Computer Vision

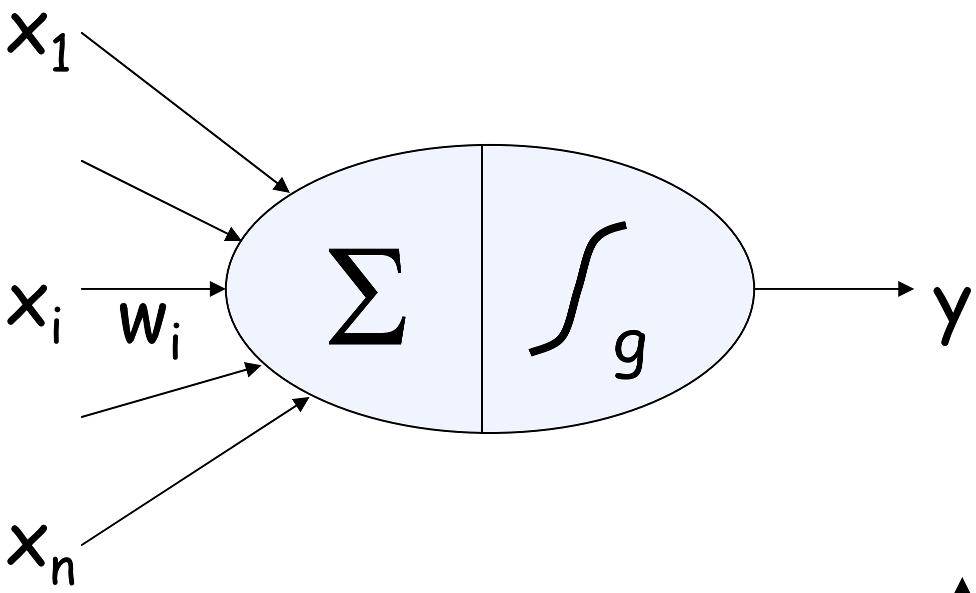
Lecture 12

Neural networks for Computer Vision

Usage of NN for Computer Vision

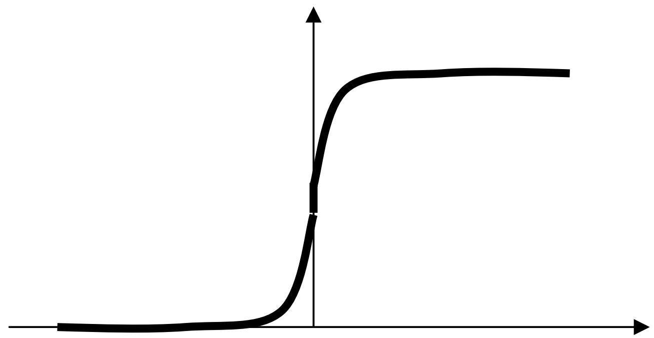
- Recognition of objects (scenes)
  - Based on classification (supervised learning)
- Categorization of objects (scenes)
  - Based on clustering (unsupervised learning)
- Recognition of motion
  - Based on prediction

Unit (Neuron)

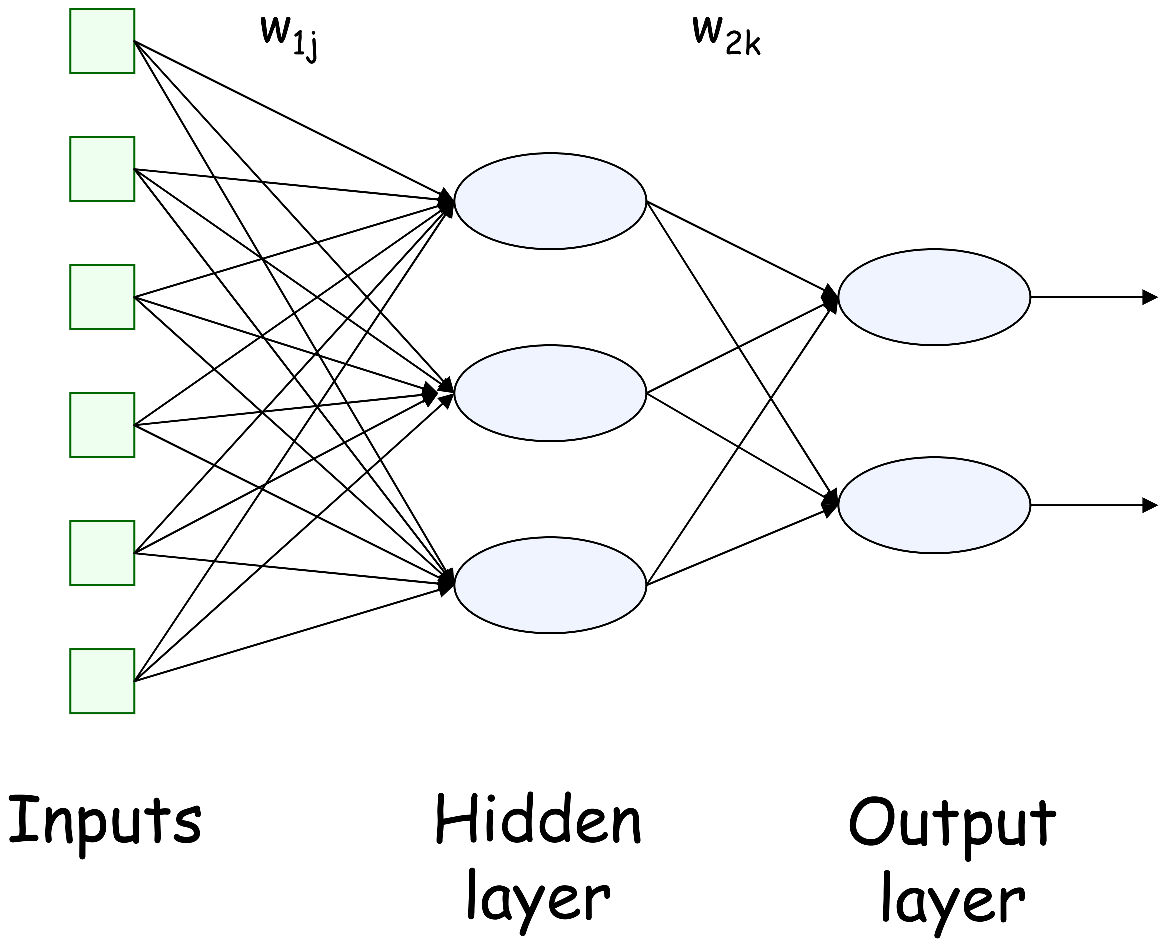


$$y = g\left(\Sigma_{i=1,\ldots,n} w_i x_i\right)$$

$$g(u) = 1/[1 + \exp(-\alpha \times u)]$$

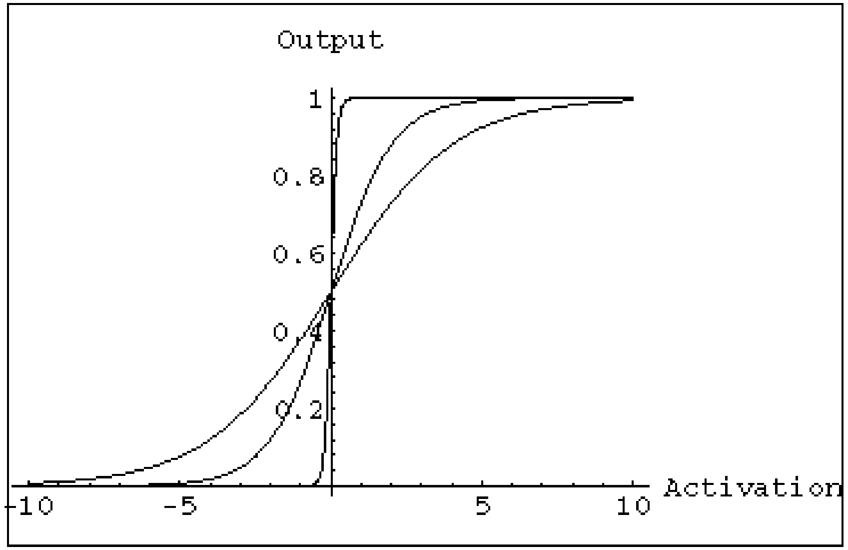


Two-Layer Feed-Forward Neural Network



Typical Activation Functions

- $F(x) = 1 / (1 + e^{-k \sum (w_i x_i)})$
- Shown for
- k = 0.5, 1 and 10



- Using a nonlinear function which approximates a linear threshold allows a network to approximate nonlinear functions

Backpropagation (Principle)

- New example $y(k) = f(x(k))$
- $\varphi(k)$ = outcome of NN with weights $w(k-1)$ for inputs $x(k)$
- Error function: $E(k)(w(k-1)) = ||\varphi(k) – y(k)||2$
- $wij(k) = wij(k-1) – \varepsilon\times\partial E/\partial wij$ $(w(k) = w(k-1) - e\times\nabla E)$
- Backpropagation algorithm:
  Update the weights of the inputs to the last layer, then the weights of the inputs to the previous layer, etc.

BP Network Details

- Forward Pass:
  - Error is calculated from outputs
  - Used to update output weights
- Backward Pass:
  - Error at hidden nodes is calculated by back propagating the error at the outputs through the new weights
  - Hidden weights updated

In Matrix Form

- For:
- n inputs, m hidden nodes
- and q outputs
- **o**lk is the output of the lth neuron
- For the kth of p patterns

$$A = \begin{pmatrix} a_{10} & a_{11} & \cdots & a_{1n} \\ a_{20} & a_{21} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m0} & a_{m1} & \cdots & a_{mn} \end{pmatrix},$$

$$B = \begin{pmatrix} b_{10} & b_{11} & \cdots & b_{1m} \\ b_{20} & b_{21} & \cdots & b_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ b_{q0} & b_{q1} & \cdots & b_{qm} \end{pmatrix}.$$

$$o_{lk} = f_H\left(\sum_{j=0}^{m} b_{lj} f_H\left(\sum_{i=0}^{n} a_{ji} x_{ik}\right)\right), \qquad 1 \leq k \leq p.$$

- **v**k is the output of the hidden layer
- **o**k is the true output vector

$$\boldsymbol{v}_k = \begin{pmatrix} 1 \\ F_H(\boldsymbol{A}\boldsymbol{x}_k) \end{pmatrix}$$

$$\boldsymbol{o}_k = F(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{x}_k) = F_H(\boldsymbol{B}\boldsymbol{v}_k)$$

Matrix Tricks

E(**A**, **B**) = k=1pΣ (**tk** – **o**k)T(**t**k – **o**k)

- tk denotes true output vectors

The optimal weight matrix of B can be computed directly if fH-1(**t**) is known

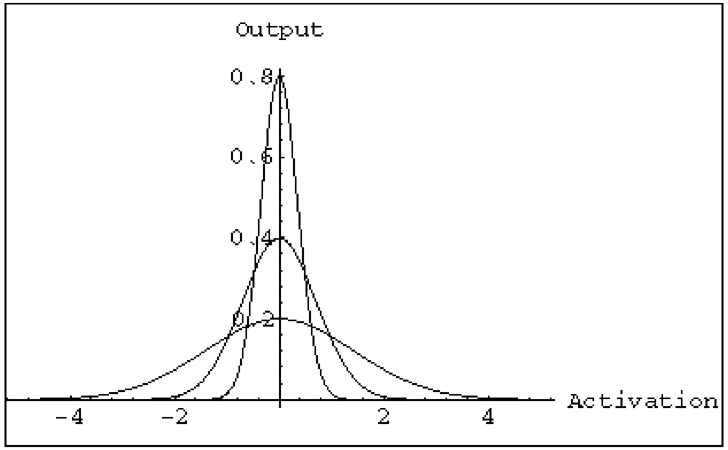
- **B**’ = fH-1(**t**)**v**T(**vv**T)*
- So…   E(**A**, **B**) = E(**A**, **B**(**A**)) = E’(**A**)
    - Which makes our weight space much smaller

Comments and Issues

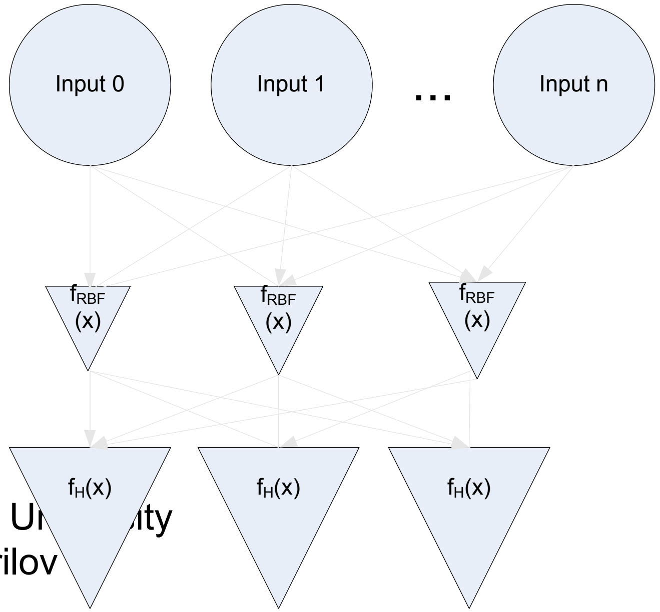
- How to choose the size and structure of networks?
  - If network is too large, risk of over-fitting (data caching)
  - If network is too small, representation may not be rich enough
- Role of representation: e.g., learn the concept of an odd number
- Incremental learning

Alternative Activation functions

- Radial Basis Functions
  - Square
  - Triangle
  - Gaussian!



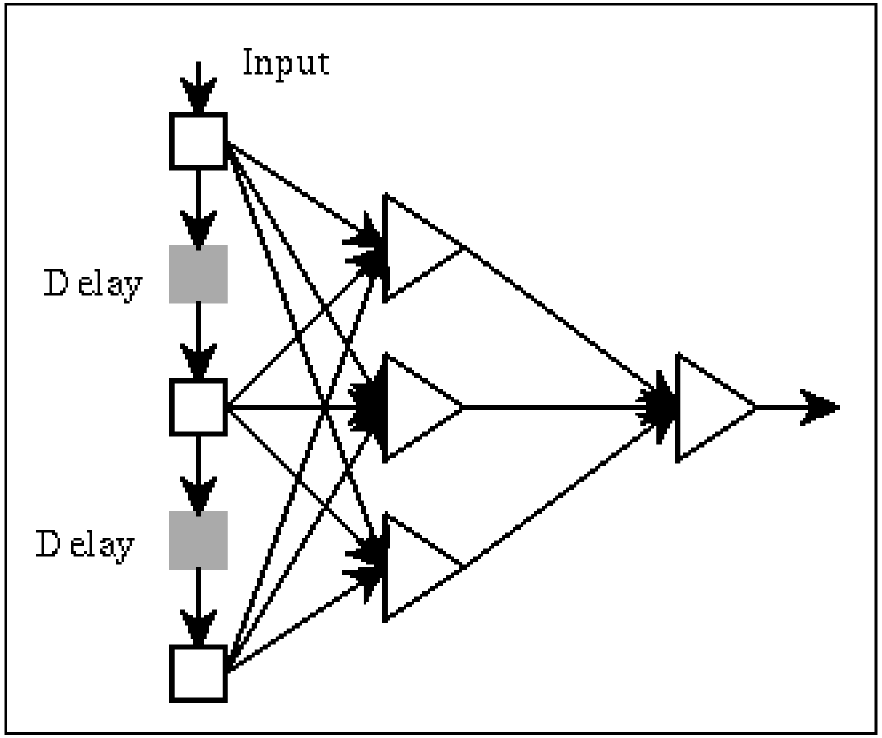
- (μ, σ) can be varied at each hidden node to guide training



Alternate Topologies

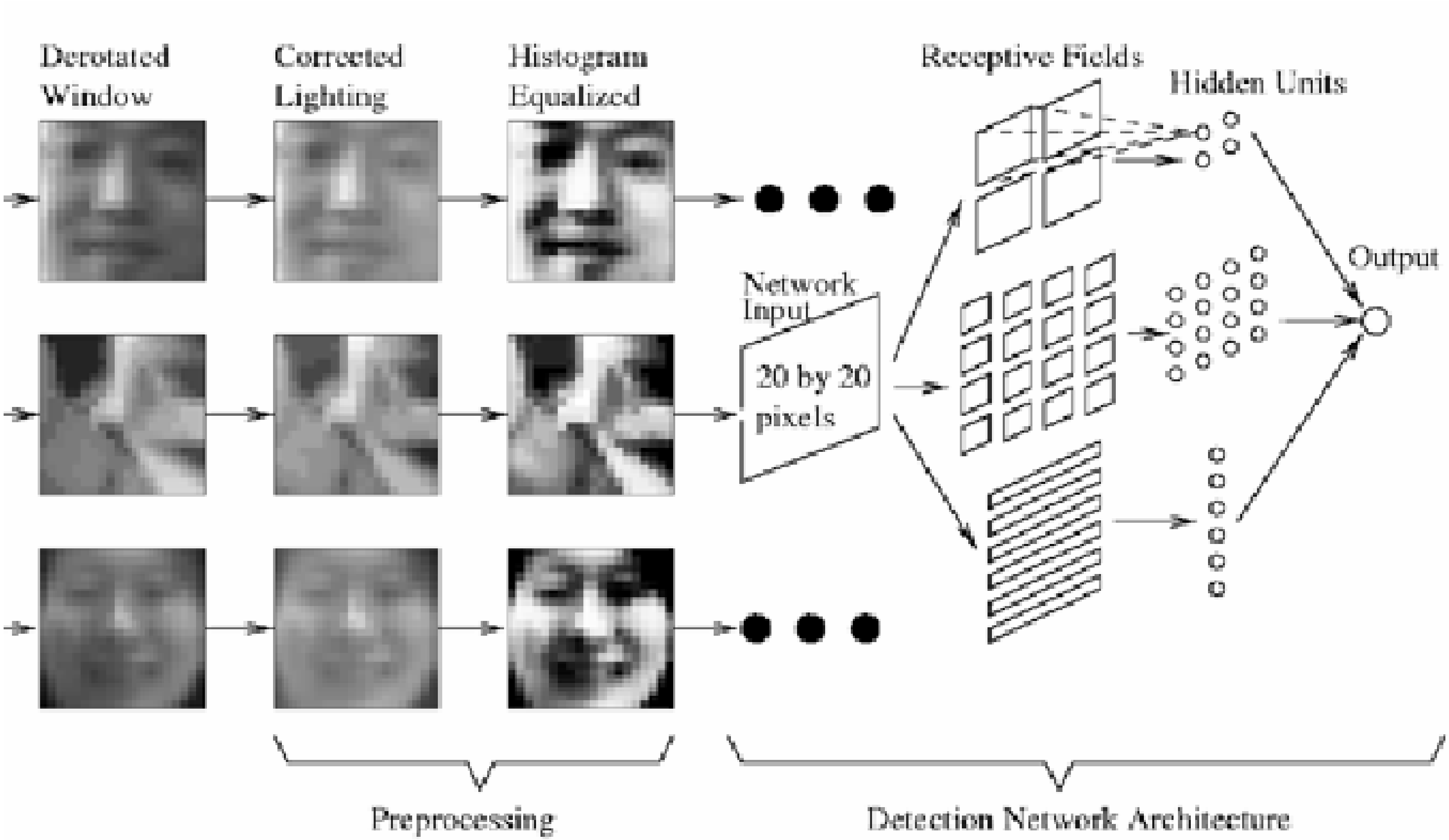
- Inputs analyze signal at multiple points in time

- RBF functions may be used to select a ‘window’ in the input data
- Invariant to translation



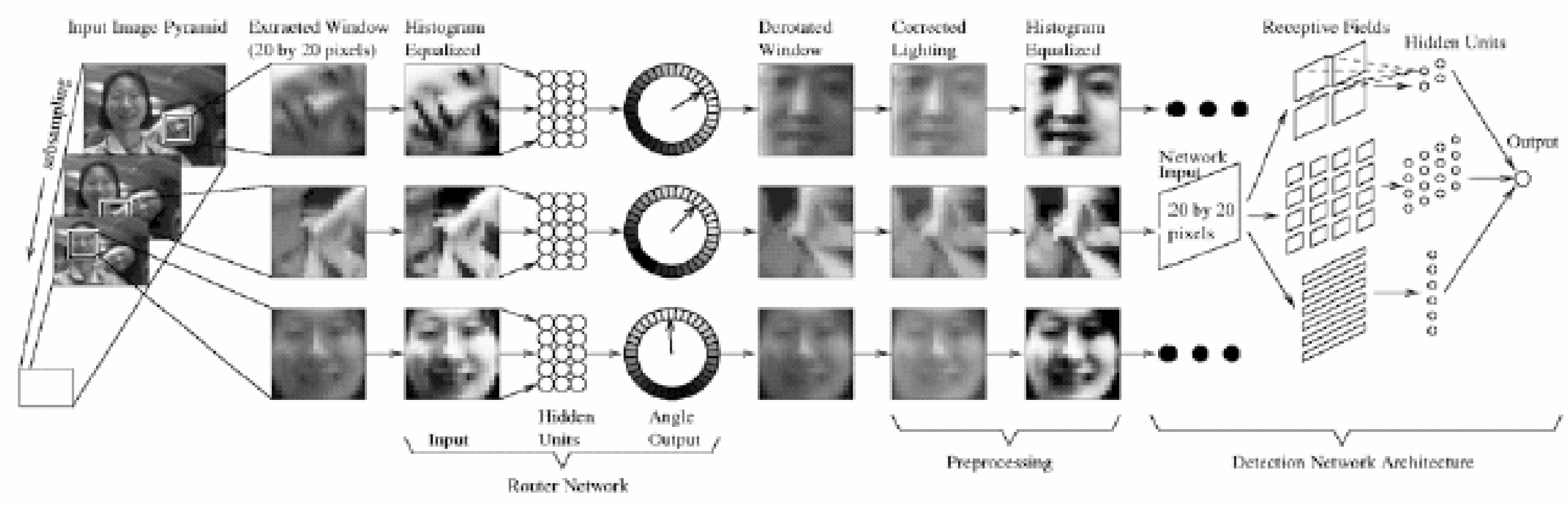
Preprocessing of image for NN

- Normalization
  - Inputs must be in (-1,1) or (0,1)
- Problem of reduction of dimensionality
- PCA
- Filtering



The vertical face-finding part of Rowley, Baluja and Kanade's system

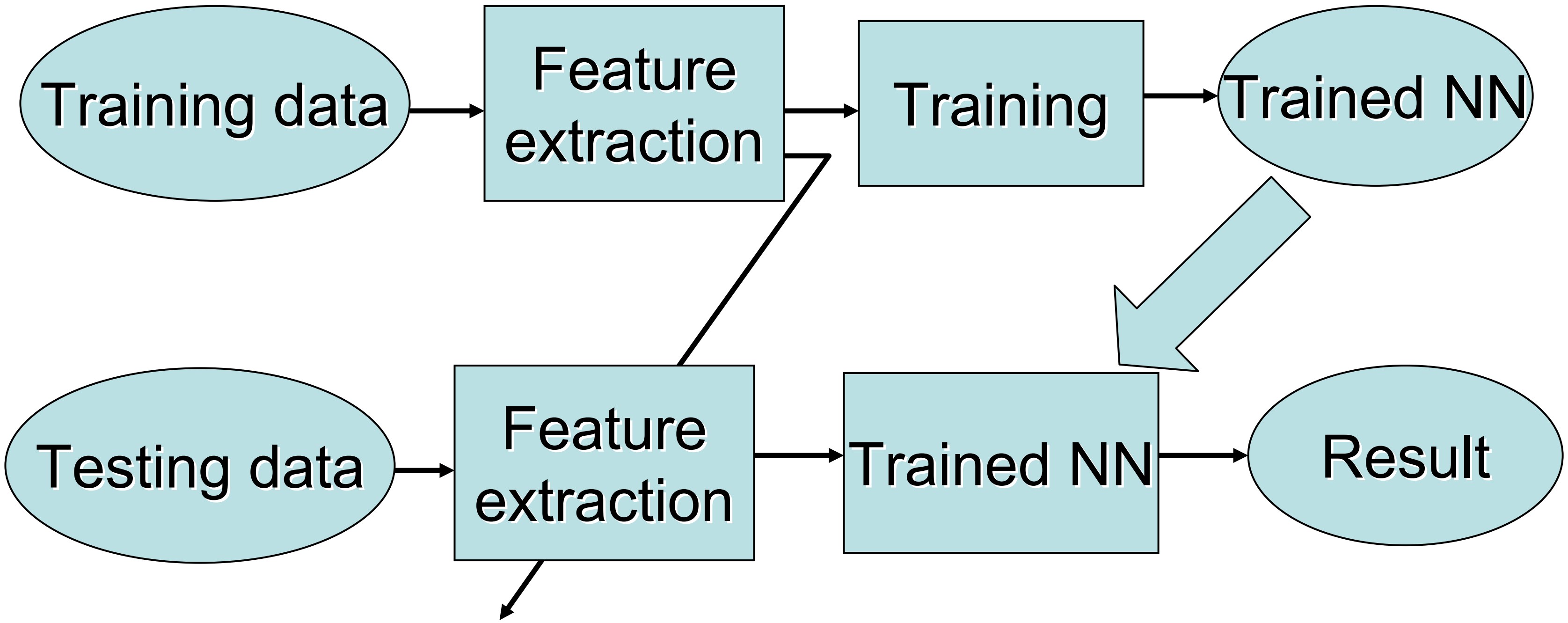
Figure from "Rotation invariant neural-network based face detection," H.A. Rowley, S. Baluja and T. Kanade, Proc. Computer Vision and Pattern Recognition, 1998, copyright 1998, IEEE



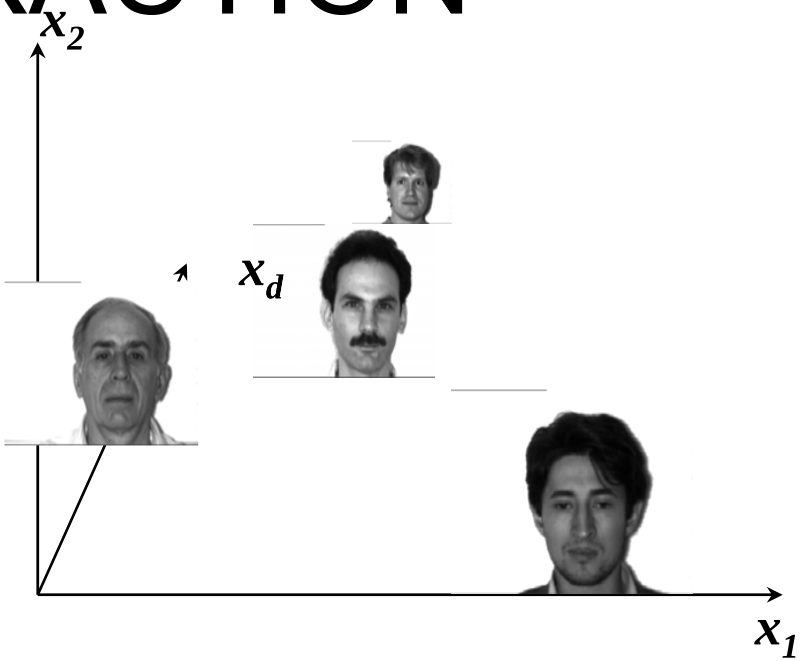
Architecture of the complete system: they use another neural net to estimate orientation of the face, then rectify it. They search over scales to find bigger/smaller faces.

Figure from "Rotation invariant neural-network based face detection," H.A. Rowley, S. Baluja and T. Kanade, Proc. Computer Vision and Pattern Recognition, 1998, copyright 1998, IEEE

Face recognition using NN system (Phan Tran Ho Truc, UClab KHU)



FEATURE EXTRACTION



- A face image 100 x 100 pixels corresponds to a point in 10000-D space.
- Similar -> near
- Different -> far

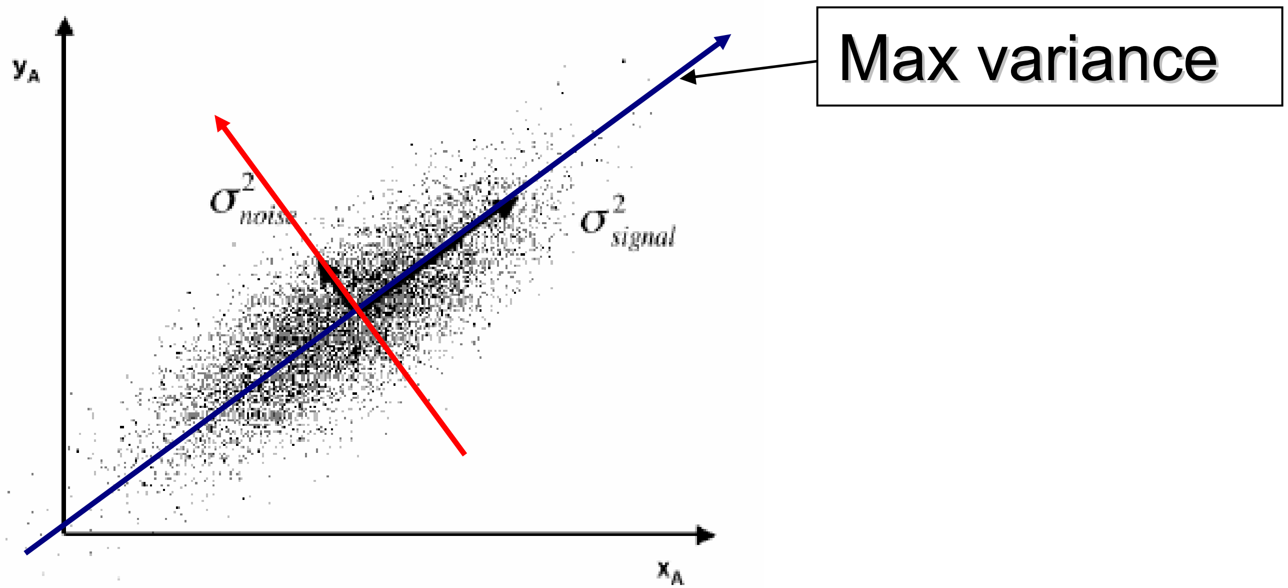
However, 10000 D => too large and redundant.

<u>Problem:</u> find out an appropriate feature space.

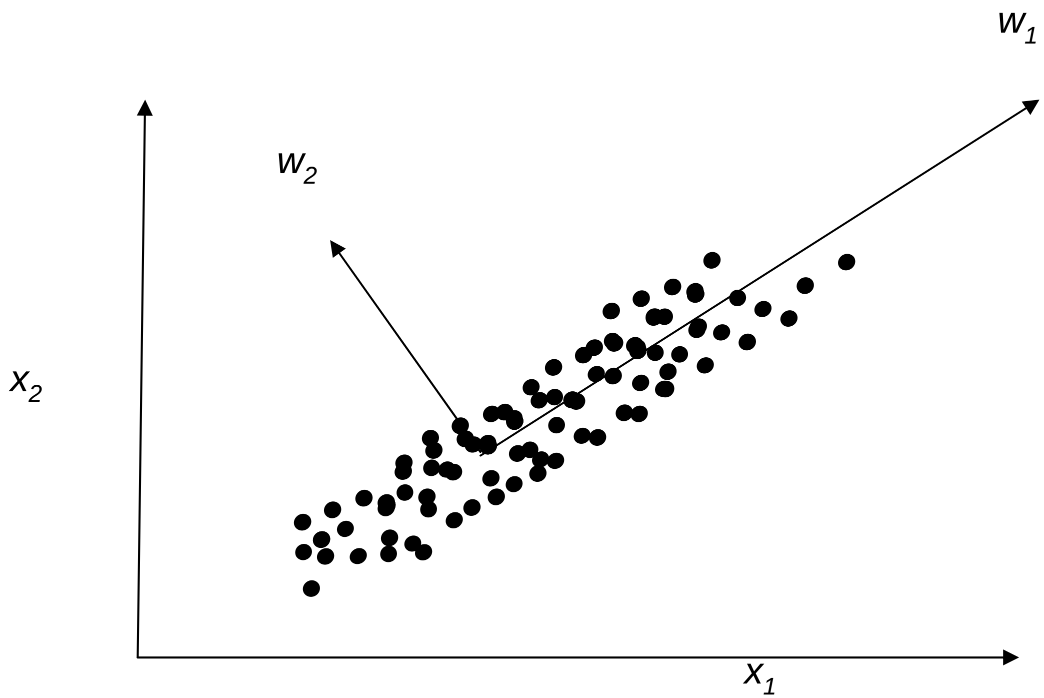
How ? → **PCA**

Principal Component Analysis (PCA)

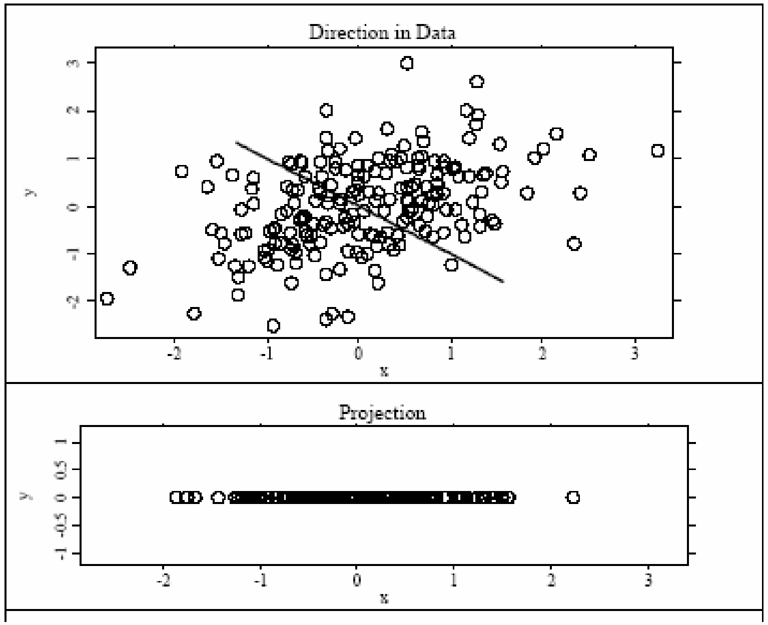
- PCA is to find a feature space in which the data have max variance

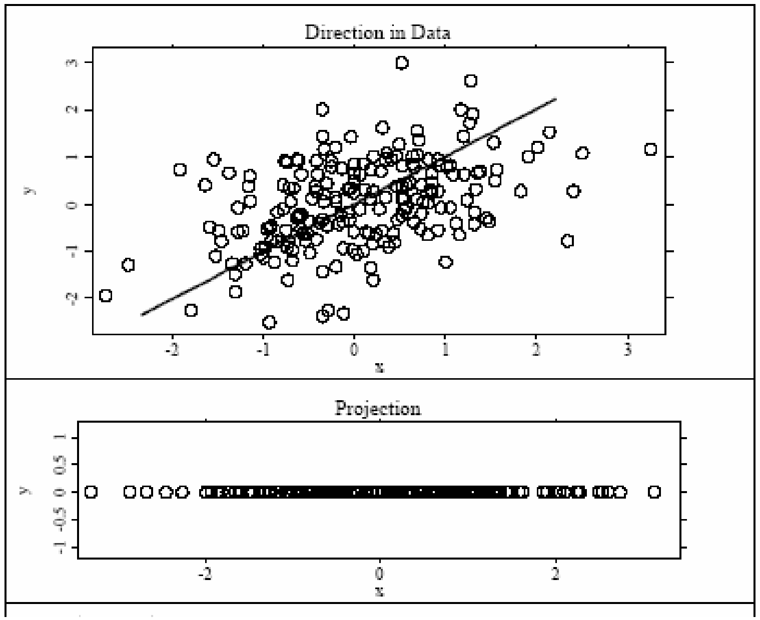


How we can find the best Principle Components



How we can find the best Principle Components (cont.)





Maximize the variance of the projection of the observations on the Y variables

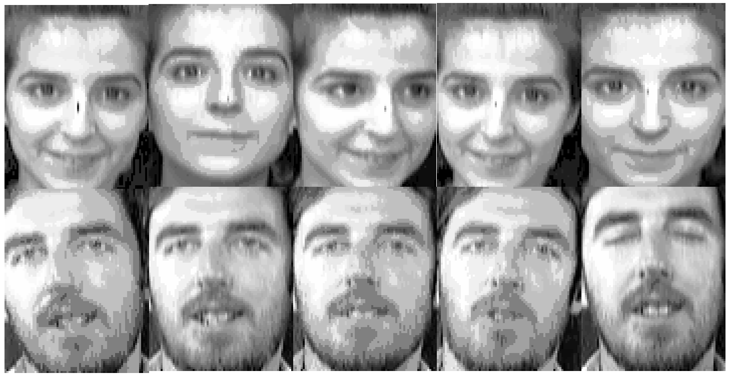
Find *w so that*

$$Var(w^T \mathbf{X}) = \mathbf{w}^T \mathbf{Var(X)}\ \mathbf{w} \quad \text{is maximal}$$

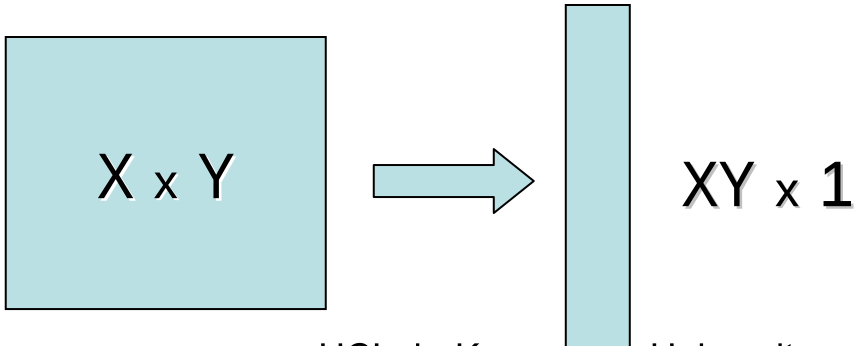
The matrix **C=Var(X)** is the covariance matrix of the *Xi* variables

Algorithm of feature extraction using PCA

- <u>Step 1:</u> collect training image set $I_1, I_2, \ldots, I_M$.



- <u>Step 2:</u> Represent image Ii as a vector Ti.



Algorithm (cont.)

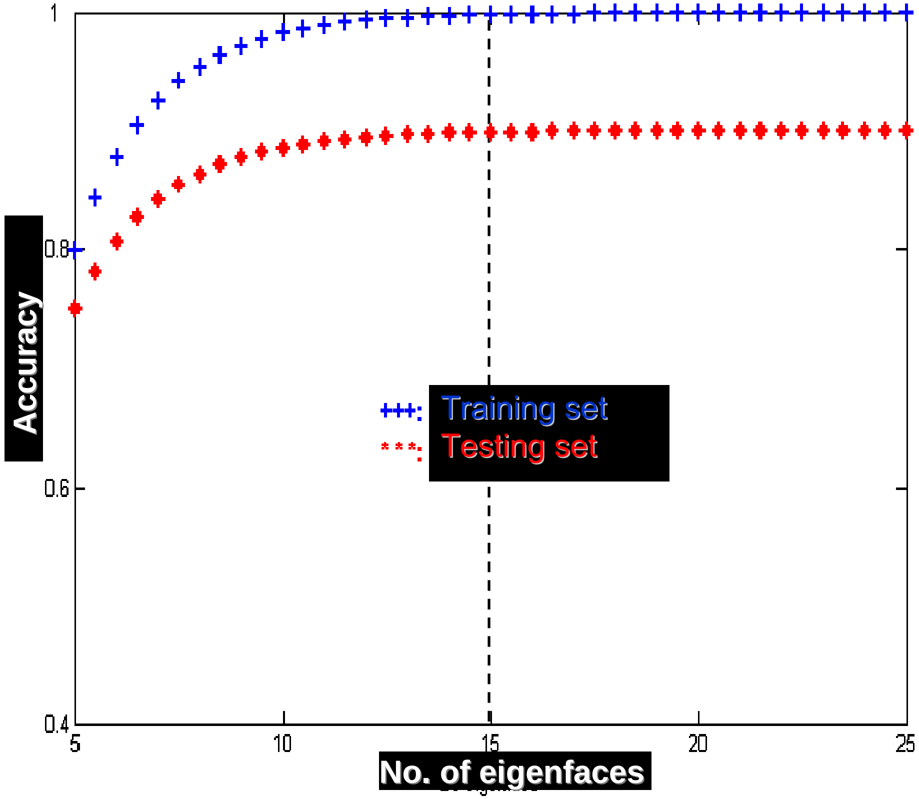
- Step 3: calculate the Mean Face Ψ
- $$\Psi = \frac{1}{M}\sum_{i=1}^{M} T_i$$
- Step 4: subtract Mean Face from each image
- $$\Phi_i = T_i - \Psi$$
- Step 5: constructing the covariance matrix C: $$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$
- where: A = [Φ_1 Φ_2 … Φ_M]

Algorithm (cont.)

- Step 6: calculate eigenvectors $\mathbf{u_i}$ of matrix C
- Step 7: select K largest eigen vectors
- Each face is a linear combination of these K eigenvectors

Φi = $\sum_{j=1}^{K} w_j u_j$ $\Rightarrow$ Inputs to the Neural Nets: $\Omega_i = \begin{bmatrix} w_1^i \\ w_2^i \\ \vdots \\ w_K^i \end{bmatrix}$

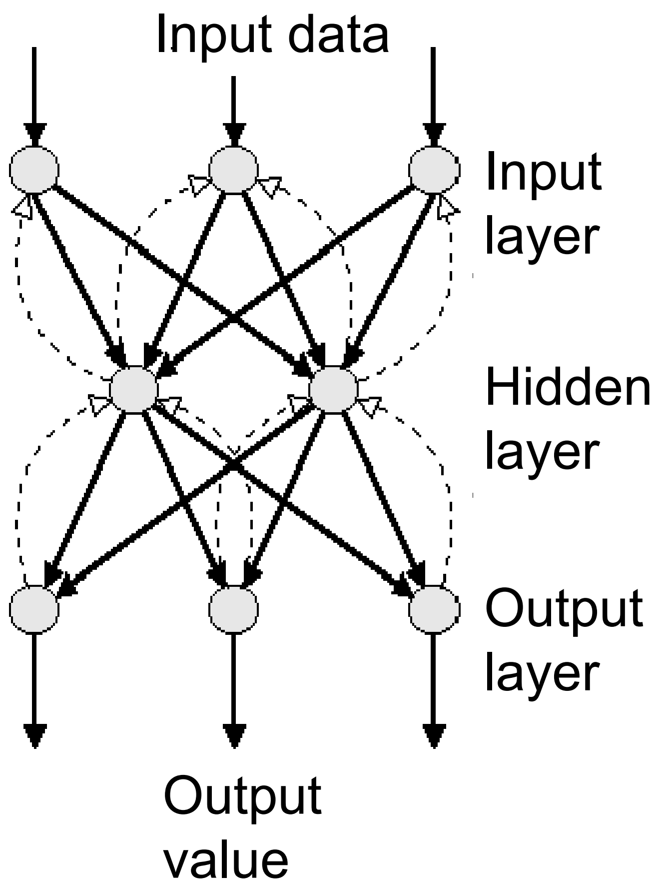
Selection of K - # dimension



- Recognition accuracy increases with number of eigenfaces till 15.
- ⇒ Later eigenfaces do not help much with recognition.

- K = 15

Neural networks



- ***Selection of parameters:***
- \# input neurons = K
- \# output neurons = # identifying people.
- <u>Output value</u>:
- 1000000000 -> 1st person;
- 0100000000 -> 2nd person, …
- With K = 15 and 10 identifying people, # hidden neurons = 20
- Learning rate is selected experimentally as 0.3

FACE DATABASE

From the Olivetti Research Laboratory (ORL), Cambridge University, UK. 400 images of 40 people with different face orientations and expressions

EXPERIMENTAL RESULTS

No. of people	Best accuracy (%)	
	Training set	Testing set
10	100	92
16	100	87
20	100	83
30	100	82

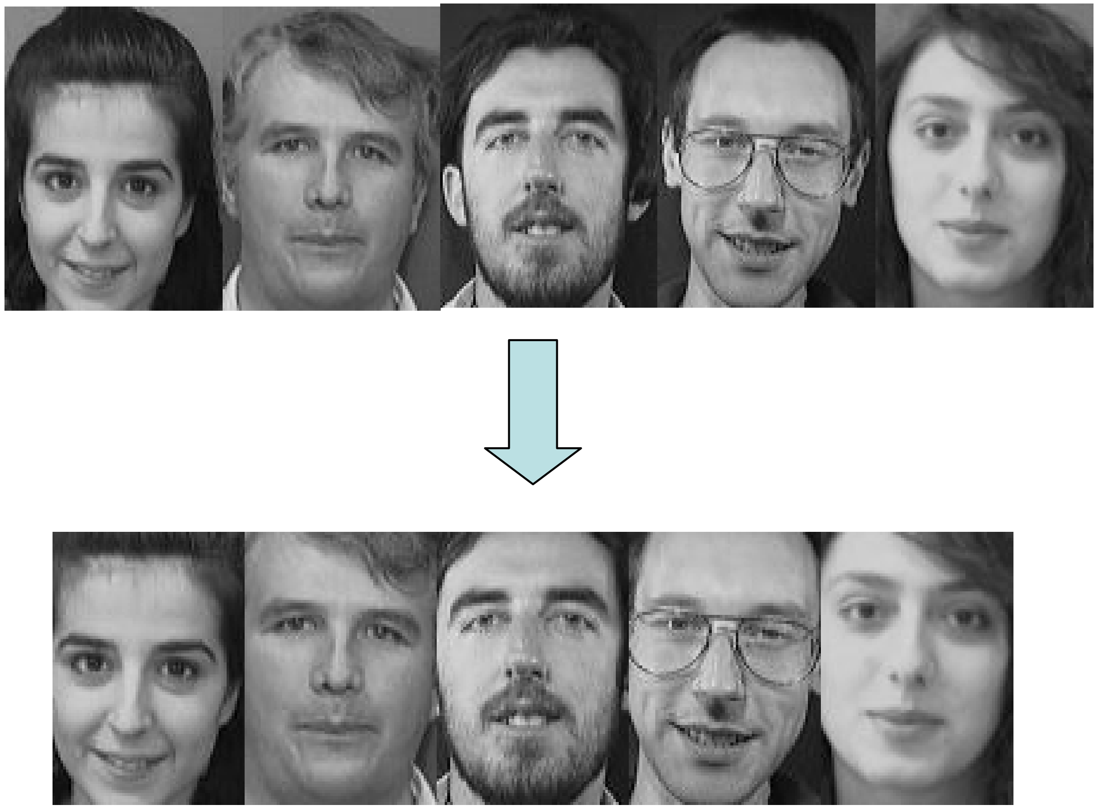
NN training time with 50 data and tolerance 10^{-4} : **3 – 4 seconds**

Recognition time: **180 – 220 ms**

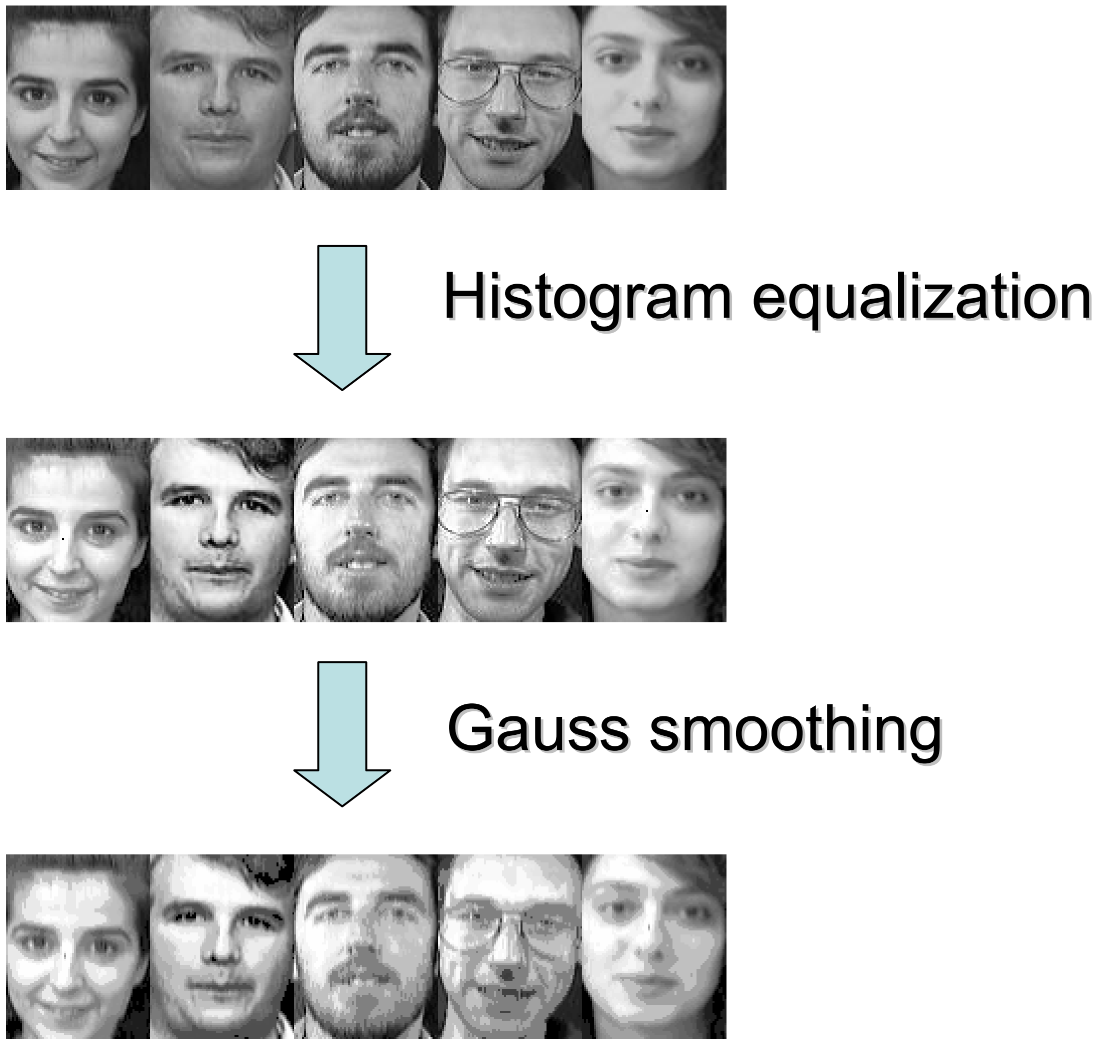
COMMENTS

- Face recognition challenges:
- - Various lighting conditions
- - Image size changes (face detection needed)
- - Appearance changes: wearing/not glasses, smiling/not, beard/not, …
- - Pose changes: straight ahead, turn left and right, …

GEOMETRY NORMALIZATION



INTENSITY NORMALIZATION



Comments (2)

- Image size and lighting changes can be partly solved by geometry and intensity normalization
- PCA is insensible to appearance changes but much sensible to noises and pose changes