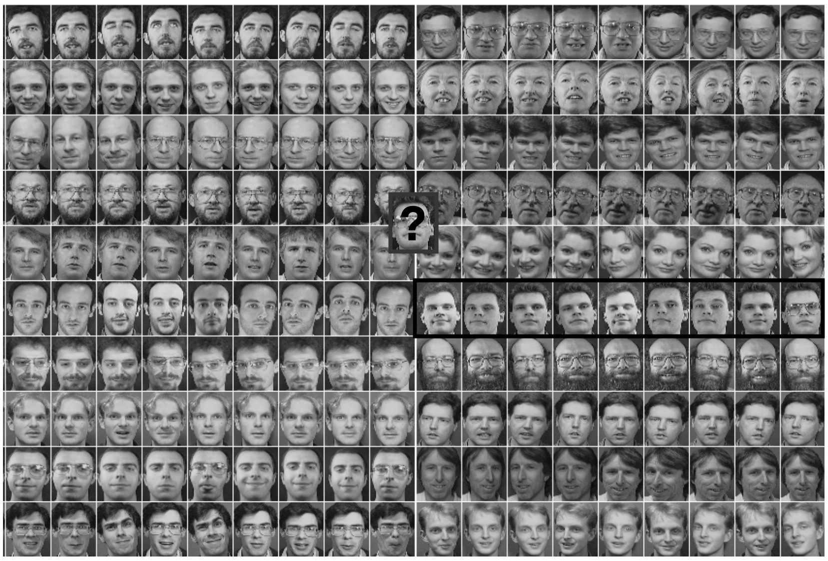
Eigenface

Outline

- Performance measurement
- Dimensionality reduction
- Face representation
  - Eigenfaces

Face Classification



20 faces (i.e., classes), 9 examples (i.e., training data) of each

Measuring Performance

- Classification accuracy:
  - The percentage of correctly labeled data by a classifier
- Validation: Split experimental data into training and test set
  - **Training set**: Data points used to build a classifier
  - **Test set**: Data points left out of training procedure

Cross-Validation

- m-fold **cross-validation**
  - Randomly split data into m equal-sized subsets
  - Train m times on m - 1 subsets, test on left-out subset
  - Error is mean test error over left-out subsets
- **Leave-one-out**: Cross-validation with 1-point subsets
  - Very accurate but expensive; variance allows confidence measuring

A Simple Method

- Idea: Search over training set for most similar image (e.g., in SSD sense) and choose its class
- This is the same as a 1-nearest neighbor classifier when feature space = image space
- Issues
  - Large storage requirements (nd, where n = image space dimensionality and d = number of faces in training set)
  - Correlation is computationally expensive

Dimension reduction: Motivation

- Using images themselves as feature vectors is easy, but has problem of high dimensionality
  - A 100 x 100 image = 10,000-dimensional feature space!
- We want features that stay in low-dimensional spaces and result in well-separated classes

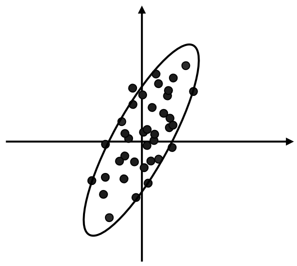
Dimensionality Reduction: Projection

- Two popular schemes
  - Principal components analysis (PCA): Projection maximizing total variance of data

$$S_T = \sum_{k=1}^{N} (x_k - u)(x_k - u)^T$$

  - Fisher's Linear Discriminant (FLD): Maximize ratio of between-class variance to within-class variance

Geometric Interpretation



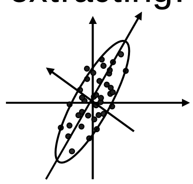
Covariance $\mathrm{C} = \mathrm{X}\,\mathrm{X}^T$

where $\mathbf{X}$ is zero-mean

adapted from Z. Dodds

Geometric Factorization of Covariance

- SVD of covariance matrix $\mathrm{C} = \mathrm{R}\ \mathrm{D}\ \mathrm{R}^T$ describes geometric components of transform by extracting:
  - Diagonal scaling matrix D
  - Rotation matrix R



- E.g., given points $\mathrm{x} = \begin{pmatrix} 2 & -2 & 1 & -1 \\ 5 & -5 & -1 & 1 \end{pmatrix}$, the covariance factors as

cos, sin of 70°

$$\mathrm{X}\,\mathrm{X}^{\mathrm{T}} = \begin{pmatrix} 2.5 & 5 \\ 5 & 13 \end{pmatrix} = \begin{pmatrix} .37 & -.93 \\ .93 & .37 \end{pmatrix} \begin{pmatrix} 15 & 0 \\ 0 & .5 \end{pmatrix} \begin{pmatrix} .37 & .93 \\ -.93 & .37 \end{pmatrix}$$

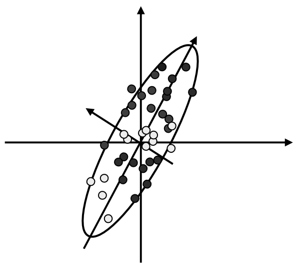
major, minor axis lengths

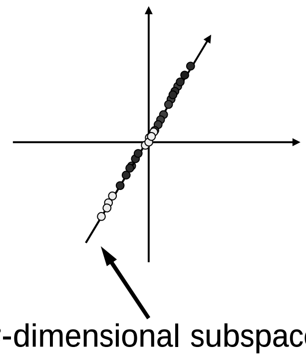
adapted from Z. Dodds

PCA for Dimensionality Reduction

- Any point in n-dimensional original space can thus be expressed as a linear combination of the n eigenvectors (the rows of $\mathbf{R}$) via a set of weights $[\omega_1\ ,\ \omega_2,\ \ldots,\ \omega_n]$
- By projecting points onto only the first k << n principal components (eigenvectors with the largest eigenvalues), we are essentially throwing away the least important feature information

Projection onto Principal Components





Full n-dimensional space (here n = 2)

k-dimensional subspace (here k = 1)

adapted from Z. Dodds

Eigenfaces

- Idea: Compress image space to "face space" by projecting onto principal components ("eigenfaces" = eigenvectors of image space)
  - Represent each face as a low-dimensional vector (weights on eigenfaces)
  - Measure similarity in face space for classification
- Advantage: Storage requirements are nk instead of nd

Eigenfaces: Initialization

- Calculate eigenfaces
  - Compute n–dimensional mean face $\mathbf{\Psi}$
  - Compute difference of every face from mean face $\mathbf{\Phi}_j = \mathbf{\Gamma}_j$ - $\mathbf{\Psi}$
  - Form covariance matrix of these $\mathbf{C} = \mathbf{A}\mathbf{A}^T$, where $\mathbf{A} = [\mathbf{\Phi}_1, \mathbf{\Phi}_2, \ldots, \mathbf{\Phi}_d]$
  - Extract eigenvectors $\mathbf{u}_i$ from $\mathbf{C}$ such that $\mathbf{C}\mathbf{u}_i = \lambda_i \mathbf{u}_i$
  - Eigenfaces are k eigenvectors with largest eigenvalues

Example eigenfaces

Eigenfaces: Initialization

- Project faces into face space
  - Get eigenface weights for every face in the training set
  - The weights $[\omega_{j1}\ , \omega_{j2},\ \ldots,\ \omega_{jk}]$ for face j are computed via dot products $\omega_{ji} = \mathbf{u}_i^T \mathbf{\Phi}_j$

Calculating Eigenfaces

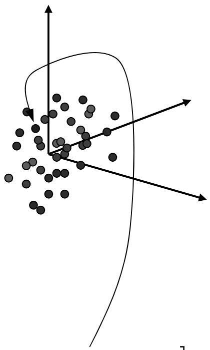
- Obvious way is to perform SVD of covariance matrix, but this is often prohibitively expensive
  - E.g., for 128 x 128 images, $\mathbf{C}$ is 16,384 x 16,384
- Consider eigenvector decomposition of d x d matrix $\mathbf{A}^T\mathbf{A}$: $\mathbf{A}^T\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$. Multiplying both sides on the left by $\mathbf{A}$, we have

$$\boxed{\mathbf{C}}\,\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{A}\mathbf{v}_i$$

- So $\mathbf{u}_i = \mathbf{A}\mathbf{v}_i$ are the eigenvectors of $\mathbf{C} = \mathbf{A}\mathbf{A}^T$

Eigenfaces: Recognition

- Project new face into face space
- Classify
  - Assign class of nearest face from training set



$[\omega_1 , \omega_2, \ldots, \omega_8]$

Original face 8 eigenfaces Weights

adapted from Z. Dodds