Lecture 2:
A Case Study of Computer Vision – Face Recognition

Professor Fei-Fei Li
Stanford Vision Lab
What we will learn today

• Recognition problems
• Face discrimination
  – Principal Component Analysis (PCA) and Eigenfaces (Problem Set 1 (Q1))
  – Linear Discriminant Analysis (LDA) and Fisherfaces (supplementary materials)
• Face detection
  – SVM (Problem Set 0 (Q2))
  – Boosting
“Faces” in the brain

Courtesy of Johannes M. Zanker
“Faces” in the brain

Stimulus
Face
House

Kanwisher, et al. 1997
Face Recognition

- Digital photography
Face Recognition

- Digital photography
- Surveillance
Face Recognition

- Digital photography
- Surveillance
- Album organization
Face Recognition

• Digital photography
• Surveillance
• Album organization
• Person tracking/id.
Face Recognition

- Digital photography
- Surveillance
- Album organization
- Person tracking/id.
- Emotions and expressions
Face Recognition

• Digital photography
• Surveillance
• Album organization
• Person tracking/id.
• Emotions and expressions
• Security/warfare
• Tele-conferencing
• Etc.
What’s ‘recognition’?
What’s ‘recognition’?

Identification

vs.

Categorization
Yes, there are faces
Yes, there is John Lennon
Detection or Localization

Localization

Identification

Categorization
Pose recognition

Detection or localization

No localization

Identification

Categorization
Milestone Face Recognition methods

1. PCA & Eigenfaces (Turk & Pentland, 1991)
2. LDA & Fisherfaces (Bellumeur et al. 1997)
3. AdaBoost (Viola & Jones, 2001)
Eigenfaces and Fishfaces

- Principle Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)

The Space of Faces

- An image is a point in a high dimensional space
  - If represented in grayscale intensity, an N x M image is a point in $\mathbb{R}^{NM}$
Key Idea

- Images in the possible set $\mathcal{X} = \{\hat{X}\}$ are highly correlated.
- So, compress them to a low-dimensional subspace that captures key appearance characteristics of the visual DOFs.

- **USE PCA for estimating the sub-space**
  (dimensionality reduction)

- Compare two faces by projecting the images into the subspace and measuring the EUCLIDEAN distance between them.

**EIGENFACES:** [Turk and Pentland 91]
USE PCA for estimating the sub-space

- Computes n-dim subspace such that the projection of the data points onto the subspace has the largest variance among all n-dim subspaces.
USE PCA for estimating the sub-space

2nd principal component

1st principal component
PCA Mathematical Formulation

PCA = eigenvalue decomposition of a data covariance matrix

Define a transformation, $W,$

$$y_j = W^T x_j \quad j = 1, 2 \ldots N$$

$m$-dimensional

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

$n$-dimensional

$$S_T = \sum_{j=1}^{N} (x_j - \bar{x})(x_j - \bar{x})^T = \text{Data Scatter matrix}$$

$$\hat{S}_T = \sum_{j=1}^{N} (y_j - \bar{y})(y_j - \bar{y})^T = W^T S_T W = \text{Transf. data scatter matrix}$$

$W_{\text{opt}} = \arg\max W |W^T S_T W|$

$= [v_1 \downarrow v_2 \downarrow \ldots \downarrow v_m]$
• Computes n-dim subspace such that the projection of the data points onto the subspace has the largest variance among all n-dim subspaces.

• Maximize the scatter of the training images in face space
Eigenfaces

• PCA extracts the eigenvectors of $S_T$
  – Gives a set of vectors $v_1, v_2, v_3, ...$
  – Each one of these vectors is a direction in face space:
Projecting onto the Eigenfaces

- The eigenfaces $v_1, \ldots, v_K$ span the space of faces

- A face is converted to eigenface coordinates by

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

$$a_1 \quad a_2 \quad a_K$$

$$\mathbf{x} \approx \bar{x} + a_1 v_1 + a_2 v_2 + \ldots + a_K v_K$$
Algorithm

Training

1. Align training images $x_1, x_2, \ldots, x_N$

   ![-aligned_images]

   Note that each image is formulated into a long vector!

2. Compute average face $u = \frac{1}{N} \sum x_i$

3. Compute the difference image $\varphi_i = x_i - u$
**Algorithm**

4. Compute the covariance matrix (total scatter matrix)

\[ S_T = (1/N) \sum \phi_i \phi_i^T = BB^T, \quad B=[\phi_1, \phi_2 \ldots \phi_N] \]

5. Compute the eigenvectors of the covariance matrix \( S_T \)

6. Compute training projections \( a_1, a_2 \ldots a_N \)

**Testing**

1. Take query image \( X \)
2. Project \( X \) into Eigenface space (\( W = \{\text{eigenfaces}\} \)) and compute projection \( \omega_i = W (X - u) \),
3. Compare projection \( \omega_i \) with all training \( N \) projections \( a_i \)
Illustration of Eigenfaces

- The visualization of eigenvectors:

These are the first 4 eigenvectors from a training set of 400 images (ORL Face Database).
Eigenfaces look somewhat like generic faces.
Reconstruction and Errors

<table>
<thead>
<tr>
<th>P = 4</th>
<th>P = 200</th>
<th>P = 400</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image 1" /></td>
<td><img src="image2" alt="Image 2" /></td>
<td><img src="image3" alt="Image 3" /></td>
</tr>
<tr>
<td><img src="image4" alt="Image 4" /></td>
<td><img src="image5" alt="Image 5" /></td>
<td><img src="image6" alt="Image 6" /></td>
</tr>
<tr>
<td><img src="image7" alt="Image 7" /></td>
<td><img src="image8" alt="Image 8" /></td>
<td><img src="image9" alt="Image 9" /></td>
</tr>
</tbody>
</table>

- Only selecting the top $P$ eigenfaces $\rightarrow$ reduces the dimensionality.
- Fewer eigenfaces result in more information loss, and hence less discrimination between faces.
Summary for Eigenface

Pros

• Non-iterative, globally optimal solution

Limitations

• PCA projection is optimal for reconstruction from a low dimensional basis, but may NOT be optimal for discrimination...

  • See supplementary materials for “Linear Discriminative Analysis”, aka “Fisherfaces”
Limitations

• Global appearance method: not robust to misalignment, or background variation
Limitations

- PCA assumes that the data has a Gaussian distribution (mean $\mu$, covariance matrix $\Sigma$)

The shape of this dataset is not well described by its principal components
Extensions

• Generalized PCA:

• Tensor Faces:

• PCA-SIFT
  *PCA-SIFT: A More Distinctive Representation for Local Image Descriptors* - Y Ke, R Sukthankar - IEEE CVPR 04
Milestone Face Recognition methods

Detection or localization

No localization

Identification

Categorization

1. PCA & Eigenfaces (Turk & Pentland, 1991)
2. LDA & Fisherfaces (Bellumeur et al. 1997)
3. AdaBoost (Viola & Jones, 2001)
Detecting foreground objects: A binary classification formulation
Linear classifiers

• Find linear function (hyperplane) to separate positive and negative examples

\[
\begin{align*}
\text{x}_i \text{ positive: } & \quad \text{x}_i \cdot \text{w} + b \geq 0 \\
\text{x}_i \text{ negative: } & \quad \text{x}_i \cdot \text{w} + b < 0
\end{align*}
\]

Which hyperplane is best?
Support vector machines

• Find hyperplane that maximizes the *margin* between the positive and negative examples.
Nonlinear SVMs

- Datasets that are linearly separable work out great:

- But what if the dataset is just too hard?

- We can map it to a higher-dimensional space:
Nonlinear SVMs

• General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

\[ \Phi: x \rightarrow \varphi(x) \]
Boosting


Each data point has a class label:

\[ y_t = \begin{cases} 
+1 \ 	ext{(•)} & \text{if } x_t \text{ is positive} \\
-1 \ 	ext{ (○)} & \text{if } x_t \text{ is negative} 
\end{cases} \]

and a weight:

\[ w_t = 1 \]

- It is a sequential procedure:
Toy example

Weak learners from the family of lines

Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases} \]

and a weight:

\[ w_t = 1 \]

\[ h \Rightarrow p(\text{error}) = 0.5 \text{ it is at chance} \]
Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases} \]

and a weight:

\[ w_t = 1 \]

This is a ‘\textbf{weak classifier}’: It performs slightly better than chance.
Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\text{red dot}) \\ -1 & (\text{blue dot}) \end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]
Each data point has a class label:

\[ y_t = \begin{cases} 
+1 & (\text{red}) \\
-1 & (\text{blue}) 
\end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]
Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\text{red}) \\ -1 & (\text{blue}) \end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]
Each data point has a class label:

\[ y_t = \begin{cases} +1 & \text{(\textcolor{red}{\bullet})} \\ -1 & \text{(\textcolor{blue}{\bullet})} \end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]
The strong (non-linear) classifier is built as the combination of all the weak (linear) classifiers.
Boosting

• Defines a classifier using an additive model:

\[ h(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) + \ldots \]
Boosting

• Defines a classifier using an additive model:

\[ h(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) + \ldots \]

• We need to define a family of weak classifiers

\[ h_{\downarrow k}(x) \] form a family of weak classifiers
Why boosting?

• A simple algorithm for learning robust classifiers
  – Freund & Shapire, 1995
  – Friedman, Hastie, Tibshhirani, 1998

• Provides efficient algorithm for sparse visual feature selection
  – Tieu & Viola, 2000
  – Viola & Jones, 2003

• Easy to implement, not requires external optimization tools.
Boosting - mathematics

• Weak learners

\[ h_j(x) = \begin{cases} 
1 & \text{if } f_j(x) > \theta_j \\
0 & \text{otherwise}
\end{cases} \]

value of rectangle feature

threshold

• Final strong classifier

\[ h(x) = \begin{cases} 
1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\
0 & \text{otherwise}
\end{cases} \]
Weak classifier

- 4 kind of Rectangle filters
- Value =

\[ \sum (\text{pixels in white area}) - \sum (\text{pixels in black area}) \]

Credit slide: S. Lazebnik
Weak classifier

Source

Result

Credit slide: S. Lazebnik
Viola & Jones algorithm

1. Evaluate each rectangle filter on each example

Weak classifier

\[ h_j(x) = \begin{cases} 
1 & \text{if } f_j(x) > \theta_j \\
0 & \text{otherwise} 
\end{cases} \]

Viola & Jones algorithm

• For a 24x24 detection region,

Viola & Jones algorithm

2. Select best filter/threshold combination

a. Normalize the weights
\[ w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \]

b. For each feature, \( j \)
\[ \varepsilon_j = \sum_i w_i |h_j(x_i) - y_i| \]

c. Choose the classifier, \( h_t \) with the lowest error \( \varepsilon_t \)

3. Reweight examples
\[ w_{t+1,i} = w_{t,i} \beta_t^{1 - |h_t(x_i) - y_i|} \]
\[ \beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t} \]

4. The final strong classifier is

\[
h(x) = \begin{cases} 
1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\
0 & \text{otherwise}
\end{cases}
\]

\[
\alpha_t = \log \frac{1}{\beta_t}
\]

The final hypothesis is a weighted linear combination of the \( T \) hypotheses where the weights are inversely proportional to the training errors.

Feature selection

Non discriminative features

Discriminative features
Viola & Jones algorithm

• A “paradigmatic” method for real-time object detection
• Training is slow, but detection is very fast
• Key ideas
  – Integral images for fast feature evaluation
  – Boosting for feature selection
  – Attentional cascade for fast rejection of non-face windows

Boosting for face detection

- For each round of boosting:
  1. Evaluate each rectangle filter on each example
  2. Select best filter/threshold combination
  3. Reweight examples
The implemented system

- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
    - 9500 non-face images
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose

System performance

- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous detector of comparable accuracy (Rowley et al., 1998)

Output of Face Detector on Test Images

Other detection tasks

Facial Feature Localization

Profile Detection

Male vs. female
Profile Detection
Profile Features
Face Image Databases

• Databases for face recognition can be best utilized as training sets
  – Each image consists of an individual on a uniform and uncluttered background
• Test Sets for face detection
  – MIT, CMU (frontal, profile), Kodak
### Experimental Results

- **Test dataset**
  - MIT+CMU frontal face test set
  - 130 images with 507 labeled frontal faces

<table>
<thead>
<tr>
<th>False detection</th>
<th>10</th>
<th>31</th>
<th>50</th>
<th>65</th>
<th>78</th>
<th>95</th>
<th>110</th>
<th>167</th>
<th>422</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AdaBoost</strong></td>
<td>78.3</td>
<td>85.2</td>
<td>88.8</td>
<td>89.8</td>
<td>90.1</td>
<td>90.8</td>
<td>91.1</td>
<td>91.8</td>
<td>93.7</td>
</tr>
<tr>
<td><strong>Neural-net</strong></td>
<td>83.2</td>
<td>86.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>89.2</td>
<td>-</td>
<td>90.1</td>
<td>89.9</td>
</tr>
</tbody>
</table>

MIT test set: 23 images with 149 faces
Sung & Poggio: detection rate 79.9% with 5 false positives
AdaBoost: detection rate 77.8% with 5 false positives
Experimental Results

- not significantly different accuracy
- but the cascade class almost 10 times faster
Sharing features with Boosting

Sharing features: efficient boosting procedures for multiclass object detection

Matlab code

• Gentle boosting

• Object detector using a part based model

http://people.csail.mit.edu/torralba/iccv2005/
What we have learned today

• Recognition problems
• Face discrimination
  – Principal Component Analysis (PCA) and Eigenfaces (Problem Set 1 (Q1))
  – Linear Discriminant Analysis (LDA) and Fisherfaces (supplementary materials)
• Face detection
  – SVM (Problem Set 0 (Q2))
  – Boosting
Supplementary materials
Fisher Faces: Linear Discriminant Analysis
Linear Discriminant Analysis (LDA)

Fisher’s Linear Discriminant (FLD)

- Eigenfaces exploit the max scatter of the training images in face space
- Fisherfaces attempt to maximise the **between class scatter**, while minimising the **within class scatter**.
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:** \( \{x_1, \ldots, x_c\} \)
- **Average of each class:**
  \[
  \mu_i = \frac{1}{N_i} \sum_{x_k \in \chi_i} x_k
  \]
- **Total average:**
  \[
  \mu = \frac{1}{N} \sum_{k=1}^{N} x_k
  \]
Scatters

• Scatter of class i:
  \[ S_i = \sum_{x_k \in \mathcal{X}_i} (x_k - \mu_i)(x_k - \mu_i)^T \]

• Within class scatter:
  \[ S_W = \sum_{i=1}^{c} S_i \]

• Between class scatter:
  \[ S_B = \sum_{i=1}^{c} |\mathcal{X}_i| (\mu_i - \mu)(\mu_i - \mu)^T \]

• Total scatter:
  \[ S_T = S_W + S_B \]
Within class scatter

\[ S_W = S_1 + S_2 \]

Between class scatter

\[
S_i = \sum_{x_k \in \mathcal{X}_i} (x_k - \mu_i)(x_k - \mu_i)^T
\]

\[
S_W = \sum_{i=1}^{c} S_i
\]

\[
S_B = \sum_{i=1}^{c} \mu_i \left( \mu_i - \mu \right) \left( \mu_i - \mu \right)^T
\]
Mathematical Formulation (1)

• After projection:
  \[ y_k = W^T x_k \]

• Between class scatter (of y’s):
  \[ \tilde{S}_B = W^T S_B W \]

• Within class scatter (of y’s):
  \[ \tilde{S}_W = W^T S_W W \]
$y_k = W^T x_k$

$\tilde{S}_W = \tilde{S}_1 + \tilde{S}_2$

$S_W = \sum_{i=1}^{c} S_i$

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

$\tilde{S}_W = W^T S_W W$

$\tilde{S}_B = W^T S_B W$
Mathematical Formulation

• The desired projection:

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

• How is it found? → Generalized Eigenvectors

\[
S_B w_i = \lambda_i S_W w_i \quad i = 1, \ldots, m
\]

• If \( S_W \) has full rank, the generalized eigenvectors are eigenvectors of \( S_W^{-1} S_B \) with largest eigenvalues
Training/ Testing

Projection in Eigenface

Projection $\omega_i = W_{\text{opt}} (X - u)$,

$W_{\text{opt}} = \{\text{fisher-faces}\}$
Results: Eigenface vs. Fisherface (1)

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

- Variation in Facial Expression, Eyewear, and Lighting
Eigenface vs. Fisherface (2)

The graph illustrates the error rate for Eigenface and Fisherface methods as a function of the number of principal components. The x-axis represents the number of principal components, while the y-axis shows the error rate in percentage. The graph compares Eigenface, Eigenface without the first three components, and Fisherface (7.3%). The error rate for Fisherface is consistently lower than for Eigenface and Eigenface w/o first three components, indicating superior performance.