COLLEGE OF ENGINEERING
ELECTRICAL ENGINEERING & COMPUTER SCIENCE
UNIVERSITY OF MICHIGAN

EigenFaces—Learn a Linear Basis

Images as Points Case Study

EECS 442 Computer Vision

Instructor: Jason Corso (jjcorso)

Starting Example: https://www.youtube.com/watch?v=wr4rx0Spihs

Materials on these slides have come from many sources in addition to myself; I am infinitely grateful to these, especially Derek Hoiem, Lana Lazebnik, Silvio Savarese and Fei-Fei Li.

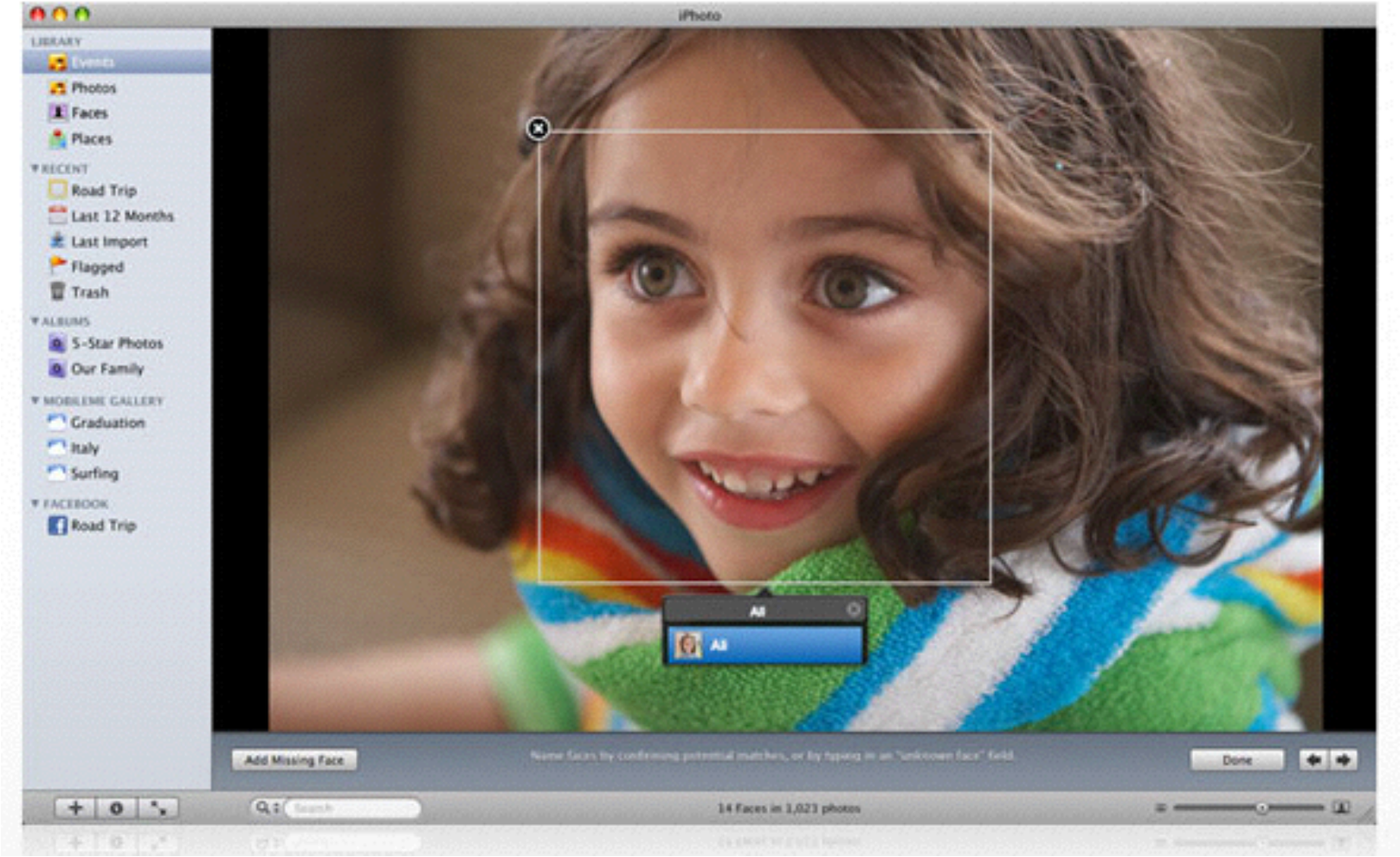
Applications of Face Recognition

- Surveillance

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Applications of Face Recognition

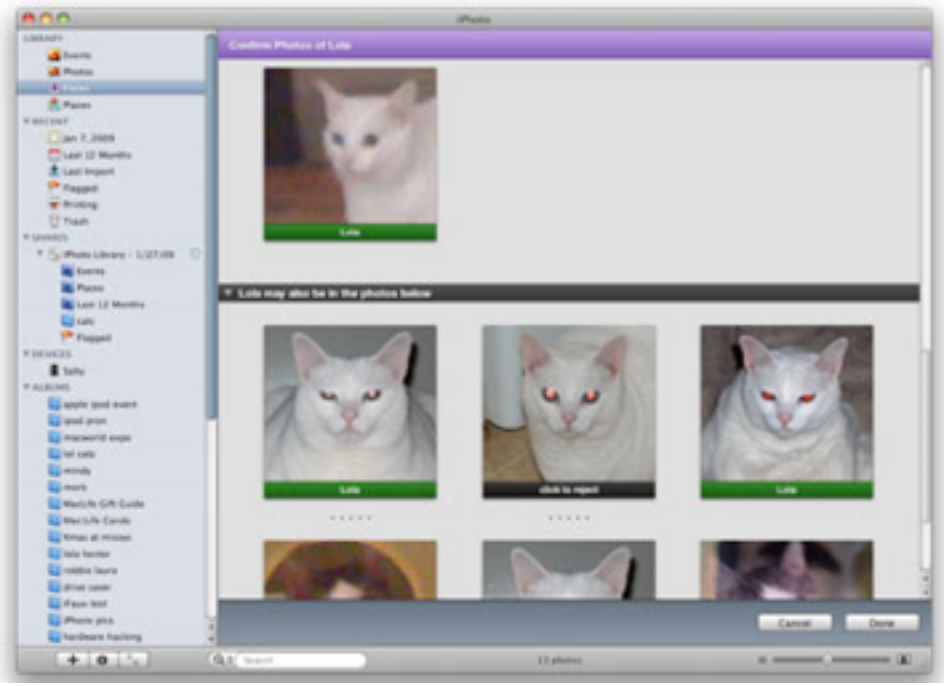
- Album organization: iPhoto 2009



http://www.apple.com/ilife/iphoto/

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

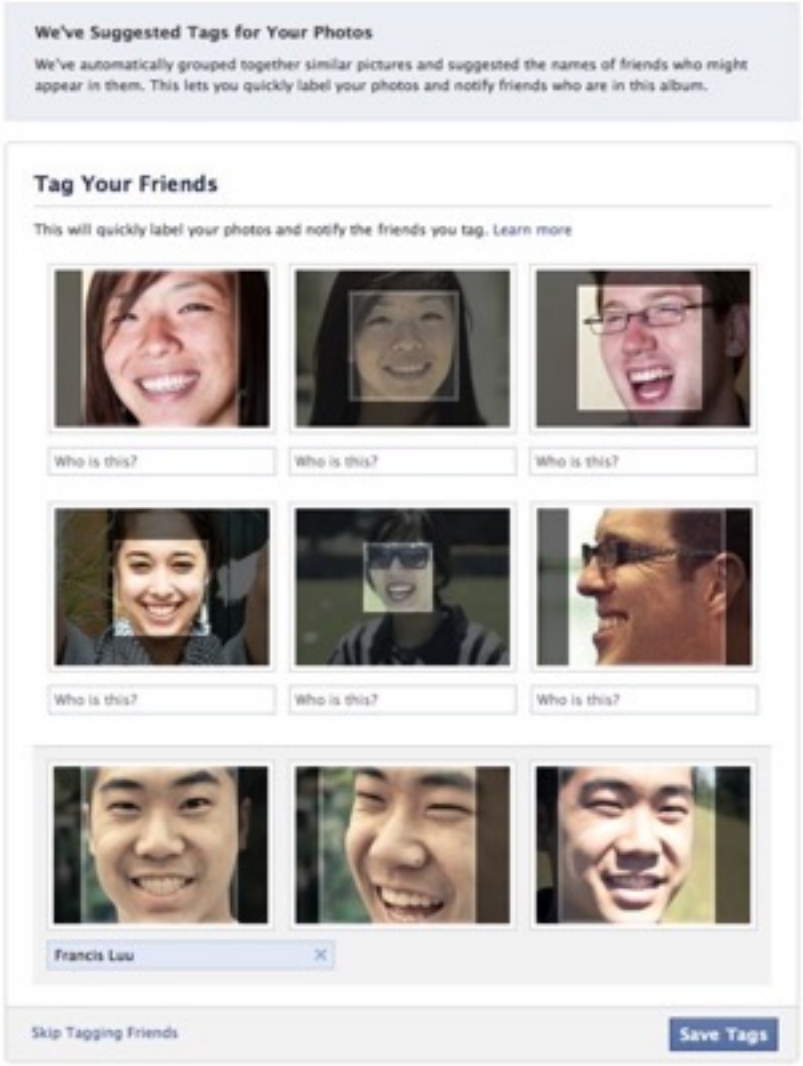
- Can be trained to recognize pets!



http://www.maclife.com/article/news/iphotos_faces_recognizes_cats

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Facebook friend-tagging with auto-suggest



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

YouTube Faces Dataset (YTF)

same **different**

- Video Face Verification: Given a pair of videos specify whether they belong to the same person
- 3425 videos, 1595 people
- Standard benchmark in the community
- Wide pose, expression and illumination variation

[Wolf, Hassner, Moaz CVPR 2011]

YouTube Faces Dataset (YTF)

same **different**

- Video Face Verification: Given a pair of videos specify whether they belong to the same person
- 3425 videos, 1595 people
- Standard benchmark in the community
- Wide pose, expression and illumination variation

[Wolf, Hassner, Moaz CVPR 2011]

Labeled Faces In the Wild Dataset (LFW)

- Face Verification: Given a pair of images specify whether they belong to the same person
- 13K images, 5.7K people
- Standard benchmark in the community
- Several test protocols depending upon availability of training data within and outside the dataset.

[G. Huang, M. Ramesh, T. Berg and E. Learned-Miller - Tech Report 0

Dataset Collection

Dataset Comparison

No.	Aim	# Persons	Total # images
1	Labeled Faces In the Wild	5,749	13,233
2	WDRef	2995	99,773
3	Celeb Faces	10177	202,599
4	Oxford	2622	1,635,159
5	Facebook	4030	4.4M
6	Google	8M	200M

From Parkhi, Vedaldi and Zisserman.

Oxford Buffy Dataset

Weakly supervised face classification

- **“Buffy The Vampire Slayer”**
  - Face tracks from 7 episodes of season 5.
  - Both frontal and profile detections
  - Weak supervision from transcript and subtitles
  - Multi Class classification for every episode

[Everingham et al. IVC 2009, Sivic et al. CVPR 2009]

Oxford Buffy Dataset

Weakly supervised face classification

- **“Buffy The Vampire Slayer”**
  - Face tracks from 7 episodes of season 5.
  - Both frontal and profile detections
  - Weak supervision from transcript and subtitles
  - Multi Class classification for every episode

[Everingham et al. IVC 2009, Sivic et al. CVPR 2009]

Face recognition: once you've detected and cropped a face, try to recognize it

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: once you've detected and cropped a face, try to recognize it

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: once you've detected and cropped a face, try to recognize it

Detection

Recognition

"Sally"

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example

Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example
- What's hard: changes in expression, lighting, age, occlusion, **viewpoint**

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example
- What's hard: changes in expression, lighting, age, occlusion, **viewpoint**
- Basic approaches (all nearest neighbor)
  1. Project into a new subspace (or kernel space) (e.g., "Eigenfaces"=PCA)

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example
- What's hard: changes in expression, lighting, age, occlusion, **viewpoint**
- Basic approaches (all nearest neighbor)
  1. Project into a new subspace (or kernel space) (e.g., "Eigenfaces"=PCA)
  2. Measure face features

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example
- What's hard: changes in expression, lighting, age, occlusion, **viewpoint**
- Basic approaches (all nearest neighbor)
  1. Project into a new subspace (or kernel space) (e.g., "Eigenfaces"=PCA)
  2. Measure face features
  3. Make 3d face model, compare shape+appearance (e.g., AAM)

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Typical face recognition scenarios

- Verification: a person is claiming a particular identity; verify whether that is true
  - E.g., security
- Closed-world identification: assign a face to one person from among a known set
- General identification: assign a face to a known person or to "unknown"

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

What makes face recognition hard?

Expression

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

What makes face recognition hard?

Lighting

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

What makes face recognition hard?

Occlusion

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

What makes face recognition hard?

Viewpoint

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Starting idea of “eigenfaces”

$\mathbf{x}$

$\mathbf{y_1} \cdots \mathbf{y_n}$

$$k = \text{argmin}_k \|\mathbf{y}_k^T - \mathbf{x}\|_2^2$$

From Hoiem’s Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Starting idea of "eigenfaces"

(Any NN Recognition)

1. Treat pixels as a vector

$\mathbf{x}$

$x \in \mathbb{R}^d$

$y_i \in \mathbb{R}^d \; \forall i$

2. Recognize face by nearest neighbor

$\mathbf{y_1} \cdots \mathbf{y_n}$

$$k = \text{argmin}_k \|\mathbf{y}_k^T - \mathbf{x}^T\|_2^2$$

$k = 1 \cdots n$

k nearest-neighbor

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

The space of all face images

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

The space of all face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

The space of all face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage
- But very few 10,000-dimensional vectors are valid face images

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

The space of all face images

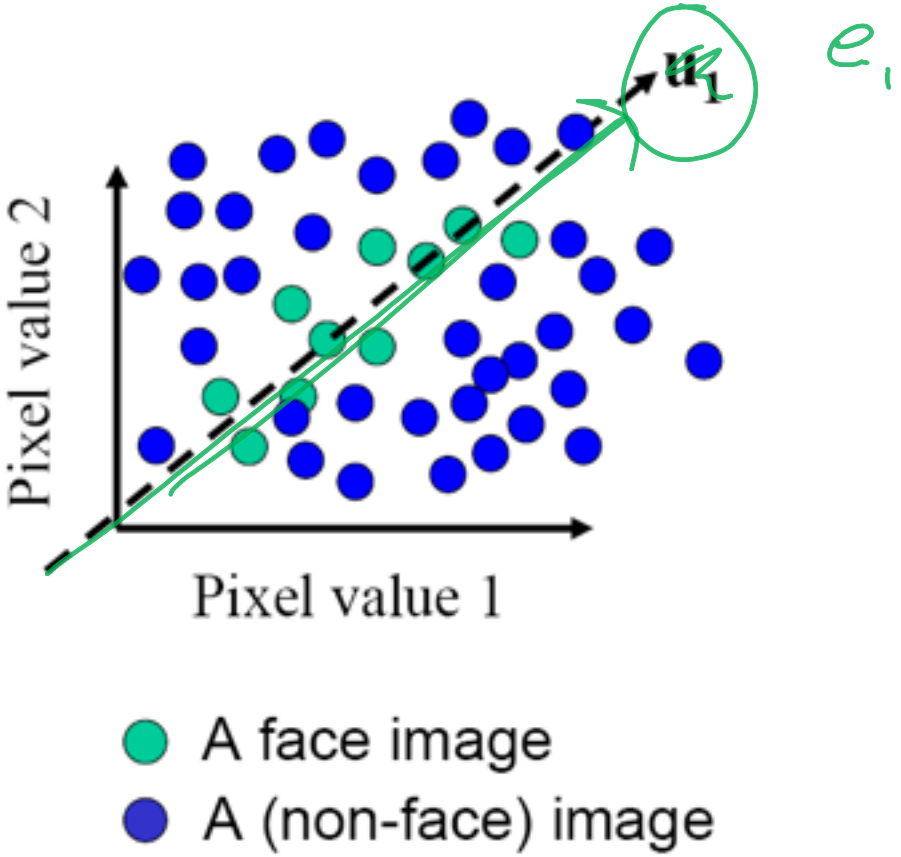
- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage
- But very few 10,000-dimensional vectors are valid face images
- We want to effectively model the subspace of face images

$2^{7^{10,800}}$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

The space of all face images

- Eigenface idea: construct a low-dimensional linear subspace that best explains the variation in the set of face images



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Implementation issue

- Covariance matrix is huge (N^2 for N pixels)

$\mathbf{X}$

$\mathbf{e}$ $\mathbf{X}\mathbf{X}^{\mathbf{T}}$ $\mathbf{X}^{\mathbf{T}}\mathbf{X}$

$\mathbf{X}^{\mathbf{T}}\mathbf{e}$ $\mathbf{X}^{\mathbf{T}}\mathbf{X}$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Implementation issue

- Covariance matrix is huge (N^2 for N pixels)
- But typically # examples << N

$$\mathbf{X}$$

$$\mathbf{e} \quad \mathbf{X}\mathbf{X}^{\mathbf{T}} \qquad \mathbf{X}^{\mathbf{T}}\mathbf{X}$$

$$\mathbf{X}^{\mathbf{T}}\mathbf{e} \qquad \mathbf{X}^{\mathbf{T}}\mathbf{X}$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Implementation issue

- Covariance matrix is huge (N^2 for N pixels)
- But typically # examples << N
- Simple trick

$\mathbf{X}$

$\mathbf{e}$ $\mathbf{X}\mathbf{X}^{\mathbf{T}}$ $\mathbf{X}^{\mathbf{T}}\mathbf{X}$

$\mathbf{X}^{\mathbf{T}}\mathbf{e}$ $\mathbf{X}^{\mathbf{T}}\mathbf{X}$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Implementation issue

- Covariance matrix is huge (N^2 for N pixels)
- But typically # examples << N
- Simple trick
  - $\mathbf{X}$ is matrix of normalized training data
  - Solve for eigenvectors $\mathbf{e}$ of $\mathbf{X}\mathbf{X}^\mathbf{T}$ instead of $\mathbf{X}^\mathbf{T}\mathbf{X}$
  - Then $\mathbf{X}^\mathbf{T}\mathbf{e}$ is eigenvector of covariance $\mathbf{X}^\mathbf{T}\mathbf{X}$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Implementation issue

- Covariance matrix is huge (N^2 for N pixels)
- But typically # examples << N
- Simple trick
  - $\mathbf{X}$ is matrix of normalized training data
  - Solve for eigenvectors $\mathbf{e}$ of $\mathbf{X}\mathbf{X}^\mathbf{T}$ instead of $\mathbf{X}^\mathbf{T}\mathbf{X}$
  - Then $\mathbf{X}^\mathbf{T}\mathbf{e}$ is eigenvector of covariance $\mathbf{X}^\mathbf{T}\mathbf{X}$
  - May need to normalize (to get unit length vector)

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Eigenfaces (PCA on face images)

0. Compute the mean face image and remove mean from dataset.

1. Compute covariance matrix of face images

2. Compute the principal components ("eigenfaces")
   - K eigenvectors with largest eigenvalues

$e_1, e_2, \ldots e_K$

3. Represent all face images in the dataset as linear combinations of eigenfaces
   - Perform nearest neighbor on these coefficients

$\rightarrow \bar{x}_1, \ldots \bar{x}_n$

$\mu = \sum_i x_i$

$A = XX^T$

$\bar{x}_i = x_i - \mu$

$A\lambda = \lambda e$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Eigenfaces example

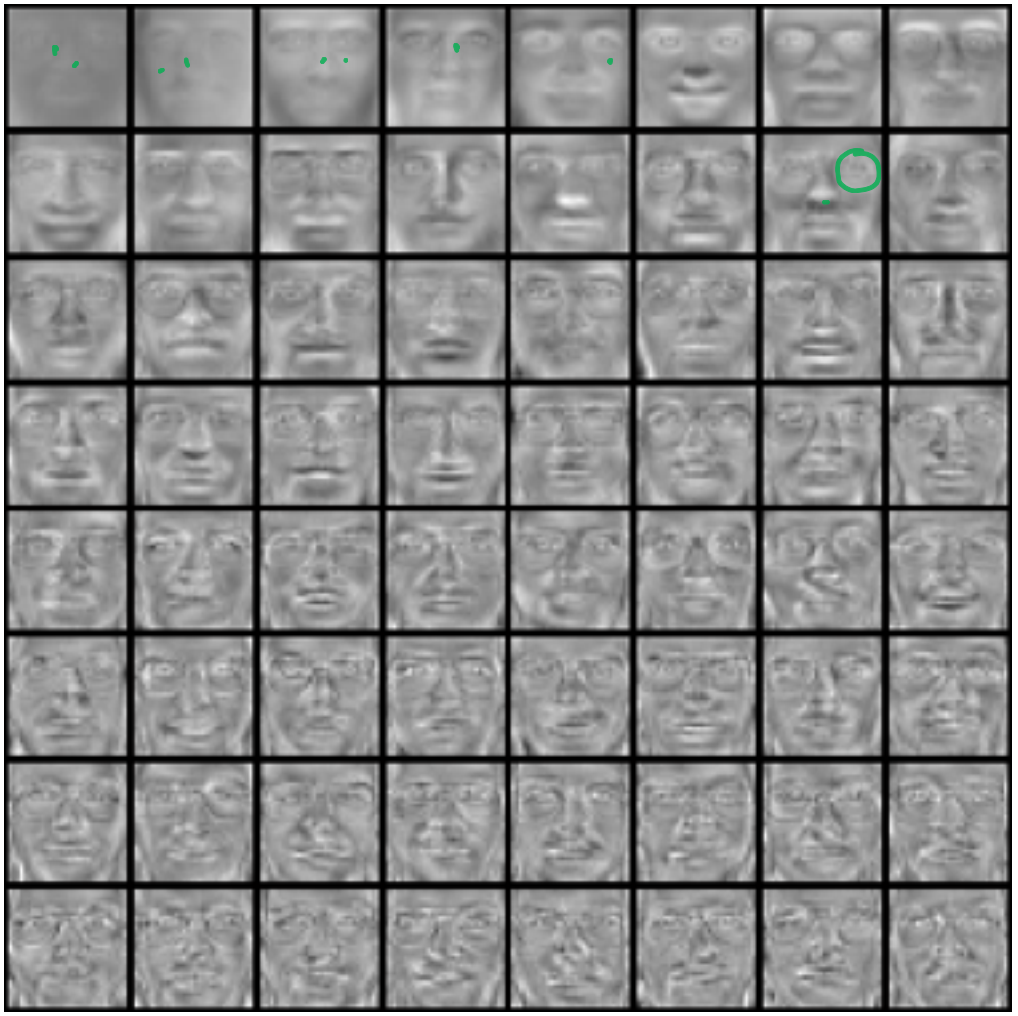
- Training images $\mathbf{x}_1, \ldots, \mathbf{x}_N$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Eigenfaces example

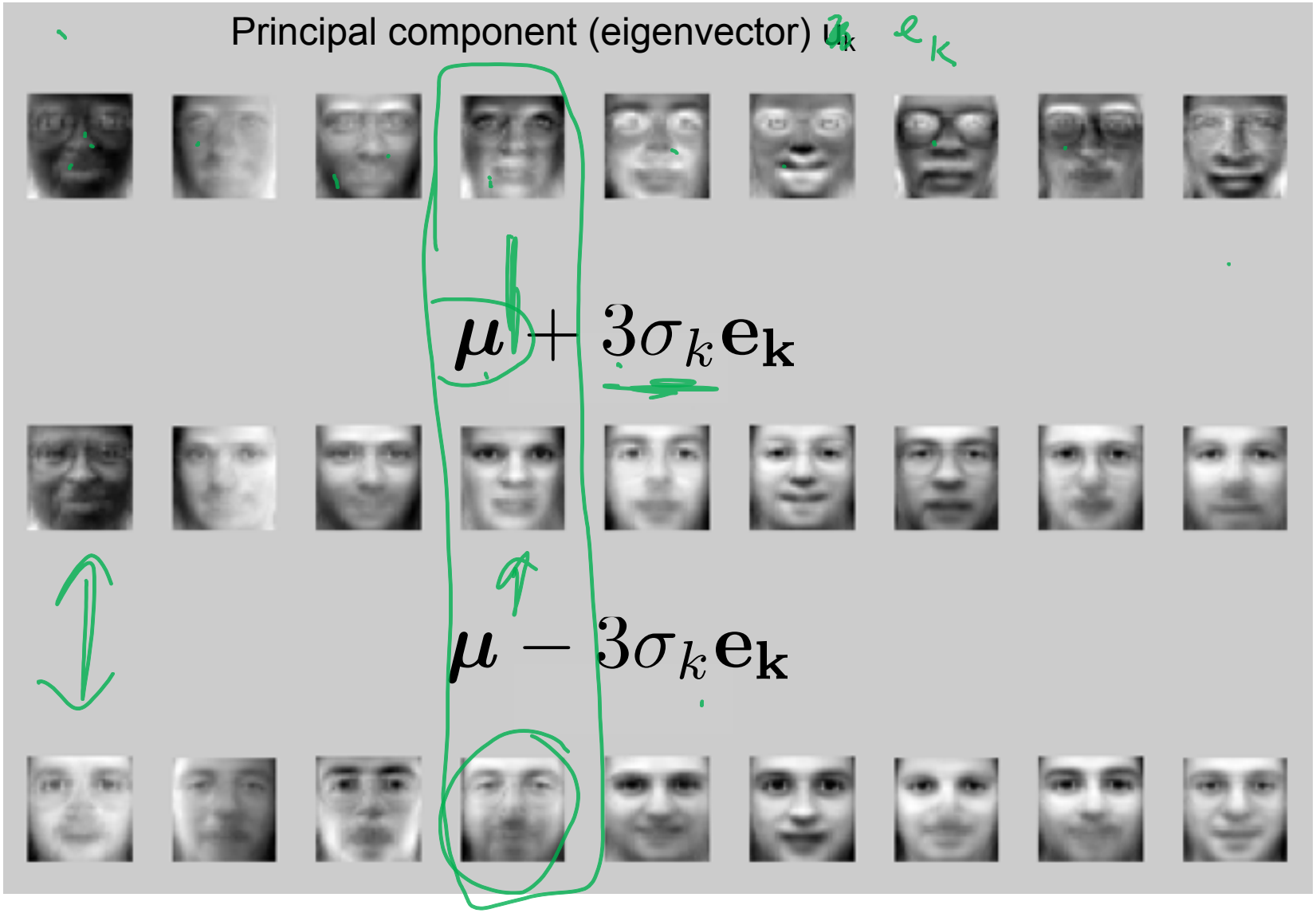
Top eigenvectors $\mathbf{e_1}, \ldots, \mathbf{e_k}$

Mean: μ



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Visualization of eigenfaces



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Representation and reconstruction

- Face $\mathbf{x}$ in "face space" coordinates:

$$\mathbf{x} \rightarrow \left[\mathbf{e_1}^T (\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{e_k}^T (\mathbf{x} - \boldsymbol{\mu}) \right]$$

$$= w_1, \ldots, w_k$$

$w_i = e_i^T (x - \mu)$

Representation and reconstruction

$\hat{x} = \|x - \hat{x}\|_2$

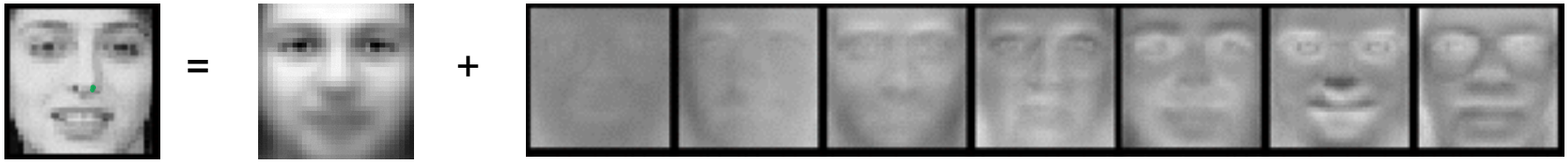
- Face $\mathbf{x}$ in "face space" coordinates:

$$\mathbf{x} \rightarrow \left[\mathbf{e_1}^T (\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{e_k}^T (\mathbf{x} - \boldsymbol{\mu}) \right]$$

$$= w_1, \ldots, w_k$$

inversion

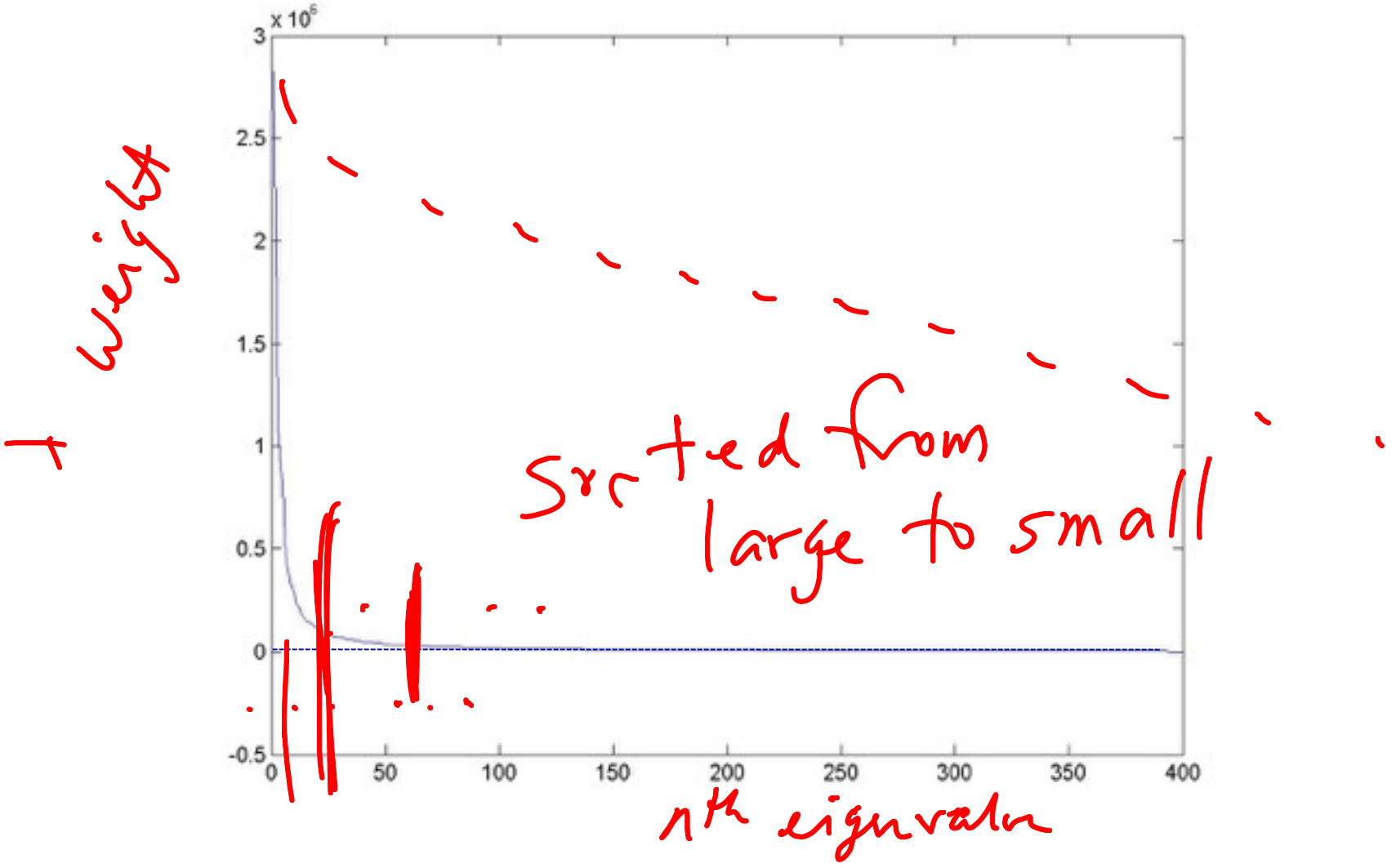
- Reconstruction



$$\hat{\mathbf{x}} = \boldsymbol{\mu} + w_1 \mathbf{e_1} + w_2 \mathbf{e_2} + \cdots + w_k \mathbf{e_k}$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Eigenvalues (variance along eigenvectors)



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean μ and covariance matrix Σ

$$\Sigma = XX^T$$

$$\left[\bar{x}_1 \vdots \bar{x}_2 \vdots \cdots \right]$$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean μ and covariance matrix Σ
- Find k principal components (eigenvectors of Σ) $\mathbf{e_1}, \ldots, \mathbf{e_k}$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean μ and covariance matrix Σ
- Find k principal components (eigenvectors of Σ) $\mathbf{e_1}, \ldots, \mathbf{e_k}$
- Project each training image $\mathbf{x_i}$ onto subspace spanned by principal components:

$$(w_{i1}, \ldots, w_{ik}) = \left(\mathbf{u_1}^T(\mathbf{x_i} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T(\mathbf{x_i} - \boldsymbol{\mu})\right)$$

$w_i \in \mathbb{R}^k$ $x_i \in \mathbb{R}^{BIG}$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean μ and covariance matrix Σ
- Find k principal components (eigenvectors of Σ) $\mathbf{e_1}, \ldots, \mathbf{e_k}$
- Project each training image $\mathbf{x_i}$ onto subspace spanned by principal components:

$$(w_{i1}, \ldots, w_{ik}) = \left(\mathbf{u_1}^T (\mathbf{x_i} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T (\mathbf{x_i} - \boldsymbol{\mu})\right)$$

Given novel image $\mathbf{x}$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean μ and covariance matrix Σ
- Find k principal components (eigenvectors of Σ) $\mathbf{e_1}, \ldots, \mathbf{e_k}$
- Project each training image $\mathbf{x_i}$ onto subspace spanned by principal components:

$$(w_{i1}, \ldots, w_{ik}) = \left(\mathbf{u_1}^T (\mathbf{x_i} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T (\mathbf{x_i} - \boldsymbol{\mu})\right)$$

Given novel image $\mathbf{x}$

- Project onto subspace:

$$(w_1, \ldots, w_k) = \left(\mathbf{u_1}^T (\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T (\mathbf{x} - \boldsymbol{\mu})\right)$$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{e_1}, \ldots, \mathbf{e_k}$
- Project each training image $\mathbf{x_i}$ onto subspace spanned by principal components:

$$(w_{i1}, \ldots, w_{ik}) = \left(\mathbf{u_1}^T(\mathbf{x_i} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T(\mathbf{x_i} - \boldsymbol{\mu})\right)$$

Given novel image $\mathbf{x}$

- Project onto subspace:

$$(w_1, \ldots, w_k) = \left(\mathbf{u_1}^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T(\mathbf{x} - \boldsymbol{\mu})\right)$$

- Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with eigenfaces

Process labeled training images

- Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{e_1}, \ldots, \mathbf{e_k}$
- Project each training image $\mathbf{x_i}$ onto subspace spanned by principal components:

$$(w_{i1}, \ldots, w_{ik}) = \left(\mathbf{u_1}^T(\mathbf{x_i} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T(\mathbf{x_i} - \boldsymbol{\mu})\right)$$

Given novel image $\mathbf{x}$

- Project onto subspace:

$$(w_1, \ldots, w_k) = \left(\mathbf{u_1}^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u_k}^T(\mathbf{x} - \boldsymbol{\mu})\right)$$

- Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face
- Classify as closest training face in k-dimensional subspace

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

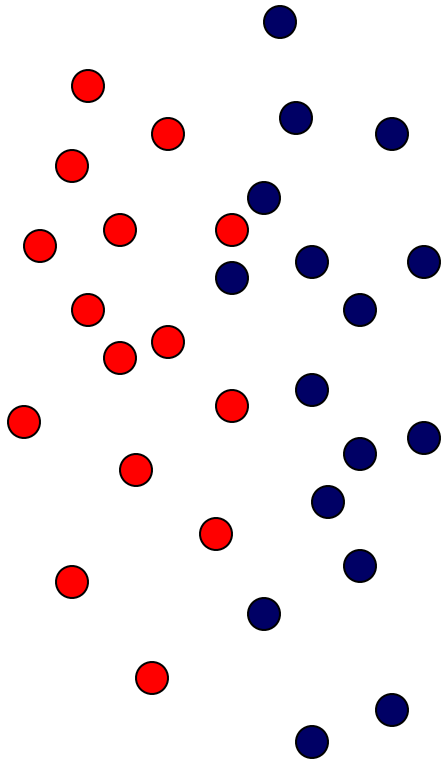
Limitations

Global appearance method: not robust to misalignment, background variation

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Limitations

- The direction of maximum variance is not always good for classification



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

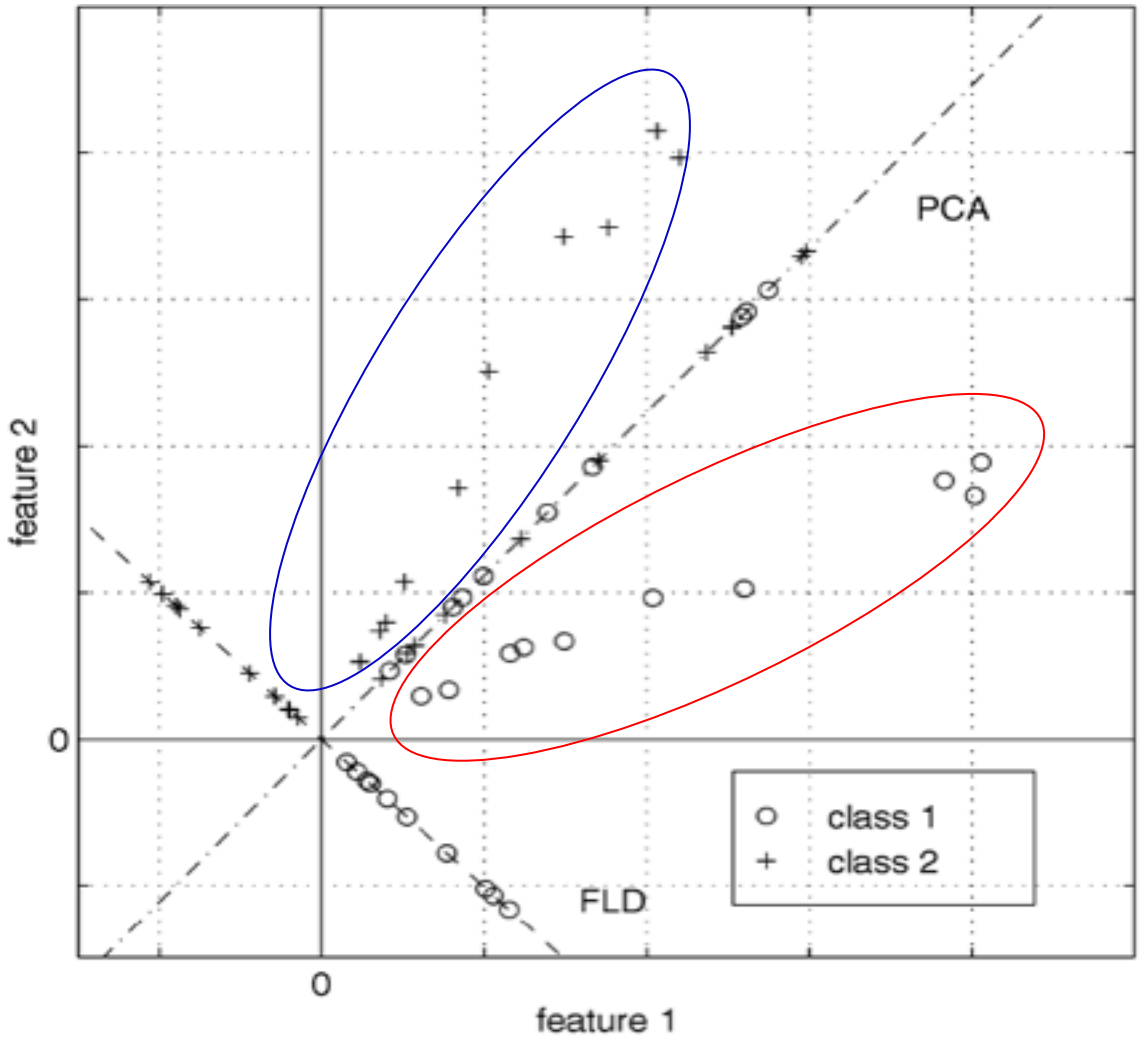
A more discriminative subspace: FLD

- Fisher Linear Discriminants → "Fisher Faces"
- PCA preserves maximum variance
- FLD preserves discrimination
  - Find projection that maximizes scatter between classes and minimizes scatter within classes

Reference: Eigenfaces vs. Fisherfaces, Belheumer et al., PAMI 1997

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Comparing with PCA



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Variables

- N Sample images: $\{x_1, \cdots, x_N\}$
- c classes: $\{\chi_1, \cdots, \chi_c\}$
- Average of each class: $\mu_i = \frac{1}{N_i} \sum_{x_k \in \chi_i} x_k$
- Average of all data: $\mu = \frac{1}{N} \sum_{k=1}^{N} x_k$

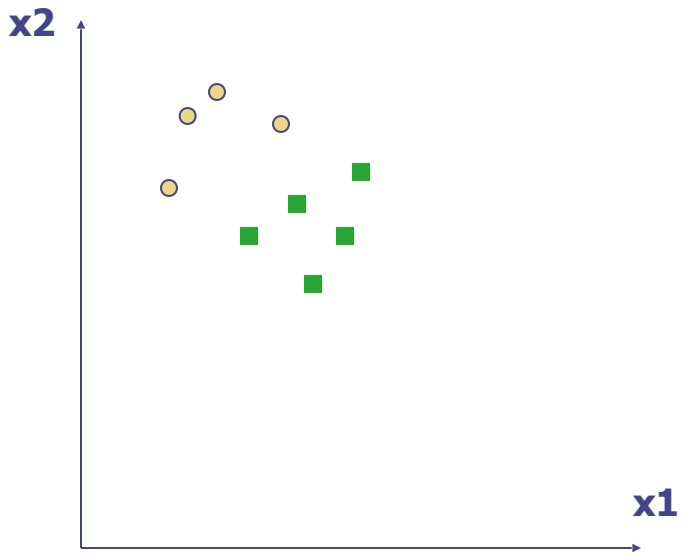
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Scatter Matrices

- Scatter of class i: $$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T$$
- Within class scatter: $$S_W = \sum_{i=1}^{c} S_i$$
- Between class scatter: $$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

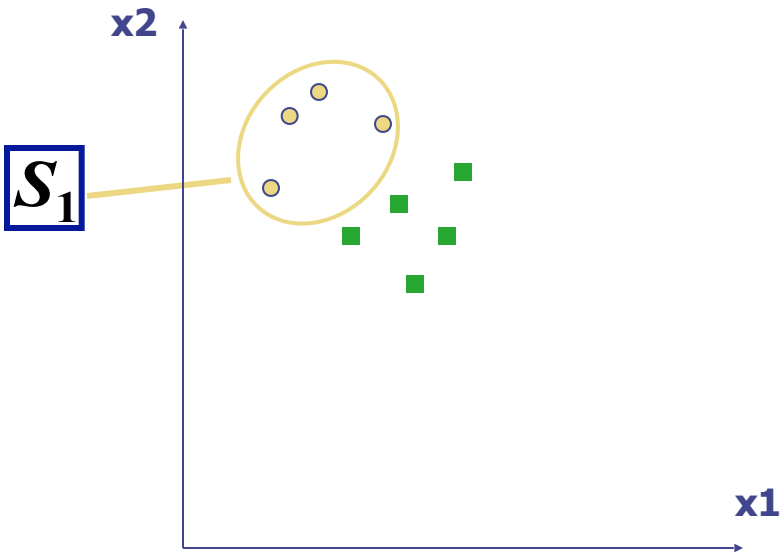
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Illustration



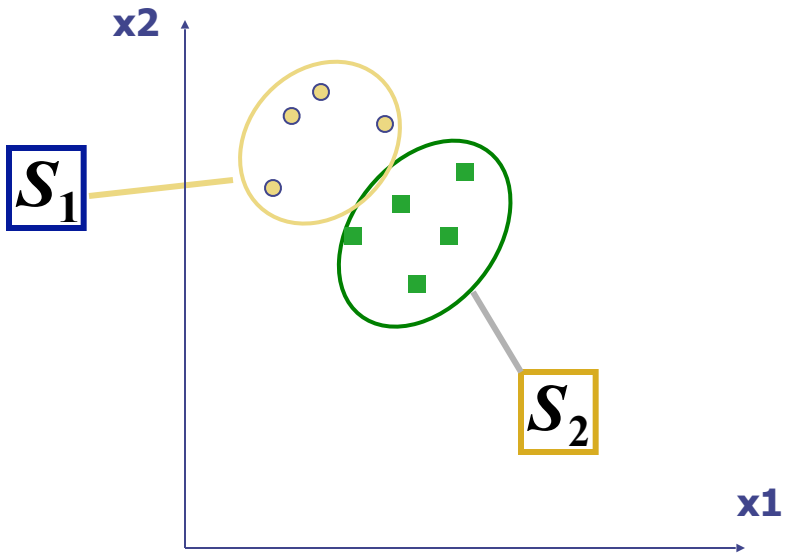
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Illustration



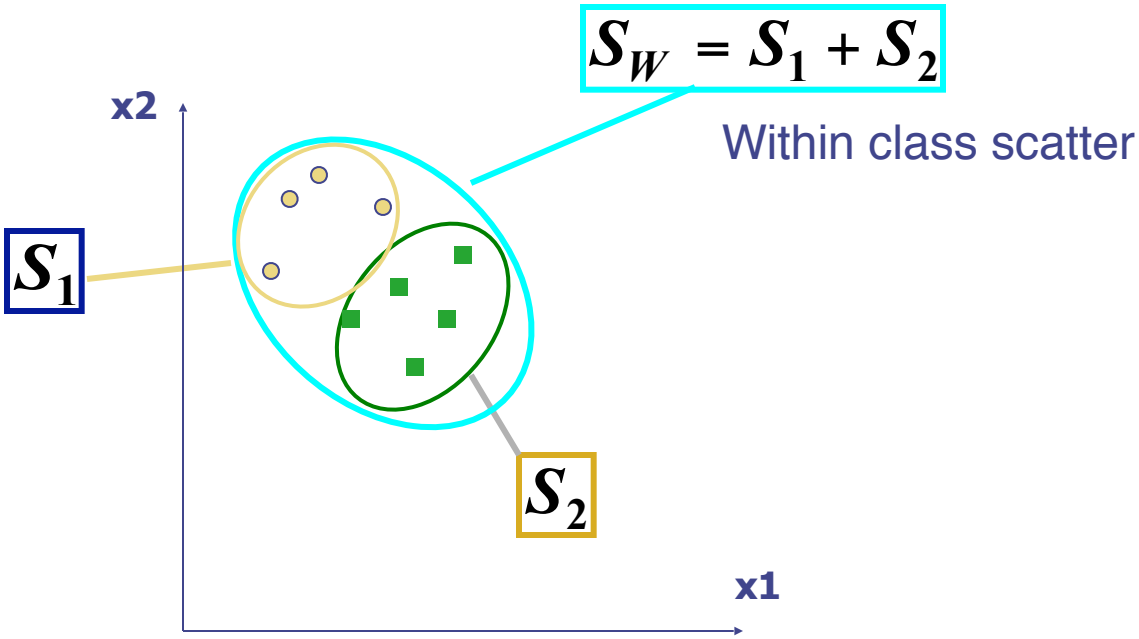
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Illustration



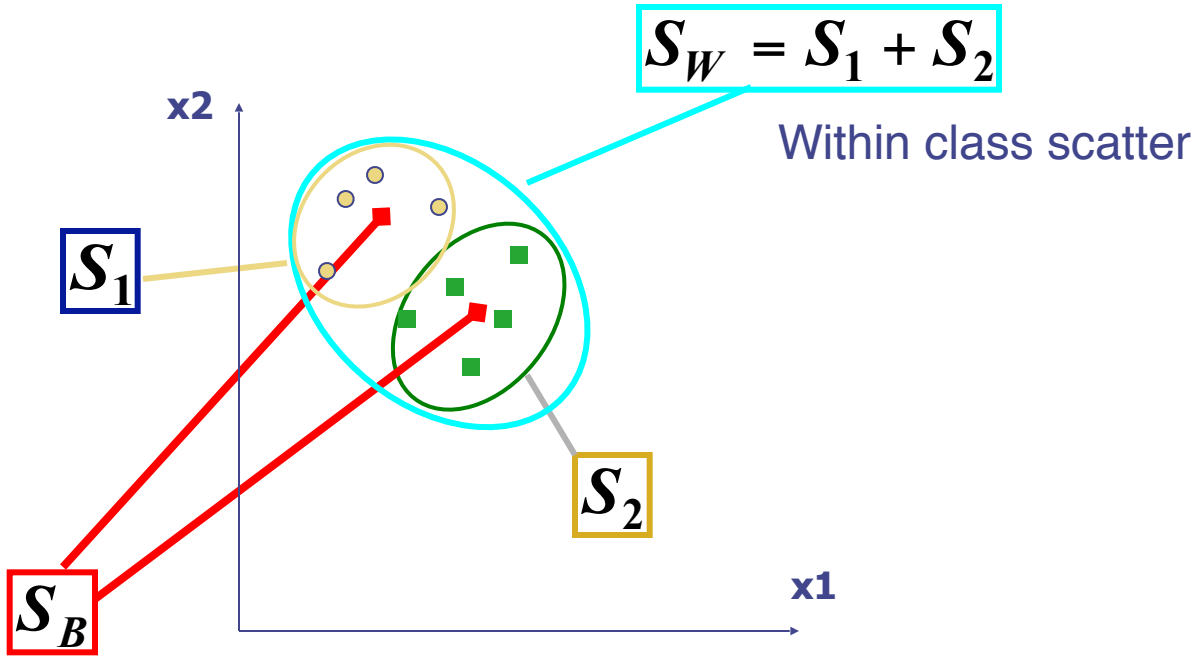
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Illustration



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Illustration



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Mathematical Formulation

- After projection

$$y_k = W^T x_k$$

  - Between class scatter $\tilde{S}_B = W^T S_B W$
  - Within class scatter $\tilde{S}_W = W^T S_W W$

- Objective:

$$W_{opt} = \arg\max_{W} \frac{\left|\tilde{S}_B\right|}{\left|\tilde{S}_W\right|} = \arg\max_{W} \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

- Solution: Generalized Eigenvectors

$$S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

- Rank of W_{opt} is limited
  - Rank(S_B) <= |Cl-1
  - Rank(S_W) <= N-C

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \qquad S_W = \sum_{i=1}^{c} S_i \qquad S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \qquad S_W = \sum_{i=1}^{c} S_i \qquad S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Solve generalized eigenvector problem

$$W_{fld} = \arg\max_{W} \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|} \qquad S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \qquad S_W = \sum_{i=1}^{c} S_i \qquad S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Solve generalized eigenvector problem

- Project to FLD subspace (c-1 dimensions)

$$W_{fld} = \arg\max_{\text{w}} \frac{|W^T S_B W|}{|W^T S_W W|} \qquad S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

$$W^T{}_{opt} = W^T{}_{fld} W^T{}_{pca} \qquad \hat{x} = W_{opt}{}^T x$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \qquad S_W = \sum_{i=1}^{c} S_i \qquad S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Solve generalized eigenvector problem

- Project to FLD subspace (c-1 dimensions)

$$W_{fld} = \arg\max_{W} \frac{|W^T S_B W|}{|W^T S_W W|} \qquad S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

- Classify by nearest neighbor

$$W^T{}_{opt} = W^T{}_{fld} W^T{}_{pca} \qquad \hat{x} = W_{opt}{}^T x$$

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \qquad S_W = \sum_{i=1}^{c} S_i \qquad S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Solve generalized eigenvector problem

- Project to FLD subspace (c-1 dimensions)

$$W_{fld} = \arg\max_{\text{w}} \frac{|W^T S_B W|}{|W^T S_W W|} \qquad S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

- Classify by nearest neighbor

$$W^T{}_{opt} = W^T{}_{fld} W^T{}_{pca} \qquad \hat{x} = W_{opt}{}^T x$$

Note: x in step 2 refers to PCA coef; x in step 4 refers to original data

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

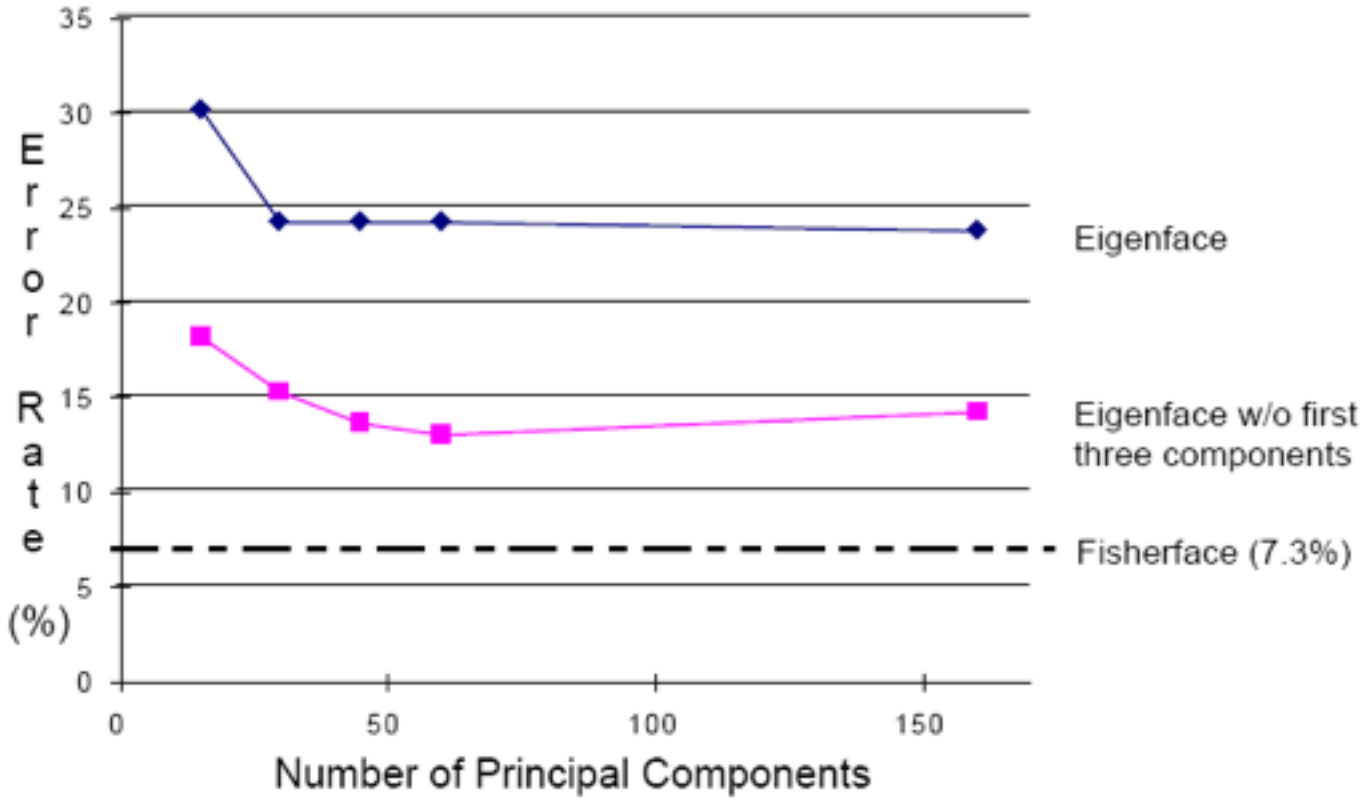
Results: Eigenface vs. Fisherface

- Input: 160 images of 16 people
- Train: 159 images
- Test: 1 image

- Variation in Facial Expression, Eyewear, and Lighting

Reference: Eigenfaces vs. Fisherfaces, Belheumer et al., PAMI 1997

Eigenfaces vs. Fisherfaces



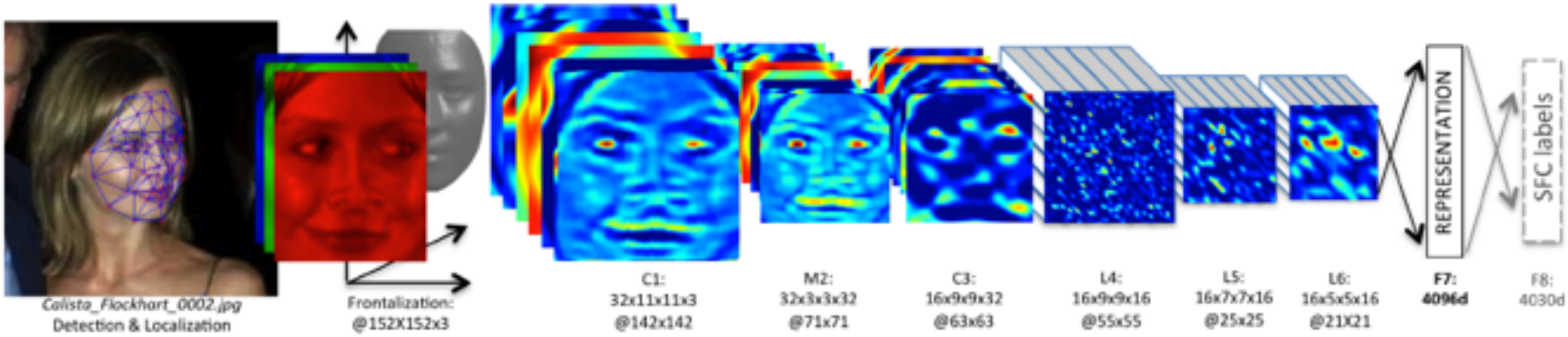
Reference: Eigenfaces vs. Fisherfaces, Belheumer et al., PAMI 1997

DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Yaniv Taigman Ming Yang Marc'Aurelio Ranzato Lior Wolf

Facebook AI Research
Menlo Park, CA, USA
{yaniv, mingyang, ranzato}@fb.com

Tel Aviv University
Tel Aviv, Israel
wolf@cs.tau.ac.il



DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Taigman, Yang, Ranzato, & Wolf (Facebook, Tel Aviv), CVPR 2014

Following slides adapted from Daphne Tsatsoulis

Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

2. Iteratively scale, rotate, and translate image until it aligns with a target face

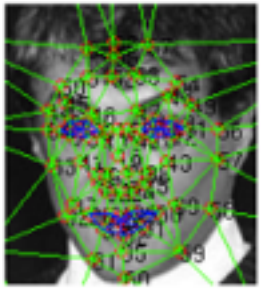
From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

2. Iteratively scale, rotate, and translate image until it aligns with a target face

3. Localize 67 fiducial points in the 2D aligned crop



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

2. Iteratively scale, rotate, and translate image until it aligns with a target face

3. Localize 67 fiducial points in the 2D aligned crop

4. Create a generic 3D shape model by taking the average of 3D scans from the USF Human-ID database and manually annotate the 67 anchor points

From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

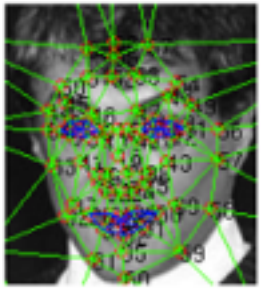
2. Iteratively scale, rotate, and translate image until it aligns with a target face

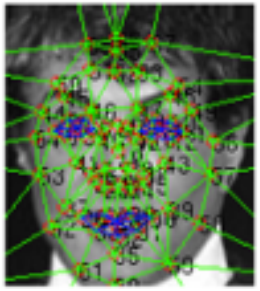
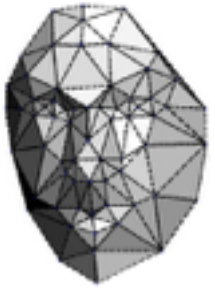
3. Localize 67 fiducial points in the 2D aligned crop

4. Create a generic 3D shape model by taking the average of 3D scans from the USF Human-ID database and manually annotate the 67 anchor points

5.Fit an affine 3D-to-2D camera and use it to direct the warping of the face






From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Face Alignment

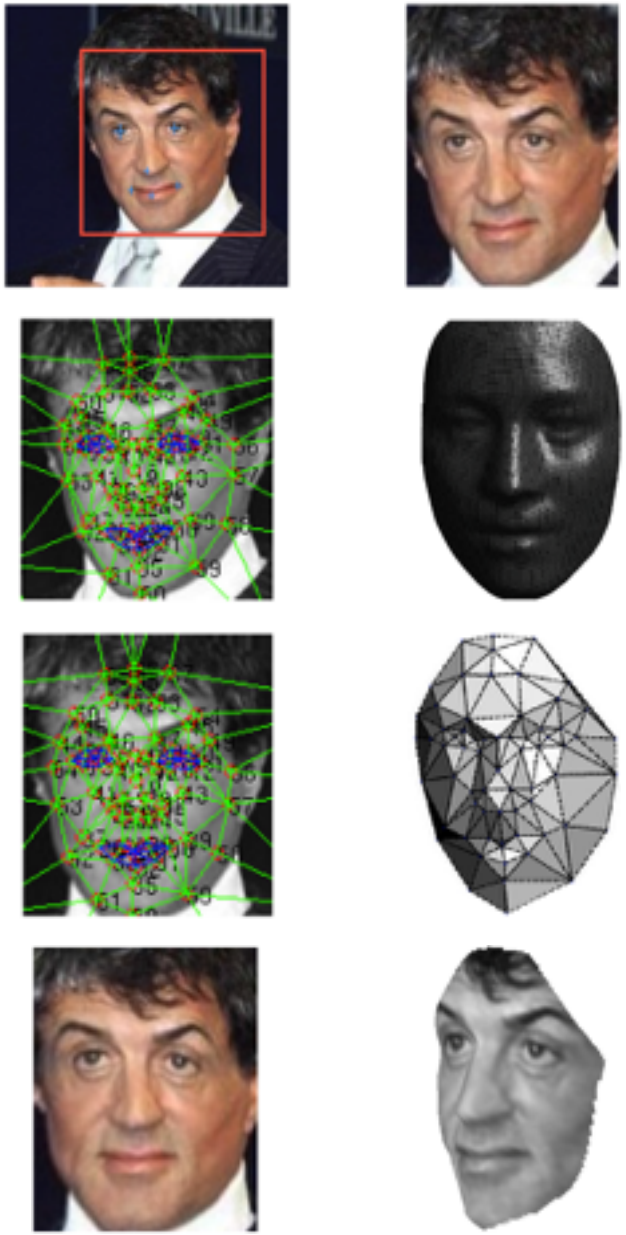
1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

2. Iteratively scale, rotate, and translate image until it aligns with a target face

3. Localize 67 fiducial points in the 2D aligned crop

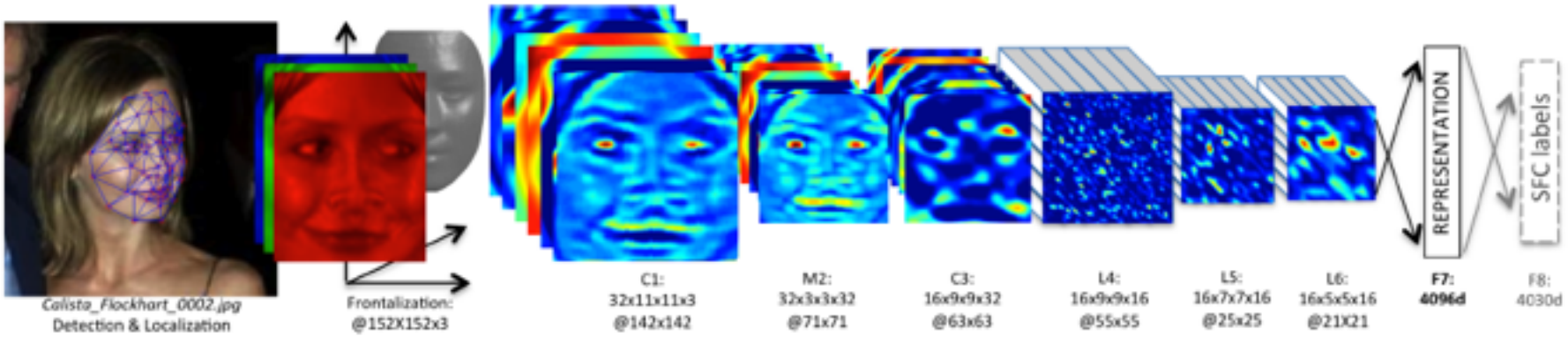
4. Create a generic 3D shape model by taking the average of 3D scans from the USF Human-ID database and manually annotate the 67 anchor points

5.Fit an affine 3D-to-2D camera and use it to direct the warping of the face



From Hoiem's Slides: http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture15%20-%20Face%20Recognition.ppt

Train DNN classifier on aligned faces



Architecture (deep neural network classifier)

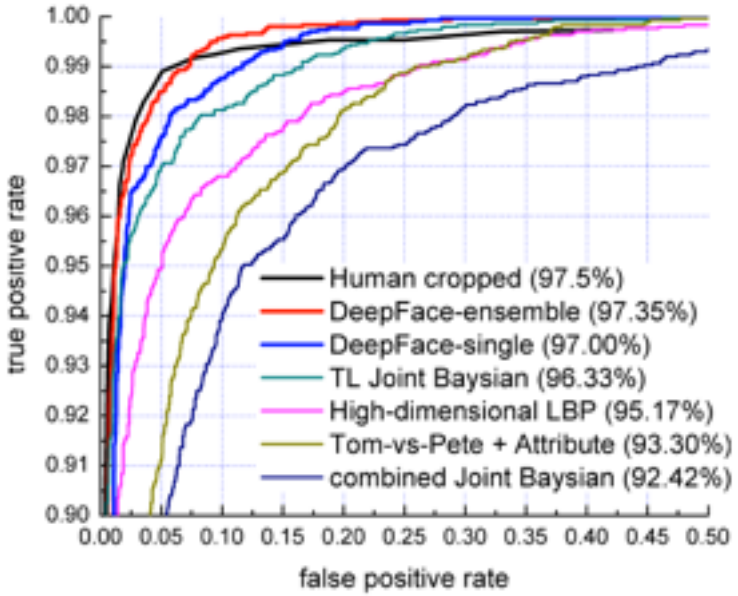
- Two convolutional layers (with one pooling layer)
- 3 locally connected and 2 fully connected layers
- > 120 million parameters

Train on dataset with 4400 individuals, ~1000 images each

- Train to identify face among set of possible people

Verification is done by comparing features at last layer for two faces

Results: Labeled Faces in the Wild Dataset



Method	Accuracy ± SE	Protocol
Joint Bayesian [6]	0.9242 ±0.0108	restricted
Tom-vs-Pete [4]	0.9330 ±0.0128	restricted
High-dim LBP [7]	0.9517 ±0.0113	restricted
TL Joint Bayesian [5]	0.9633 ±0.0108	restricted
DeepFace-single	**0.9592** ±0.0029	unsupervised
DeepFace-single	**0.9700** ±0.0028	restricted
DeepFace-ensemble	**0.9715** ±0.0027	restricted
DeepFace-ensemble	**0.9735** ±0.0025	unrestricted
Human, cropped	0.9753	

Performs similarly to humans!

(note: humans would do better with uncropped faces)

Experiments show that alignment is crucial (0.97 vs 0.88) and that deep features help (0.97 vs. 0.91)

Images as Points: Learning a Linear Basis

Computer Vision
Technical Derivation and Background
Case Study: Faces

Jason Corso

University of Michigan

Motivation

- We discussed first two classes of images basis for considering images as points:
  1. Cartesian Basis
  2. Analytical Bases
     - ★ Fourier Basis
     - ★ Haar Wavelet Basis
     - ★ (Gabor Basis)
- While the Cartesian Basis is useful primarily for conceptualization, the analytical bases each have their own appropriate use. For example, Fourier Basis is useful for analyzing periodic textures in images while the Haar basis has compact support and is multiscale.
- However, none of these bases let the data define its structure. This may be useful, for example, when the image content lives on a lower dimensional subspace than the original Cartesian Basis.
- When this lower dimensional subspace is linear, this becomes a relatively straightforward problem of learning the basis.

Learning a Linear Basis Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?

Learning a Linear Basis Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?
- First, we need a distance function on the sample space. Let's use the Euclidean distance and the sum of squared distances criterion:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n} \|\mathbf{x}_0 - \mathbf{x}_k\|^2 \tag{1}$$

Learning a Linear Basis Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?
- First, we need a distance function on the sample space. Let's use the Euclidean distance and the sum of squared distances criterion:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n} \|\mathbf{x}_0 - \mathbf{x}_k\|^2 \tag{1}$$

- Then, we seek a value of $\mathbf{x}_0$ that minimizes J_0.

Learning a Linear Basis Version 0

- We can show that the minimizer is indeed the sample mean:

$$\mathbf{m} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k \tag{2}$$

- We can verify it by adding $\mathbf{m} - \mathbf{m}$ into J_0:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n}\|(\mathbf{x}_0 - \mathbf{m}) - (\mathbf{x}_k - \mathbf{m})\|^2 \tag{3}$$

$$= \sum_{k=1}^{n}\|\mathbf{x}_0 - \mathbf{m}\|^2 + \sum_{k=1}^{n}\|\mathbf{x}_k - \mathbf{m}\|^2 \tag{4}$$

- Thus, J_0 is minimized when $\mathbf{x}_0 = \mathbf{m}$. Note, the second term is independent of $\mathbf{x}_0$.

Learning a Linear Basis Version 0

- So, the sample mean is an initial dimension-reduced representation of the data (a zero-dimensional one).
- It is simple, but it not does reveal any of the variability in the data.
- Let's try to obtain a one-dimensional representation: i.e., let's project the data onto a line running through the sample mean.

Learning a Linear Basis Version 1

- Let $\mathbf{e}$ be a unit vector in the direction of the line.
- The standard equation for a line is then

$$\mathbf{x} = \mathbf{m} + a\mathbf{e} \tag{5}$$

where scalar $a \in \mathbb{R}$ governs which point along the line we are and hence corresponds to the distance of any point $\mathbf{x}$ from the mean $\mathbf{m}$.

Learning a Linear Basis Version 1

- Let $\mathbf{e}$ be a unit vector in the direction of the line.
- The standard equation for a line is then

$$\mathbf{x} = \mathbf{m} + a\mathbf{e} \tag{5}$$

  where scalar $a \in \mathbb{R}$ governs which point along the line we are and hence corresponds to the distance of any point $\mathbf{x}$ from the mean $\mathbf{m}$.
- Represent point $\mathbf{x}_k$ by $\mathbf{m} + a_k\mathbf{e}$.
- We can find an optimal set of coefficients by again minimizing the squared-error criterion

$$J_1(a_1, \ldots, a_n, \mathbf{e}) = \sum_{k=1}^{n} \|(\mathbf{m} + a_k\mathbf{e}) - \mathbf{x}_k\|^2 \tag{6}$$

$$= \sum_{k=1}^{n} a_k^2 \|\mathbf{e}\|^2 - 2\sum_{k=1}^{n} a_k \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m}) + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{7}$$

Learning a Linear Basis Version 1

- Differentiating for a_k and equating to 0 yields

$$a_k = \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m}) \tag{8}$$

- This indicates that the best value for a_k is the projection of the point $\mathbf{x}_k$ onto the line $\mathbf{e}$ that passes through $\mathbf{m}$.

Learning a Linear Basis Version 1

- Differentiating for a_k and equating to 0 yields

$$a_k = \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m}) \tag{8}$$

- This indicates that the best value for a_k is the projection of the point $\mathbf{x}_k$ onto the line $\mathbf{e}$ that passes through $\mathbf{m}$.
- How do we find the best direction for that line?

Learning a Linear Basis Version 1

- What if we substitute the expression we computed for the best a_k directly into the J_1 criterion:

$$J_1(\mathbf{e}) = \sum_{k=1}^{n} a_k^2 - 2\sum_{k=1}^{n} a_k^2 + \sum_{k=1}^{n} \|x_k - \mathbf{m}\|^2 \tag{9}$$

$$= -\sum_{k=1}^{n} \left[\mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m})\right]^2 + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{10}$$

$$= -\sum_{k=1}^{n} \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}}\mathbf{e} + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{11}$$

Learning a Linear Basis Version 1

The Scatter Matrix

- Define the scatter matrix $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{12}$$

Learning a Linear Basis Version 1

The Scatter Matrix

- Define the scatter matrix $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{12}$$

- This should be familiar – this is a multiple of the sample covariance matrix.

Learning a Linear Basis Version 1

The Scatter Matrix

- Define the scatter matrix $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{12}$$

- This should be familiar – this is a multiple of the sample covariance matrix.
- Putting it in:

$$J_1(\mathbf{e}) = -\mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e} + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{13}$$

- The $\mathbf{e}$ that maximizes $\mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e}$ will minimize J_1.

Learning a Linear Basis Version 1

Solving for $\mathbf{e}$

- We use Lagrange multipliers to maximize $\mathbf{e}^T\mathbf{S}\mathbf{e}$ subject to the constraint that $\|\mathbf{e}\| = 1$.

$$u = \mathbf{e}^T\mathbf{S}\mathbf{e} - \lambda(\mathbf{e}^T\mathbf{e} - 1) \tag{14}$$

Learning a Linear Basis Version 1

Solving for $\mathbf{e}$

- We use Lagrange multipliers to maximize $\mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e}$ subject to the constraint that $\|\mathbf{e}\| = 1$.

$$u = \mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e} - \lambda(\mathbf{e}^{\mathsf{T}}\mathbf{e} - 1) \tag{14}$$

- Differentiating w.r.t. $\mathbf{e}$ and setting equal to 0.

$$\frac{\partial u}{\partial \mathbf{e}} = 2\mathbf{S}\mathbf{e} - 2\lambda\mathbf{e} \tag{15}$$

$$\mathbf{S}\mathbf{e} = \lambda\mathbf{e} \tag{16}$$

- Does this form look familiar?

Learning a Linear Basis Version 1

Eigenvectors of $\mathbf{S}$

$$\mathbf{S}\mathbf{e} = \lambda\mathbf{e}$$

- This is an eigenproblem.
- Hence, it follows that the best one-dimensional estimate (in a least-squares sense) for the data is the eigenvector corresponding to the largest eigenvalue of $\mathbf{S}$.
- So, we will project the data onto the largest eigenvector of $\mathbf{S}$ and translate it to pass through the mean.

Principal Component Analysis

- We're already done...basically.

Principal Component Analysis

- We're already done...basically.
- This idea readily extends to multiple dimensions, say $d' < d$ dimensions.
- We replace the earlier equation of the line with

$$\mathbf{x} = \mathbf{m} + \sum_{i=1}^{d'} a_i \mathbf{e}_i \tag{17}$$

- And we have a new criterion function

$$J_{d'} = \sum_{k=1}^{n} \left\| \left(\mathbf{m} + \sum_{i=1}^{d'} a_{ki} e_i \right) - \mathbf{x}_k \right\|^2 \tag{18}$$

Principal Component Analysis

- $J_{d'}$ is minimized when the vectors $\mathbf{e}_1, \ldots, \mathbf{e}_{d'}$ are the d' eigenvectors fo the scatter matrix having the largest eigenvalues.
- These vectors are orthogonal.
- They form a natural set of basis vectors for representing any feature $\mathbf{x}$.
- The coefficients a_i are called the principal components.
- Visualize the basis vectors as the principal axes of a hyperellipsoid surrounding the data (a cloud of points).
- Principle components reduces the dimension of the data by restricting attention to those directions of maximum variation, or scatter.

Fisher Linear Discriminant

- Description vs. Discrimination
- PCA is likely going to be useful for representing data.
- But, there is no reason to assume that it would be good for discriminating between two classes of data.
  - 'Q' versus 'O'.
- Discriminant Analysis seeks directions that are efficient for discrimination.

Let's Formalize the Situation

- Suppose we have a set of n d-dimensional samples with n_1 in set $\mathcal{D}_1$, and similarly for set $\mathcal{D}_2$.

$$\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\}, \quad i = \{1, 2\} \tag{19}$$

Let's Formalize the Situation

- Suppose we have a set of n d-dimensional samples with n_1 in set $\mathcal{D}_1$, and similarly for set $\mathcal{D}_2$.

$$\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\}, \quad i = \{1, 2\} \tag{19}$$

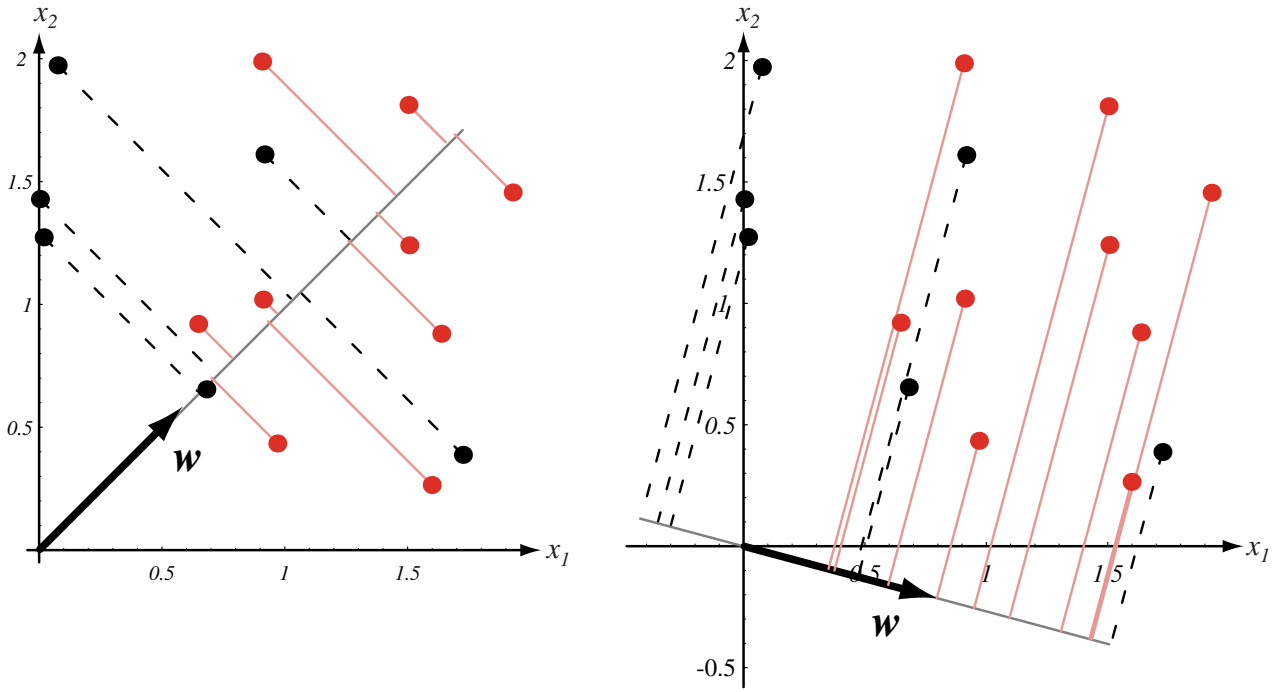
- We can form a linear combination of the components of a sample $\mathbf{x}$:

$$y = \mathbf{w}^\mathsf{T}\mathbf{x} \tag{20}$$

which yields a corresponding set of n samples $y_1, \ldots, y_n$ split into subsets $\mathcal{Y}_1$ and $\mathcal{Y}_2$.

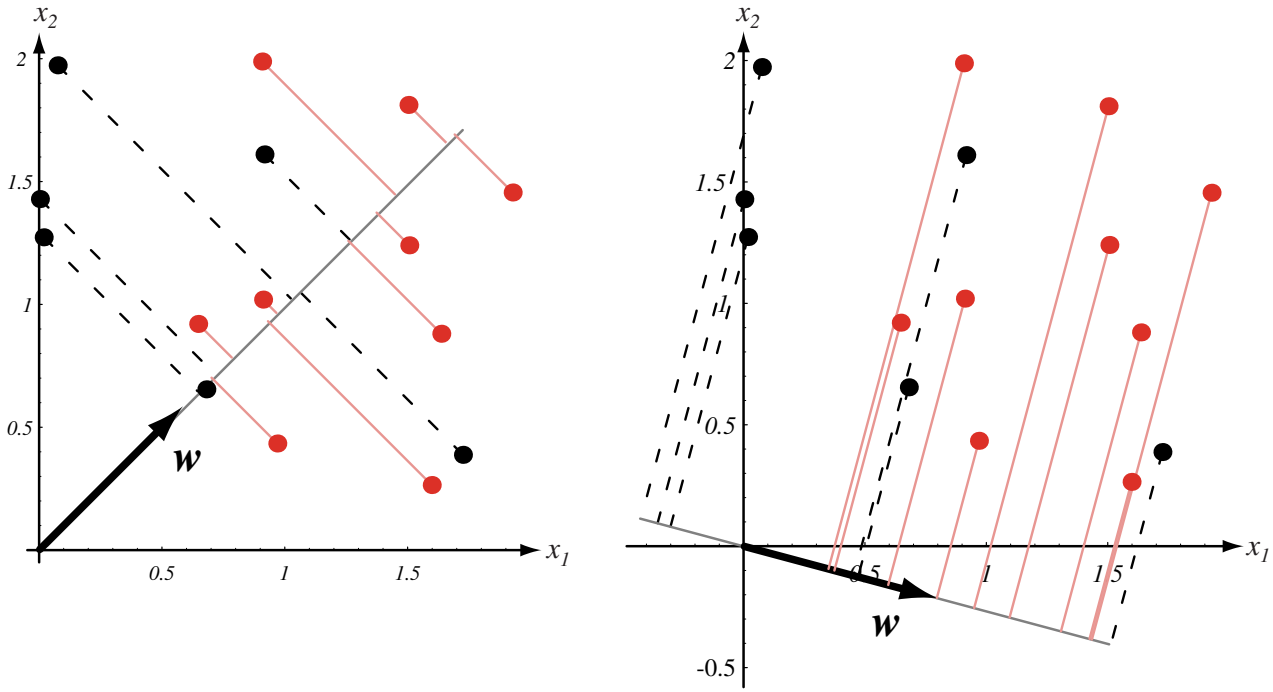
Geometrically, This is a Projection

- If we constrain the norm of $\mathbf{w}$ to be 1 (i.e., $\|\mathbf{w}\| = 1$) then we can conceptualize that each y_i is the projection of the corresponding $\mathbf{x}_i$ onto a line in the direction of $\mathbf{w}$.



Geometrically, This is a Projection

- If we constrain the norm of $\mathbf{w}$ to be 1 (i.e., $\|\mathbf{w}\| = 1$) then we can conceptualize that each y_i is the projection of the corresponding $\mathbf{x}_i$ onto a line in the direction of $\mathbf{w}$.



- Does the magnitude of $\mathbf{w}$ have any real significance?

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).
- However, this may not be possible depending on our underlying classes.

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).
- However, this may not be possible depending on our underlying classes.
- So, how do we find the best direction $\mathbf{w}$?

Separation of the Projected Means

- Let $\mathbf{m}_i$ be the d-dimensional sample mean for class i:

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{x} \ . \tag{21}$$

Separation of the Projected Means

- Let $\mathbf{m}_i$ be the d-dimensional sample mean for class i:

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{x} \ . \tag{21}$$

- Then the sample mean for the projected points is

$$\tilde{m}_i = \frac{1}{n_i} \sum_{y \in \mathcal{Y}_i} y \tag{22}$$

$$= \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{w}^\mathsf{T} \mathbf{x} \tag{23}$$

$$= \mathbf{w}^\mathsf{T} \mathbf{m}_i \ . \tag{24}$$

And, thus, is simply the projection of $\mathbf{m}_i$.

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{25}$$

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{25}$$

- The scale of $\mathbf{w}$: we can make this distance arbitrarily large by scaling $\mathbf{w}$.

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{25}$$

- The scale of $\mathbf{w}$: we can make this distance arbitrarily large by scaling $\mathbf{w}$.
- Rather, we want to make this distance large relative to some measure of the standard deviation. ...This story we've heard before.

The Scatter

- To capture this variation, we compute the scatter

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m}_i)^2 \tag{26}$$

which is essentially a scaled sampled variance.

The Scatter

- To capture this variation, we compute the scatter

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m}_i)^2 \tag{26}$$

  which is essentially a scaled sampled variance.
- From this, we can estimate the variance of the pooled data:

$$\frac{1}{n} \left(\tilde{s}_1^2 + \tilde{s}_2^2 \right) \quad . \tag{27}$$

- $\tilde{s}_1^2 + \tilde{s}_2^2$ is called the total within-class scatter of the projected samples.

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{28}$$

- It does so independently of the magnitude of $\mathbf{w}$.

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{28}$$

- It does so independently of the magnitude of $\mathbf{w}$.
- This term is the ratio of the distance between the projected means scaled by the within-class scatter (the variation of the data).
- Recall the similar term from earlier in the lecture which indicated the amount a feature will reduce the probability of error is proportional to the ratio of the difference of the means to the variance. FLD will choose the maximum...

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{28}$$

- It does so independently of the magnitude of $\mathbf{w}$.
- This term is the ratio of the distance between the projected means scaled by the within-class scatter (the variation of the data).
- Recall the similar term from earlier in the lecture which indicated the amount a feature will reduce the probability of error is proportional to the ratio of the difference of the means to the variance. FLD will choose the maximum...
- We need to rewrite $J(\cdot)$ as a function of $\mathbf{w}$.

Within-Class Scatter Matrix

- Define scatter matrices $\mathbf{S}_i$:

$$\mathbf{S}_i = \sum_{x \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^\mathsf{T} \tag{29}$$

and

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2 \ . \tag{30}$$

Within-Class Scatter Matrix

- Define scatter matrices $\mathbf{S}_i$:

$$\mathbf{S}_i = \sum_{x \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^\mathsf{T} \tag{29}$$

and

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2 \ . \tag{30}$$

- $\mathbf{S}_W$ is called the within-class scatter matrix.
- $\mathbf{S}_W$ is symmetric and positive semidefinite.
- In typical cases, when is $\mathbf{S}_W$ nonsingular?

Within-Class Scatter Matrix

- Deriving the sum of scatters.

$$\tilde{s}_i^2 = \sum_{x \in \mathcal{D}_i} \left(\mathbf{w}^\mathsf{T} \mathbf{x} - \mathbf{w}^\mathsf{T} \mathbf{m}_i \right)^2 \tag{31}$$

$$= \sum_{x \in \mathcal{D}_i} \mathbf{w}^\mathsf{T} \left(\mathbf{x} - \mathbf{m}_i \right) \left(\mathbf{x} - \mathbf{m}_i \right)^\mathsf{T} \mathbf{w} \tag{32}$$

$$= \mathbf{w}^\mathsf{T} \mathbf{S}_i \mathbf{w} \tag{33}$$

- We can therefore write the sum of the scatters as an explicit function of $\mathbf{w}$:

$$\tilde{s}_1^2 + \tilde{s}_2^2 = \mathbf{w}^\mathsf{T} \mathbf{S}_W \mathbf{w} \tag{34}$$

Between-Class Scatter Matrix

- The separation of the projected means obeys

$$(\tilde{m}_1 - \tilde{m}_2)^2 = \left(\mathbf{w}^\mathsf{T}\mathbf{m}_1 - \mathbf{w}^\mathsf{T}\mathbf{m}_2\right)^2 \tag{35}$$

$$= \mathbf{w}^\mathsf{T}\left(\mathbf{m}_1 - \mathbf{m}_2\right)\left(\mathbf{m}_1 - \mathbf{m}_2\right)^\mathsf{T}\mathbf{w} \tag{36}$$

$$= \mathbf{w}^\mathsf{T}\mathbf{S}_B\mathbf{w} \tag{37}$$

- Here, $\mathbf{S}_B$ is called the between-class scatter matrix:

$$\mathbf{S}_B = \left(\mathbf{m}_1 - \mathbf{m}_2\right)\left(\mathbf{m}_1 - \mathbf{m}_2\right)^\mathsf{T} \tag{38}$$

- $\mathbf{S}_B$ is also symmetric and positive semidefinite.
- When is $\mathbf{S}_B$ nonsingular?

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^\mathsf{T}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^\mathsf{T}\mathbf{S}_W\mathbf{w}} \tag{39}$$

- This is the generalized Rayleigh quotient.

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^\mathsf{T}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^\mathsf{T}\mathbf{S}_W\mathbf{w}} \tag{39}$$

- This is the generalized Rayleigh quotient.
- The vector $\mathbf{w}$ that maximizes $J(\cdot)$ must satisfy

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w} \tag{40}$$

which is a generalized eigenvalue problem.

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^\mathsf{T}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^\mathsf{T}\mathbf{S}_W\mathbf{w}} \tag{39}$$

- This is the generalized Rayleigh quotient.
- The vector $\mathbf{w}$ that maximizes $J(\cdot)$ must satisfy

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w} \tag{40}$$

  which is a generalized eigenvalue problem.
- For nonsingular $\mathbf{S}_W$ (typical), we can write this as a standard eigenvalue problem:

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{w} = \lambda\mathbf{w} \tag{41}$$

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{42}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{42}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.
- The FLD converts a many-dimensional problem to a one-dimensional one.

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{42}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.
- The FLD converts a many-dimensional problem to a one-dimensional one.
- One still must find the threshold. This is easy for known densities, but not so easy in general.

A Classic PCA vs. FLD comparison

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- PCA maximizes the total scatter across all classes.
- PCA projections are thus optimal for reconstruction from a low dimensional basis, but not necessarily from a discrimination standpoint.
- FLD maximizes the ratio of the between-class scatter and the within-class scatter.
- FLD tries to "shape" the scatter to make it more effective for classification.

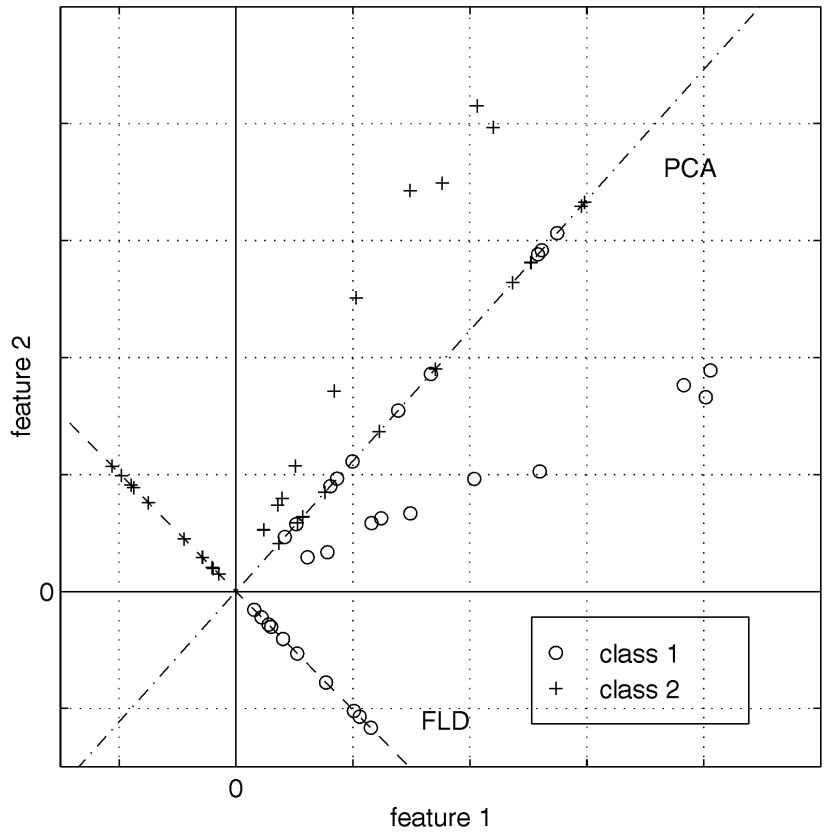


Fig. 2. A comparison of principal component analysis (PCA) and Fisher's linear discriminant (FLD) for a two class problem where data for each class lies near a linear subspace.

Case Study: EigenFaces versus FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Analysis of classic pattern recognition techniques (PCA and FLD) to do face recognition.
- Fixed pose but varying illumination.
- The variation in the resulting images caused by the varying illumination will nearly always dominate the variation caused by an identity change.

Fig. 1. The same person seen under different lighting conditions can appear dramatically different: In the left image, the dominant light source is nearly head-on; in the right image, the dominant light source is from above and to the right.

Case Study: EigenFaces versus FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

Is Linear Okay For Faces?

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Fact: all of the images of a Lambertian surface, taken from a fixed viewpoint, but under varying illumination, lie in a 3D linear subspace of the high-dimensional image space.

Is Linear Okay For Faces?

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Fact: all of the images of a Lambertian surface, taken from a fixed viewpoint, but under varying illumination, lie in a 3D linear subspace of the high-dimensional image space.
- But, in the presence of shadowing, specularities, and facial expressions, the above statement will not hold. This will ultimately result in deviations from the 3D linear subspace and worse classification accuracy.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- Pre-Processing – each image is normalized to have zero mean and unit variance.
- Why?

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- Pre-Processing – each image is normalized to have zero mean and unit variance.
- Why?
- Gets rid of the light source intensity and the effects of a camera's automatic gain control.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- Pre-Processing – each image is normalized to have zero mean and unit variance.
- Why?
- Gets rid of the light source intensity and the effects of a camera's automatic gain control.
- For a query image $\mathbf{x}$, we select the class of the training image that is the nearest neighbor in the image space:

$$\mathbf{x}^* = \arg\min_{\{\mathbf{x}_i\}} \|\mathbf{x} - \mathbf{x}_i\| \quad \text{then decide } C(\mathbf{x}^*) \tag{43}$$

where we have vectorized each image.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- What are the advantages and disadvantages of the correlation based method in this context?

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- What are the advantages and disadvantages of the correlation based method in this context?
- Computationally expensive.
- Require large amount of storage.
- Noise may play a role.
- Highly parallelizable.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- Quickly recall the main idea of PCA.
- Define a linear projection of the original n-d image space into an m-d space with $m < n$ or $m \ll n$, which yields new vectors $\mathbf{y}$:

$$\mathbf{y}_k = W^{\mathsf{T}} \mathbf{x}_k \qquad k = 1, 2, \ldots, N \tag{44}$$

where $W \in \mathbb{R}^{n \times m}$.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- Quickly recall the main idea of PCA.
- Define a linear projection of the original n-d image space into an m-d space with $m < n$ or $m \ll n$, which yields new vectors $\mathbf{y}$:

$$\mathbf{y}_k = W^{\mathsf{T}} \mathbf{x}_k \qquad k = 1, 2, \ldots, N \tag{44}$$

  where $W \in \mathbb{R}^{n \times m}$.
- Define the total scatter matrix S_T as

$$S_T = \sum_{k=1}^{N} (\mathbf{x}_k - \boldsymbol{\mu}) (\mathbf{x}_k - \boldsymbol{\mu})^{\mathsf{T}} \tag{45}$$

  where μ is the sample mean image.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- The scatter of the projected vectors $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is

$$W^{\mathsf{T}} S_T W \tag{46}$$

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- The scatter of the projected vectors $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is

$$W^{\mathsf{T}} S_T W \tag{46}$$

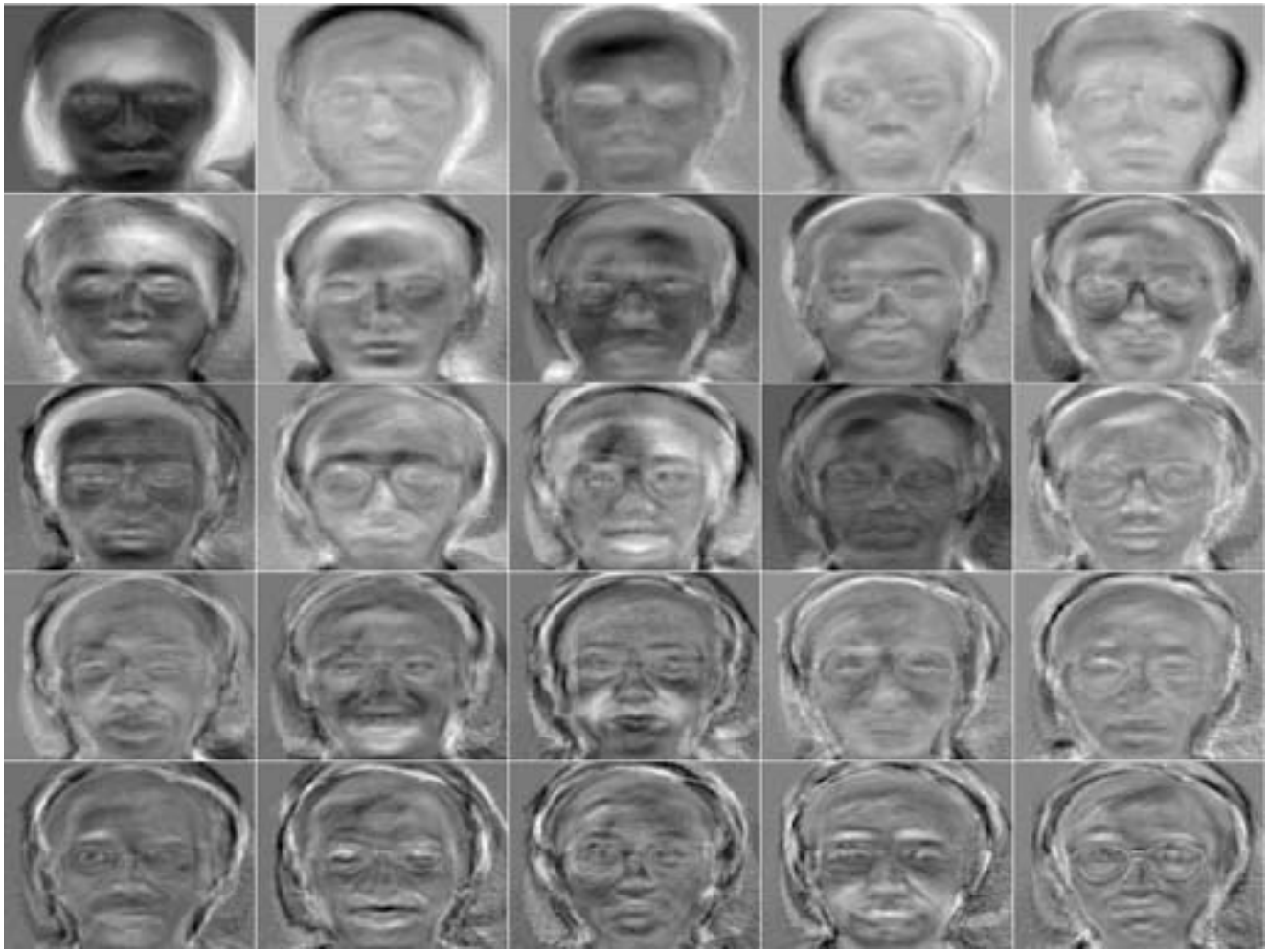
- W_{opt} is chosen to maximize the determinant of the total scatter matrix of the projected vectors:

$$W_{\text{opt}} = \arg\max_W |W^{\mathsf{T}} S_T W| \tag{47}$$

$$= \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \ldots & \mathbf{w}_m \end{bmatrix} \tag{48}$$

where $\{\mathbf{w}_i | i = 1, d, \ldots, m\}$ is the set of n-d eigenvectors of S_T corresponding to the largest m eigenvalues.

- An Example:



Source: http://www.cs.princeton.edu/ cdecoro/eigenfaces/. (not sure if this dataset include lighting variation...)

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- What are the advantages and disadvantages of the eigenfaces method in this context?

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- What are the advantages and disadvantages of the eigenfaces method in this context?
- The scatter being maximized is due not only to the between-class scatter that is useful for classification but also to the within-clas scatter, which is generally undesirable for classification.
- If PCA is presented faces with varying illumination, the projection matrix W_{opt} will contain principal components that retain the variation in lighting. If this variation is higher than the variation due to class identity, then PCA will suffer greatly for classification.
- Yields a more compact representation than the correlation-based method.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Use Lambertian model directly.
- Consider a point p on a Lambertian surface illuminated by a point light source at infinity.
- Let $\mathbf{s} \in \mathbb{R}^3$ signify the product of the light source intensity with the unit vector for the light source direction.
- The image intensity of the surface at p when viewed by a camera is

$$E(p) = a(p)\mathbf{n}(p)^{\mathsf{T}}\mathbf{s} \tag{49}$$

  where $\mathbf{n}(p)$ is the unit inward normal vector to the surface at point p, and $a(p)$ is the albedo of the surface at p (a scalar).
- Hence, the image intensity of the point p is linear on $\mathbf{s}$.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- So, if we assume no shadowing, given three images of a Lambertian surface from the same viewpoint under three known, linearly independent light source directions, the albedo and surface normal can be recovered.
- Alternatively, one can reconstruct the image of the surface under an arbitrary lighting direction by a linear combination of the three original images.
- This fact can be used for classification.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- So, if we assume no shadowing, given three images of a Lambertian surface from the same viewpoint under three known, linearly independent light source directions, the albedo and surface normal can be recovered.
- Alternatively, one can reconstruct the image of the surface under an arbitrary lighting direction by a linear combination of the three original images.
- This fact can be used for classification.
- For each face (class) use three or more images taken under different lighting conditions to construct a 3D basis for the linear subspace.
- For recognition, compute the distance of a new image to each linear subspace and choose the face corresponding to the shortest distance.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Pros and Cons?

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Pros and Cons?
- If there is no noise or shadowing, this method will achieve error free classification under any lighting conditions (and if the surface is indeed Lambertian).
- Faces inevitably have self-shadowing.
- Faces have expressions...
- Still pretty computationally expensive (linear in number of classes).

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Recall the Fisher Linear Discriminant setup.
- The between-class scatter matrix

$$S_B = \sum_{i=1}^{c} N_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^{\mathsf{T}} \tag{50}$$

- The within-class scatter matrix

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \mathcal{D}_i} (x_k - \boldsymbol{\mu}_i)(x_k - \boldsymbol{\mu}_i)^{\mathsf{T}} \tag{51}$$

- The optimal projection W_{opt} is chosen as the matrix with orthonormal columns which maximizes the ratio of the determinant of the between-class scatter matrix of the projected vectors to the determinant of the within-class scatter of the projected vectors:

$$W_{\text{opt}} = \arg\max_{W} \frac{|W^{\mathsf{T}} S_B W|}{|W^{\mathsf{T}} S_W W|} \tag{52}$$

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{53}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{53}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.
- In face recognition, things get a little more complicated because the within-class scatter matrix S_W is always singular.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{53}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.
- In face recognition, things get a little more complicated because the within-class scatter matrix S_W is always singular.
- This is because the rank of S_W is at most $N - c$ and the number of images in the learning set are commonly much smaller than the number of pixels in the image.
- This means we can choose W such that the within-class scatter is exactly zero.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- To overcome this, project the image set to a lower dimensional space so that the resulting within-class scatter matrix S_W is nonsingular.
- How?

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- To overcome this, project the image set to a lower dimensional space so that the resulting within-class scatter matrix S_W is nonsingular.
- How?
- PCA to first reduce the dimension to $N - c$ and the FLD to reduce it to $c - 1$.
- $W_{\text{opt}}^{\mathsf{T}}$ is given by the product $W_{\text{FLD}}^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}}$ where

$$W_{\text{PCA}} = \arg\max_{W} |W^{\mathsf{T}} S_T W| \tag{54}$$

$$W_{\text{FLD}} = \arg\max_{W} \frac{|W^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}} S_B W_{\text{PCA}} W|}{|W^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}} S_W W_{\text{PCA}} W|} \tag{55}$$

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Hypothesis: face recognition algorithms will perform better if they exploit the fact that images of a Lambertian surface lie in a linear subspace.
- Used Hallinan's Harvard Database which sampled the space of light source directions in 15 degree incrememements.
- Used 330 images of five people (66 of each) and extracted five subsets.
- Classification is nearest neighbor in all cases.

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

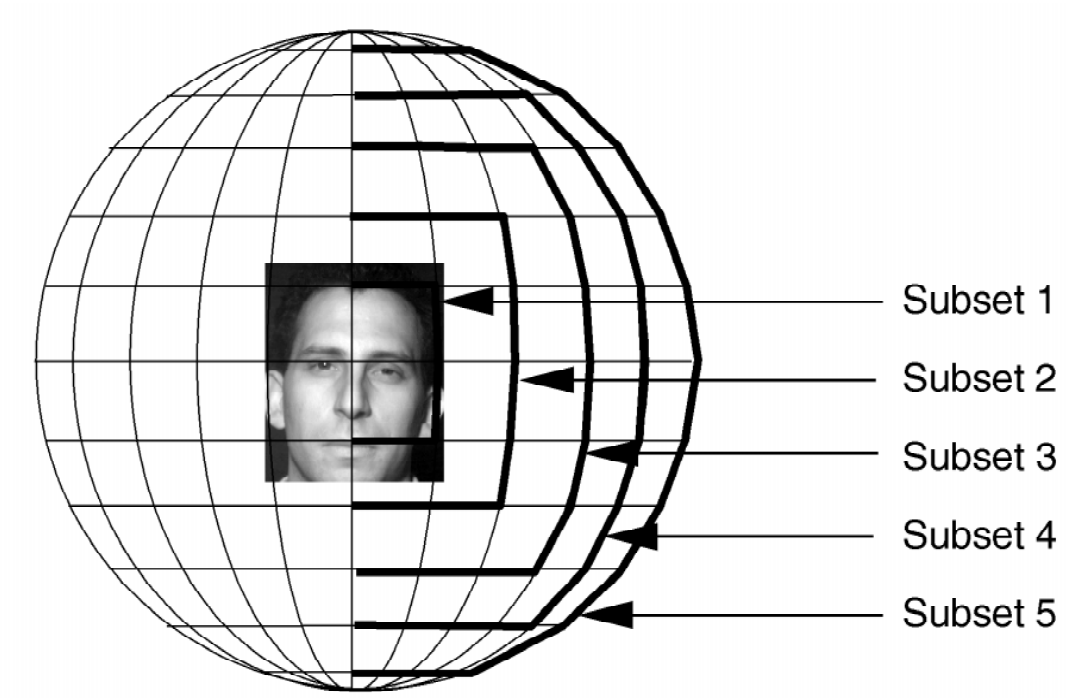
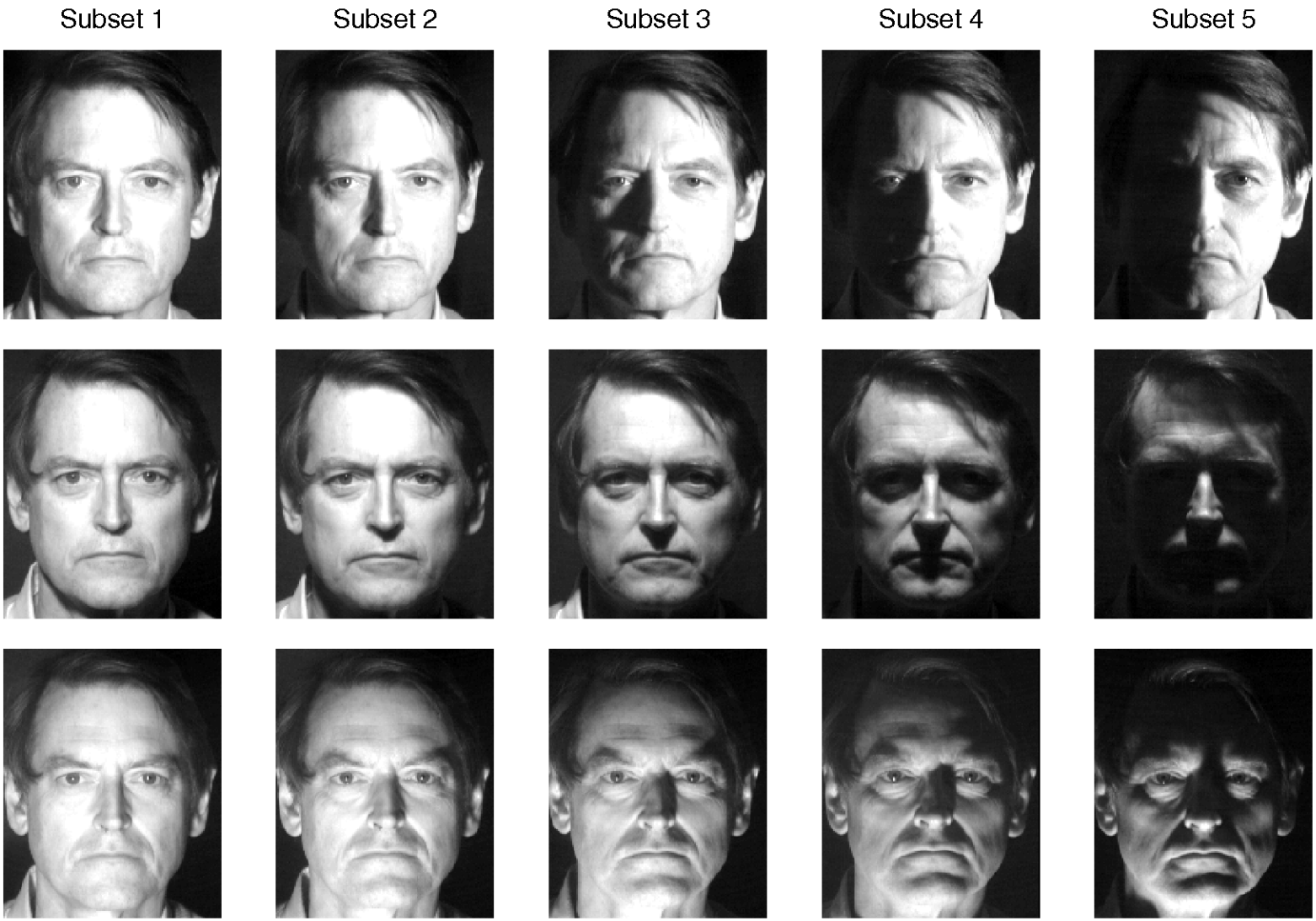


Fig. 3. The highlighted lines of longitude and latitude indicate the light source directions for Subsets 1 through 5. Each intersection of a longitudinal and latitudinal line on the right side of the illustration has a corresponding image in the database.

Experiment 1: Variation in Lighting

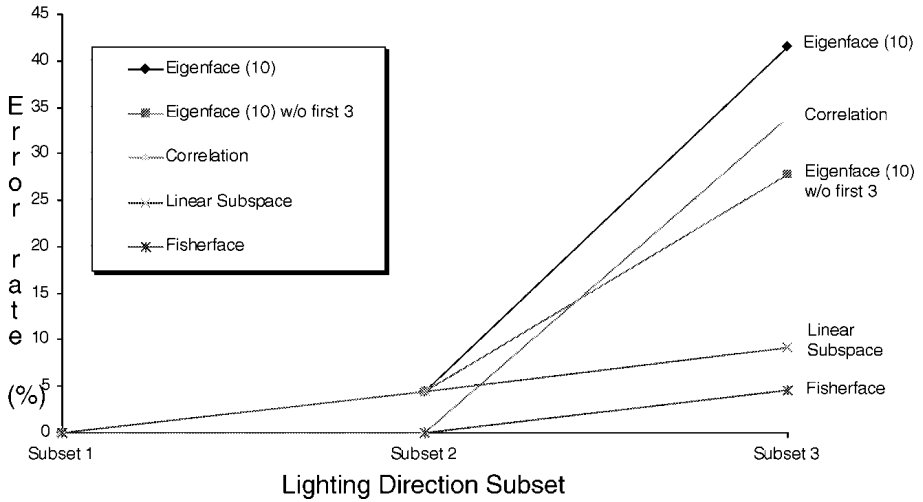
Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.



Experiment 1: Variation in Lighting – Extrapolation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Train on Subset 1.
- Test of Subsets 1,2,and 3.

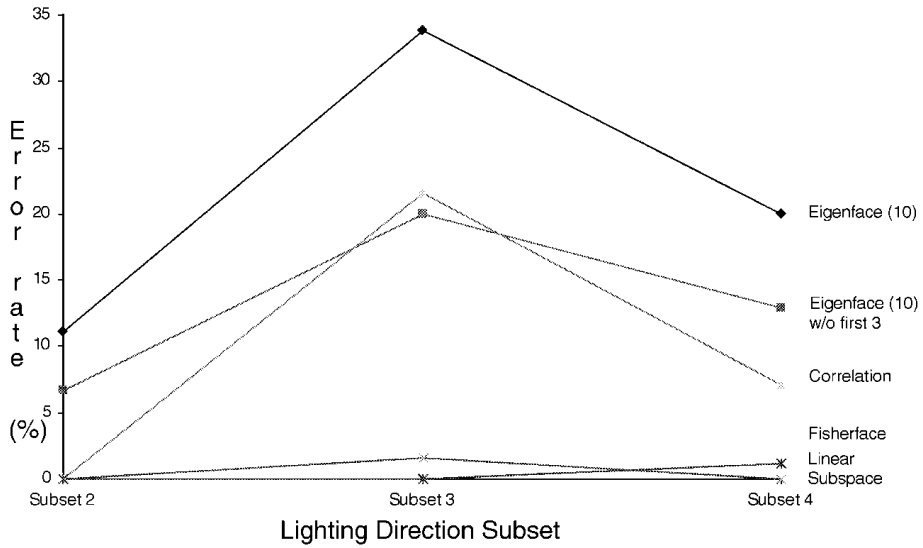


Extrapolating from Subset 1				
Method	Reduced Space	Error Rate (%)		
		Subset 1	Subset 2	Subset 3
Eigenface	4	0.0	31.1	47.7
	10	0.0	4.4	41.5
Eigenface w/o 1st 3	4	0.0	13.3	41.5
	10	0.0	4.4	27.7
Correlation	29	0.0	0.0	33.9
Linear Subspace	15	0.0	4.4	9.2
Fisherface	4	0.0	0.0	4.6

Experiment 2: Variation in Lighting – Interpolation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Train on Subsets 1 and 5.
- Test of Subsets 2, 3, and 4.



Interpolating between Subsets 1 and 5				
Method	Reduced Space	Error Rate (%)		
		Subset 2	Subset 3	Subset 4
Eigenface	4	53.3	75.4	52.9
	10	11.11	33.9	20.0
Eigenface w/o 1st 3	4	31.11	60.0	29.4
	10	6.7	20.0	12.9
Correlation	129	0.0	21.54	7.1
Linear Subspace	15	0.0	1.5	0.0
Fisherface	4	0.0	0.0	1.2

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.
- Linear Subspace has comparable error rates with the FisherFace method, but it requires 3x as much storage and takes three times as long.

Experiment 1 and 2: Variation in Lighting

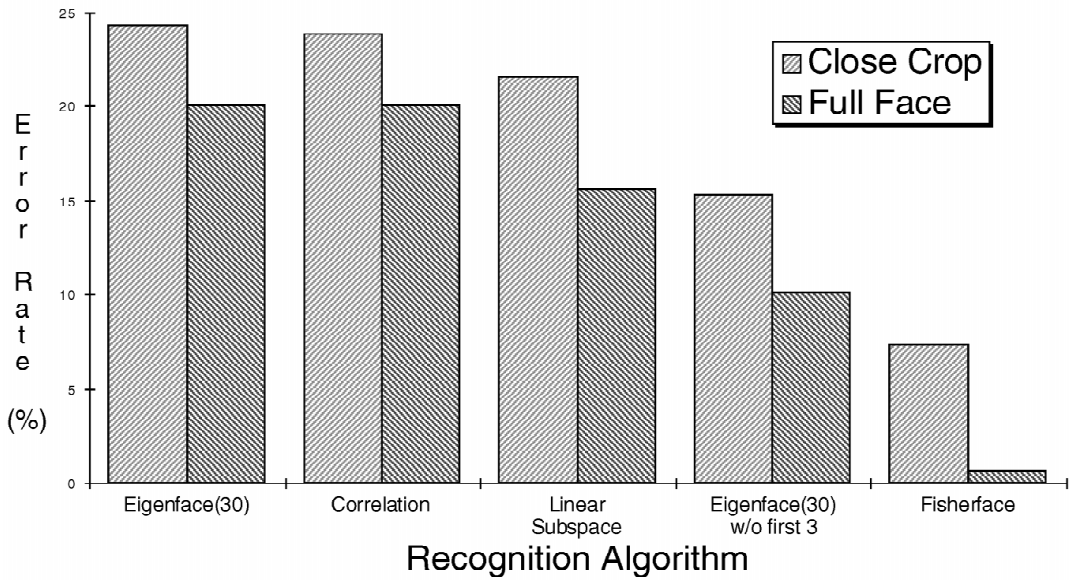
Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.
- Linear Subspace has comparable error rates with the FisherFace method, but it requires 3x as much storage and takes three times as long.
- The Fisherface method had error rates lower than the Eigenface method and required less computation time.

Experiment 3: Yale DB

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

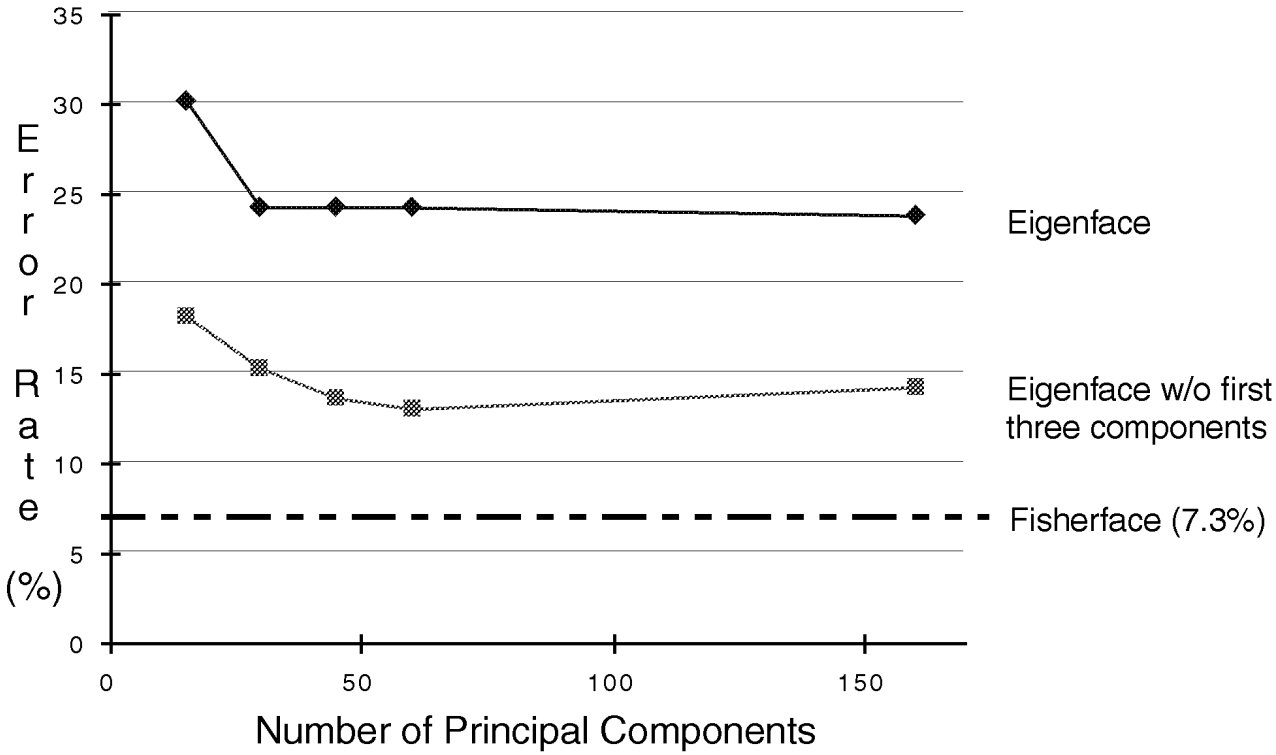
- Uses a second database of 16 subjects with ten images of images (5 varying in expression).



"Leaving-One-Out" of Yale Database			
Method	Reduced Space	Error Rate (%)	
		Close Crop	Full Face
Eigenface	30	24.4	19.4
Eigenface w/o 1st 3	30	15.3	10.8
Correlation	160	23.9	20.0
Linear Subspace	48	21.6	15.6
Fisherface	15	7.3	0.6

Experiment 3: Comparing Variation with Number of Principal Components

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.



Can PCA outperform FLD for recognition?

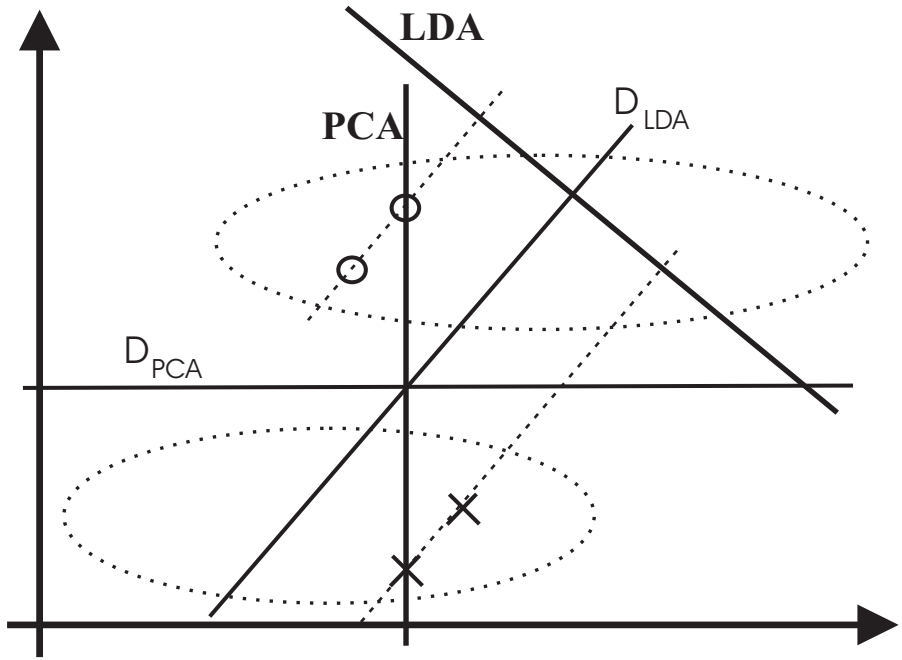
Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- There may be situations in which PCA might outperform FLD.
- Can you think of such a situation?

Can PCA outperform FLD for recognition?

Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- There may be situations in which PCA might outperform FLD.
- Can you think of such a situation?



- When we have few training data, then it may be preferable to describe total scatter.

Can PCA outperform FLD for recognition?

Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- They tested on the AR face database and found affirmative results.