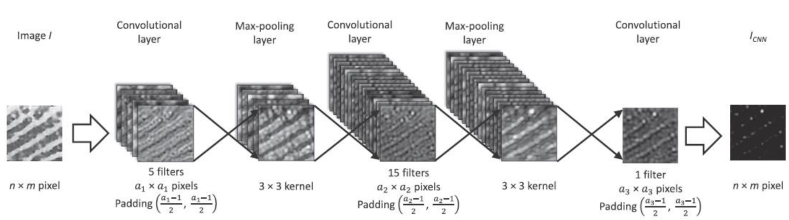
Minutiae Extraction



L. N. Darlow and B. Rosman, Fingerprint minutiae extraction using deep learning, IEEE International Joint Conference on Biometrics (IJCB), 2017, pp. 22-30

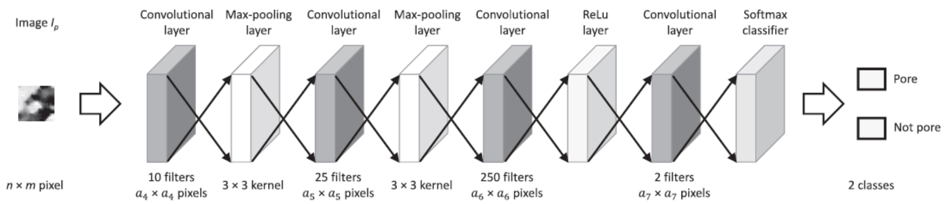
Pore Extraction (1/2)

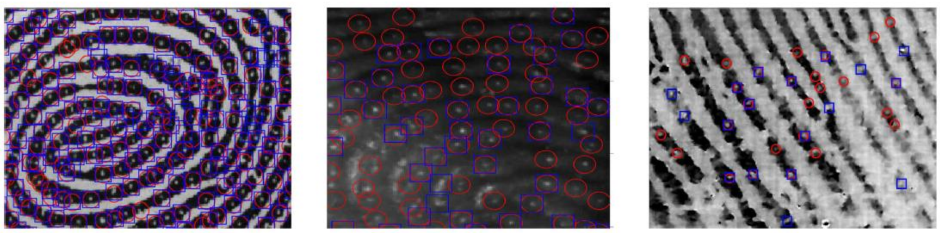




R. Donida Labati, A. Genovese, E. Muñoz, V. Piuri, F. Scotti, A novel pore extraction method for heterogeneous fingerprint images using Convolutional Neural Networks, Pattern Recognition Letters, Vol. 113, 2018, pp. 58-66.

Pore Extraction (2/2)

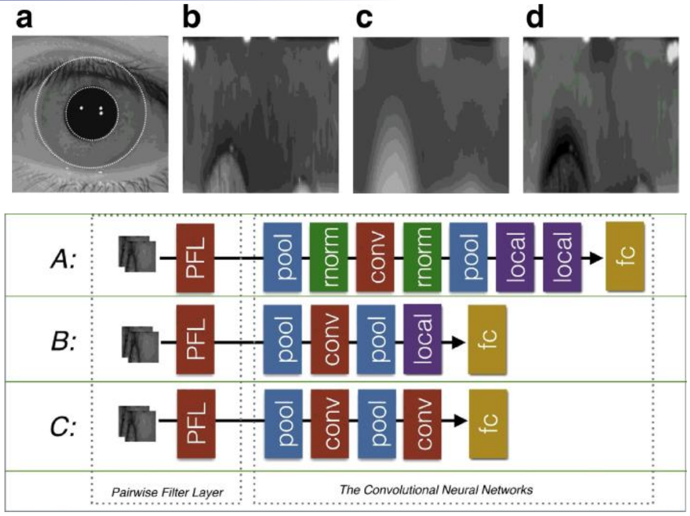




R. Donida Labati, A. Genovese, E. Muñoz, V. Piuri, F. Scotti, A novel pore extraction method for heterogeneous fingerprint images using Convolutional Neural Networks, Pattern Recognition Letters, Vol. 113, 2018, pp. 58-66.

6.3 Iris

Iris Recognition



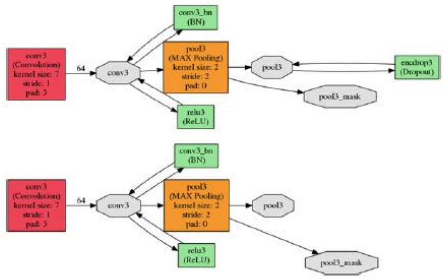
N. Liu, M. Zhang, H. Li, Z. Sun, T. Tan, DeepIris: Learning pairwise filter bank for heterogeneous iris verification, Pattern Recognition Letters, vol. 82, 2016, pp. 154-161.

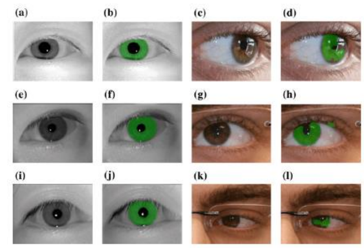
Some Results for Iris Recognition

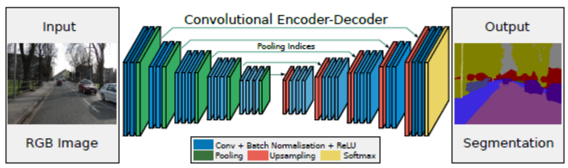
Dataset	Methods	Representation	EER(%)
MICHE-I	Daugman et al. [24]	Gabor	8.35
	Raghavendra et al. [113]	LBP	5.22
	Raja et al. [114]	SAE*	**3.93**
VSSIRIS	Daugman et al. [24]	Gabor	3.57
	Raghavendra et al. [113]	LBP	9.45
	Raja et al. [114]	SAE*	**1.70**
Q-FIRE	Weinberger et al. [152]	LMNN	1.73
	Liu et al. [85]	MDML	1.67
	Liu et al. [86]	CNN*	**0.15**
LG2200	Daugman et al. [24]	Gabor	7.12
	Gangwar et al. [41]	CNN*	**2.40**
LG4000	Daugman et al. [24]	Gabor	5.30
	Gangwar et al. [41]	CNN*	**1.82**

*Deep learning approach.

Iris Segmentation



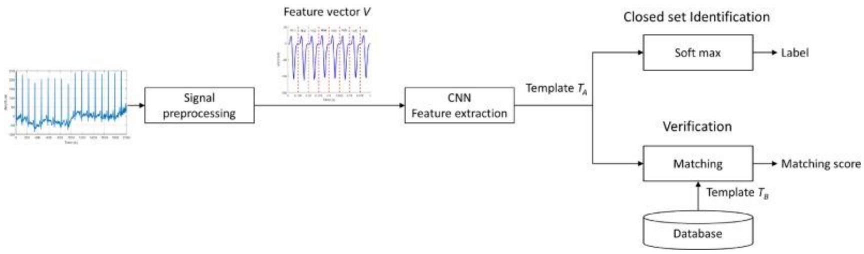


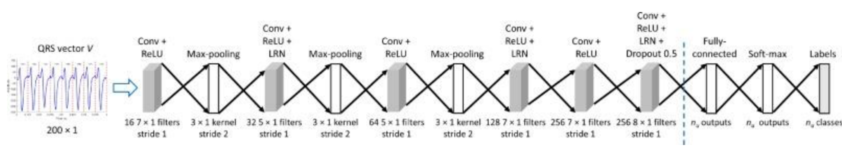


E. Jalilian, A. Uhl, Iris Segmentation Using Fully Convolutional Encoder–Decoder Networks. In Deep Learning for Biometrics, Springer, Cham, 2017.

6.4 Other Biometric Traits

Electrocardiographic Signals (ECG)



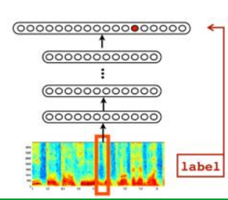


R. Donida Labati, E. Muñoz, V. Piuri, R. Sassi, F. Scotti, Deep-ECG: Convolutional Neural Networks for ECG biometric recognition, Pattern Recognition Letters, 2018.

Some Results for Speaker Recognition

Testing					Training			
Dataset	#trials	#subj	Methods	Arch.	Dataset	#trials	#subj	EER(%)
SRE 2012	C2,C5	-	Li et al. [78]	UBM-EM (4096)	SRE 2012	-	1,918	2.18
				DNN*				**1.66**
	C2	1,040	Kenny et al. [70]	DNN*	SRE 2012	4,432	1,040	2.16
SRE 2012 training data	male	1,000	Vasilakakis et al. [148]	GMM	SRE 2012 training data	28,920	1,818	**0.45**
				DBN*				0.58
SRE 2010 telephone	-	7,196	Garcia et al. [42]	GMM	Switchboard I & II	33,039	3,114	6.92
				DNN*				4.20
			Saleem et al. [125]	DNN*	SRE 2004 & 2005 & 2006			**2.18**
SRE 2006	51,068	816	Ghahabi et al. [44]	i-vector	SRE 2004 & 2005	6,000	-	7.18
				DBN*				6.44
			Ghahabi et al. [46]	RBM*	SRE 2004 & 2005	6,125	-	7.58
			Ghahabi et al. [47]	DBN & DNN*	SRE 2004 & 2005 & 2006	-	-	**4.76**

*Deep learning approach.



Some Results for Signature Verification

Datasets	Methods	Representation	EER(%)	FAR(%)	FRR(%)	Acc(%)
SVS 2004	Fallah et al. [36]	Mellin transform, MFCC, etc.	3.0	-	-	-
	Ansari et al. [6]	Fuzzy modeling	2.46	-	-	-
	Fayyaz et al. [38]	SAE*	**2.15**	-	-	-
	Lai et al. [75]	RNN*	2.37	-	-	-
GPDS-300	Ferrer et al. [39]	Geometric features	-	13.12	15.41	86.65
	Vargas et al. [145]	High-pressure pts	-	14.66	10.01	**87.67**
	Ribeiro et al. [118]	DBN*	-	14.67	20.25	82.85
	Hafemann et al. [51]	CNN*	10.70	9.08	20.60	-
	Haemann et al. [50]	CNN*	**3.47**	**5.13**	**6.55**	-
	Dey et al. [29]	CNN*	-	23.17	23.17	76.83

*Deep learning approach.

Some Results for Palmprint Verification

Datasets	Methods	Representation	Recognition Accuracy(%)	EER(%)
PolyU	Lu et al. [90]	Enhanced GRCM	98.0	-
	Xu et al. [157]	Quaternion PCA+Quaternion DWT	98.83	-
	Jia et al. [65]	KPCA on HOL	99.73	-
	Jalali et al. [64]	CNN*	99.98	-
	Minaee et al. [102]	Scattering networks*	**100.0**	-
	Dian et al. [30]	CNN*	-	0.0443

*Deep learning approach.

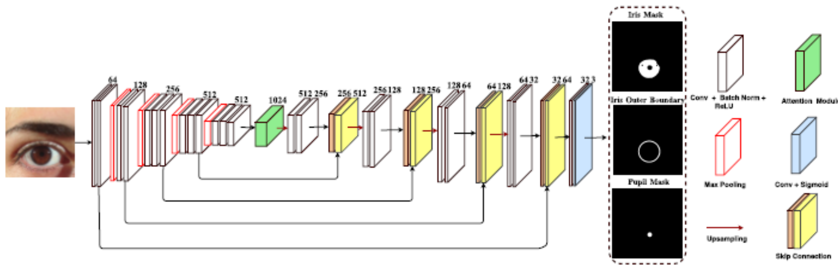
Some Results for Gait Recognition

Datasets	Methods	Representation	Accuracy(%)
	Different views		
CASIA-B	Kusakunniran et al. [74]	CCA	68.5
	Yu et al. [164]	GEI+NN	23.76
	Wu et al. [156]	CNN*	84.67
	Yan et al. [159]	CNN*	30.55
	Alotaibi et al. [5]	CNN*	85.51
	Hossain et al. [58]	RBM*	92.50
	Wolf et al. [155]	3D-CNN*	**97.35**
	Different scenes		
	Hu et al. [60]	LF+iHMM	71.76
	Kusakunniran et al. [73]	STIP	79.66
	Yan et al. [159]	CNN*	**95.0**
	Alotaibi et al. [5]	CNN*	86.70
OU-ISIR	Muramatsu et al. [103]	TCM+	72.80
	Muramatsu et al. [104]	wQVTM	70.51
	Wu et al. [156]	CNN*	94.8
	Zhang et al. [166]	CNN*	80.50
	Shiraga et al. [129]	CNN*	90.45
	Li et al. [80]	CNN*	**95.04**

*Deep learning approach.

6.4 Other Applications

Iris Segmentation

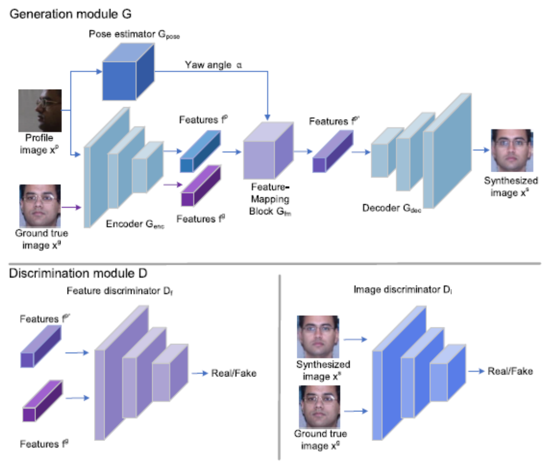


C. Wang, J. Muhammad, Y. Wang, Z. He and Z. Sun, "Towards Complete and Accurate Iris Segmentation Using Deep Multi-Task Attention Network for Non-Cooperative Iris Recognition," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 2944-2959, 2020.

Aging

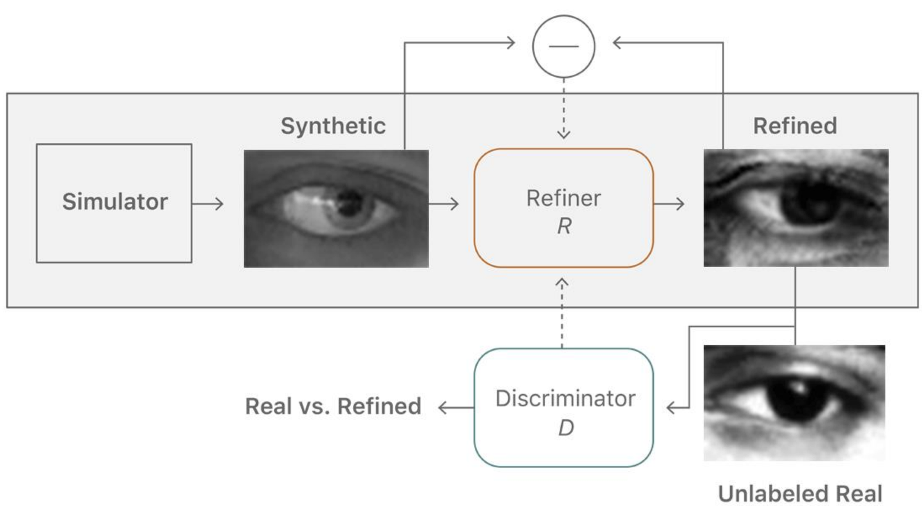
A. Genovese, V. Piuri, F. Scotti, "Towards explainable face aging with Generative Adversarial Networks", in Proc. of the 26th IEEE Int. Conf. on Image Processing (ICIP 2019), Taipei, Taiwan, pp. 3806-3810, September 22-25, 2019

Pose Compensation



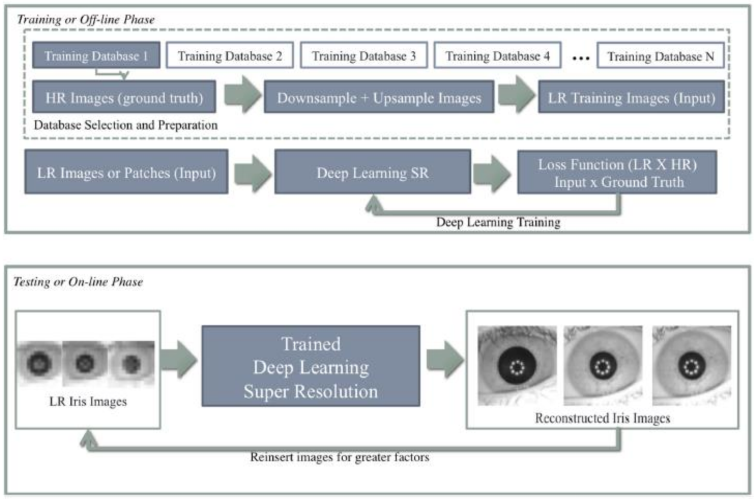
C. Rong, X. Zhang and Y. Lin, "Feature-Improving Generative Adversarial Network for Face Frontalization," in IEEE Access, vol. 8, pp. 68842-68851, 2020, doi: 10.1109/ACCESS.2020.2986079.

Generative Adversarial Networks for Data Augmentation



https://machinelearning.apple.com/research/gan

Super-resolution

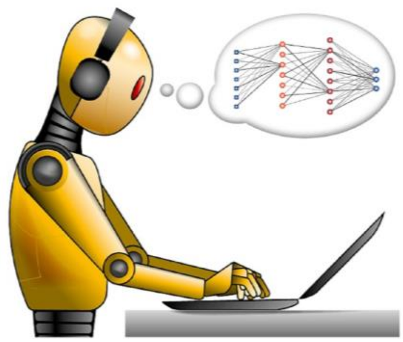


E. Ribeiro, A. Uhl and F. Alonso-Fernandez, "Iris super-resolution using CNNs: is photo-realism important to iris recognition?," in IET Biometrics, vol. 8, no. 1, pp. 69-78, 1 2019

7. Summary

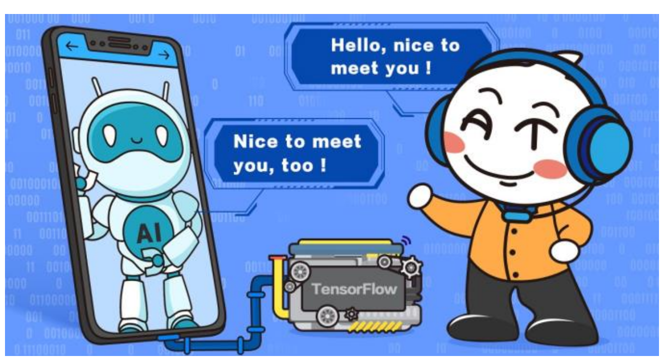
Deep Learning

- Convolutional neural networks
- Greedy layer-wise training
- Software for deep learning

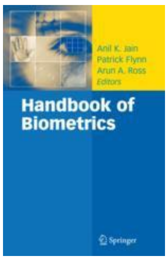


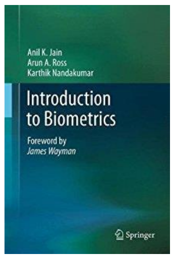
Deep Learning for Biometrics

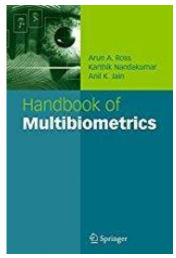
- Design of biometric systems
- Applications of artificial intelligence in biometrics

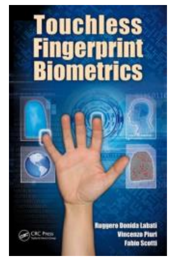


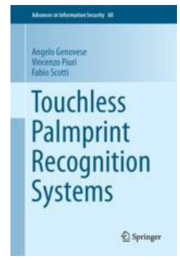
Thank you!


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