Face Recognition

The “Margaret Thatcher Illusion”, by Peter Thompson

Read 14.2
Face Recognition

The "Margaret Thatcher Illusion", by Peter Thompson
Lab2
Lab3

• Use online storage.
• Send only the link via email.
• Make sure that the file is “shared”.
• Check your emails regularly.
Face detection and recognition

Detection

Recognition

“Sally”
Applications of Face Recognition

- Digital photography
- Surveillance
Applications of Face Recognition

- Digital photography
- Surveillance
- Album organization
Starting idea of “eigenfaces”

1. Treat pixels as a vector
Starting idea of “eigenfaces”

1. Treat pixels as a vector

\[ x \rightarrow y_1 \cdots y_n \]
Starting idea of “eigenfaces”

1. Treat pixels as a vector

2. Recognize face by nearest neighbor

\[ k = \underset{k}{\text{argmin}} \| y_k^T x - x \| \]
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
Classification can be expensive

- Must either search (e.g., nearest neighbors) or store large PDF’s

Suppose the data points are arranged as above

- Idea—fit a line, classifier measures distance to line

Convert $x$ into $v_1, v_2$ coordinates

$$x \rightarrow ((x - \bar{x}) \cdot v_1, (x - \bar{x}) \cdot v_2)$$

What does the $v_2$ coordinate measure?
- distance to line
- use it for classification—near 0 for orange pts

What does the $v_1$ coordinate measure?
- position along line
- use it to specify which orange point it is

$\bar{x}$ is the mean of the orange points
Principal Component Analysis

Dimensionality reduction

- We can represent the orange points with *only* their $v_1$ coordinates
  - since $v_2$ coordinates are all essentially 0
- This makes it much cheaper to store and compare points
- A bigger deal for higher dimensional problems
Consider the variation along direction $\mathbf{v}$ among all of the orange points:

$$\text{var}(\mathbf{v}) = \sum_{\text{orange point } x} \| (x - \bar{x})^T \cdot \mathbf{v} \|^2$$

What unit vector $\mathbf{v}$ minimizes $\text{var}$?

$$\mathbf{v}_2 = \min_{\mathbf{v}} \{ \text{var}(\mathbf{v}) \}$$

What unit vector $\mathbf{v}$ maximizes $\text{var}$?

$$\mathbf{v}_1 = \max_{\mathbf{v}} \{ \text{var}(\mathbf{v}) \}$$

Solution: $\mathbf{v}_1$ is eigenvector of $\mathbf{A}$ with largest eigenvalue

$\mathbf{v}_2$ is eigenvector of $\mathbf{A}$ with smallest eigenvalue
Principal component analysis

Suppose each data point is N-dimensional

• Same procedure applies:

\[
\text{var}(v) = \sum_x \| (x - \bar{x})^T \cdot v \|
\]

\[
= v^T A v \quad \text{where} \quad A = \sum_x (x - \bar{x})(x - \bar{x})^T
\]
Principal component analysis

Suppose each data point is N-dimensional

• Same procedure applies:

\[ \text{var}(v) = \sum_x \| (x - \bar{x})^T \cdot v \| \]

\[ = v^T A v \quad \text{where} \quad A = \sum_x (x - \bar{x})(x - \bar{x})^T \]

• The eigenvectors of \( A \) define a new coordinate system
  – eigenvector with largest eigenvalue captures the most variation among training vectors \( x \)
  – eigenvector with smallest eigenvalue has least variation
Principal component analysis

• Compress data by the top few eigenvectors
  – Choosing a “linear subspace”
  – Eigenvectors known as the principal components

\[\overline{x}\] is the mean of the orange points
Implementation issue

• Covariance matrix is huge \((N^2 \text{ for } N \text{ pixels})\)

\[
\text{var}(v) = \sum_x \|(x - \bar{x})^T \cdot v\| = v^T A v \quad \text{where } A = \sum_x (x - \bar{x})(x - \bar{x})^T
\]

• But typically \# examples \(<\ll N\)

• Simple trick
  – \(X\) is matrix of normalized training data
  – Solve for eigenvectors \(u\) of \(XX^T\) instead of \(X^TX\)
  – Then \(X^Tu\) is eigenvector of covariance \(X^TX\)
  – May need to normalize (to get unit length vector)
Eigenfaces (PCA on face images)

1. Compute covariance matrix of face images

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

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

1. Compute covariance matrix of face images

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

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

Eigenfaces example

- Training images
- $x_1, \ldots, x_N$
Eigenfaces example

Top eigenvectors: $u_1, \ldots, u_k$

Mean: $\mu$
Visualization of eigenfaces

Principal component (eigenvector) $u_k$
Visualization of eigenfaces

Principal component (eigenvector) $u_k$

$\mu + 3\sigma_k u_k$
Visualization of eigenfaces

Principal component (eigenvector) $u_k$

$\mu + 3\sigma_k u_k$

$\mu - 3\sigma_k u_k$
Representation and reconstruction

• Face $\mathbf{x}$ in “face space” coordinates:

\[
x \rightarrow \begin{bmatrix} u_1^T (x - \mu), \ldots, u_k^T (x - \mu) \end{bmatrix} = w_1, \ldots, w_k
\]
Representation and reconstruction

• Face $x$ in “face space” coordinates:

$$x \rightarrow [u_1^T (x - \mu), \ldots, u_k^T (x - \mu)]$$

$$= w_1, \ldots, w_k$$

• Reconstruction:

$$\hat{x} = \mu + w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + \ldots$$
Reconstruction

After computing eigenfaces using 400 face images from ORL face database
Reconstruction

After computing eigenfaces using 400 face images from ORL face database
Eigenvalues (variance along eigenvectors)
Preserving variance (minimizing MSE) does not necessarily lead to qualitatively good reconstruction.

$P = 200$
Recognition with eigenfaces

Process labeled training images

• Find mean $\mu$ and covariance matrix $\Sigma$
• Find $k$ principal components (eigenvectors of $\Sigma$) $u_1, \ldots, u_k$
• Project each training image $x_i$ onto subspace spanned by principal components:
  
  $$(w_{i1}, \ldots, w_{ik}) = (u_1^T(x_i - \mu), \ldots, u_k^T(x_i - \mu))$$

Given novel image $x$

• Project onto subspace:
  
  $$(w_1, \ldots, w_k) = (u_1^T(x - \mu), \ldots, u_k^T(x - \mu))$$

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

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 $\rightarrow$ “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, Belhumer et al., PAMI 1997
Illustration of the Projection

- Using two classes as example:

Poor Projection

Good
Comparing with PCA
Variables

• N Sample images: \( \{x_1, \ldots, x_N\} \)

• c classes: \( \{\chi_1, \ldots, \chi_c\} \)

• 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 $i$: $S_i = \sum_{x_k \in C_i} (x_k - \mu_i)(x_k - \mu_i)^T$

- Within class scatter: $S_W = \sum_{i=1}^{c} S_i$

- Between class scatter: $S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$
• 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:
  \[ S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T \]

\[ S_W = S_1 + S_2 \]
Mathematical Formulation

• After projection
  – Between class scatter
  – Within class scatter

• Objective

\[
y_k = W^T x_k \\
\tilde{S}_B = W^T S_B W \\
\tilde{S}_W = W^T S_W W
\]

\[
W_{opt} = \arg \max_W \frac{\tilde{S}_B}{\tilde{S}_W} = \arg \max_W \frac{W^T S_B W}{W^T S_W W}
\]

• Solution: Generalized Eigenvectors

\[
S_B w_i = \lambda_i S_W w_i \quad i = 1, K, m
\]
Recognition with FLD

• Similar to “eigenfaces”

• Compute within-class and between-class scatter matrices

\[ s_i = \sum_{x_k \in \mathcal{X}_i} (x_k - \mu_i)(x_k - \mu_i)^T \quad s_W = \sum_{i=1}^{c} s_i \quad s_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T \]
Recognition with FLD

- Similar to “eigenfaces”

- Compute within-class and between-class scatter matrices

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

- Solve generalized eigenvector problem

\[ W_{opt} = \arg \max_w \frac{|W^T S_B W|}{|W^T S_W W|} \quad S_B w_i = \lambda_i S_W w_i \quad i = 1, K, m \]

- Project to FLD subspace and classify by nearest neighbor

\[ \hat{x} = W_{opt}^T x \]
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
Face recognition by humans

Face recognition by humans: 20 results
Result 1

- Humans can recognize faces in extremely low resolution images.
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 3

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

Jim Carrey and Kevin Costner.
Result 5

- Eyebrows are among the most important for recognition
Result 5

- Eyebrows are among the most important for recognition
Result 5

- Eyebrows are among the most important for recognition
Eyebrows are among the most important for recognition

Richard M. Nixon and Winona Ryder
Result 6

- Both internal and external facial cues are important and they exhibit non-linear interactions.
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
Human memory for briefly seen faces is rather poor
The list goes on

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

Things to remember

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

• FLD better for discrimination

• Computer face recognition works very well under controlled environments – still room for improvement in general conditions (but deep learning will solve this...)
Project 3
• Compute eigenfaces (eigen vectors)
• Project a face to eigen-space to get coeffs
• Project faces to eigen-space
• Is this a face?: Project to eigen-space, reconstruct, and see the difference from original
• Verify user: Compare coeffs after projection
• Recognition: Find the image in the database with the closest coeffs
• Find a face: Sliding window. Project, reconstruct, see the difference. Find peaks.
Face class

- Face -> Image -> Vector
- A public member function of Vector is also a member function of Face...
Face class

- Vector v0(20), v1(20), v2(20);
  init(v0, v1);
  v2 = v0 + v1;
  v0.add(v1, v2); // same as above
  v2 = v0 - v1;
  v0.sub(v1, v2); // same as above
  double d0 = v0.dot(v1); // Dot product.
  double d1 = v0.mse(v1); // mse
Face face0, face1, sum, diff;
init(face0, face1, sum, diff);
sum = face0 + face1;
face0.add(face1, sum);
diff = face0 - face1;
face0.sub(face1, diff);
sum = 0.3 * face0 + 0.7 * face1;
double d0 = face0.dot(face1);
double d1 = face0.mse(face1);
Resize a face

- Face face(300, 200, 1);
  init(face);
  Face rescaled_face(75, 50, 1);  // a quarter
  face.resample(rescaled_face);
  // rescaled_face is a smaller version of face!
Drawing boxes after face detection

- Image image;
  init(image);
  image.line(50, 80, 90, 120, 100, 255, 100);
  // Draw a box whose top-left corner is at (50, 80), bottom-right corner is at (90, 120), with a color (100, 255, 100).
Many many utility functions...

- Use them wisely.
Standard Template Library
http://www.sgi.com/tech/stl/stl_index.html

• vector
• list
• set
• map
• ...

vector

- vector<double> v;
  v.resize(10);
  cout << v.size() << endl;  // 10
  v.clear();
  cout << v.size() << endl;  // 0
  v.push_back(2.3);
  v.push_back(4.7);
  cout << v.size() << endl;  // 2
  cout << v[0] << endl;      // 2.3
  cout << v[1] << endl;      // 4.7
  v.erase(remove(v.begin(), v.end(), 4.7));
  OR v.erase(v.begin() + 1);
• list<int> v;
v.push_back(3);    // 3
v.push_front(5);   // 5 3
v.push_back(9);    // 5 3 9
v.insert(v.begin() + 2, 12);   // 5 3 12 9
// erase, pop_back, pop_front to remove