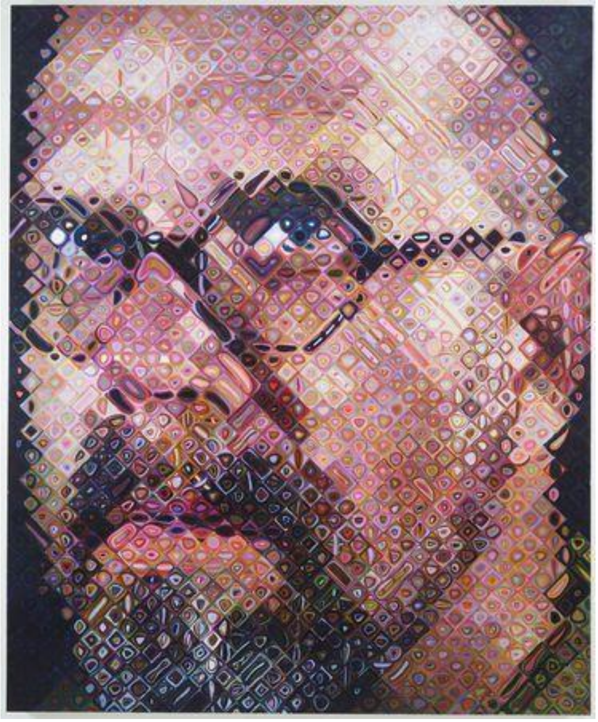
Face Recognition and Feature Subspaces

Lucas by Chuck Close



Chuck Close, self portrait

Slides from Derek Hoiem, Lana Lazebnik, Silvio Savarese, Fei-Fei Li

This class: face recognition

- Two methods: “Eigenfaces” and “Fisherfaces”
  - Feature subspaces: PCA and FLD
- Recent method: DeepFace
- Look at interesting findings about human face recognition

Applications of Face Recognition

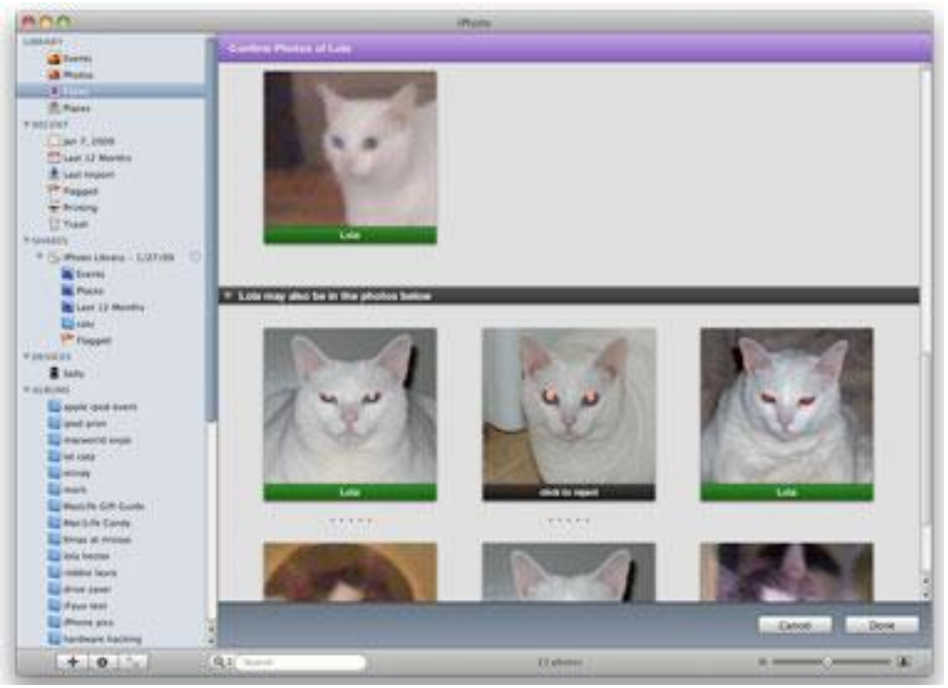
- Surveillance

Applications of Face Recognition

- Album organization

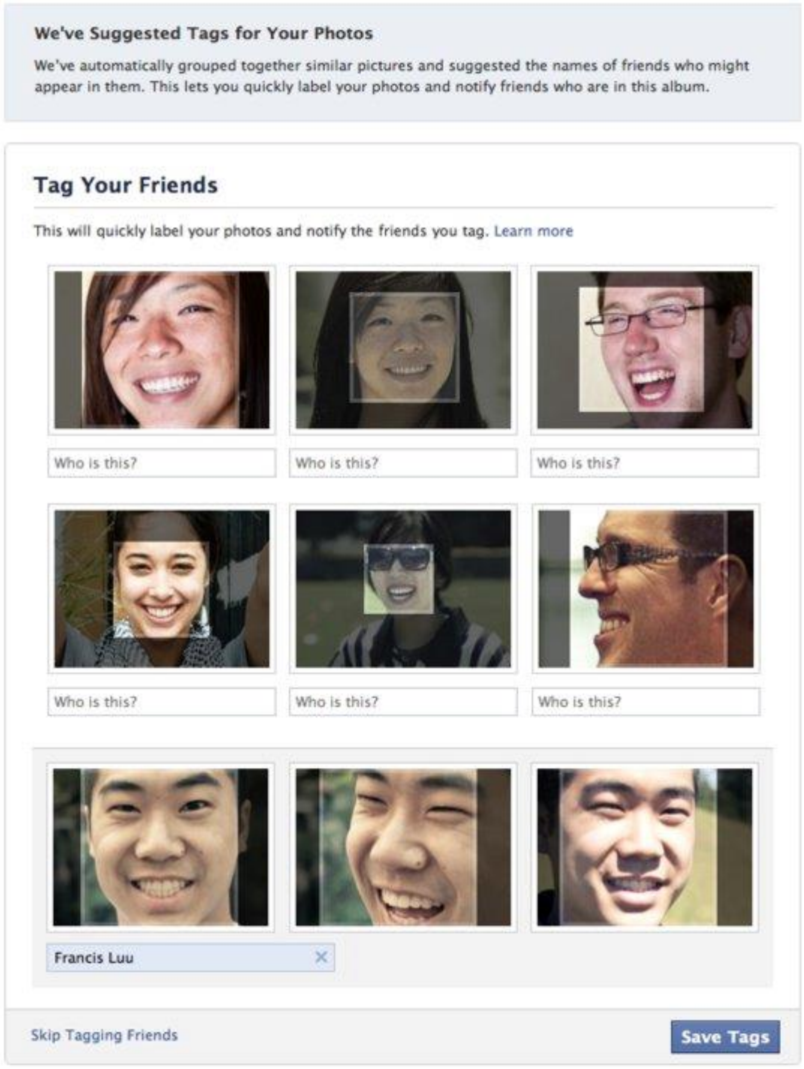
http://www.apple.com/ilife/iphoto/

- Can be trained to recognize pets!



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

Facebook friend-tagging with auto-suggest



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

Detection

Recognition

"Sally"

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)

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”

What makes face recognition hard?

- Expression

What makes face recognition hard?

- Lighting

What makes face recognition hard?

- Occlusion

What makes face recognition hard?

- Viewpoint

Simple idea for face recognition

1. Treat face image as a vector of intensities

$\rightarrow \mathbf{x}$

2. Recognize face by nearest neighbor in database

$\rightarrow \mathbf{y}_1 \ldots \mathbf{y}_n$

$$k = \underset{k}{\operatorname{argmin}} \left\| \mathbf{y}_k - \mathbf{x} \right\|$$

The space of all face images

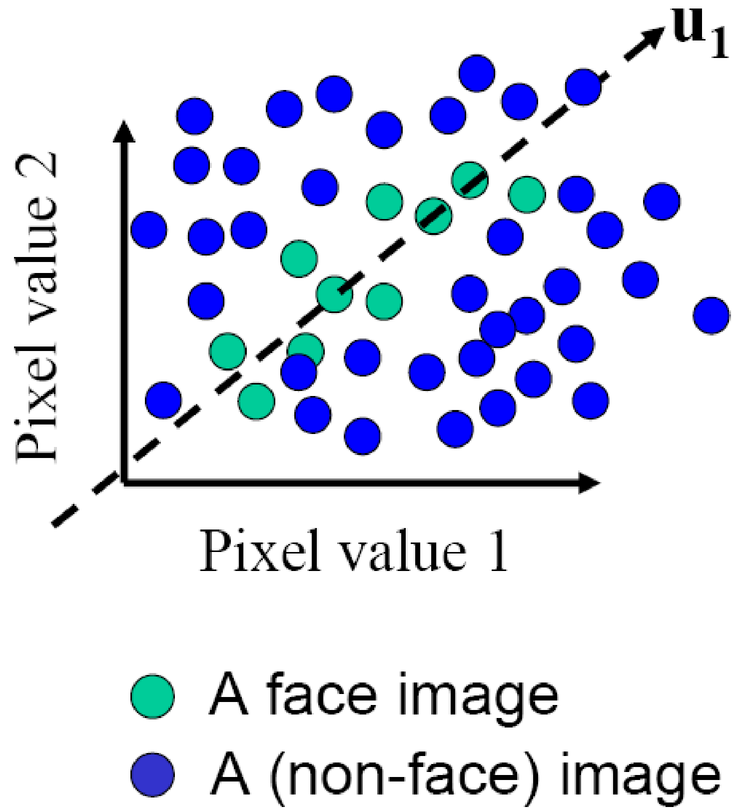
- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage

- But very few 10,000-dimensional vectors are valid face images

- We want to effectively model the subspace of face images

The space of all face images

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



Principal Component Analysis (PCA)

- Given: N data points $\boldsymbol{x_1}, \ldots, \boldsymbol{x_N}$ in R^d

- Goal: find a new set of features that are linear combinations of the original ones:

$$u(\mathbf{x}_i) = \mathbf{u}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

  ($\boldsymbol{\mu}$: mean of data points)

- Choose unit vector $\boldsymbol{u}$ in R^d that captures the most data variance

Forsyth & Ponce, Sec. 22.3.1, 22.3.2

Principal Component Analysis

- Direction that maximizes the variance of the projected data:

Maximize $$\frac{1}{N}\sum_{i=1}^{N} \underbrace{\mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu)}_{\text{Projection of data point}}(\mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu))^{\mathrm{T}}$$ subject to ||**u**||=1

$$= \mathbf{u}^{\mathrm{T}} \underbrace{\left[1/N \sum_{i=1}^{N} (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^{\mathrm{T}} \right]}_{\text{Covariance matrix of data}} \mathbf{u}$$

$$= \mathbf{u}^{\mathrm{T}} \Sigma \mathbf{u}$$

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of Σ (can be derived using Raleigh's quotient or Lagrange multiplier)

Implementation issue

- Covariance matrix is huge (M^2 for M pixels)
- But typically # examples << M
- Simple trick
  - **X** is MxN matrix of normalized training data
  - Solve for eigenvectors **u** of $\mathbf{X}^T\mathbf{X}$ instead of $\mathbf{X}\mathbf{X}^T$
  - Then **Xu** is eigenvector of covariance $\mathbf{X}\mathbf{X}^T$
  - Need to normalize each vector of **Xu** into unit length

Eigenfaces (PCA on face images)

1. Compute the principal components ("eigenfaces") of the covariance matrix

$$\boldsymbol{X} = [(\boldsymbol{x_1} - \boldsymbol{\mu}) \ \ (\boldsymbol{x_2} - \boldsymbol{\mu}) \ \ \ldots \ \ \ (\boldsymbol{x_n} - \boldsymbol{\mu})]$$
$$[\boldsymbol{U}, \boldsymbol{\lambda}] = \mathrm{eig}(\boldsymbol{X}^T \boldsymbol{X})$$
$$\boldsymbol{V} = \boldsymbol{X}\boldsymbol{U}$$

2. Keep K eigenvectors with largest eigenvalues

$$\boldsymbol{V} = \boldsymbol{V}(:, \mathrm{largest\_eig})$$

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

$$\boldsymbol{X_{pca}} = \boldsymbol{V}\left(:, \mathrm{largest_{eig}}\right)^{\boldsymbol{T}} \boldsymbol{X}$$

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

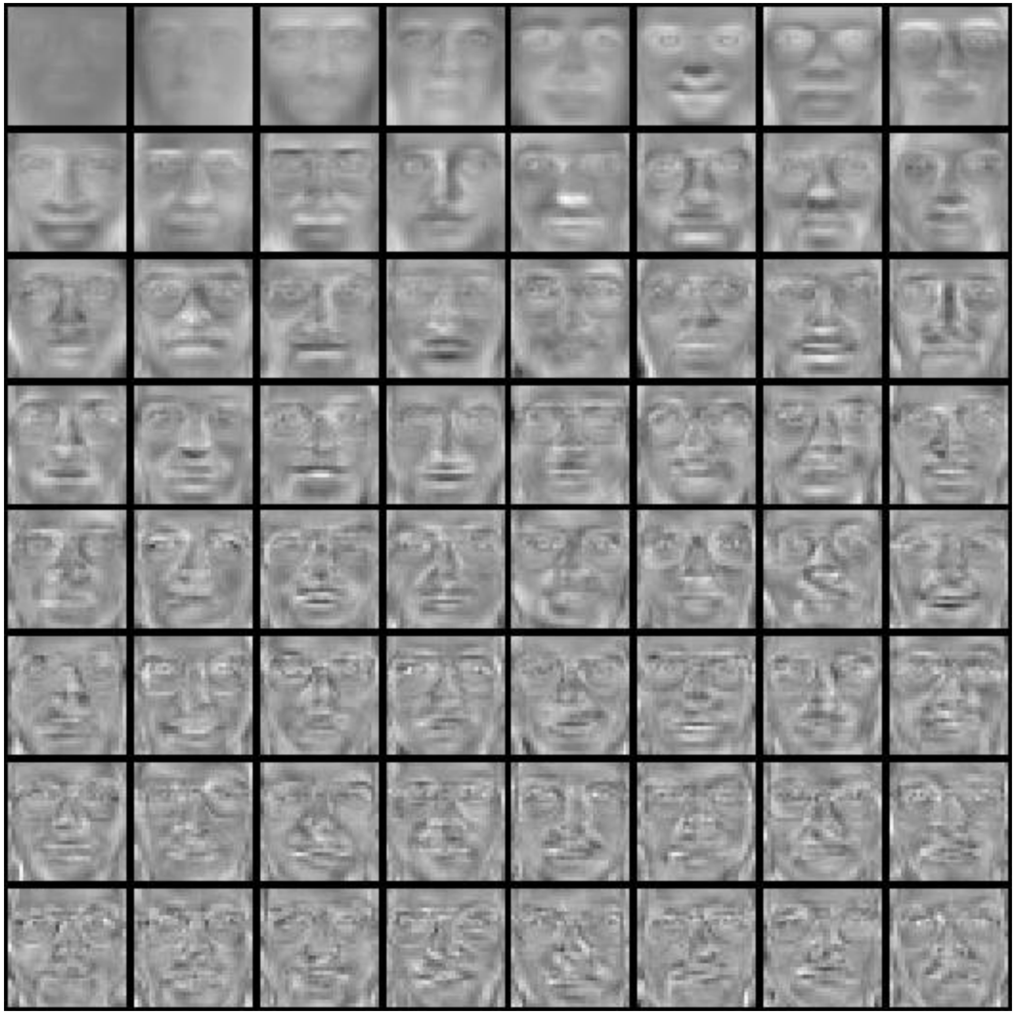
Eigenfaces example

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

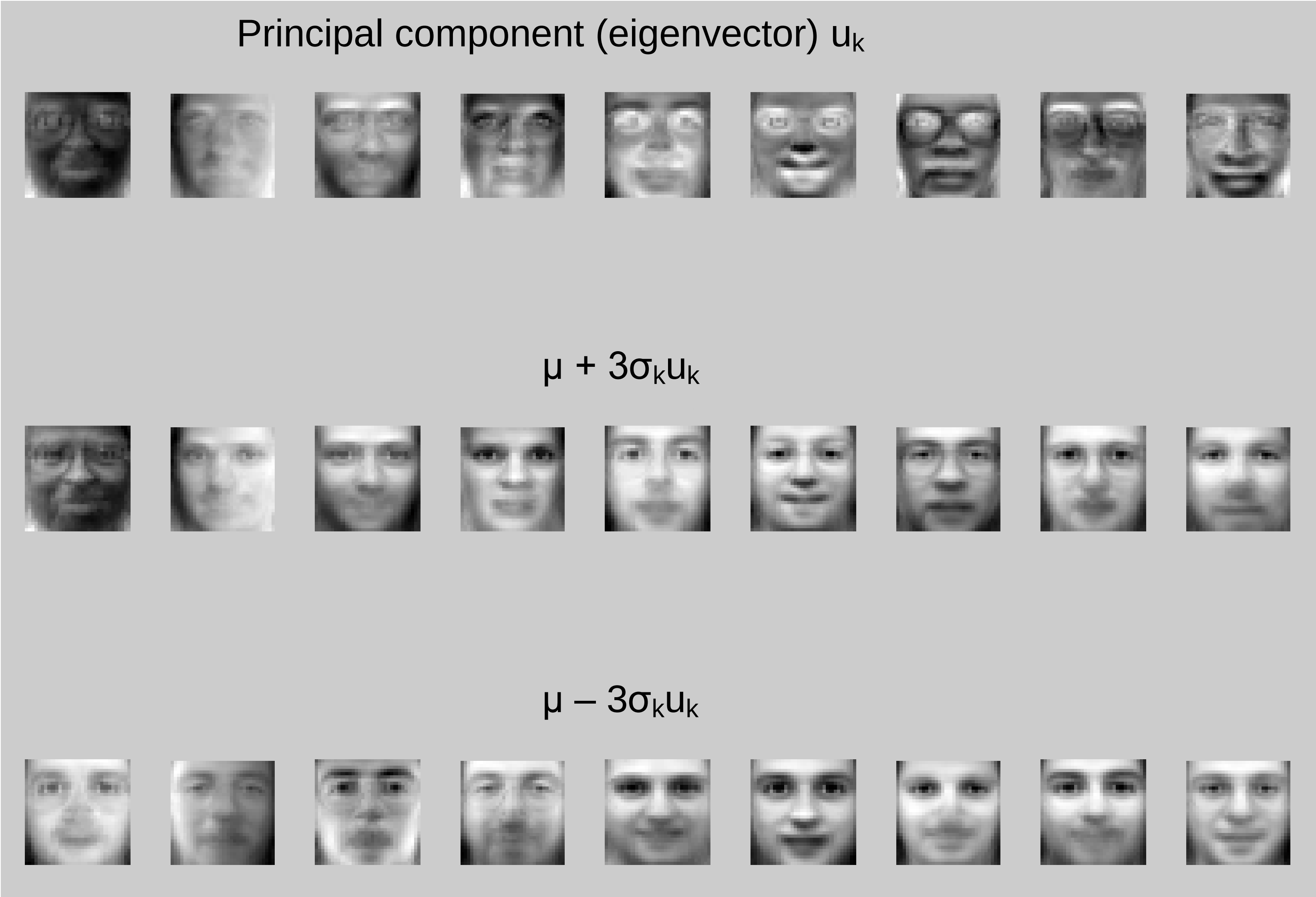
Eigenfaces example

Mean: μ

Top eigenvectors: $u_1, \ldots u_k$



Visualization of eigenfaces



Representation and reconstruction

- Face **x** in “face space” coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x} - \mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x} - \mu)]$$
$$= \quad w_1, \ldots, w_k$$

Representation and reconstruction

- Face **x** in “face space” coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$
$$= \quad w_1, \ldots, w_k$$

- Reconstruction:

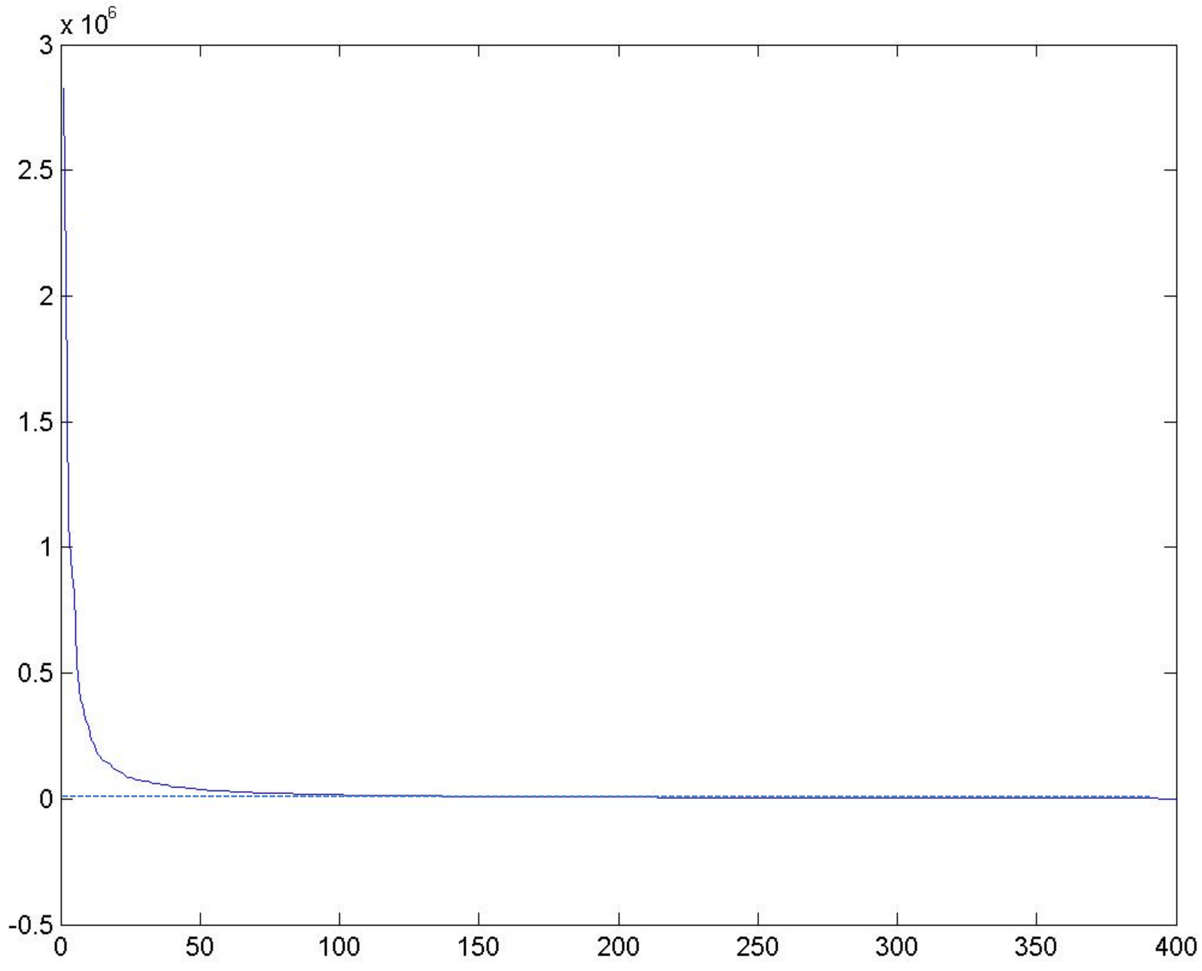
 =  + 

$\hat{x} \quad = \quad \mu \quad + \quad w_1u_1+w_2u_2+w_3u_3+w_4u_4+ \ldots$

Reconstruction

After computing eigenfaces using 400 face images from ORL face database

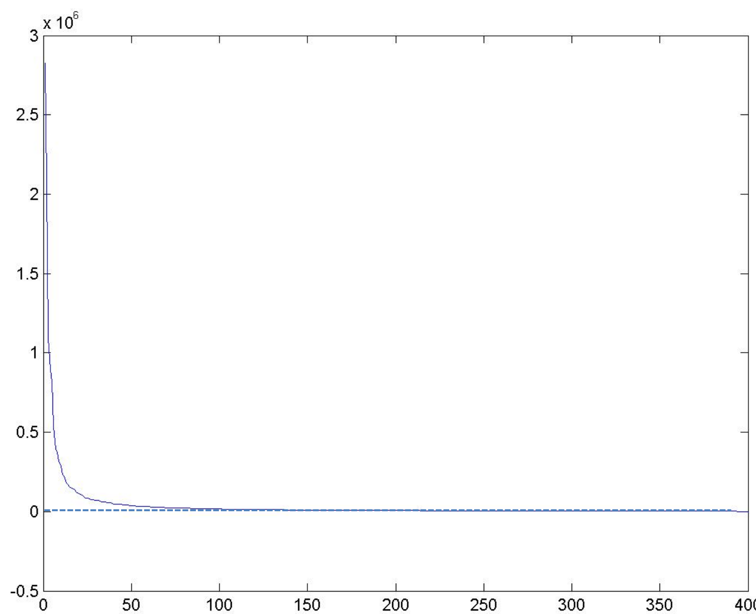
Eigenvalues (variance along eigenvectors)



Note

Preserving variance (minimizing MSE) does not necessarily lead to qualitatively good reconstruction.

P = 200



Recognition with eigenfaces

Process labeled training images

- Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{u}_1, \ldots \mathbf{u}_k$
- Project each training image $\mathbf{x}_i$ onto subspace spanned by principal components:
  $(w_{i1}, \ldots, w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x}_i - \boldsymbol{\mu}))$

Given novel image $\mathbf{x}$

- Project onto subspace:
  $(w_1, \ldots, w_k) = (\mathbf{u}_1^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu}))$
- 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

PCA

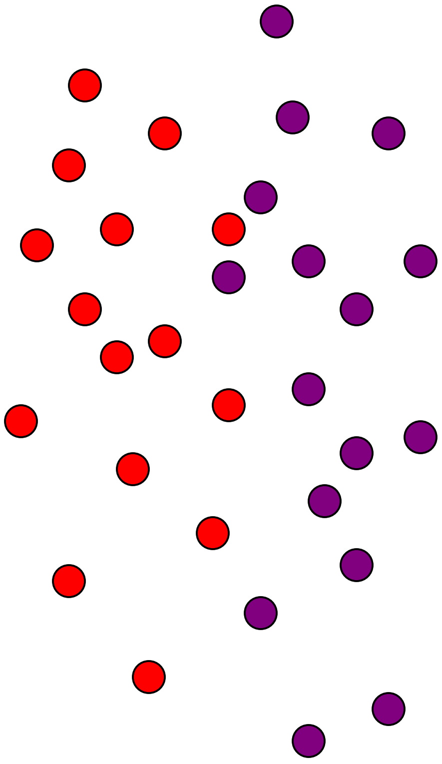
- General dimensionality reduction technique
- Preserves most of variance with a much more compact representation
  - Lower storage requirements (eigenvectors + a few numbers per face)
  - Faster matching
- What are the problems for face recognition?

Limitations

Global appearance method: not robust to misalignment, background variation

Limitations

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



A more discriminative subspace: FLD

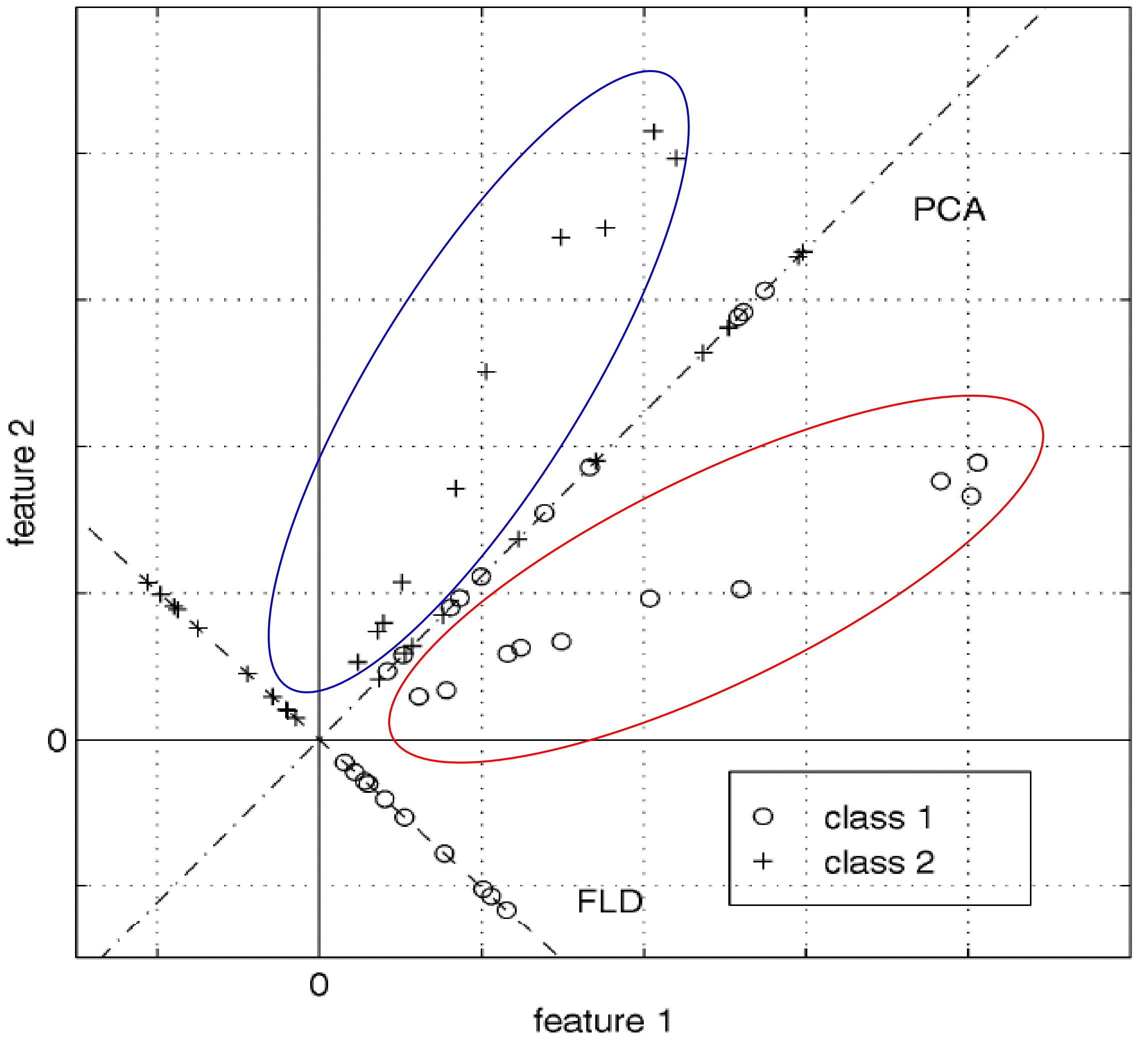
- Fisher Linear Discriminants → "Fisher Faces"

- 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]

Comparing with PCA



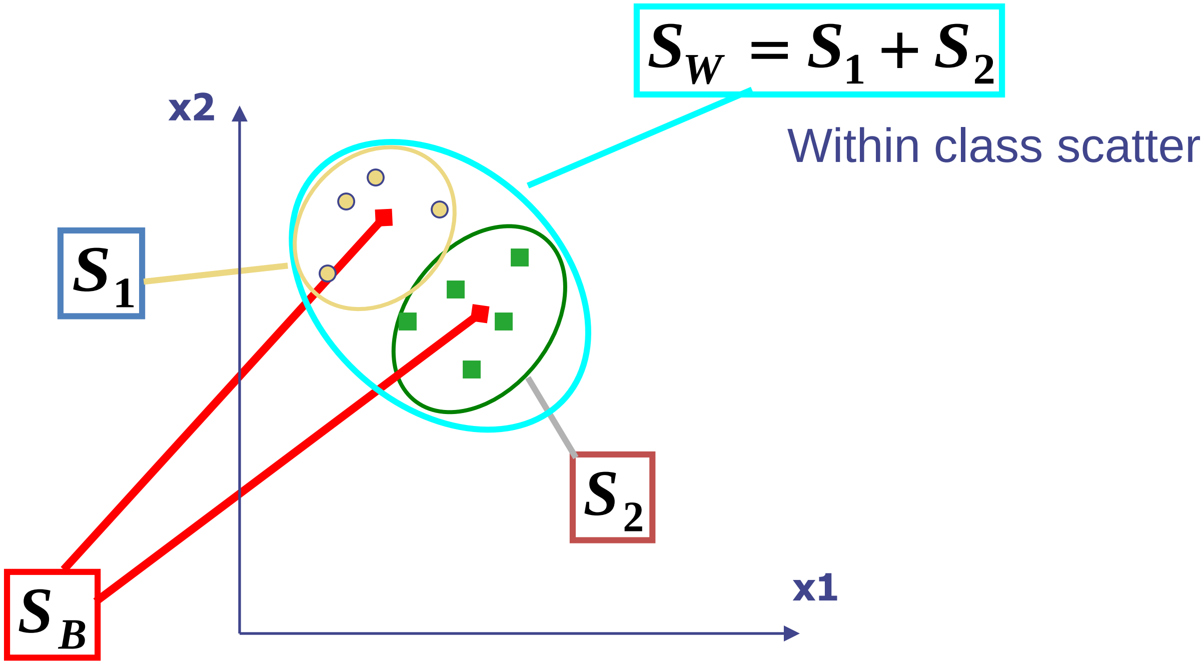
Variables

- 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: $\mu = \frac{1}{N} \sum_{k=1}^{N} x_k$

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

Illustration



Between class scatter

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 $\boldsymbol{W_{opt}}$ is limited
  - Rank(S_B) <= |C|-1
  - Rank(S_W) <= N-C

Recognition with FLD

- Use PCA to reduce dimensions to N-C

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

- 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

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

- Project to FLD subspace (c-1 dimensions)

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

- Classify by nearest neighbor

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

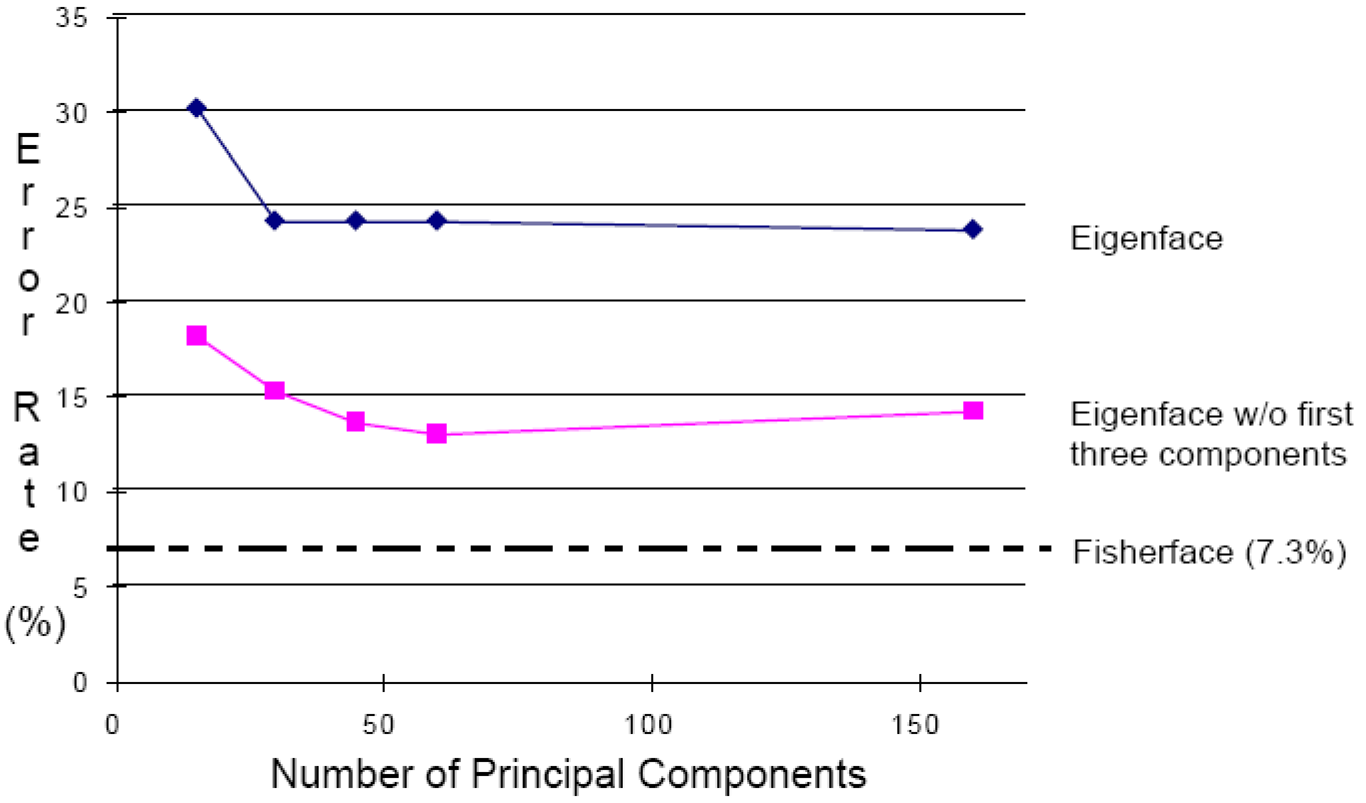
Results: Eigenface vs. Fisherface

- 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

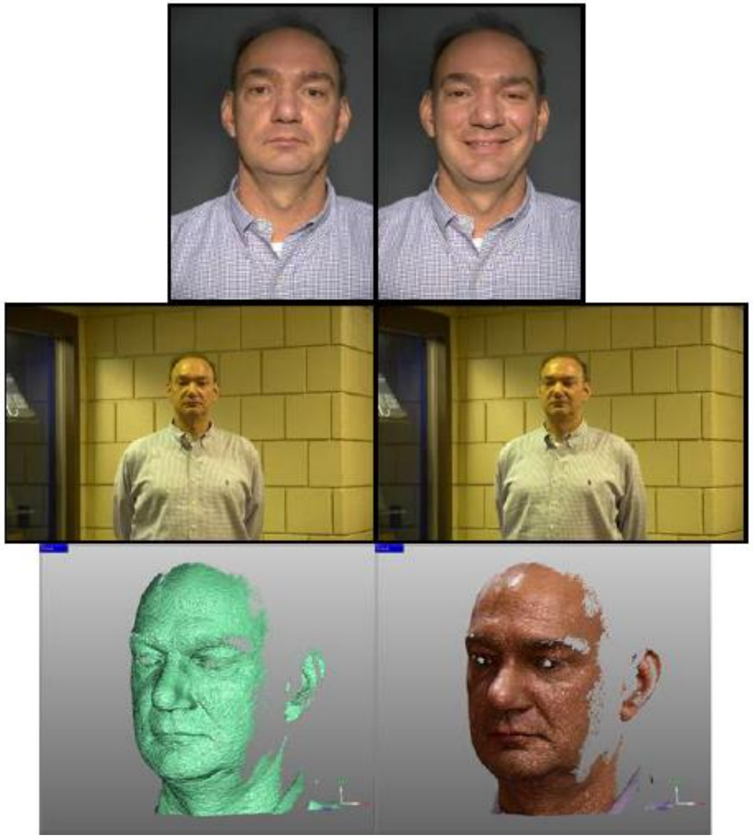


Reference: [Eigenfaces vs. Fisherfaces, Belheumer et al., PAMI 1997]

1997 till today, what has changed?

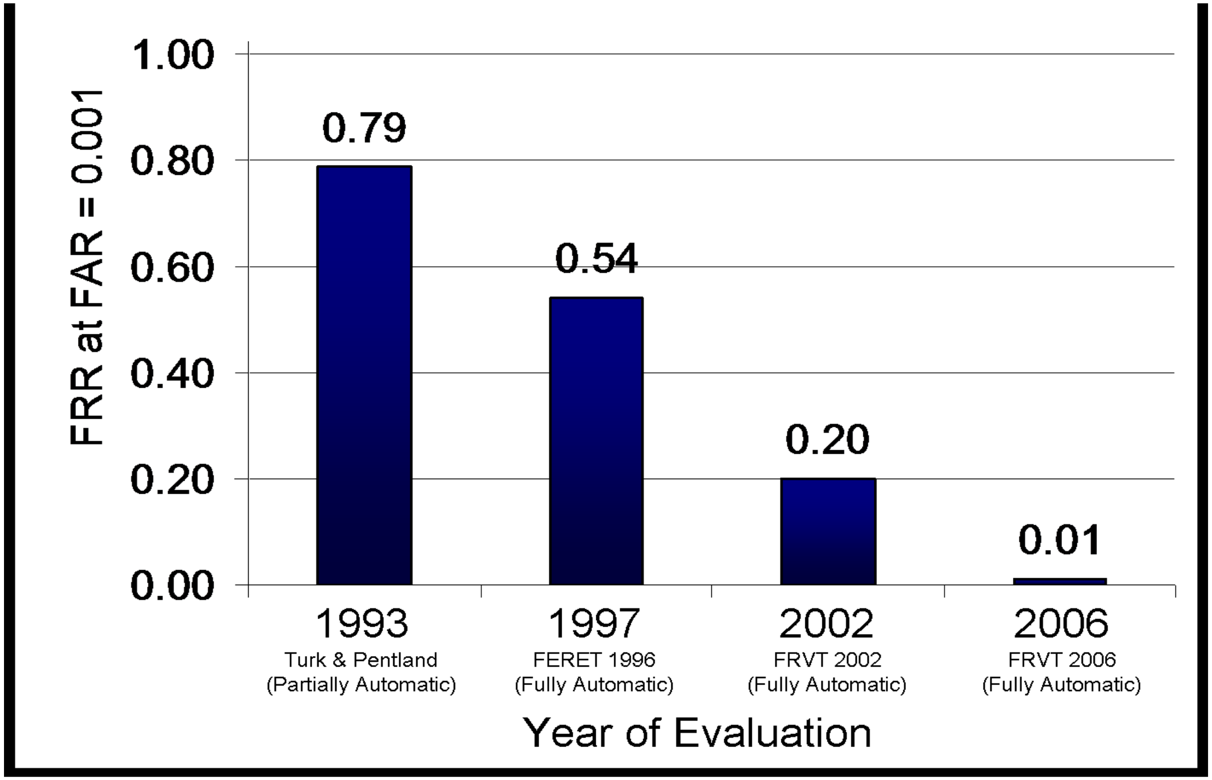
- 2006:
  Face Recognition Vendor Test 2006
  http://www.nist.gov/itl/iad/ig/frvt-2006.cfm

- Controlled Data



1997 till today, what has changed?

- 2006:
  Face Recognition Vendor Test 2006
  http://www.nist.gov/itl/iad/ig/frvt-2006.cfm



State-of-the-art Face Recognizers

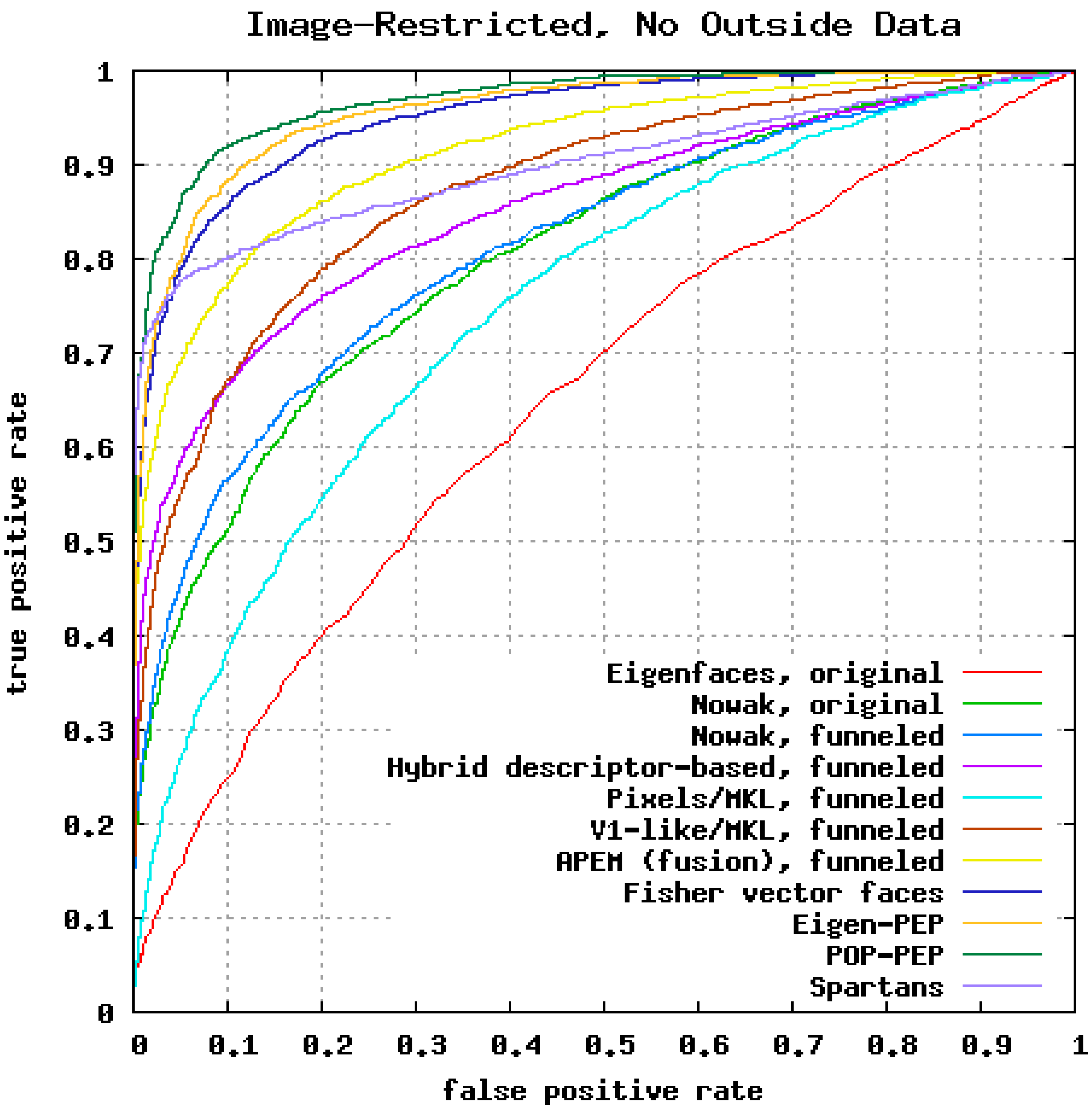
- Most recent research focuses on “faces in the wild”, recognizing faces in normal photos
  - Classification: assign identity to face
  - Verification: say whether two people are the same

- Important steps
  1. Detect
  2. Align
  3. Represent
  4. Classify

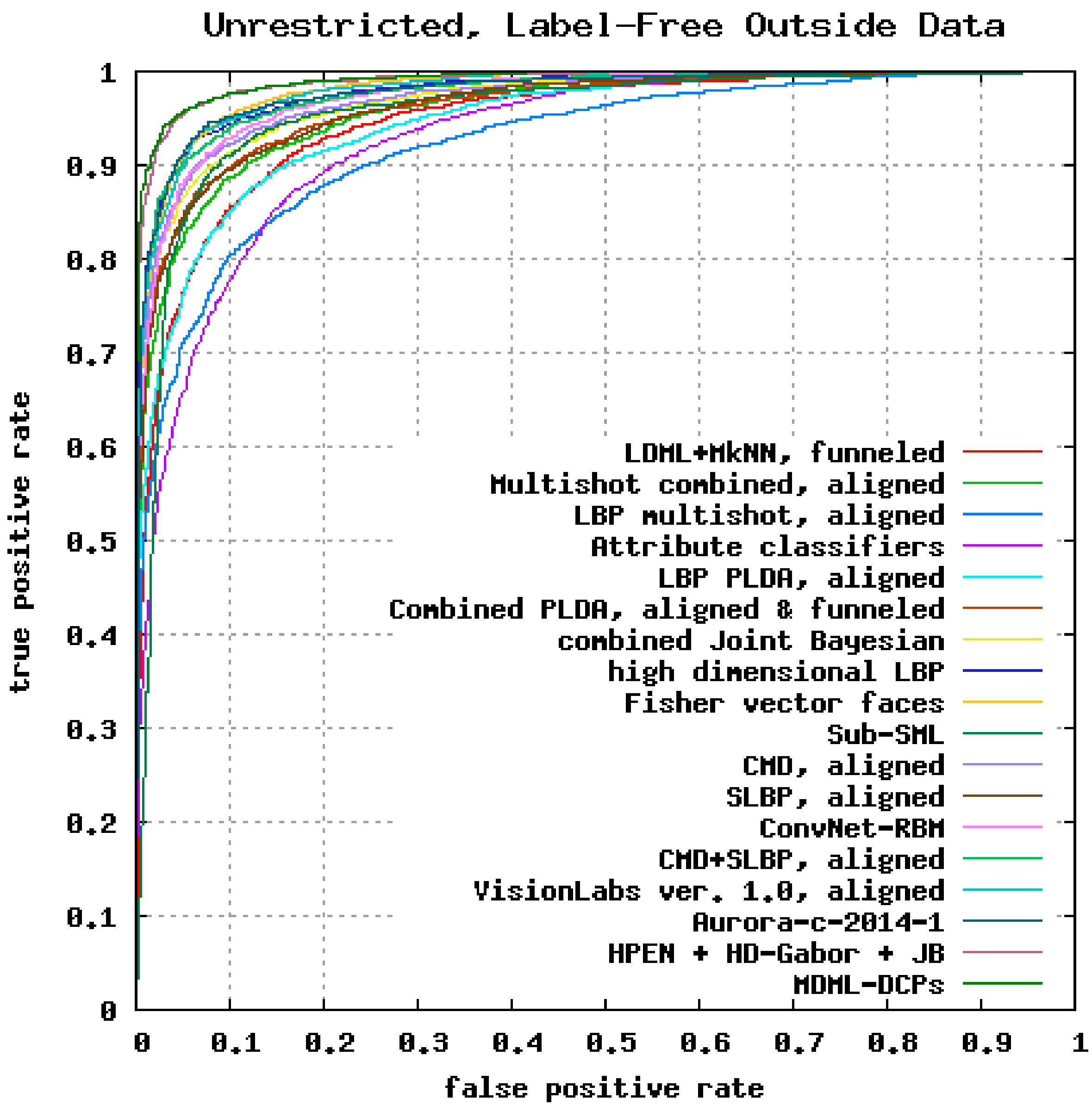
Labeled Faces in the Wild (LFW)

- http://vis-www.cs.umass.edu/lfw/

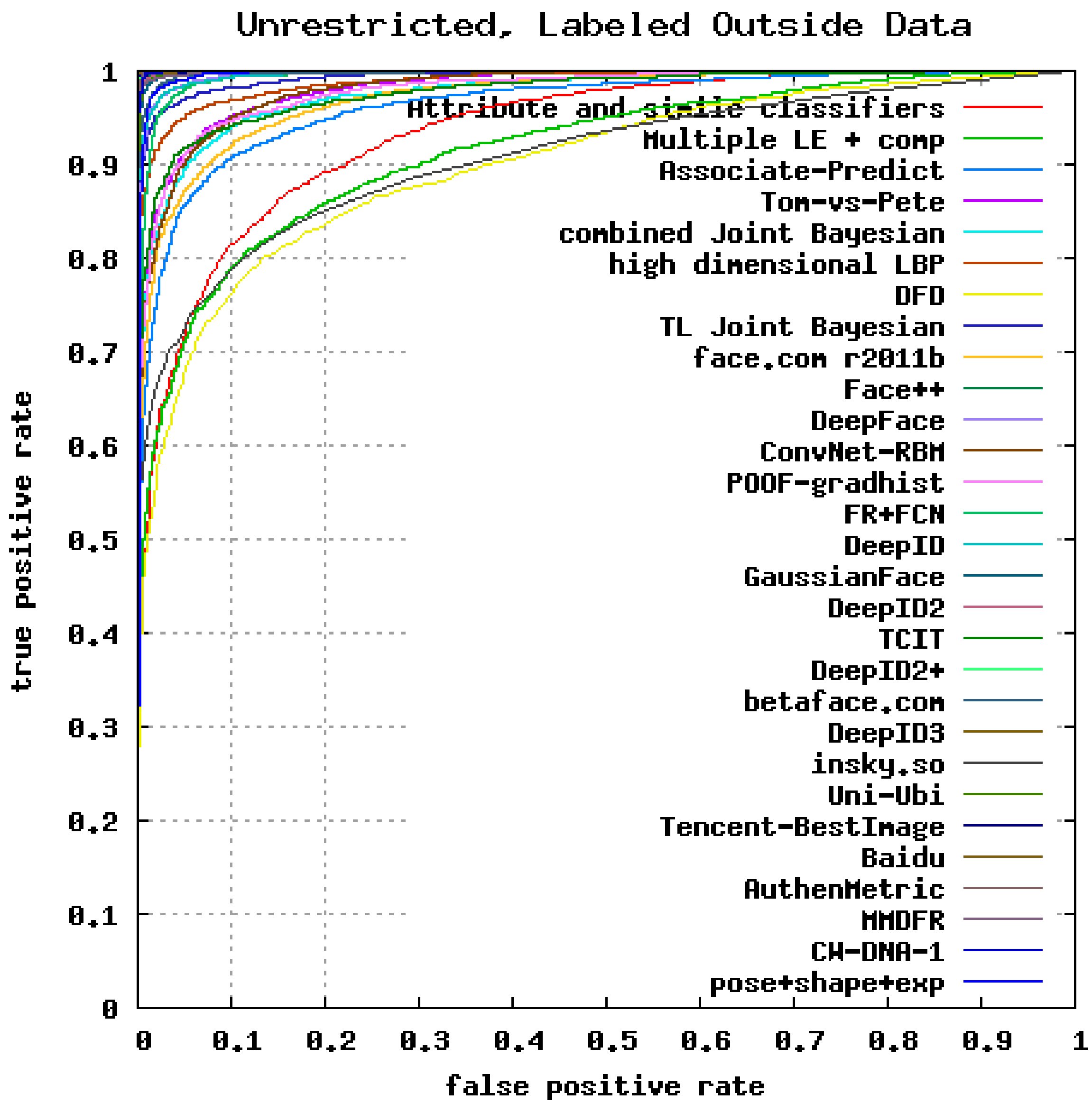
Labeled Faces in the Wild (LFW)



Labeled Faces in the Wild (LFW)



Labeled Faces in the Wild (LFW)

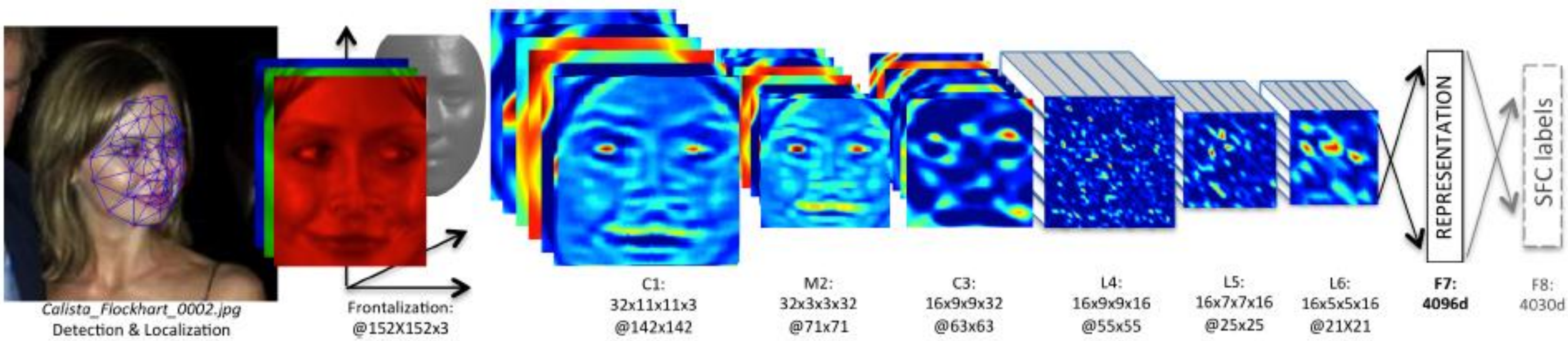


DeepFace: Closing the Gap to Human-Level Performance in Face Verification

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

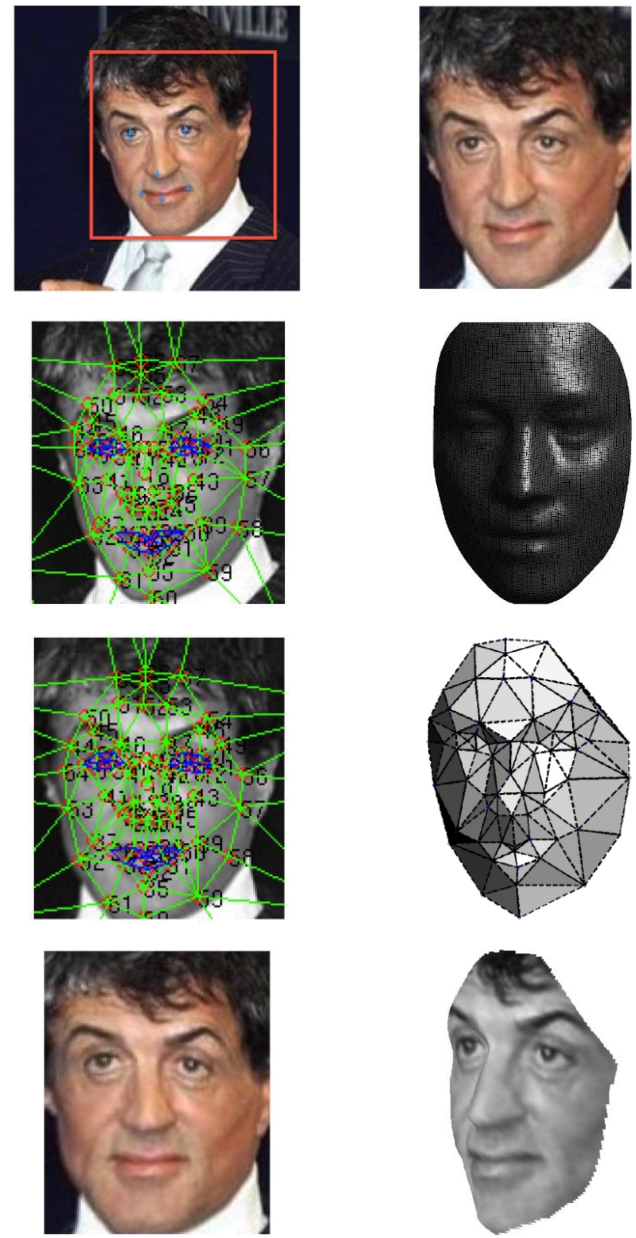
1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)

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

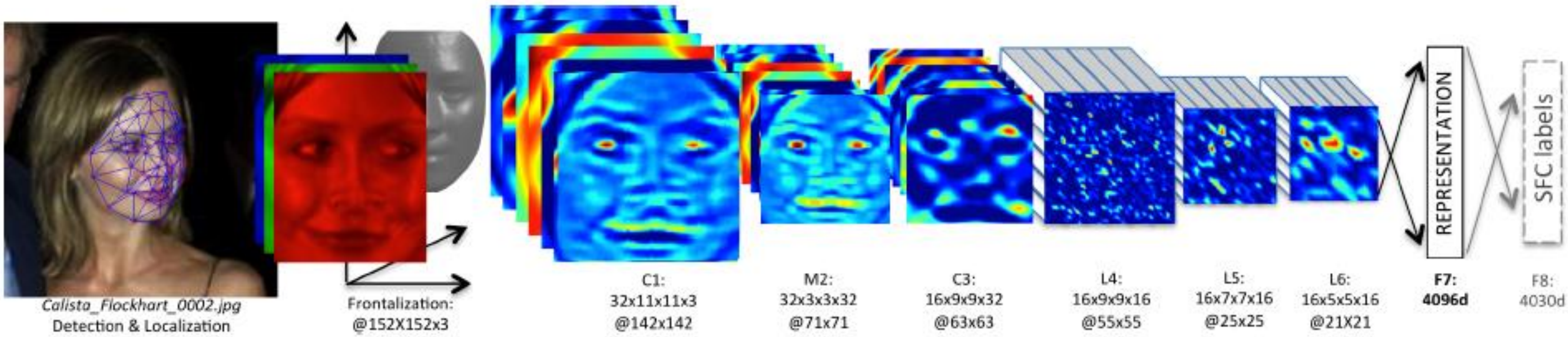
3. Localize 67 fiducial points in the 2D aligned crop

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



Train DNN classifier on aligned faces



Architecture (deep neural network classifier)

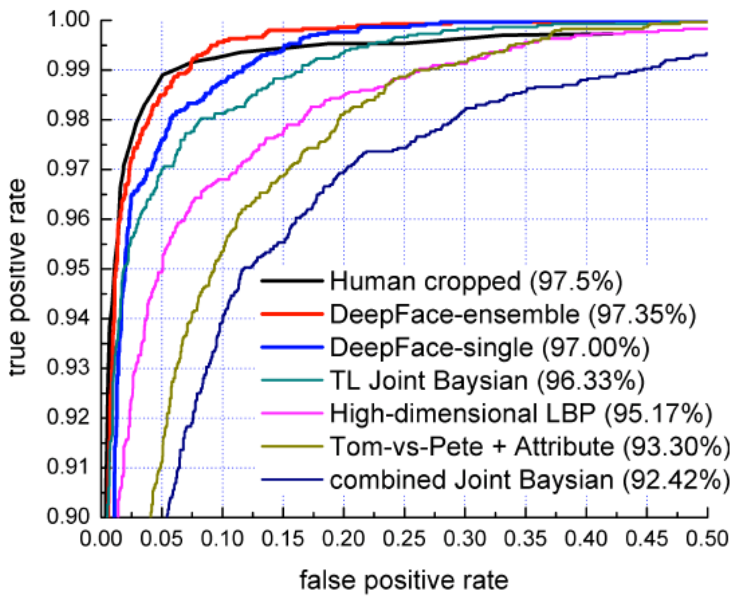
- Two convolutional layers (with one pooling layer)
- 3 locally connected and 2 fully connected layers
- > 120 million parameters

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)

Face recognition by humans

Face Recognition by Humans: Nineteen Results All Computer Vision Researchers Should Know About

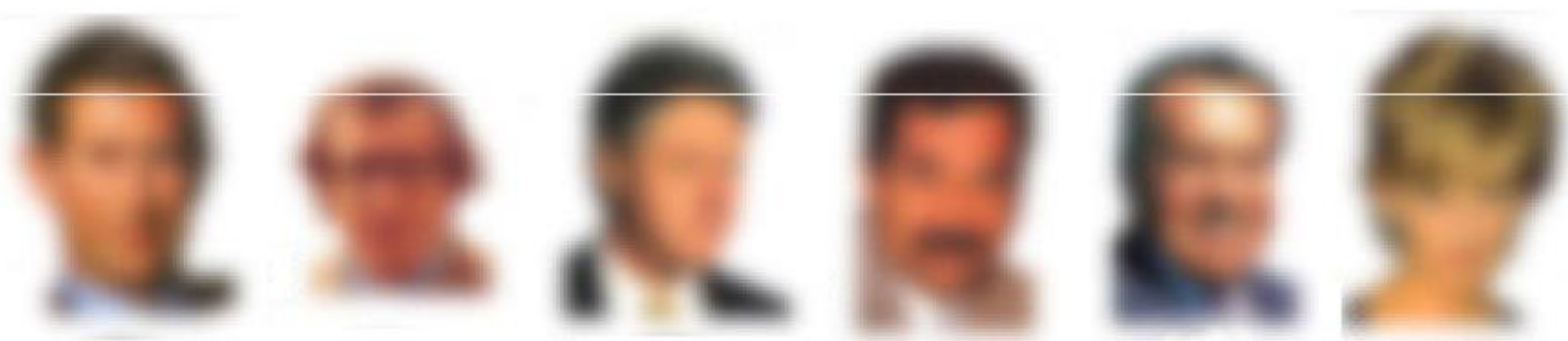
By Pawan Sinha, Benjamin Balas, Yuri Ostrovsky, and Richard Russell

Proc. IEEE, 2006

http://web.mit.edu/sinhalab/Papers/19results_sinha_etal.pdf

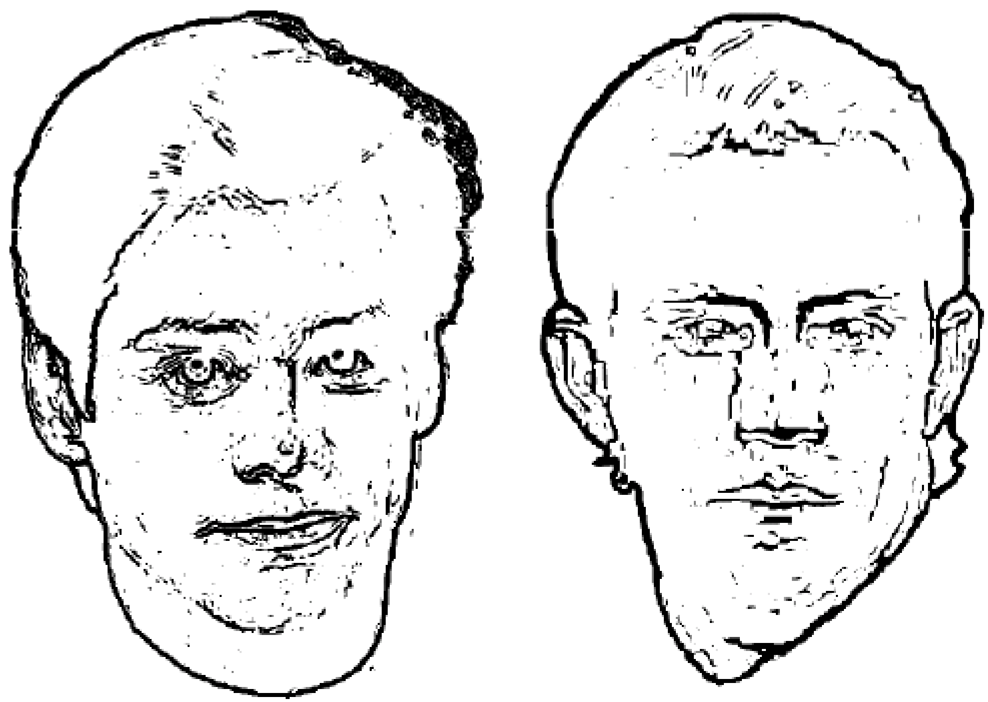
Result 1

- Humans can recognize faces in extremely low resolution images.



Result 3

- High-frequency information by itself does not lead to good face recognition performance



Result 4

- Facial features are processed holistically

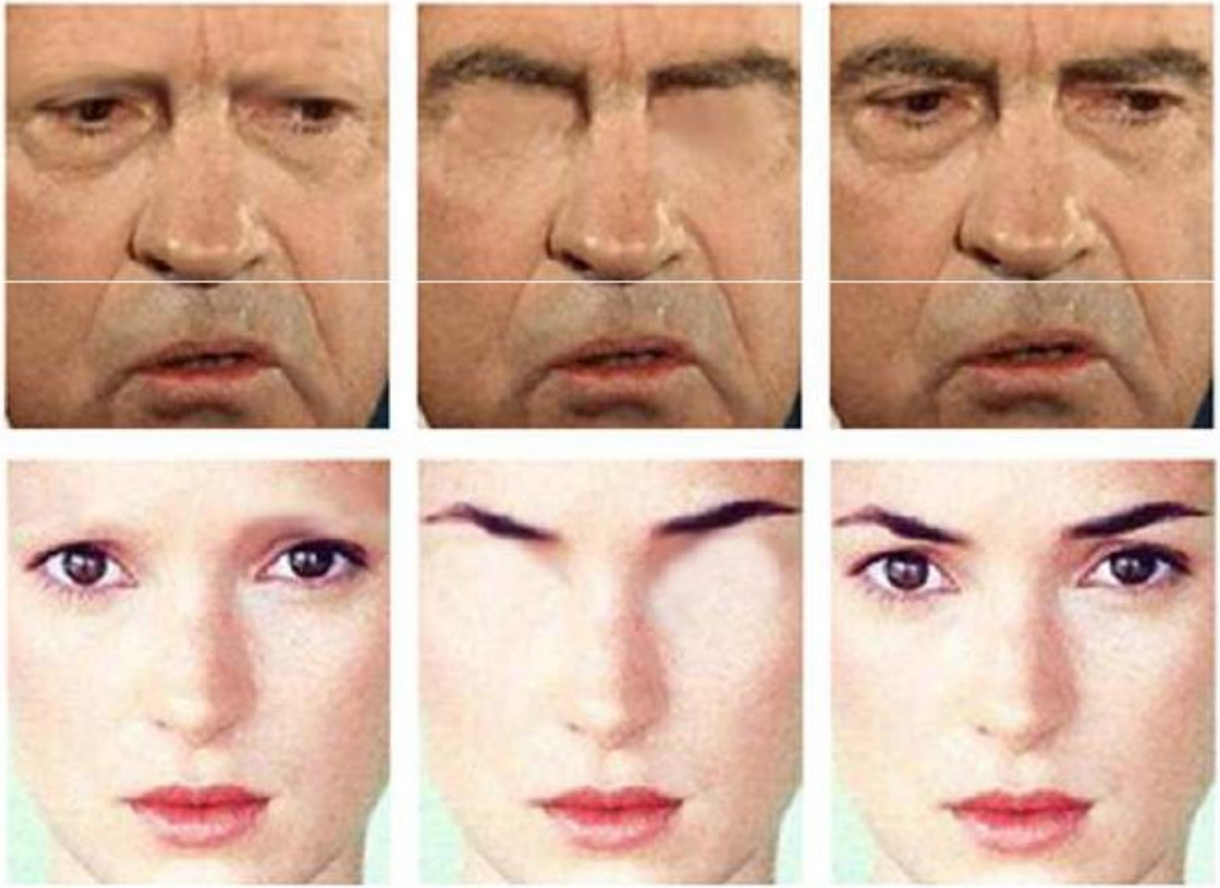
Who's in the picture?

hard

Easier

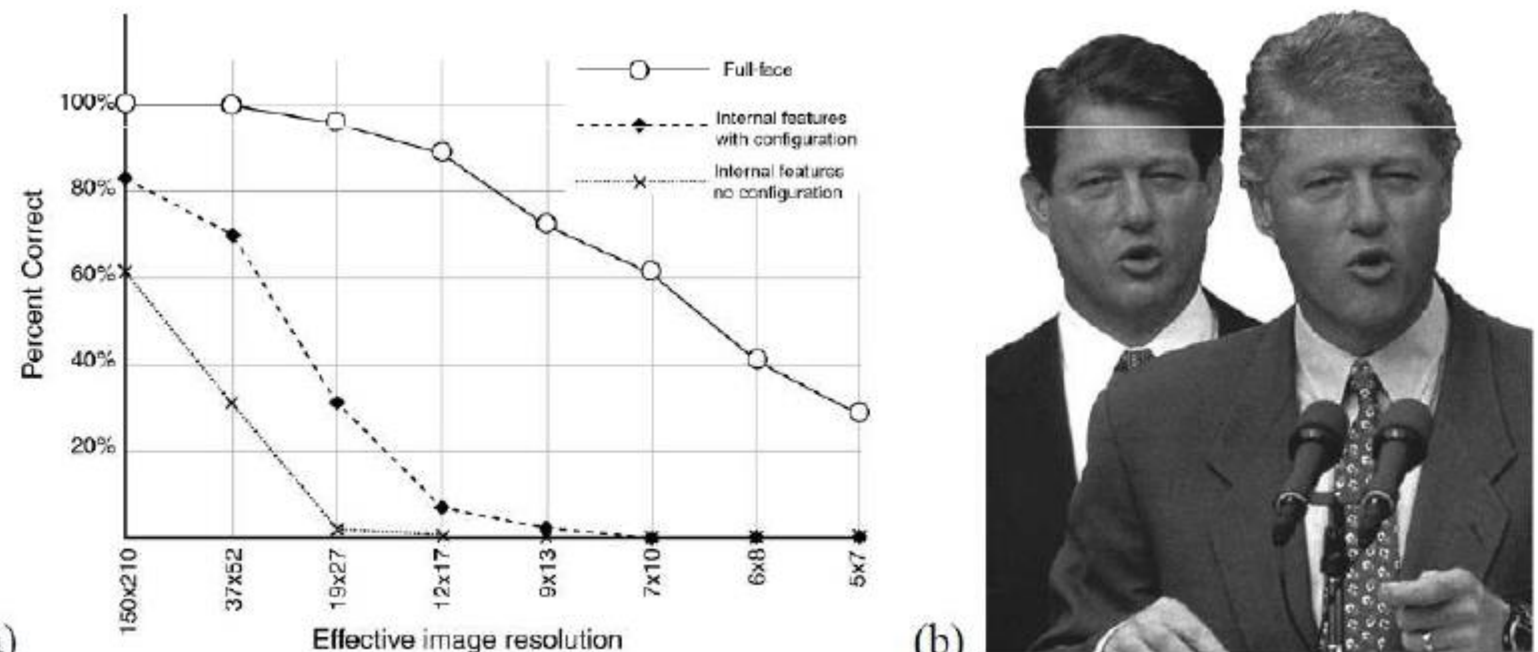
Result 5

- Eyebrows are among the most important for recognition



Result 6

- Both internal and external facial cues are important and they exhibit non-linear interactions



(a) (b)

Result 7

- The important configural relations appear to be independent across the width and height dimensions

Result 8

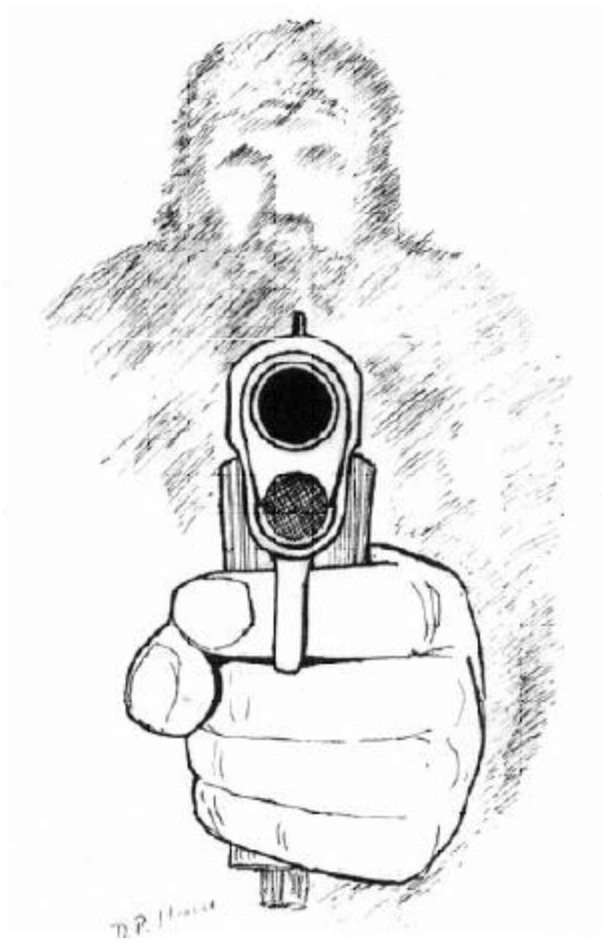
- Vertical inversion dramatically reduces recognition performance

Result 12

- Contrast polarity inversion dramatically impairs recognition performance, possibly due to compromised ability to use pigmentation cues

Result 20

- Human memory for briefly seen faces is rather poor



Things to remember

- PCA is a generally useful dimensionality reduction technique
  - But not ideal for discrimination

- FLD better for discrimination, though only ideal under Gaussian data assumptions

- Computer face recognition works very well under controlled environments (since 2006)

- Also starting to perform at human level in uncontrolled settings (recent progress: better alignment, features, more data)