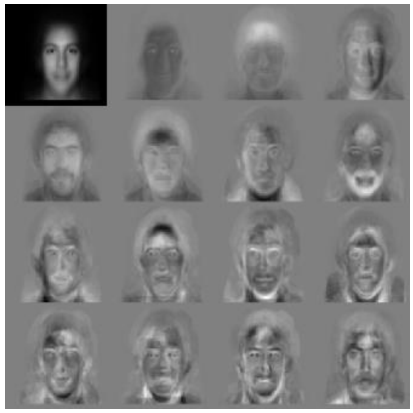
CS4670/5670: Intro to Computer Vision

Noah Snavely

Eigenfaces



What makes face recognition hard?

Expression

slide courtesy from Derek Hoiem

What makes face recognition hard?

Lighting

slide courtesy from Derek Hoiem

What makes face recognition hard?

Occlusion

slide courtesy from Derek Hoiem

What makes face recognition hard?

Viewpoint

slide courtesy from Derek Hoiem

Face detection

- Do these images contain faces? Where?

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

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

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

slide courtesy from Derek Hoiem

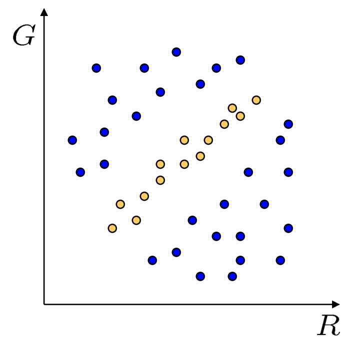
The space of all face images

- 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

slide courtesy from Derek Hoiem

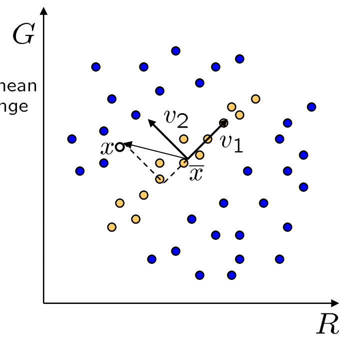
The space of all face images

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



Linear subspaces

$\overline{x}$ is the mean of the orange points



convert **x** into $\mathbf{v_1}$, $\mathbf{v_2}$ coordinates

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

What does the $\mathbf{v_2}$ coordinate measure?

- distance to line
- use it for classification—near 0 for orange pts

What does the $\mathbf{v_1}$ coordinate measure?

- position along line
- use it to specify which orange point it is

Classification can be expensive

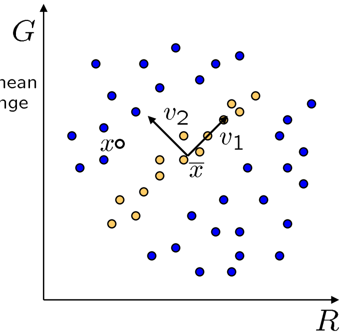
- Must either search (e.g., nearest neighbors) or store large PDF's

Suppose the data points are arranged as above

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

Dimensionality reduction

$\overline{x}$ is the mean of the orange points



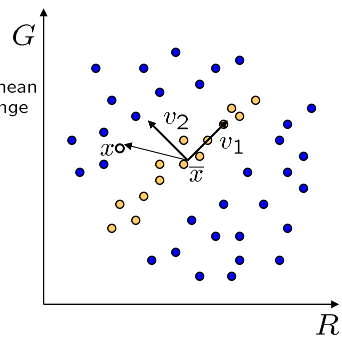
How to find $\mathbf{v_1}$ and $\mathbf{v_2}$?

Dimensionality reduction

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

Linear subspaces

$\overline{x}$ is the mean of the orange points



Consider the variation along direction $\mathbf{v}$ among all of the orange points:

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

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

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

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

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

$$\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\|^2 \\
&= \sum_{\mathbf{x}} \mathbf{v}^{\mathrm{T}}(\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}}\mathbf{v} \\
&= \mathbf{v}^{\mathrm{T}} \left[\sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}}\right] \mathbf{v} \\
&= \mathbf{v}^{\mathrm{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}}
\end{aligned}$$

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

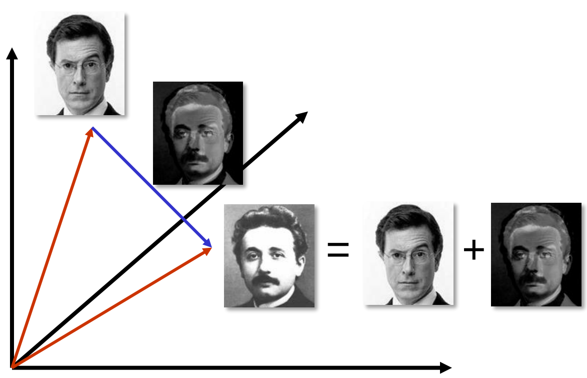
Suppose each data point is N-dimensional

- Same procedure applies:

$$\begin{aligned} var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\| \\ &= \mathbf{v}^{\mathrm{T}} \mathbf{A} \mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \end{aligned}$$

- 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
- We can compress the data by only using the top few eigenvectors
  - corresponds to choosing a “linear subspace”
    - » represent points on a line, plane, or “hyper-plane”
  - these eigenvectors are known as the ***principal components***

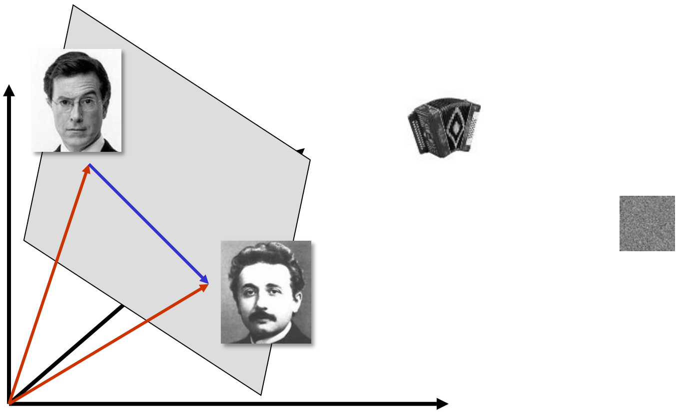
The space of faces



An image is a point in a high dimensional space

- An N x M intensity image is a point in R^{NM}
- We can define vectors in this space as we did in the 2D case

Dimensionality reduction



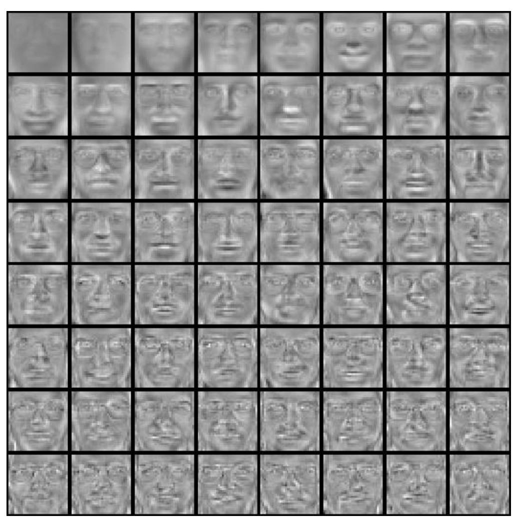
The set of faces is a "subspace" of the set of images

- Suppose it is K dimensional
- We can find the best subspace using PCA
- This is like fitting a "hyper-plane" to the set of faces
  - spanned by vectors $\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_K}$
  - any face $\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_k\mathbf{v_k}$

Eigenfaces example

Mean: μ

Top eigenvectors: $u_1, \ldots u_k$



slide courtesy from Derek Hoiem

Representation and reconstruction

- Face **x** in "face space" coordinates:

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

slide courtesy from Derek Hoiem

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:

 =  + 

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

slide courtesy from Derek Hoiem

Reconstruction

P = 4

P = 200

P = 400

After computing eigenfaces using 400 face images from ORL face database

slide courtesy from Derek Hoiem

Detection and recognition with eigenfaces

Algorithm

1. Process the image database (set of images with labels)
   - Run PCA—compute eigenfaces
   - Calculate the K coefficients for each image
2. Given a new image (to be recognized) **x**, calculate K coefficients

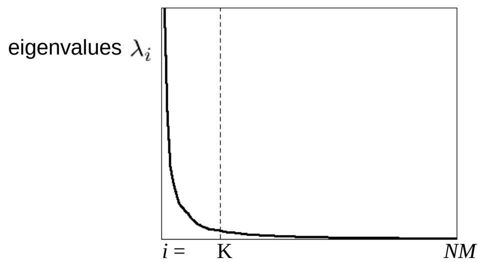
$$\mathbf{x} \rightarrow (a_1, a_2, \ldots, a_K)$$

3. Detect if x is a face

$$\|\mathbf{x} - (\overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K})\| < \text{threshold}$$

4. If it is a face, who is it?
   - Find closest labeled face in database
     - nearest-neighbor in K-dimensional space

Choosing the dimension K



How many eigenfaces to use?

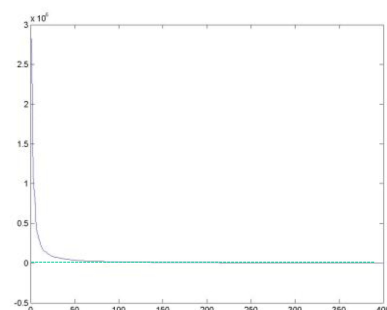
Look at the decay of the eigenvalues

- the eigenvalue tells you the amount of variance “in the direction” of that eigenface
- ignore eigenfaces with low variance

Note

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

P = 200



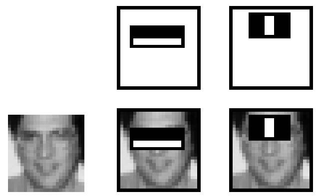
slide courtesy from Derek Hoiem

Issues: metrics

What's the best way to compare images?

- need to define appropriate features
- depends on goal of recognition task

exact matching
complex features work well
(SIFT, MOPS, etc.)



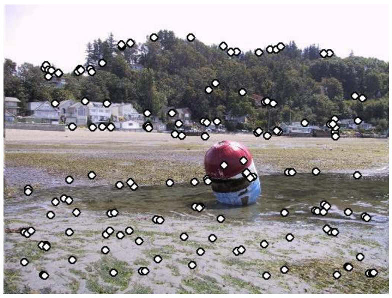
classification/detection
simple features work well
(Viola/Jones, etc.)

Metrics

Lots more feature types that we haven't mentioned

- moments, statistics
  - metrics: Earth mover's distance, ...
- edges, curves
  - metrics: Hausdorff, shape context, ...
- 3D: surfaces, spin images
  - metrics: chamfer (ICP)
- ...

Issues: feature selection



If all you have is one image:
non-maximum suppression, etc.

If you have a training set of images:
AdaBoost, etc.

Issues: data modeling

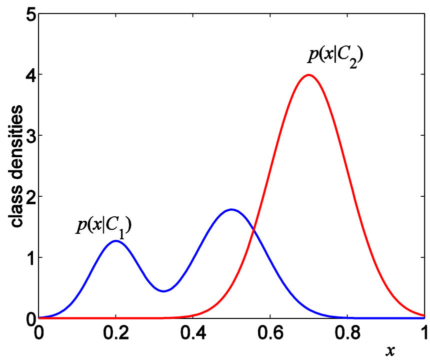
Generative methods

- model the “shape” of each class
  - histograms, PCA, mixtures of Gaussians
  - graphical models (HMM’s, belief networks, etc.)
  - ...

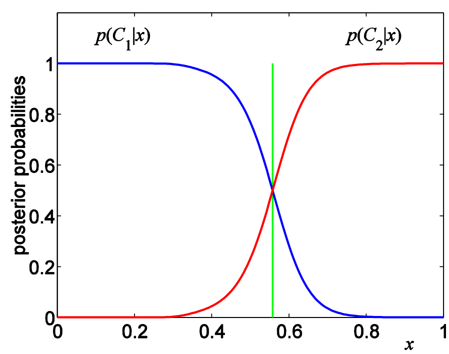
Discriminative methods

- model boundaries between classes
  - perceptrons, neural networks
  - support vector machines (SVM’s)

Generative vs. Discriminative



Generative Approach
model individual classes, priors



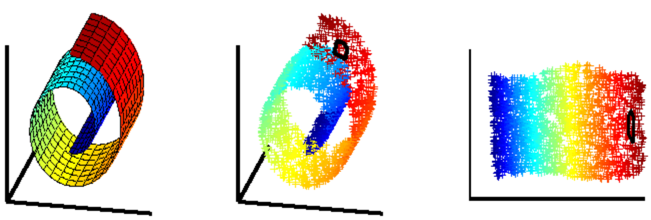
Discriminative Approach
model posterior directly

from Chris Bishop

Issues: dimensionality

What if your space isn't *flat*?

- PCA may not help



Nonlinear methods
LLE, MDS, etc.

Moving forward

- Faces are pretty well-behaved
  - Mostly the same basic shape
  - Lie close to a low-dimensional subspace
- Not all objects are as nice

Different appearance, similar parts