Recognition
(Part 2)

Introduction to Computer Vision

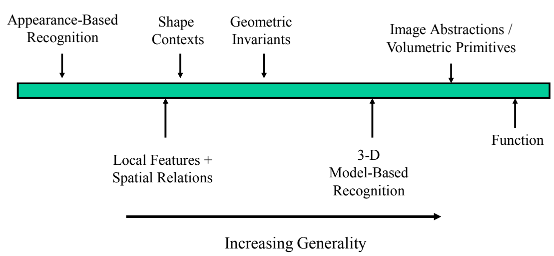
CSE 152

Lecture 15

Announcements

- Homework 3 is due May 18, 11:59 PM
- Homework 4 will be assigned this week
- Reading:
  - Chapter 15: Learning to Classify
  - Chapter 16: Classifying Images
  - Chapter 17: Detecting Objects in Images

A Rough Recognition Spectrum



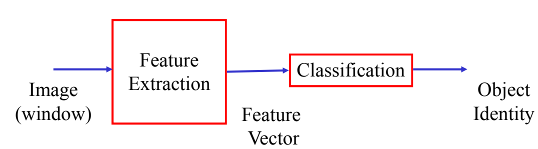
Appearance-Based Recognition

Appearance-Based Vision for Instances Level Recognition

- A Pattern Classification Viewpoint
  1. Bayesian Classification
  2. Appearance Manifolds
  3. Feature Space
  4. Dimensionality Reduction

Feature Space

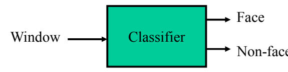
- Sketch of a Pattern Recognition Architecture



Sliding window approaches

Example: Face Detection

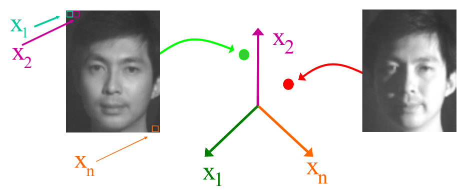
- Scan window over image
- Search over position & scale
- Classify window as either:
  - Face
  - Non-face



Feature Space

- **What are the features?**
- What is the classifier?

The Space of Images



- We will treat an d-pixel image as a point in an d-dimensional space, $\mathbf{x} \in \mathbf{R}^d$.
- Each pixel value is a coordinate of **x**.

More features

- Filtered image
- Filter with multiple filters (bank of filters)
- Histogram of colors
- Histogram of Gradients (HOG)
- Haar wavelets
- Scale Invariant Feature Transform (SIFT)
- Speeded Up Robust Feature (SURF)

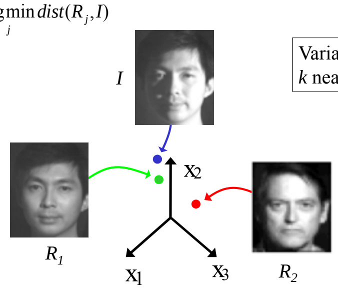
Feature Space

- What are the features?
- **What is the classifier?**

Nearest Neighbor Classifier

{ R_j } are set of training images.

$$ID = \arg\min_j dist(R_j, I)$$



Variation of this:
k nearest neighbors

Comments on Nearest Neighbor

- Sometimes called “Template Matching”
- Variations on distance function (e.g., L_1, robust distances)
- Multiple templates per class - perhaps many training images per class
- Expensive to compute *k* distances, especially when each image is big (*d*-dimensional)
- May not generalize well to unseen examples of class
- No worse than twice the error rate of the optimal classifier (if enough training samples)
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction

Do features vectors have structure in the image space?

- Faces of individuals cluster in the image space. (Not true)
- Faces of individuals are confined to a linear or affine subspace of $\mathbf{R}^{\mathbf{d}}$
- Faces of an individual are approximated by a linear subspace
- Faces and objects lie on or near a manifold in the space of images

An idea:

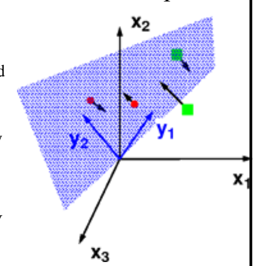
Represent the set of images as a linear subspace

What is a linear subspace?

Let *V* be a vector space and let *W* be a subset of *V*. Then *W* is a subspace if and only if:

1. The null vector **0** is in *W*
2. If **u** and **v** are elements of *W*, then any linear combination of **u** and **v** is an element of *W*; a**u** + b**v** $\in W$
3. If **u** is an element of *W* and *c* is a scalar, then the scalar product *c***u** $\in W$

- A *k*-dimensional subspace is spanned by *k* linearly independent vectors. It is spanned by a *k*-dimensional orthogonal basis



Example: A 2-D linear subspace of $\mathbf{R}^3$ spanned by y_1 and y_2

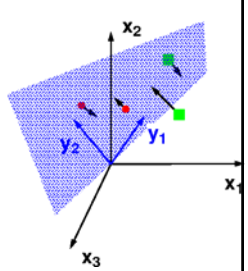
Linear Subspaces & Linear Projection

- A *d*-pixel image $\boldsymbol{x} \in \mathbf{R}^d$ can be projected to a low-dimensional feature space $\boldsymbol{y} \in \mathbf{R}^k$ by

$$\boldsymbol{y} = W\boldsymbol{x}$$

where *W* is an *k* by *d* matrix

- Each training image is projected to the subspace
- Recognition is performed in $\mathbf{R}^k$ using, for example, nearest neighbor
- How do we choose a good *W*?



Example: A 2-D linear subspace of $\mathbf{R}^3$ spanned by y_1 and y_2

Linear Subspaces & Recognition

1. Eigenfaces: Approximate all training images as a single linear subspace
2. Distance to subspace: Represent lighting variation without shadowing for a single individual as a 3D linear subspace. *n* individuals are modeled as as *n* 3D linear subspaces
3. Fisherfaces: Project all training images to a single subspace that enhances discriminability

Eigenfaces: Principal Component Analysis (PCA)

Assume we have a set of n feature vectors $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$ in $\mathbb{R}^d$. Write

$$\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{x}_i$$

$$\Sigma = \frac{1}{n-1}\sum_i (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

The unit eigenvectors of Σ — which we write as $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_d$, where the order is given by the size of the eigenvalue and $\boldsymbol{v}_1$ has the largest eigenvalue — give a set of features with the following properties:

- They are independent.
- Projection onto the basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ gives the k-dimensional set of linear features that preserves the most variance.

Algorithm 22.5: *Principal components analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset.*

Eigen decomposition of covariance matrix.

Alternative: singular value decomposition of (mean-deviation form of) data matrix.

Singular value decomposition and its relationship to eigen decomposition

- Any m by n matrix $\boldsymbol{A}$ may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

[m x n] = [m x m][m x n][n x n]

- $\boldsymbol{U}$: m by m, orthogonal matrix
  - Columns of **U** are the eigenvectors of $\boldsymbol{AA}^T$
- $\boldsymbol{V}$: n by n, orthogonal matrix,
  - columns are the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$
- **Σ:** m by n, diagonal with non-negative entries ($\sigma_1, \sigma_2, \ldots, \sigma_s$) with s=min(m,n) are called the called the singular values
  - **Singular values are the square roots of eigenvalues of both $\boldsymbol{AA}^T$ and $\boldsymbol{A}^T\boldsymbol{A}$**
  - *Result of SVD algorithm:* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_s$

SVD Properties

- In Matlab [u s v] = svd(A), and you can verify that: A=u*s*v'
- r=Rank($\boldsymbol{A}$) = # of non-zero singular values.
- $\boldsymbol{U}$, $\boldsymbol{V}$ give an orthonormal bases for the subspaces of $\boldsymbol{A}$:
  - 1st r columns of $\boldsymbol{U}$: Column space of $\boldsymbol{A}$
  - Last $m - r$ columns of $\boldsymbol{U}$: Left nullspace of $\boldsymbol{A}$
  - 1st r columns of $\boldsymbol{V}$: Row space of $\boldsymbol{A}$
  - 1st $n - r$ columns of $\boldsymbol{V}$: (Right) nullspace of $\boldsymbol{A}$
- *For some d where $d \leq r$, the first d column of U provide the best d-dimensional basis for columns of A in least squares sense.*

Performing PCA with SVD

- Singular values of A are the square roots of eigenvalues of AA^T (and A^TA)
- Columns of U are corresponding Eigenvectors of AA^T
- And $\sum_{i=1}^{n} a_i a_i^T = [a_1 \ a_2 \ \cdots \ a_n][a_1 \ a_2 \ \cdots \ a_n]^T = AA^T$
- Covariance matrix is:

$$\Sigma = \frac{1}{n-1}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T$$

- So, ignoring 1/(n-1), subtract mean image μ from each input image, create a d by n data matrix, and perform thin SVD on the data matrix. $D = [x_1 - \mu \mid x_2 - \mu \mid \ldots \ x_n - \mu]$

Economy SVD

- Any m by n matrix $\boldsymbol{A}$ may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

[m x n] = [m x m][m x n][n x n]

- If m>n, then one can view Σ as: (i.e., more pixels than images)

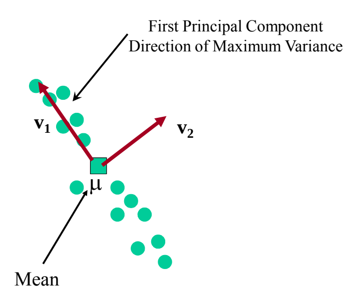
$$\begin{bmatrix} \Sigma' \\ 0 \end{bmatrix}$$

- Where Σ'=diag($\sigma_1, \sigma_2, \ldots, \sigma_s$) with s=min(m,n), and lower matrix is (n-m by m) of zeros.
- Alternatively, you can write:

$$\boldsymbol{A} = \boldsymbol{U}'\Sigma'\boldsymbol{V}^T$$

- In Matlab, economy SVD is:[U, S, V] = svd(A,'econ')

PCA Example



Eigenfaces

Modeling

1. Given a collection of n training images x_i, represent each one as a d-dimensional column vector
2. Compute the mean image and covariance matrix
3. Compute k Eigenvectors of the covariance matrix corresponding to the k largest Eigenvalues and form matrix $W^T=[u_1, u_2,\dots,u_k]$ (Or perform using SVD)
   - Note that the Eigenvectors are images
4. Project the training images to the k-dimensional Eigenspace. $y_i=Wx_i$

Recognition

1. Given a test image x, project the vectorized image to the Eigenspace by $y=Wx$
2. Perform classification of y to the projected training images

Why is W a good projection?

- The linear subspace spanned by W maximizes the variance (i.e., the spread) of the projected data.
- W spans a subspace that is the best approximation to the data in a least squares sense. E.g., W is the subspace that minimizes the the sum of the squared distances from each datapoint to the the subspace.

Eigenfaces: Training Images

[Turk, Pentland 91]

Eigenfaces

Mean Image

Basis Images

Difficulties with PCA

- Projection may suppress important detail
  - smallest variance directions may not be unimportant
- Method does not take discriminative task into account
  - typically, we wish to compute features that allow good discrimination
  - not the same as largest variance or minimizing reconstruction error.

Alternative projections

Fisherfaces: Class specific linear projection

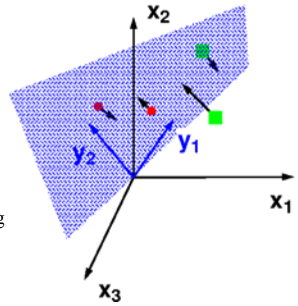
P. Belhumeur, J. Hespanha, D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, PAMI, July 1997, pp. 711--720.

- An *n*-pixel image $x \in \mathbf{R}^d$ can be projected to a low-dimensional feature space $y \in \mathbf{R}^k$ by

$$y = Wx$$

where W is an k by d matrix

- Recognition is performed using nearest neighbor in $\mathbf{R}^k$
- How do we choose a good W?



PCA & Fisher's Linear Discriminant

- Between-class scatter

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

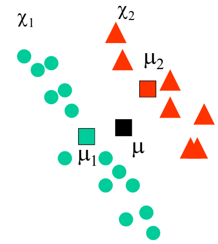
- Within-class scatter

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

- Total scatter

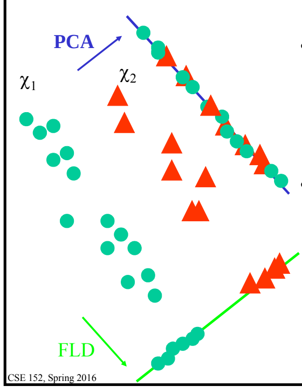
$$S_T = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu)^T = S_B + S_W$$

- Where
  - c is the number of classes
  - μ_i is the mean of class χ_i
  - $|\chi_i|$ is number of samples of χ_i.



If the data points x_i are projected by $y_i = Wx_i$ and the scatter of x_i is S, then the scatter of the projected points y_i is $W^T S W$

PCA & Fisher's Linear Discriminant



- PCA (Eigenfaces)

$$W_{PCA} = \arg\max_W |W^T S_T W|$$

Maximizes projected total scatter

- Fisher's Linear Discriminant

$$W_{fld} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

Maximizes ratio of projected between-class to projected within-class scatter

Computing the Fisher Projection Matrix

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|} = [\mathbf{w}_1 \ \mathbf{w}_2 \ \ldots \ \mathbf{w}_m] \quad (4)$$

where $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ is the set of generalized eigenvectors of S_B and S_W corresponding to the m largest generalized eigenvalues $\{\lambda_i | i = 1, 2, \ldots, m\}$, i.e.,

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

- The w_i are orthonormal
- There are at most *c*-1 non-zero generalized Eigenvalues, so $m \le c-1$
- Can be computed with *eig* in Matlab

Fisherfaces

$$W = W_{fld} W_{PCA}$$

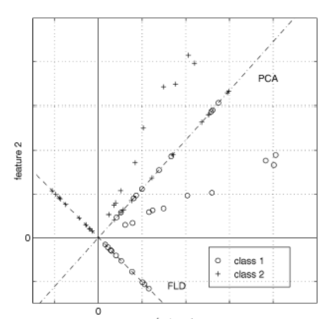
$$W_{PCA} = \arg\max_W |W^T S_T W|$$

$$W_{fld} = \arg\max_W \frac{|W^T W_{PCA}^T S_B W_{PCA} W|}{|W^T W_{PCA}^T S_W W_{PCA} W|}$$

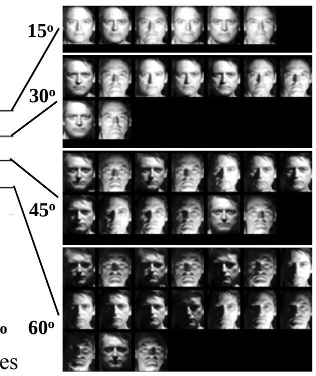
- Since S_W is rank N-c, project training set to subspace spanned by first *N-c* principal components of the training set.
- Apply FLD to *N-c* dimensional subspace yielding *c-1* dimensional feature space.

- Fisher's Linear Discriminant projects away the within-class variation (lighting, expressions) found in training set.
- Fisher's Linear Discriminant preserves the separability of the classes.

PCA vs. FLD

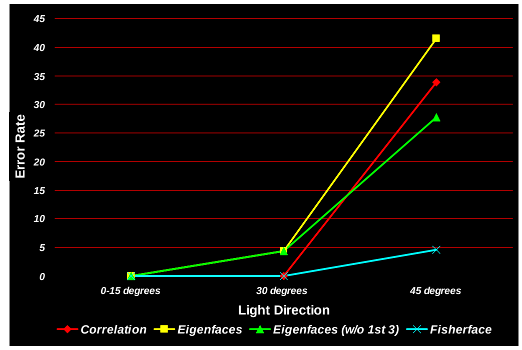


Harvard Face Database



- 10 individuals
- 66 images per person
- Train on 6 images at 15º
- Test on remaining images

Recognition Results: Lighting Extrapolation



Next Lecture

- Recognition, detection, and classification
- Reading:
  - Chapter 15: Learning to Classify
  - Chapter 16: Classifying Images
  - Chapter 17: Detecting Objects in Images