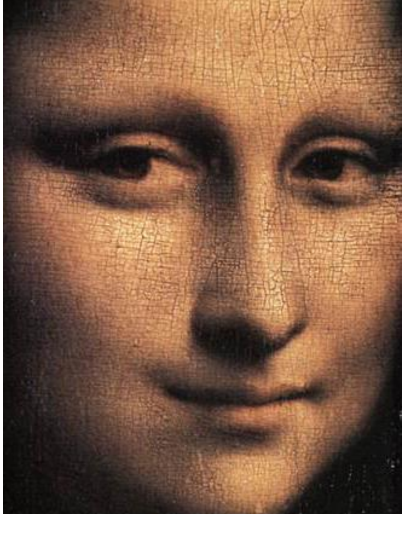
ECS797P - Machine Learning for Visual Data Analysis

Tao Xiang
t.xiang@qmul.ac.uk

Lecture 3

An instance classification problem: Face Recognition



Face Recognition

- Face is the most common biometric used by humans
- Applications range from static, mug-shot verification to a dynamic, uncontrolled face identification in a cluttered background
- Challenges:
  - automatically locate the face
  - recognize the face from a general view point under different
    - illumination conditions
    - facial expressions
    - aging effects

Authentication vs. Identification

- Face Authentication/Verification (1:1 matching)

- Face Identification/Recognition (1:N matching)

Applications

- Access Control

www.viisage.com

www.visionics.com

Applications

- Video Surveillance (On-line or off-line)

Face Scan at Airports

www.facesnap.de

Why is Face Recognition Hard?

Many faces of Madonna

Face Recognition Difficulties

- Identify similar faces (inter-class similarity)
- Accommodate intra-class variability due to:
  - head pose
  - illumination conditions
  - expressions
  - facial accessories
  - aging effects

Inter-class Similarity

- Different persons may have very similar appearance


www.marykateandashley.com

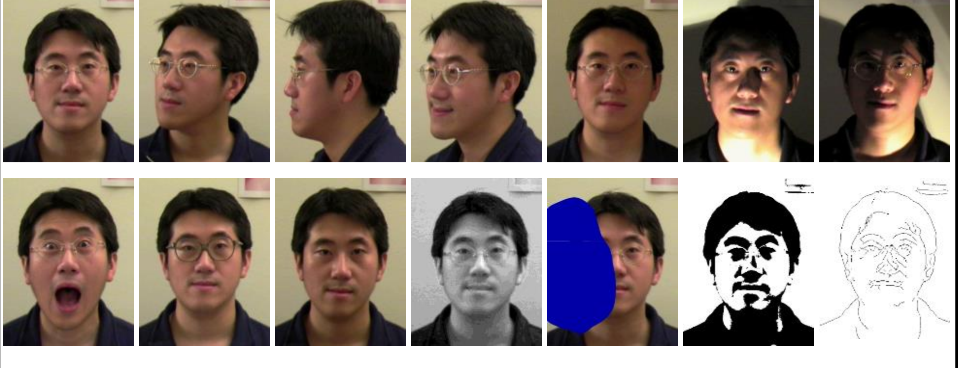

news.bbc.co.uk/hi/english/in_depth/americas/2000/us_elections

Twins

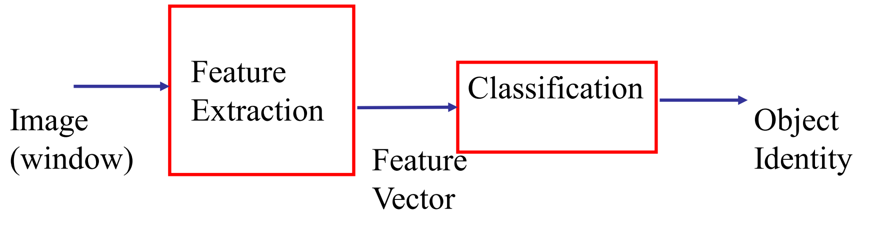
Father and son

Intra-class Variability

- Faces with intra-subject variations in pose, illumination, expression, accessories, color, occlusions, and brightness

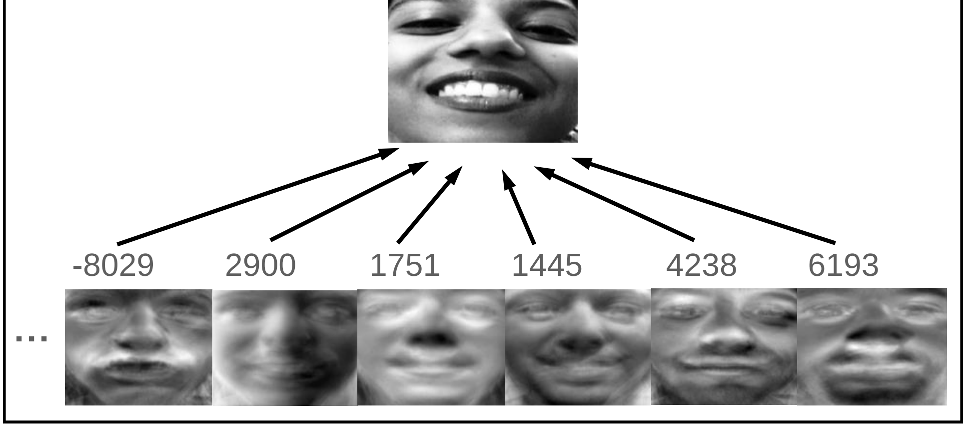


Sketch of a Pattern Recognition Architecture



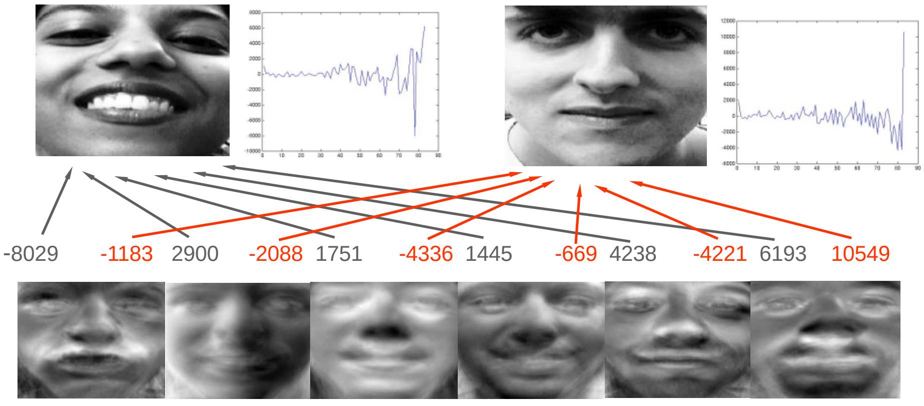
Eigenfaces: the idea

- Think of a face as being a weighted combination of some "component" or "basis" faces
- These basis faces are called eigenfaces



Eigenfaces: representing faces

- These basis faces can be differently weighted to represent any face
- So we can use different vectors of weights to represent different faces



Learning Eigenfaces

Q: How do we pick the set of basis faces?

A: We take a set of real training faces

Then we find (learn) a set of basis faces which best represent the differences between them

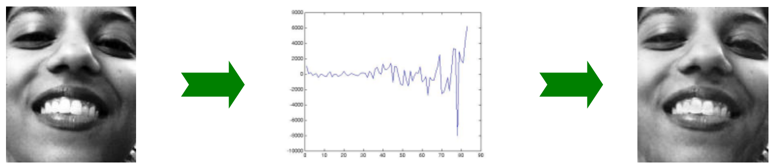
We'll use a statistical criterion for measuring this notion of "best representation of the differences between the training faces"

We can then store each face as a set of weights for those basis faces

Using Eigenfaces: recognition & reconstruction

- We can use the eigenfaces in two ways

1: we can store and then reconstruct a face from a set of weights



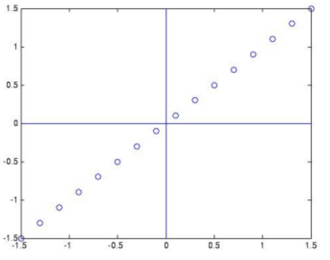
2: we can recognise a new picture of a familiar face

Learning Eigenfaces

- How do we learn them?
- We use a method called Principle Components Analysis (PCA)
- To understand this we will need to understand
  - What an eigenvector is
  - What covariance is
- But first we will look at what is happening in PCA qualitatively

Subspaces

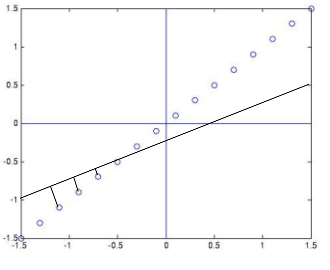
- Imagine that our face is simply a (high dimensional) vector of pixels
- We can think more easily about 2D vectors



- Here we have data in two dimensions
- But we only really need one dimension to represent it

Finding Subspaces

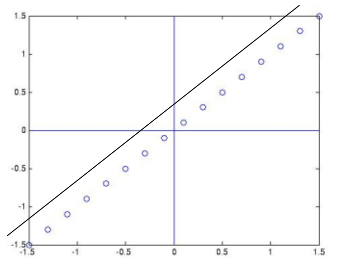
- Suppose we take a line through the space

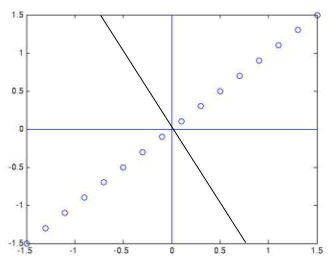


- And then take the projection of each point onto that line
- This could represent our data in “one” dimension

Finding Subspaces

- Some lines will represent the data in this way well, some badly



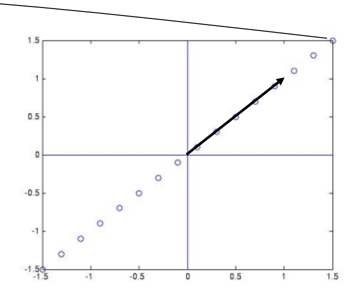


- This is because the projection onto some lines separates the data well, and the projection onto some lines separates it badly

Finding Subspaces

- Rather than a line we can perform roughly the same trick with a vector ν

$$\Phi_{16} = \frac{3}{2}\nu$$



$$\nu = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- Now we have to scale the vector to obtain any point on the line

$$\Phi_i = \mu\nu$$

Eigenvectors

- An eigenvector is a vector v that obeys the following rule:

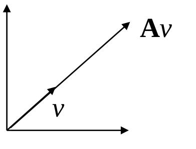
$$\mathbf{A}v = \mu v$$

Where $\mathbf{A}$ is a matrix, μ is a scalar (called the eigenvalue)

e.g. $\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$ one eigenvector of $\mathbf{A}$ is $v = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$ since

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

so for this eigenvector of this matrix the eigenvalue is 4

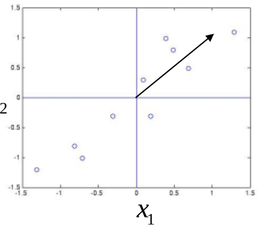


Eigenvectors

- We can think of matrices as performing transformations on vectors (e.g rotations, reflections)
- We can think of the eigenvectors of a matrix as being special vectors (for that matrix) that are scaled by that matrix
- Different matrices have different eigenvectors
- <u>Only square matrices</u> have eigenvectors
- Not all square matrices have eigenvectors
- An *n by n* matrix has <u>at most</u> *n* distinct eigenvectors
- All the distinct eigenvectors of a matrix are orthogonal (i.e., perpendicular)

Covariance

- Which single vector can be used to separate these points as much as possible?

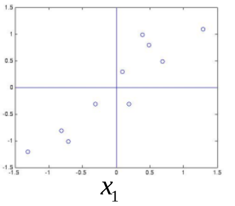


- This vector turns out to be a vector expressing the direction of the correlation
- Here I have two variables x_1 and x_2
- They co-vary (y tends to change in roughly the same direction as x)

Covariance

- The covariances can be expressed as a matrix

$$C = \begin{array}{cc} x_1 & x_2 \\ \end{array}$$
$$C = \begin{bmatrix} .617 & .615 \\ .615 & .717 \end{bmatrix} \begin{array}{c} x_1 \\ x_2 \end{array}$$

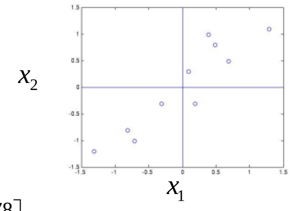


- The diagonal elements are the variances e.g. Var(x_1)
- The covariance of two variables is:

$$\operatorname{cov}(x_1, x_2) = \frac{\sum_{i=1}^{n} (x_1^i - \overline{x}_1)(x_2^i - \overline{x}_2)}{n-1}$$

Eigenvectors of the covariance matrix

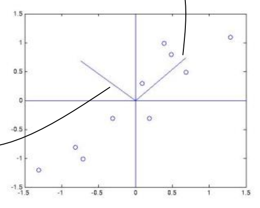
- The covariance matrix has eigenvectors



covariance matrix $C = \begin{bmatrix} .617 & .615 \\ .615 & .717 \end{bmatrix}$

eigenvectors $v_1 = \begin{bmatrix} -.735 \\ .678 \end{bmatrix}$ $v_2 = \begin{bmatrix} .678 \\ .735 \end{bmatrix}$

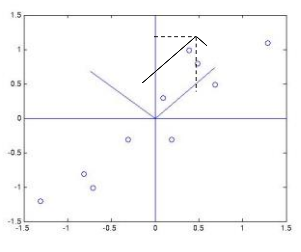
eigenvalues $\mu_1 = 0.049$ $\mu_2 = 1.284$



- Eigenvectors with larger eigenvectors correspond to directions in which the data varies more
- Finding the eigenvectors and eigenvalues of the covariance matrix for a set of data is termed principle components analysis

Expressing points using eigenvectors

- Suppose you think of your eigenvectors as specifying a new vector space



- i.e. I can reference any point in terms of those eigenvectors
- A point's position in this new coordinate system is what we earlier referred to as its "weight vector"
- For many data sets you can cope with fewer dimensions in the new space than in the old space

Eigenfaces

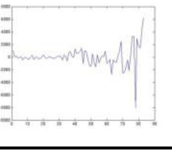
- All we are doing in the face case is treating the face as a point in a high-dimensional space, and then treating the training set of face pictures as our set of points
- To train:
  - We calculate the covariance matrix of the faces
  - We then find the eigenvectors of that covariance matrix
- These eigenvectors are the eigenfaces or basis faces
- Eigenfaces with bigger eigenvalues will explain more of the variation in the set of faces, i.e. will be more distinguishing

Eigenfaces: image space to face space

- When we see an image of a face we can transform it to face space

$$\mathbf{w}_k = \mathbf{x}^{\mathbf{i}} . v_k$$

- There are k=1…n eigenfaces v_k
- The i^{th} face in image space is a vector $\mathbf{x}^{\mathbf{i}}$
- The corresponding weight is $\mathbf{w}_k$
- We calculate the corresponding weight for every eigenface

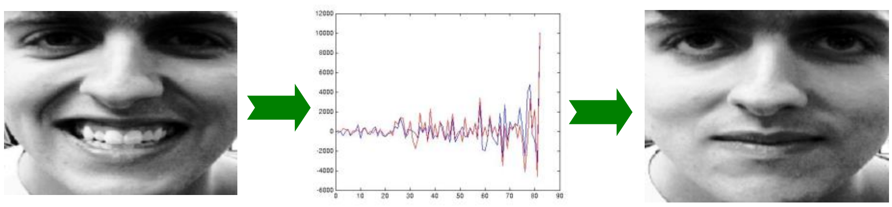


Recognition in face space

- Recognition is now simple. We find the euclidean distance ***d*** between our face and all the other stored faces in face space:

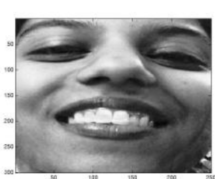
$$d(w^1, w^2) = \sqrt{\sum_{i=1}^{n} \left(w_i^1 - w_i^2 \right)^2}$$

- The closest face in face space is the chosen match

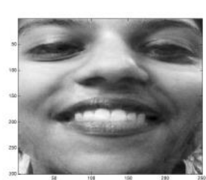


Reconstruction

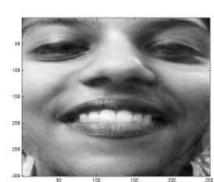
- The more eigenfaces you have the better the reconstruction, but you can have high quality reconstruction even with a small number of eigenfaces



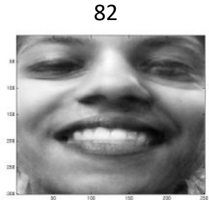
82



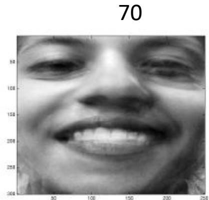
70



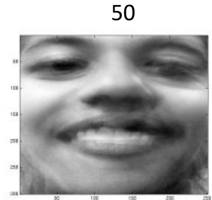
50



30



20

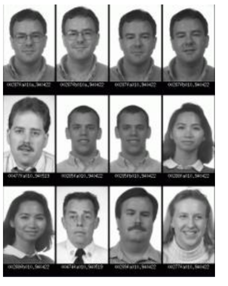


10

Image Representation

- Training set of m images of size $N*N$ are represented by vectors of size N^2

  $x_1,x_2,x_3,\dots,x_m$



<u>Example</u>

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 4 & 5 & 1 \end{bmatrix}_{3\times 3} \longrightarrow \begin{bmatrix} 1 \\ 2 \\ 3 \\ 3 \\ 1 \\ 2 \\ 4 \\ 5 \\ 1 \end{bmatrix}_{9\times 1}$$

Average Image and Difference Images

- The average training set is defined by

  $\mu = (1/m) \sum_{i=1}^{m} x_i$

- Each face differs from the average by vector

  $r_i = x_i - \mu$

Covariance Matrix

- The covariance matrix is constructed as

  $C = AA^T$ where $A=[r_1,\ldots,r_m]$

  Size of this matrix is N^2 x N^2

- Finding eigenvectors of N^2 x N^2 matrix is intractable. Hence, use the matrix A^TA of size m x m and find eigenvectors of this small matrix.

Eigenvalues and Eigenvectors - Definition

- If v is a nonzero vector and λ is a number such that

  Av = λv, then

  v is said to be an *eigenvector* of A with *eigenvalue* λ.

<u>Example</u>

λ (eigenvalues)

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 3 \times \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

A V (eigenvectors)

Eigenvectors of Covariance Matrix

- The eigenvectors v_i of A^TA are:

$$\underset{v_1}{\begin{bmatrix} 0.3923 \\ 0.5060 \\ 0.7681 \end{bmatrix}} \quad \underset{v_2}{\begin{bmatrix} 0.9087 \\ -0.0835 \\ -0.4091 \end{bmatrix}} \quad \underset{v_3}{\begin{bmatrix} 0.1429 \\ -0.8585 \\ 0.4926 \end{bmatrix}}$$

- Consider the eigenvectors v_i of A^TA such that

$$A^TAv_i = \mu_i v_i$$

- Premultiplying both sides by A, we have

$$AA^T(Av_i) = \mu_i(Av_i)$$

Face Space

- The eigenvectors of covariance matrix are

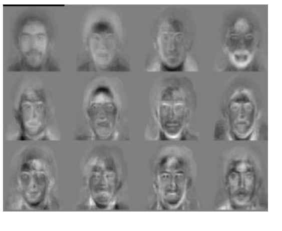
$$u_i = Av_i$$

$$u_i = A\,v_i$$

$$\underset{u_1}{\begin{bmatrix} -0.2621 \\ -0.2621 \\ -0.6527 \\ -0.1137 \\ -0.5589 \\ 0.6015 \\ -0.4895 \\ 0 \\ 0.6379 \end{bmatrix}} \quad \underset{u_2}{\begin{bmatrix} 0.3256 \\ 0.3256 \\ -3.3773 \\ 0.9922 \\ -1.0076 \\ -0.5080 \\ 2.3100 \\ 0 \\ -1.6434 \end{bmatrix}} \quad \underset{u_3}{\begin{bmatrix} -1.3511 \\ -1.3511 \\ 2.2735 \\ 1.0014 \\ -6.0561 \\ -5.4206 \\ 0.6517 \\ 0 \\ 1.7008 \end{bmatrix}}$$

Face Space

$$U = \begin{array}{c} \begin{bmatrix} -0.2621 & 0.3256 & -1.3511 \\ -0.2621 & 0.3256 & -1.3511 \\ -0.6527 & -3.3773 & 2.2735 \\ -0.1137 & 0.9922 & 1.0014 \\ -0.5589 & -1.0076 & -6.0561 \\ 0.6015 & -0.5080 & -5.4206 \\ -0.4895 & 2.3100 & 0.6517 \\ 0 & 0 & 0 \\ 0.6379 & -1.6434 & 1.7008 \end{bmatrix} \\ \begin{matrix} u_1 & \quad u_2 & \quad u_3 \end{matrix} \end{array}$$



- u_i resemble facial images which look ghostly, hence called Eigenfaces

Projection into Face Space

- A face image can be projected into this face space by

$$p_k = U^T(x_k - \mu) \quad \text{where } k=1,\ldots,m$$

Recognition

- The test image x is projected into the face space to obtain a vector p:

$$p = U^T(x - \mu)$$

- The distance of p to each face class is defined by

$$\epsilon_k^2 = ||p-p_k||^2; k = 1,\ldots,m$$

- A distance threshold Θ_c, is half the largest distance between any two face images:

$$\Theta_c = ½ \max_{j,k} \{||p_j-p_k||\}; j,k = 1,\ldots,m$$

Recognition

- Find the distance ϵ between the original image x and its reconstructed image from the eigenface space, x_f,

  $\epsilon^2 = || x - x_f ||^2$, where $x_f = U * p + \mu$

- Recognition process:
  - IF $\epsilon \geq \Theta_c$
    then input image is not a face image;
  - IF $\epsilon < \Theta_c$ AND $\epsilon_k \geq \Theta_c$ for all k
    then input image contains an unknown face;
  - IF $\epsilon < \Theta_c$ AND $\epsilon_k{*} = \min_k\{ \epsilon_k \} < \Theta_c$
    then input image contains the face of individual k*

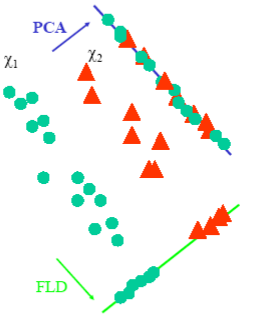
Limitations of Eigenfaces Approach

- **Variations in lighting conditions**
  - Different lighting conditions for enrolment and query.
  - Bright light causing image saturation.

- **Differences in pose – Head orientation**
  - 2D feature distances appear to distort.
- **Expression**
  - Change in feature location and shape.

Linear Discriminant Analysis

- PCA does not use class information
  - PCA projections are optimal for reconstruction from a low dimensional basis, they may not be optimal from a discrimination standpoint.
- LDA is an enhancement to PCA
  - Constructs a discriminant subspace that minimizes the scatter between images of same class and maximizes the scatter between different class images



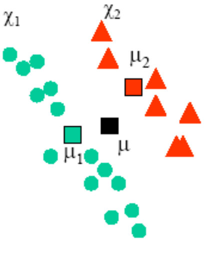
Mean Images

- Let X_1, X_2,…, X_c be the face classes in the database and let each face class X_i, i = 1,2,…,c has k facial images x_j, j=1,2,…,k.
- We compute the mean image μ_i of each class X_i as:

$$\mu_i = \frac{1}{k}\sum_{j=1}^{k} x_j$$

- Now, the mean image μ of all the classes in the database can be calculated as:

$$\mu = \frac{1}{c}\sum_{i=1}^{c} \mu_i$$



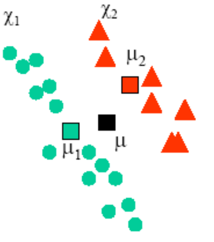
Scatter Matrices

- We calculate within-class scatter matrix as:

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- We calculate the between-class scatter matrix as:

$$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$



Multiple Discriminant Analysis

We find the projection directions as the matrix W that maximizes

$$\hat{W} = \operatorname{argmax} J(W) = \frac{|W^T S_B W|}{|W^T S_W W|}$$

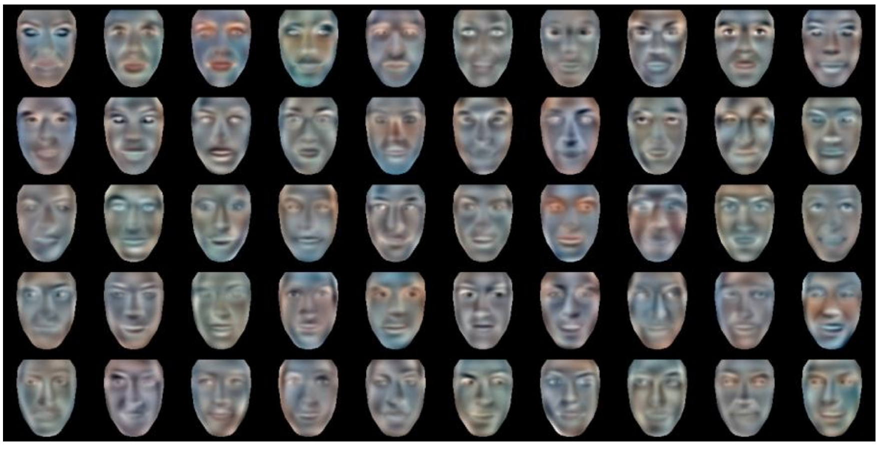
This is a generalized Eigenvalue problem where the columns of W are given by the vectors w_i that solve

$$S_B w_i = \lambda_i S_W w_i$$

Fisherface Projection

- We find the product of S_W^{-1} and S_B and then compute the Eigenvectors of this product ($S_W^{-1} S_B$) - AFTER REDUCING THE DIMENSION OF THE FEATURE SPACE.
- Use same technique as Eigenfaces approach to reduce the dimensionality of scatter matrix to compute eigenvectors.
- Form a matrix W that represents all eigenvectors of $S_W^{-1} S_B$ by placing each eigenvector w_i as a column in W.
- Each face image $x_j \in X_i$ can be projected into this face space by the operation

$$p_i = W^T(x_j - \mu)$$



Testing

- Same as Eigenfaces Approach

References

- Turk, M., Pentland, A.: *Eigenfaces for recognition*. J. Cognitive Neuroscience **3** (1991) 71–86.
- Belhumeur, P.,Hespanha, J., Kriegman, D.: *Eigenfaces vs. Fisherfaces: recognition using class specific linear projection*. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997) 711–720.

State of the art

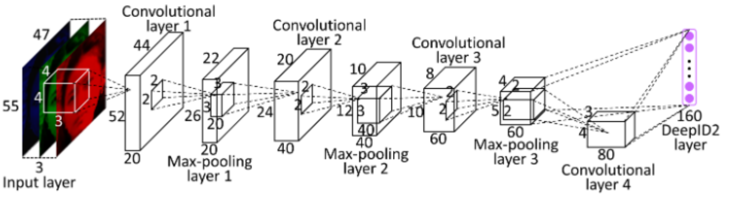


Figure 1: The ConvNet structure for DeepID2 extraction.

- Prof. X. Tang's group: DeepID2
  http://arxiv.org/pdf/1406.4773v1.pdf
- Updated version: DeepID3
  http://arxiv.org/pdf/1502.00873.pdf