View-Based Models

Goal: Explore ways to model the image appearance of objects under a wide range of viewing conditions.

Motivation:

A central question in vision concerns how we represent objects. One simple approach is to let images themselves be the representation.

- We consider the construction of low-dimensional bases for an ensemble of training images of the object(s) in question using principal components analysis (PCA).
- We introduce PCA, its derivation, its properties, and some of its uses.
- We examine its suitability for object detection, and briefly discuss some alternatives.

Readings: Sections 22.1–22.3 of the Forsyth and Ponce.
Matlab Tutorials: trainEigenEyes.m and detectEigenEyes.m

Template Matching – Straw Man

What if we just stored all images (templates) of the object(s) in each *characteristic view* available (the simplest possible view-based model).

For detection we could compute a *matching score* based on cross-correlation of each template with every image neighbourhood.

Left Eyes Right Eyes

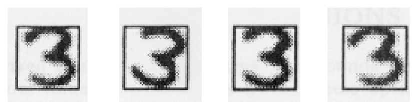
Problems:

- cross-correlation and related detectors are very sensitive to small variations in object pose, lighting, occlusions, and small variations in object shape and appearance.
- we'd certainly need an extremely large training set of images.
- storage and computation costs become unreasonable as the number of objects and views increases.

Question: How can we find a more efficient representation for the ensemble of views, and more effectve methods for matching?

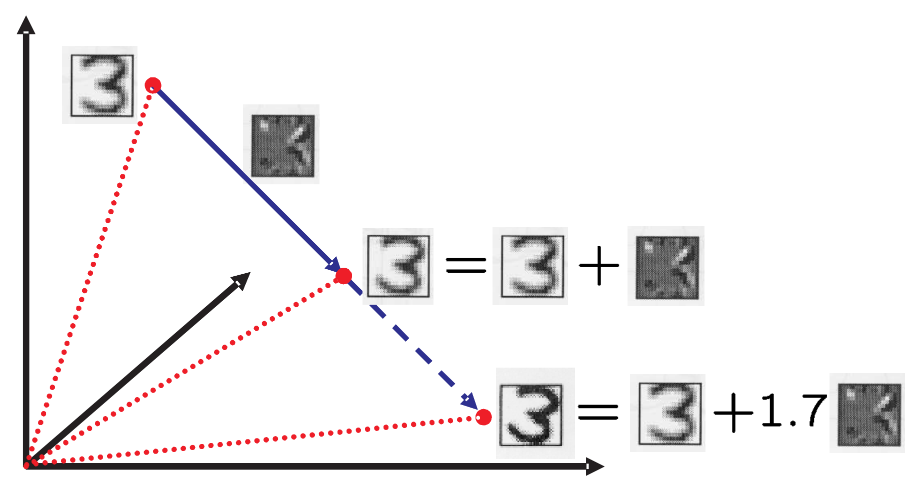
Subspace Appearance Models

Idea: Images are not random, especially those of an object, or similar objects, under different viewing conditions.



Rather, than storing every image, we might try to represent the images more effectively, e.g., in a lower dimensional *subspace*.

For example, let's represent each $N \times N$ image as a point in an N^2-dim vector space (e.g., ordering the pixels lexicographically to form the vectors).



(red points denote images, blue vectors denote image differences)

How do we find a low-dimensional basis to accurately model (approximate) each image of the training ensemble (as a linear combination of basis images)?

Linear Subspace Models

We seek a linear basis with which each image in the ensemble is approximated as a linear combination of basis images $b_k(\vec{\mathbf{x}})$

$$I(\vec{\mathbf{x}}) \;\approx\; m(\vec{x}) + \sum_{k=1}^{K} a_k\, b_k(\vec{\mathbf{x}}), \tag{1}$$

variants which emphasize detection instead of image generation here $m(\vec{x}\,)$ is the mean of the image ensemble. The *subspace coefficients* $\vec{\mathbf{a}} = (a_1, ..., a_K)$ comprise the representaion.

With some abuse of notation, assuming basis images $b_k(\vec{\mathbf{x}})$ with N^2 pixels, let's define

$\vec{\mathbf{b}}_k$ – an $N^2 \times 1$ vector with pixels arranged in lexicographic order
$\mathbf{B}$ – a matrix with columns $\vec{\mathbf{b}}_k$, *i.e.*, $\mathbf{B} = [\vec{\mathbf{b}}_1, ..., \vec{\mathbf{b}}_K] \in \mathcal{R}^{N^2 \times K}$

With this notation we can rewrite Eq. (1) in matrix algebra as

$$\vec{\mathbf{I}} \;\approx\; \vec{m} + \mathbf{B}\,\vec{\mathbf{a}} \tag{2}$$

In what follows, we assume that the mean of the ensemble is $\vec{0}$. (Otherwise, if the ensemble we have is not mean zero, we can estimate the mean and subtract it from each image.)

Choosing The Basis

Orthogonality: Let's assume orthogonal basis functions,

$$\| \vec{\mathbf{b}}_k \|_2 \;=\; 1 \quad , \qquad \vec{\mathbf{b}}_j^{\,T} \vec{\mathbf{b}}_k \;=\; \delta_{jk} \; .$$

Subspace Coefficients: It follows from the linear model in Eq. (2) and the orthogonality of the basis functions that

$$\vec{\mathbf{b}}_k^{\,T} \vec{\mathbf{I}} \;\approx\; \vec{\mathbf{b}}_k^{\,T} \mathbf{B}\,\vec{\mathbf{a}} \;=\; \vec{\mathbf{b}}_k^{\,T} [\vec{\mathbf{b}}_1, ..., \vec{\mathbf{b}}_K]\,\vec{\mathbf{a}} \;=\; a_k$$

This selection of coefficients, $\vec{\mathbf{a}} = \mathbf{B}^T \vec{\mathbf{I}}$, minimizes the sum of squared errors (or sum of squared pixel differences, SSD):

$$\min_{\vec{\mathbf{a}} \in \mathcal{R}^K} \| \vec{\mathbf{I}} - \mathbf{B}\,\vec{\mathbf{a}} \|_2^2$$

Basis Images: In order to select the basis functions $\{\vec{\mathbf{b}}_k\}_{k=1}^K$, suppose we have a training set of images

$$\{ \vec{\mathbf{I}}_l \}_{l=1}^L \quad , \quad \text{with } \; L \gg K$$

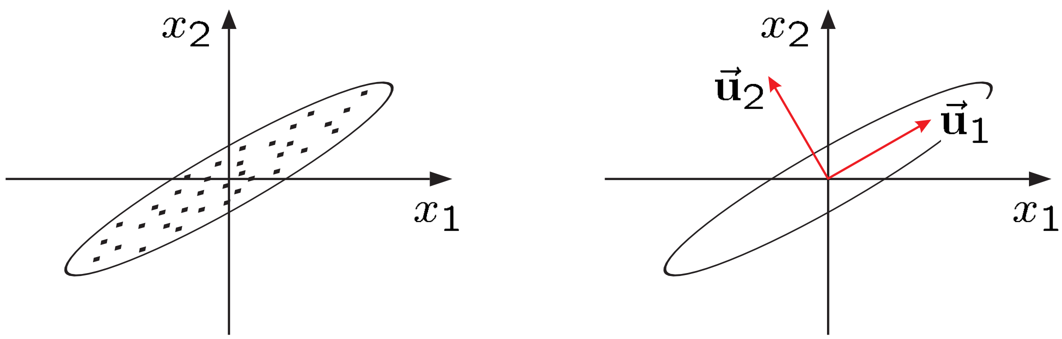
If the mean is nonzero, subtract the mean image, $\frac{1}{L} \sum_l \vec{\mathbf{I}}_l$, from each training image. (Recall we are assuming the images are mean zero.)

Finally, let's select the basis, $\{\vec{\mathbf{b}}_k\}_{k=1}^K$, to minimize squared reconstruction error:

$$\sum_{l=1}^{L} \min_{\vec{\mathbf{a}}_l} \| \vec{\mathbf{I}}_l - \mathbf{B}\,\vec{\mathbf{a}}_l \|_2^2$$

Intuitions

Example: let's consider a set of images $\{\,\vec{\mathbf{I}}_l\,\}_{l=1}^{L}$, each with only two pixels. So, each image can be viewed as a 2D point, $\vec{\mathbf{I}}_l \in \mathcal{R}^2$.



For a model with only one basis image, what should $\vec{\mathbf{b}}_1$ be?

Approach: Fit an ellipse to the distribution of the image data, and choose $\vec{\mathbf{b}}_1$ to be a unit vector in the direction of the major axis.

Define the ellipse as $\vec{\mathbf{x}}^T C^{-1} \vec{\mathbf{x}} = 1$, where C is the sample covariance matrix of the image data,

$$\mathbf{C} = \frac{1}{L} \sum_{l=1}^{L} \vec{\mathbf{I}}_l \, \vec{\mathbf{I}}_l^{\,T}$$

The eigenvectors of $\mathbf{C}$ provide the major axis, i.e.,

$$\mathbf{C}\,\mathbf{U} \;=\; \mathbf{U}\,\mathbf{D}$$

for orthogonal matrix $\mathbf{U} = [\vec{\mathbf{u}}_1, \, \vec{\mathbf{u}}_2]$, and diagonal matrix $\mathbf{D}$ with elements $d_1 \geq d_2 \geq 0$. The direction $\vec{\mathbf{u}}_1$ associated with the largest eigenvalue is the direction of the major axis, so let $\vec{\mathbf{b}}_1 = \vec{\mathbf{u}}_1$.

Principal Components Analysis

Theorem: *(Minimum reconstruction error)* The orthogonal basis $\mathbf{B}$, of rank $K < N^2$, that minimizes the squared reconstruction error over training data, $\{\vec{\mathbf{I}}_l\}_{l=1}^{L}$, i.e.,

$$\sum_{l=1}^{L} \min_{\vec{\mathbf{a}}_l} \| \vec{\mathbf{I}}_l - \mathbf{B}\,\vec{\mathbf{a}}_l \|_2^2$$

is given by the first K eigenvectors of the data covariance matrix

$$\mathbf{C} = \frac{1}{L} \sum_{l=1}^{L} \vec{\mathbf{I}}_l \, \vec{\mathbf{I}}_l^{\,T} \;\in\; \mathcal{R}^{N^2 \times N^2} \quad , \quad \text{for which} \quad \mathbf{C}\,\mathbf{U} \;=\; \mathbf{U}\,\mathbf{D}$$

where $\mathbf{U} = [\vec{\mathbf{u}}_1, ..., \vec{\mathbf{u}}_{N^2}]$ is orthogonal, and $\mathbf{D} = \text{diag}(d_1, ..., d_{N^2})$ with $d_1 \geq d_2 \geq ... \geq d_{N^2}$.

That is, the optimal basis vectors are $\vec{\mathbf{b}}_k = \vec{\mathbf{u}}_k$, for $k = 1...K$. The corresponding basis images $\{b_k(\vec{\mathbf{x}})\}_{k=1}^{K}$ are often called eigen-images.

Proof: see the derivation below.

Derivation of PCA

To begin, we want to find $\mathbf{B}$ in order to minimize squared error in subspace approximations to the images of the training ensemble.

$$E \;=\; \sum_{l=1}^{L} \min_{\vec{\mathbf{a}}_l} \parallel \vec{\mathbf{I}}_l - \mathbf{B}\,\vec{\mathbf{a}}_l \parallel_2^2$$

Given the assumption that the columns of $\mathbf{B}$ are orthonormal, the optimal coefficients are $\vec{\mathbf{a}}_l = \mathbf{B}^T\vec{\mathbf{I}}_l$, so

$$E \;=\; \sum_{l=1}^{L} \min_{\vec{\mathbf{a}}_l} \parallel \vec{\mathbf{I}}_l - \mathbf{B}\,\vec{\mathbf{a}}_l \parallel_2^2 \;\;=\;\; \parallel \vec{\mathbf{I}}_l - \mathbf{B}\,\mathbf{B}^T\vec{\mathbf{I}}_l \parallel_2^2 \tag{3}$$

Furthermore, writing the each training image as a column in a matrix $\mathbf{A} = \left[\vec{\mathbf{I}}_1, ..., \vec{\mathbf{I}}_L\right]$, we have

$$E \;=\; \sum_{l=1}^{L} \parallel \vec{\mathbf{I}}_l - \mathbf{B}\,\mathbf{B}^T\vec{\mathbf{I}}_l \parallel_2^2 \;\;=\;\; \parallel \mathbf{A} - \mathbf{B}\,\mathbf{B}^T\mathbf{A} \parallel_F^2 \;\;=\; trace\left[\,\mathbf{A}\,\mathbf{A}^T\,\right] - trace\left[\,\mathbf{B}^T\mathbf{A}\,\mathbf{A}^T\mathbf{B}\,\right]$$

You get this last step by expanding the square and noting $\mathbf{B}^T\mathbf{B} = \mathbf{I}_K$, and using the properties of $trace$, e.g., $trace[\mathbf{A}] = trace[\mathbf{A}^T]$, and also $trace[\mathbf{B}^T\mathbf{A}\mathbf{A}^T\mathbf{B}] = trace[\mathbf{A}^T\mathbf{B}\mathbf{B}^T\mathbf{A}]$.
So the minmize the average squared error in the approximation we want to find $\mathbf{B}$ to maximize

$$E' \;=\; trace\left[\,\mathbf{B}^T\mathbf{A}\,\mathbf{A}^T\mathbf{B}\,\right] \tag{4}$$

Now, let's use the fact that for the data covariance, $\mathbf{C}$ we have $\mathbf{C} = \frac{1}{L}\mathbf{A}\,\mathbf{A}^T$. Moreover, as defined above the SVD of $\mathbf{C}$ can be written as $\mathbf{C} = \mathbf{U}\,\mathbf{D}\,\mathbf{U}^T$. So, let's substitute the SVD into E':

$$E' \;=\; trace\left[\,\mathbf{B}^T\mathbf{U}\,\mathbf{D}\,\mathbf{U}^T\mathbf{B}\,\right] \tag{5}$$

where of course $\mathbf{U}$ is orthogonal, and $\mathbf{D}$ is diagonal.

Now we just have to show that we want to choose $\mathbf{B}$ such that the trace strips off the first K elements of $\mathbf{D}$ to maximize E'. Intuitively, note that $\mathbf{B}^T\mathbf{U}$ must be rank K since $\mathbf{B}$ is rank K. And note that the diagonal elements of $\mathbf{D}$ are ordered. Also the trace is invariant under matrix rotation. So, the highest rank K trace we can hope to get is by choosing $\mathbf{B}$ so that, when combined with $\mathbf{U}$ we keep the first K columns of $\mathbf{D}$. That is, the columns of $\mathbf{B}$ should be the first K orthonormal rows of $\mathbf{U}$. We need to make this a little more rigorous, but that's it for now...

Other Properties of PCA

Maximum Variance: The K-D subspace approximation captures the greatest possible variance in the training data.

- For $a_1 = \vec{\mathbf{b}}_1^T \vec{\mathbf{I}}$, the direction $\vec{\mathbf{b}}_1$ that maximizes the variance $\mathrm{E}[a_1^2] = \vec{\mathbf{b}}_1^{\,T} \mathbf{C} \vec{\mathbf{b}}_1$, subject to $\vec{\mathbf{b}}_1^T \vec{\mathbf{b}}_1 = 1$, is the first eigenvector of $\mathbf{C}$.
- The second maximizes $\vec{\mathbf{b}}_2^T \mathbf{C} \vec{\mathbf{b}}_2$ subject to $\vec{\mathbf{b}}_2^T \vec{\mathbf{b}}_2 = 1$ and $\vec{\mathbf{b}}_1^T \vec{\mathbf{b}}_2 = 0$.
- For $a_k = \vec{\mathbf{b}}_k^{\,T} \vec{\mathbf{I}}$, and $\vec{\mathbf{a}} = (a_1, ..., a_K)$, the subspace coefficient covariance is $\mathrm{E}[\vec{\mathbf{a}}\,\vec{\mathbf{a}}^T] = \mathrm{diag}(d_1, ..., d_K)$. That is, the diagonal entries of $\mathbf{D}$ are marginal variances of the subspace coefficients:

  $$\sigma_k^2 \;\equiv\; \mathrm{E}[a_k^2] \;=\; d_k\ .$$

  So the total variance *captured* in the subspace is sum of first K eigenvalues of $\mathbf{C}$.
- Total variance *lost* owing to the subspace projection (i.e., the out-of-subspace variance) is the sum of the last $N^2 - K$ eigenvalues:

  $$\frac{1}{L} \sum_{l=1}^{L} \left[\min_{\vec{\mathbf{a}}_l} \| \, \vec{\mathbf{I}}_l - \mathbf{B}\, \vec{\mathbf{a}}_l \, \|_2^2 \right] \;=\; \sum_{k=K+1}^{N^2} \sigma_k^2$$

Decorrelated Coefficients: $\mathbf{C}$ is diagonalized by its eigenvectors, so $\mathbf{D}$ is diagonal, and the subspace coefficients are uncorrelated.

- Under a Gaussian model of the images (where the images are drawn from an N^2-dimensional Gaussian pdf), this means that the coefficients are also statistically independent.

PCA and Singular Value Decomposition

The singular value decomposition of the data matrix $\mathbf{A}$,

$$\mathbf{A} = \left[\vec{\mathbf{I}}_1, ..., \vec{\mathbf{I}}_L\right] \ , \quad \mathbf{A} \in \mathcal{R}^{N^2 \times L} \ , \quad \text{where usually } \ L \ll N^2 \ .$$

is given by

$$\mathbf{A} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T$$

where $\mathbf{U} \in \mathcal{R}^{N^2\times L}$, $\mathbf{S} \in \mathcal{R}^{L\times L}$, $\mathbf{V} \in \mathcal{R}^{L\times L}$. The columns of $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, i.e., $\mathbf{U}^T\mathbf{U} = \mathbf{I}_{L\times L}$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}_{L\times L}$, and matrix $\mathbf{S}$ is diagonal, $\mathbf{S} = \text{diag}(s_1, ..., s_L)$ where $s_1 \geq s_2 \geq ... \geq s_L \geq 0$.

Theorem: The best rank-K approximation to $\mathbf{A}$ under the Frobenius norm, $\tilde{\mathbf{A}}$, is given by

$$\tilde{\mathbf{A}} = \sum_{k=1}^{K} s_k \vec{\mathbf{u}}_k \vec{\mathbf{u}}_k^{\,T} = \mathbf{B}\,\mathbf{B}^T\mathbf{A} \ , \ \text{ where } \min_{rank(\tilde{\mathbf{A}})=K} \|\, \mathbf{A} - \tilde{\mathbf{A}} \,\|_F^2 = \sum_{k=K+1}^{N^2} s_k^2 \ ,$$

and $\mathbf{B} = [\vec{\mathbf{u}}_1, ..., \vec{\mathbf{u}}_K]$. $\tilde{\mathbf{A}}$ is also the best rank-K approximation under the L_2 matrix norm.

What's the relation to PCA and the covariance of the training images?

$$\mathbf{C} = \frac{1}{L} \sum_{l=1}^{L} \vec{\mathbf{I}}_l \, \vec{\mathbf{I}}_l^{\,T} = \frac{1}{L} \mathbf{A}\,\mathbf{A}^T = \frac{1}{L} \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T\mathbf{V}\,\mathbf{S}^T\,\mathbf{U}^T = \frac{1}{L} \mathbf{U}\,\mathbf{S}^2\,\mathbf{U}^T$$

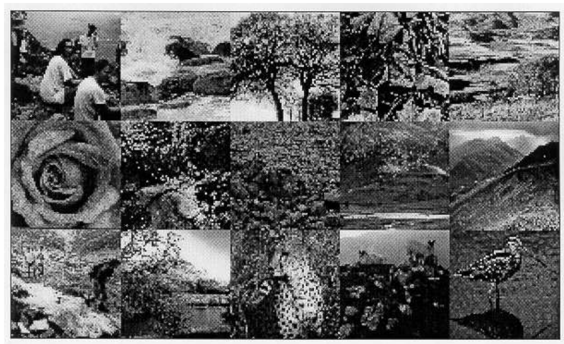
So the squared singular values of $\mathbf{A}$ are proportional to the first L eigenvalues of $\mathbf{C}$:

$$d_k = \begin{cases} \frac{1}{L} s_k^2 & \text{for } k = 1, ..., L \\ 0 & \text{for } k > L \end{cases}$$

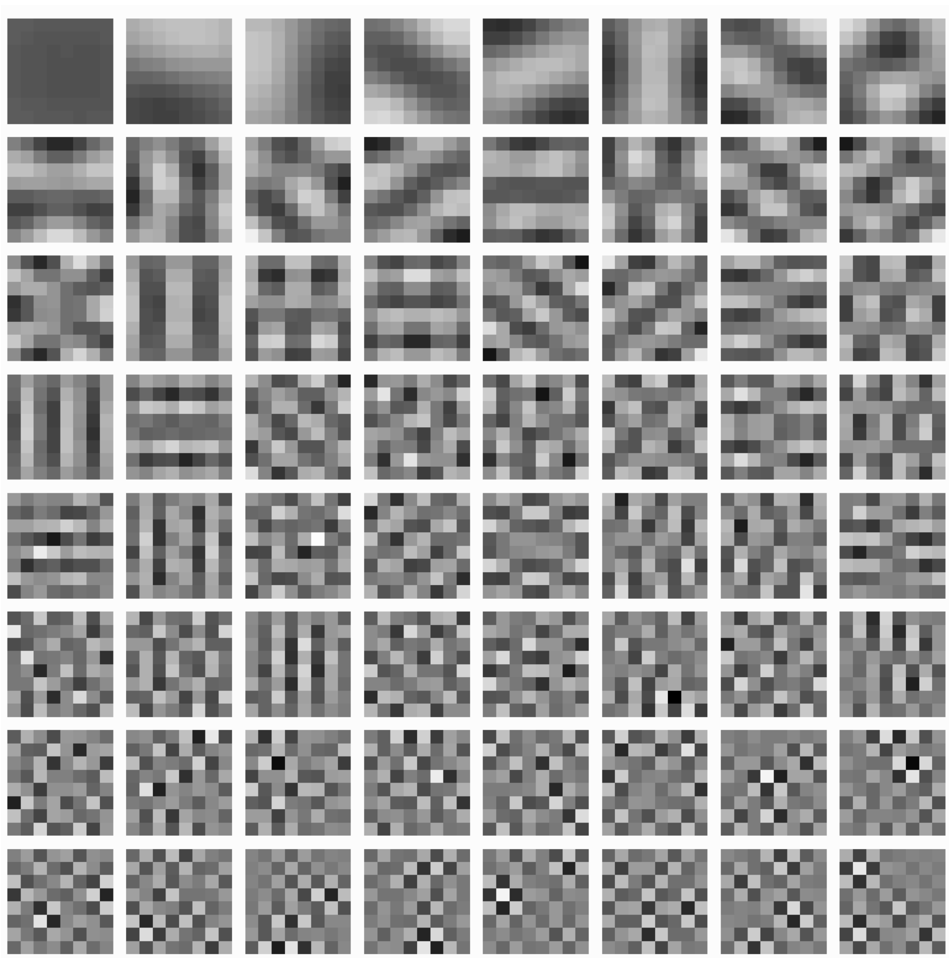
And the singular vectors of $\mathbf{A}$ are just the first L eigenvectors of $\mathbf{C}$.

Eigen-Images for Generic Images?

Fourier components are eigenfunctions of generic image ensembles.



Why? Covariance matrices for stationary processes are Toeplitz.



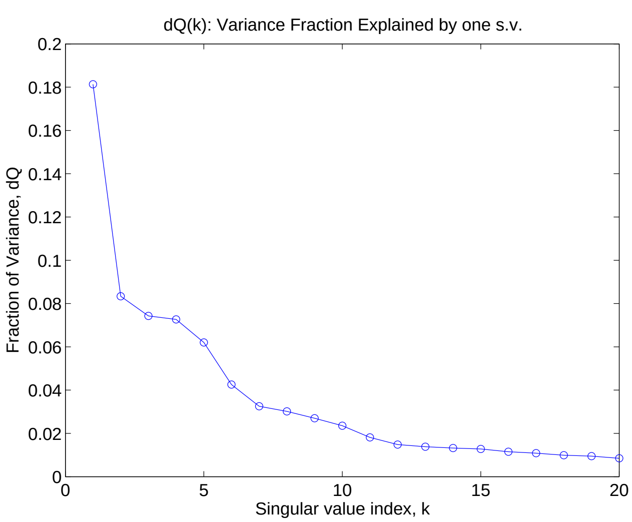
PCA yields unique eigen-images up to rotations of invariant subspaces (e.g., Fourier components with the same marginal variance).

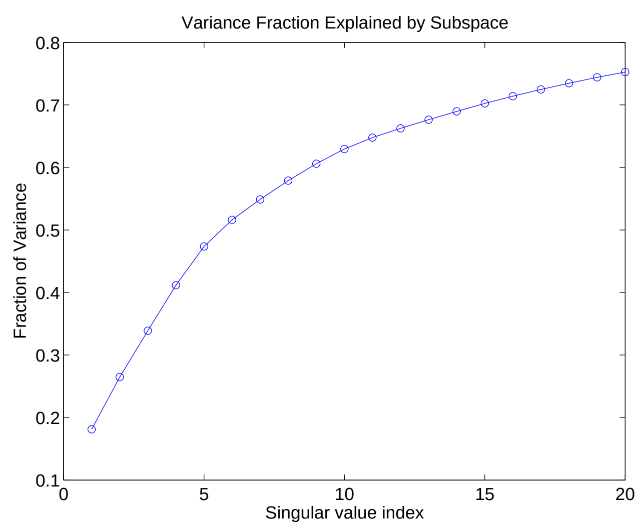
Eye Subspace Model

Subset of 1196 eye images (25×20**):**

Left Eyes　　　　　　　　Right Eyes

Defn: Let $V_k \equiv \sum_{j=1}^{k} s_j^2$, $dQ_k \equiv s_k^2 / V_L$, and $Q_k \equiv V_k / V_L$:



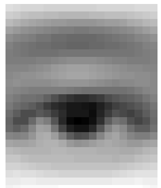


Left plot shows dQ_k, the fraction of the total variance contributed by the k^{th} principal component.

Right plot shows Q_k the fraction of the total variance captured by the subspace formed from the first k principal components.

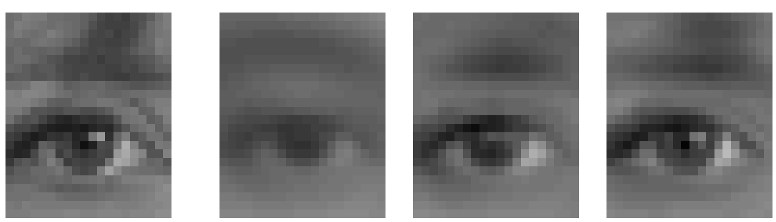
Eye Subspace Model

Mean Eye:

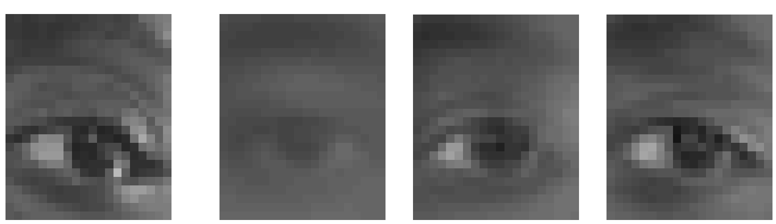


Basis Images $(1-6, \;\text{ and }\; 10:5:35)$**:**

Reconstructions (for $K = 5, 20, 50$)**:**



Eye Image | Reconstruction (K = 5) | Reconstruction (K = 20) | Reconstruction (K = 50)



Eye Image | Reconstruction (K = 5) | Reconstruction (K = 20) | Reconstruction (K = 50)

Generative Eye Model

Generative model, $\mathcal{M}$, for random eye images:

$$\vec{\mathbf{I}} \; = \; \vec{\mathbf{m}} + \left(\sum_{k=1}^{K} a_k \vec{\mathbf{b}}_k \right) \; + \; \vec{\mathbf{e}}$$

where $\vec{\mathbf{m}}$ is the mean eye image, $a_k \sim \mathcal{N}(0, \sigma_k^2)$, σ_k^2 is the sample variance associated with the k^{th} principal direction in the training data, and $\vec{\mathbf{e}} \sim \mathcal{N}(0, \sigma_e^2 \, \mathbf{I}_{N^2})$ where $\sigma_e^2 = \frac{1}{N^2} \sum_{k=K+1}^{N^2} \sigma_k^2$ is the per pixel out-of-subspace variance.

Random Eye Images:

Random draws from generative model (with K = 5, 10, 20, 50, 100, 200)

So the likelihood of an image of a eye given this model $\mathcal{M}$ is

$$p(\vec{\mathbf{I}} \,|\, \mathcal{M}) \; = \; \left(\prod_{k=1}^{K} p(a_k | \mathcal{M}) \right) \; p(\vec{\mathbf{e}} \,|\, \mathcal{M})$$

where

$$p(a_k|\mathcal{M}) \; = \; \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{a_k^2}{2\sigma_k^2}} \quad , \qquad p(\vec{\mathbf{e}} \,|\, \mathcal{M}) = \prod_{j=1}^{N^2} \frac{1}{\sqrt{2\pi}\sigma_e} e^{-\frac{e_j^2}{2\sigma_e^2}} \quad .$$

Eye Detection

The log likelihood of the model is given by

$$
\begin{aligned}
L(\mathcal{M}) \;\equiv\; \log p(\vec{\mathbf{I}}\,|\,\mathcal{M}) \;&=\; \left(\sum_{k=1}^{K} \log p(a_k|\mathcal{M})\right) \;+\; \log p(\vec{\mathbf{e}}\,|\,\mathcal{M}) \\
&=\; \left(\sum_{k=1}^{K} \frac{-a_k^2}{2\sigma_k^2}\right) \;+\; \left(\sum_{j=1}^{N^2} \frac{-e_j^2}{2\sigma_e^2}\right) \;+\; const \\
&\equiv\; S_{in}(\vec{\mathbf{a}}) \;+\; S_{out}(\vec{\mathbf{e}}) \;+\; const
\end{aligned}
$$

Detector:

1. Given an image $\vec{\mathbf{I}}$
2. Compute the subspace coefficients $\vec{\mathbf{a}} = \mathbf{B}^T(\vec{\mathbf{I}} - \vec{\mathbf{m}})$
3. Compute residual $\vec{\mathbf{e}} = \vec{\mathbf{I}} - \vec{\mathbf{m}} - \mathbf{B}\vec{\mathbf{a}}$
4. For $S(\vec{\mathbf{a}}, \vec{\mathbf{e}}) = S_{in}(\vec{\mathbf{a}}) + S_{out}(\vec{\mathbf{e}})$, and a given threshold τ, the image patch is classified as an eye when

$$S(\vec{\mathbf{a}}, \vec{\mathbf{e}}) \;>\; \tau\;.$$

Eye Detection

Terminology:

- true positive = hit
- true negative = correct rejection
- false positive = false alarm (type I error)
- false negative = miss (type II error)

	classified positives	classified negatives	
true examples	true positives, T_{pos}	false negatives, F_{neg}	$N_{pos} = T_{pos} + F_{neg}$
false examples	false positives, F_{pos}	true negatives, T_{neg}	$N_{neg} = F_{pos} + T_{neg}$
	C_{pos}	C_{neg}	N

Definitions:

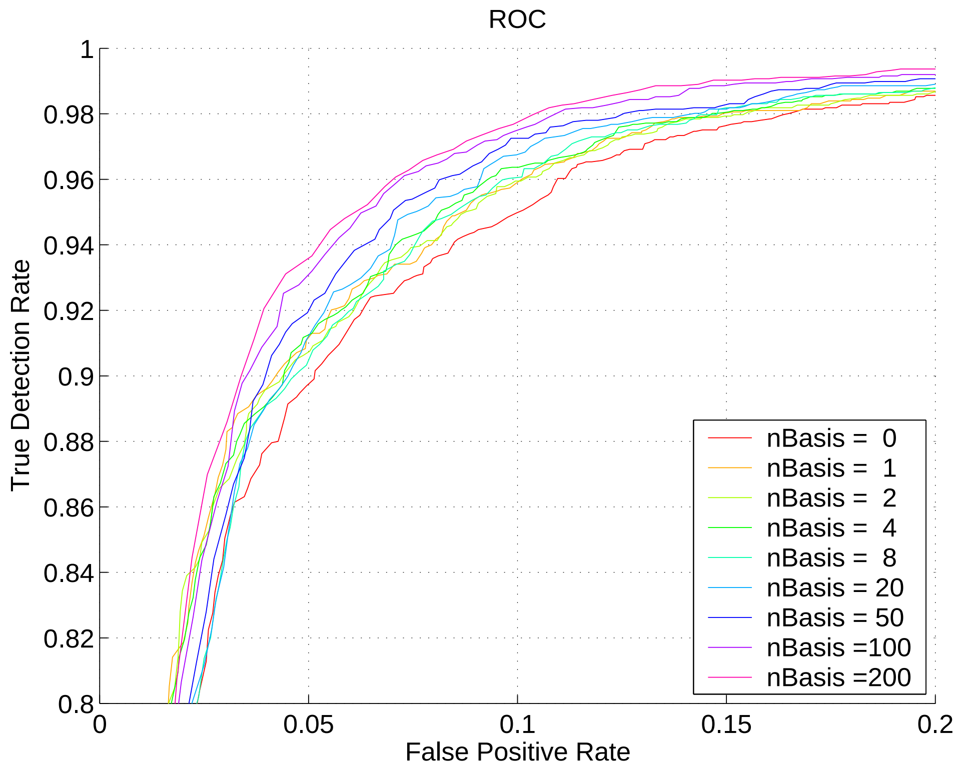
- true positive (hit) rate: $\rho_{tp} = T_{pos}/N_{pos}$ (sensitivity)
  (i.e., what fraction of the true eyes do we find?)
- true negative (reject) rate: $\rho_{tn} = T_{neg}/N_{neg}$ (specificity)
- false positive rate: $\rho_{fp} = F_{pos}/N_{neg} = 1 - \rho_{tn}$ (1 - specificity)

- precision: T_{pos}/C_{pos}
  (i.e., what fraction of positive response are correct hits? ...
  i.e., how noisy is the detector?)
- recall: $\rho_{tp} = T_{pos}/N_{pos}$
  (i.e., what fraction of the true eyes do we actually find?)

Eye Detection

ROC Curves:

- true detection rate (sensitivity) vs false positive rate (1-specificity)
- trade-off (as a function of decision threshold τ) between sensitivity (hit rate) and specificity (responding only to positive cases)



Here the eye images in the test set were different from the those in the training set. Non-eyes were drawn at random from images.

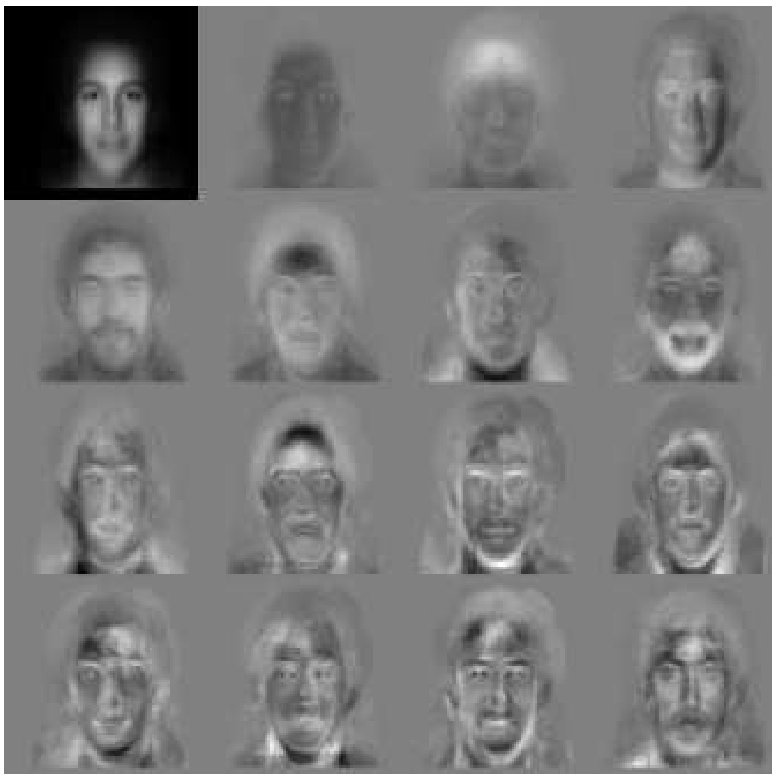
Precision-Recall Curves:

- precision vs true detection rate (sensitivity)
- better whan ROC when the $N_{neg} \gg N_{pos}$, so even a low false positive rate can yield many more false alarms than hits.
- that's why precision divides true hits by total number of hits rather than total number of positives.

Face Detection

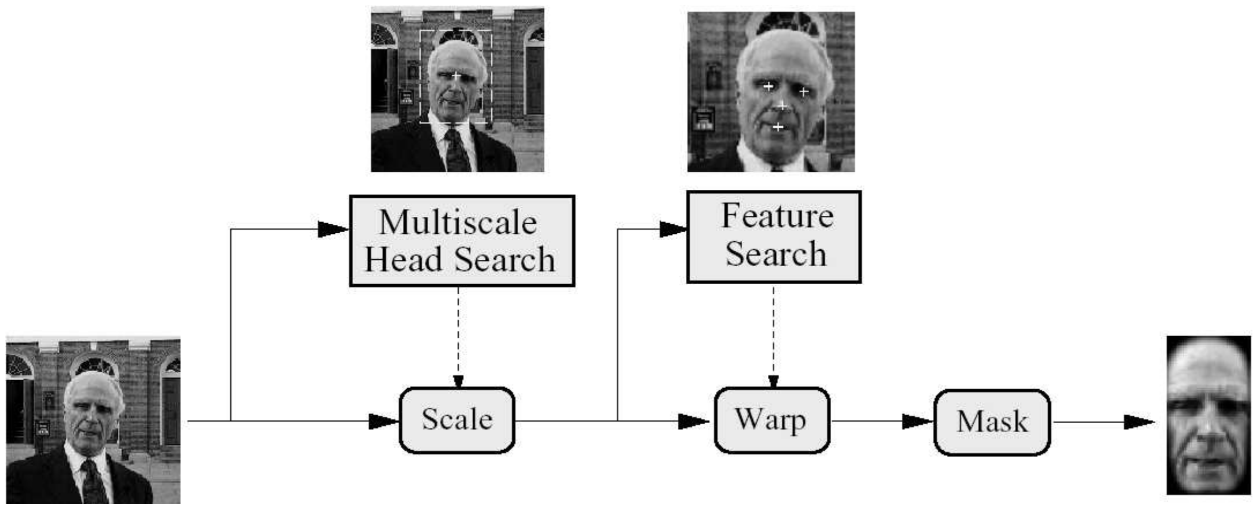
The wide-spread use of PCA for object recognition began with the work Turk and Pentland (1991) for face detection and recognition.

Shown below is the model learned from a collection of frontal faces, normalized for contrast, scale, and orientation, with the backgrounds removed prior to PCA.



Here are the mean image (upper-left) and the first 15 eigen-images. The first three show strong variations caused by illumination. The next few appear to correspond to the occurrence of certain features (hair, hairline, beard, clothing, etc).

Face Detection/Recognition



Moghaddam, Jebara and Pentland (2000): Subspace methods are used for head detection and then feature detection to normalize (warp) the facial region of the image.

Recognition: Are these two images (test and target) the same?

Approach 1: *Single Image Subspace Recognition:*
Project test and target faces onto the face subspace, and look at distance within the subspace.

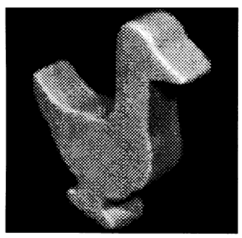
Approach 2: *Intra/Extra-Personal Subspace Recognition:*

- An intra-personal subspace is learned from difference images of the same persion under variation in lighting and expression.
- The extra-personal subspace learned from difference between images of different people under similar conditions.

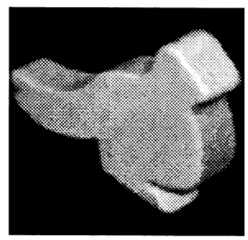
Object Recognition

Murase and Nayar (1995)

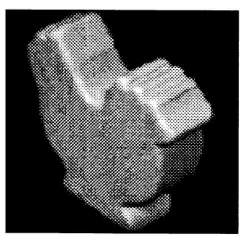
- images of multiple objects, taken from different positions on the viewsphere
- each object occupies a manifold in the subspace (as a function of position on the viewsphere)
- recognition: nearest neighbour assuming dense sampling of object pose variations in the training set.



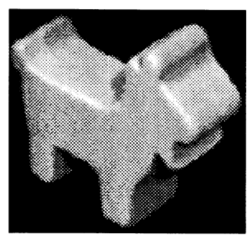
A



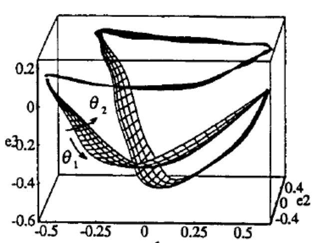
B



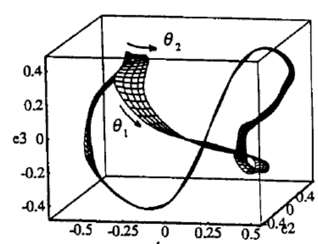
C



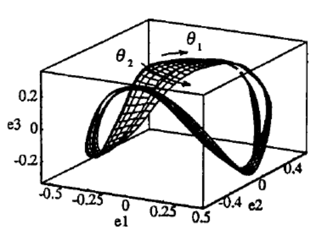
D



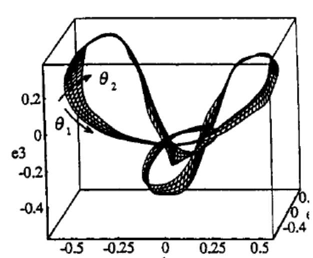
A



B



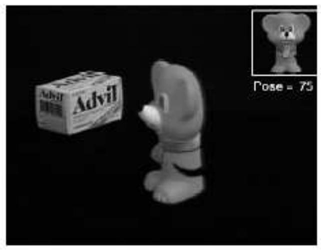
C

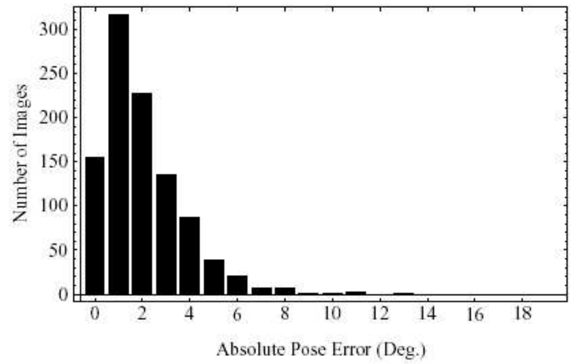


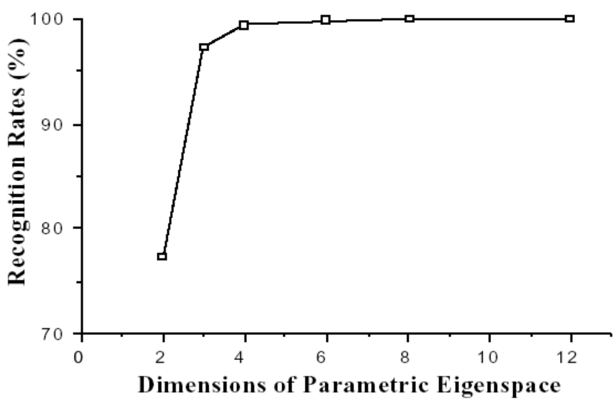
D

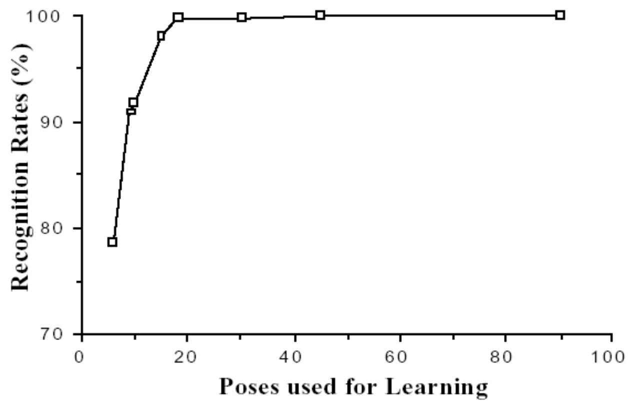
Object Recognition

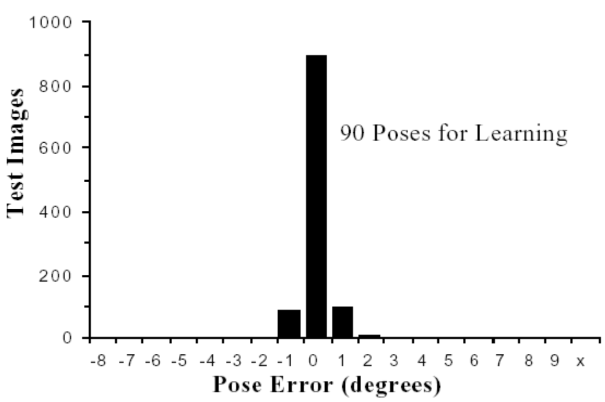
Object Recognition

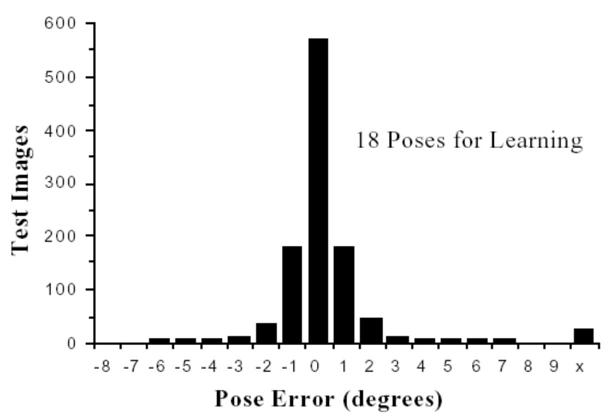












Quadratic Discriminant Analysis (QDA)

Classification. PCA was originally optimized for good reconstruction, for maximizing variance and decorrelating subspace coefficients, but not for classification.

Suppose we have a mixture of Gaussian model for an image $\vec{d}$

$$p(\vec{d}\,) = p(M_1)G(\vec{d}; \vec{\mu}_1, C_1) \; + \; p(M_2)G(\vec{d}; \vec{\mu}_2, C_2).$$

Here $p(M_1) + p(M_2) = 1$, with $p(M_k)$ the prior for model M_k. The ownership probability for $\vec{d}$ is then

$$p(M_k|\vec{d}) \; = \; \frac{p(M_k)G(\vec{d}; \vec{\mu}_k, C_k)}{p(\vec{d})}.$$

Finally, the log odds $\delta(\vec{d}\,)$ for model M_2 over M_1 is defined to be

$$\delta(\vec{d}) \; \equiv \; \log\left[\frac{p(M_2|\vec{d}\,)}{p(M_1|\vec{d}\,)}\right] \; = \; \log\left[\frac{p(M_2)|C_1|^{1/2}}{p(M_1)|C_2|^{1/2}}\right] \; + \; \frac{1}{2}\left[(\vec{d} - \vec{\mu}_1)^T C_1^{-1}(\vec{d} - \vec{\mu}_1) - (\vec{d} - \vec{\mu}_2)^T C_2^{-1}(\vec{d} - \vec{\mu}_2)\right]$$

A natural classifier for M_2 is to threshold the log odds. Simplifying the previous expression, we see we should choose model M_2 if and only if

$$(\vec{d} - \vec{\mu}_1)^T C_1^{-1}(\vec{d} - \vec{\mu}_1) - (\vec{d} - \vec{\mu}_2)^T C_2^{-1}(\vec{d} - \vec{\mu}_2) > \tau_q. \qquad (6)$$

Here τ_q is a selected constant.

Notice this decision boundary is a quadratic surface in $\vec{d}$ space.

Linear Discriminant Analysis (LDA)

In the special case where the two classes have the same covariance, i.e., $C_1 = C_2$, we see the quadratic terms in (6) cancel. As a result, the decision criterion simplifies to (for some constant τ_l)

$$\vec{d}^T C_1^{-1}(\vec{\mu}_2 - \vec{\mu}_1) > \tau_l. \tag{7}$$

Notice this decision boundary is simply a hyperplane in $\vec{d}$ space.

Often we assume the form of the decision boundary (e.g., linear or quadratic), and fit the parameters to maximize the log-odds of some training data (e.g., see [Hastie et al, 2001]).

For our purposes here,

- we note the direct connection of both LDA and QDA with thresholding the ownership probability of a Gaussian mixture model, with one Gaussian component modeling each class;
- LDA and QDA both generate simple decision boundaries (cf. the eigenspace plots from Murase & Nayar, on p.20).

Towards Better Detectors

PCA Summary

- PCA finds the subspace (of a specified dimension) that maximizes (projected) signal variance.
- A single Gaussian model is naturally associated with a PCA representation. The principal axes are the principal directions of the Gaussian's covariance.

Issues:

- The single Gaussian model is often rather crude. PCA coeff's can exhibit significantly more structure (cf. Murase & Nayar).
- As a result of this unmodelled structure, detectors based on single Gaussian models are often poor.

Alternatives:

- An alternative approach is to consider warped and aligned view based models (see Cootes, Edwards, & Taylor, 1998).
- Richer density models of the subspace coefficients are possible (e.g., nearest neighbour as in Murase & Nayar, or mixture models).
- More flexible detectors are very successful (see Viola & Jones, 2004).

Binary Classification Problem

Given training data $\{\vec{x}_k, y_k\}_{k=1}^{K}$, where

- $\vec{x}_k \in \Re^d$ is the feature vector for the k^{th} data item,
- $y_k \in \{-1, 1\}$ denotes the class membership of the k^{th} item $\vec{x}_k$,

we seek a classifier $F(\vec{x})$ such that $y(x) \equiv \mathrm{sign}(F(\vec{x}))$ approximates (in some sense) the training data.

In particular, on the training data, the given class indicator y_k should agree with the model $\mathrm{sign}(F(\vec{x}_k))$ as much as possible.

AdaBoost is an algorithm for greedily training classifiers $F(\vec{x})$ which take the form of additive linear models.

Additive Linear Models

An additive linear model has the form

$$F_m(\vec{x}) = \sum_{j=1}^{m} \alpha_j f_j(\vec{x}; \vec{\theta}_j) \tag{8}$$
$$= F_{m-1}(\vec{x}) + \alpha_m f_m(\vec{x}; \vec{\theta}_m).$$

Here $m \geq 1$ and

- $F_m(\vec{x})$ is a weighted (i.e. α_j) sum of simpler functions $f_j(\vec{x}; \vec{\theta}_j)$.
- Note the simpler functions depend on parameters $\vec{\theta}_j$, which we need to fit along with the weights α_j.
- Here we take the simpler functions $f_j(\vec{x}; \vec{\theta}_j)$ to be weak classifiers, providing values in $\{-1, 1\}$.
- We use $F_0(\vec{x}) \equiv 0$ in the recursive definition above.

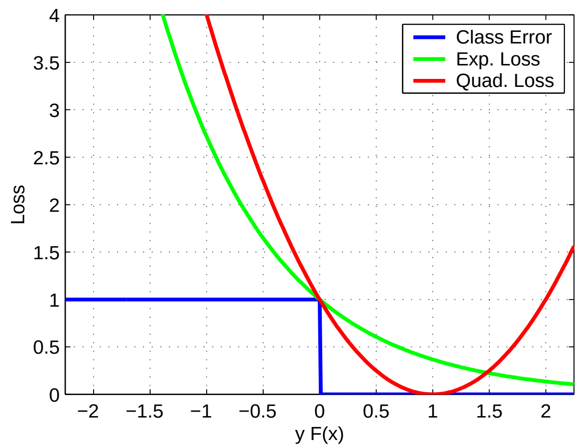
Eigen-appearance models, wavelets, and mixtures of Gaussians models provide simple examples of additive linear models (although, the basis functions are not simply weak classifiers, as they are here).

Exponential Loss

We seek a model $F_m(\vec{x})$ such that $\mathrm{sign}(F_m(\vec{x}_k))$ typically agrees with the class indicator $y_k \in \{-1, 1\}$ in the training data.

How should we measure agreement/disagreement?

Since y_k should have the same sign as $F_m(\vec{x}_k)$, it is convenient to consider $y_k F_m(\vec{x}_k)$, which should be positive.



Possible loss (cost) functions of $z \equiv yF(\vec{x})$ are:

- **Classification Error.** $C(z) = 1$ if $z <= 0$, else 0. Hard to optimize due to discontinuity.
- **Quadratic Loss.** $C(z) = (z-1)^2$. Easy to optimize but penalizes $F(\vec{x})$ for being the correct sign but too large (i.e. confident and correct).
- **Exponential Loss.** $C(z) = exp(-z)$. Smooth and monotonic in z. Large costs for $F(\vec{x})$ large in absolute value, but the wrong sign (i.e. confident and wrong, e.g. data outliers).

Greedy Fitting and AdaBoost

Suppose we have trained a classifier $F_{m-1}(\vec{x})$ with $m-1$ weak components, and wish to add one more linear component,

$$F_m(\vec{x}) \;=\; F_{m-1}(\vec{x}) + \alpha_m f_m(\vec{x}; \vec{\theta}_m).$$

Suppose we choose α_m and $\vec{\theta}_m$ to minimize the exponential loss

$$\begin{aligned}
\sum_{k=1}^{K} C(y_k F_m(\vec{x}_k)) \;&\equiv\; \sum_{k=1}^{K} e^{-y_k F_m(\vec{x}_k)} \\
&=\; \sum_{k=1}^{K} e^{-y_k F_{m-1}(\vec{x}_k)} e^{-y_k \alpha_m f_m(\vec{x}_k, \vec{\theta}_m)} \\
&=\; \sum_{k=1}^{K} w_k^{m-1} e^{-y_k \alpha_m f_m(\vec{x}_k, \vec{\theta}_m)}
\end{aligned}$$

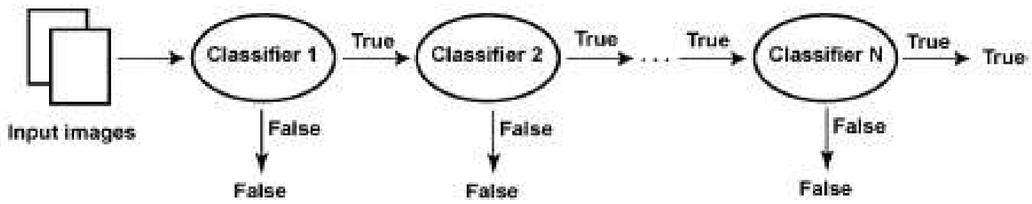
Here the weights $w_k^{m-1} = e^{-y_k F_{m-1}(\vec{x}_k)}$ are just the exponential losses for the previous function $F_{m-1}(\vec{x})$ on each training item.

Note the weights are largest for data points which the previous function $F_{m-1}(\vec{x})$ confidently classifies incorrectly (i.e. $y_k F_{m-1}(\vec{x}_k)$ significantly negative), and are smallest for points confidently classified correctly (i.e. $y_k F_{m-1}(\vec{x}_k)$ significantly positive).

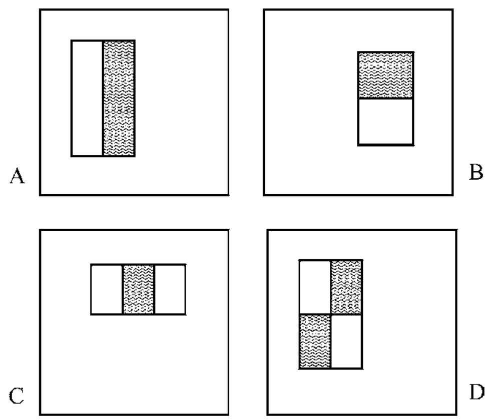
This greedy fitting of the weak classifiers in an additive model leads to the AdaBoost learning algorithm (see Friedman et al, 2000).

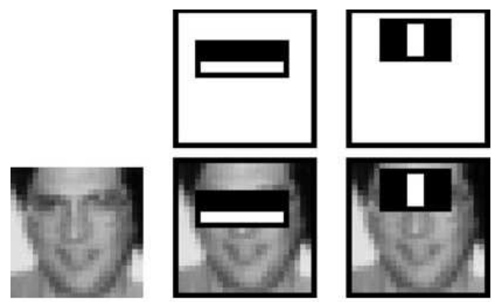
Viola and Jones Face Detector

Rejection cascade architecture:

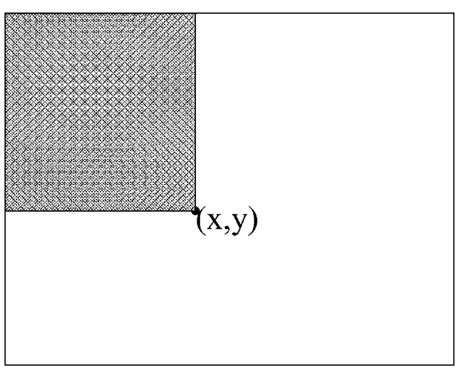


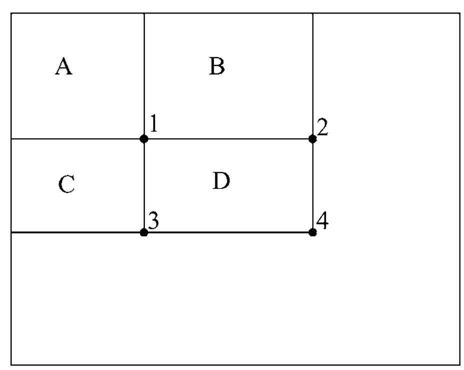
Features are formed from Haar filters...





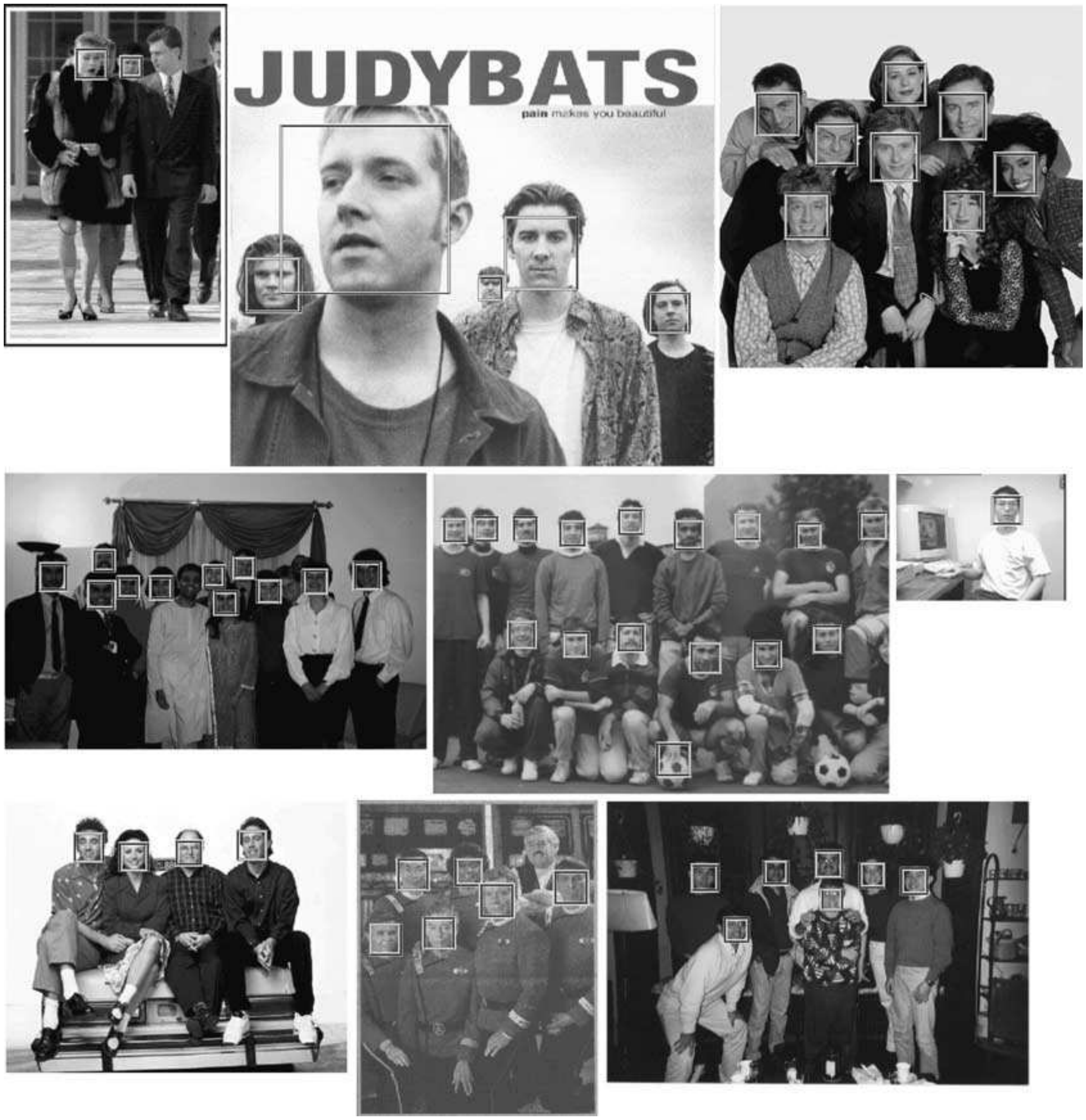
These features can be computed extremely rapidly using integral images.





The result is a real-time face detector with good classification performance (Viola and Jones, 2004).

Viola and Jones, Results



Further Readings

Belhumeur et al (1997) Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. PAMI*, 19(7):711-720

Chennubhotla, C. and Jepson, A.D. (2001) Sparse PCA (S-PCA): Extracting multi-scale structure from data. *Proc. IEEE ICCV*, Vancouver, pp. 641-647.

T. Cootes, G. Edwards, and C.J. Taylor, Active Appearance Models, *Proc. IEEE ECCV*, 1998.

J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: a statistical view of boosting, Annals of Statistics 28, 2000, pp. 337-407.

Golub, G. and van Loan, C. (1984) Matrix Computations. Johns Hopkins Press, Baltimore.

T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction, Springer, 2001.

Murase, H. and Nayar, S. (1995) Visual learning and recognition of 3D objects from appearance. *Int. J. Computer Vision* 14:5–24.

M. Black and A. Jepson (1998) EigenTracking: Robust matching and tracking of articulated objects using a view-based representation. *Int. J. Computer Vision* 26(1):63-84.

Moghaddam, B., Jebara, T. and Pentland, A. (2000) Bayesian face recognition. *Pattern Recognition*, 33(11):1771-1782

Turk, M. and Pendland, A. (1991) Face recognition using eigenfaces, *J. Cognitive Neuroscience*, 3(1):71–86.

P. Viola, and M.J. Jones, Robust real-time face detection, Int. J. of Computer Vision, 2004.