Announcements

- HW1 returned
- New attendance policy
  - On time: 1 point
  - Five minutes or more late: 0.5 points
  - Absent: 0 points

Example: Face Detection

- Scan window over image.
- Classify window as either:
  - Face
  - Non-face

Discriminative Model

- Learn face and non-face models from examples
  \( P(\text{face} | \text{Window}) \) and \( P(\text{non-face} | \text{Window}) \)
- Cluster samples of each class to create subclasses, and project the examples to a lower dimensional space based on multi-discriminant analysis.
- Detect faces in lower-dimensional space when \( P(\text{Face} | \text{Window}) > P(\text{Non-face} | \text{Window}) \)
- Add non-face examples using bootstrapping

“State of the Art Method:” Viola Jones

- [Viola and Jones CVPR 01]:
  - A face is modeled as a set of Harr-like features
  - A fast way to compute simple rectangle features
  - Use Adaboost to focus on a small set of features
  - Cascade of simple classifiers
- Error rate comparable to the best
- Fast: 15 Fps, 700Mhz Pentium, half resolution video images

[Sung and Poggio 98]
Image as a Feature Vector

- Consider an n-pixel image to be a point in an n-dimensional space, \( x \in \mathbb{R}^n \).
- Each pixel value is a coordinate of \( x \).

Nearest Neighbor Classifier

\( \{ R_j \} \) are set of training images.
\( ID = \arg \min \text{dist}(R_j, I) \)

Comments

- Sometimes called “Template Matching”
- Variations on distance function (e.g. L₁, robust distances)
- Multiple templates per class- perhaps many training images per class.
- Expensive to compute k distances, especially when each image is big (N dimensional).
- May not generalize well to unseen examples of class.
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction

The Curse of Dimensionality

Eigenface (Turk, Pentland, 91) -1

- Use Principle Component Analysis (PCA) to determine the most discriminating features between images of faces.

Eigenfaces: linear projection

- An n-pixel image \( x \in \mathbb{R}^n \) can be projected to a low-dimensional feature space \( y \in \mathbb{R}^m \) by
  \[ y = Wx \]
  where \( W \) is an \( n \) by \( m \) matrix.
- Recognition is performed using nearest neighbor in \( \mathbb{R}^m \).
- How do we choose a good \( W \)?
Eigenfaces: Principal Component Analysis (PCA)

Assume we have a set of n feature vectors \( \{x_1, \ldots, x_n\} \) in \( \mathbb{R}^d \). Write

\[
\mu = \frac{1}{n} \sum x_i, \\
\Sigma = \frac{1}{n-1} \sum (x_i - \mu)(x_i - \mu)^T
\]

The unit eigenvectors of \( \Sigma \) — which we write as \( v_1, v_2, \ldots, v_n \), where the order is given by the size of the eigenvalue and \( v_1 \) has the largest eigenvalue — give a set of features with the following properties:

- They are independent.
- Projection onto the basis \( \{v_1, \ldots, v_n\} \) gives the d-dimensional set of linear features that preserves the most variance.

Some details:

- How big is \( \Sigma \)?
- Use Singular value decomposition, "trick" to compute basis when \( n < d \)

How do you construct Eigenspace?

[\[ x_1 \ x_2 \ x_3 \ x_4 \ x_5 \]]

Construct data matrix by stacking vectorized images and then apply Singular Value Decomposition (SVD)

Matrix Decompositions

- Definition: The factorization of a matrix \( M \) into two or more matrices \( M_1, M_2, \ldots, M_p \) such that \( M = M_1 \cdots M_p \).
- Many decompositions exist...
  - QR Decomposition
  - LU Decomposition
  - LDL Decomposition
  - Etc.

SVD Properties

- In Matlab \([U \Sigma V] = \text{svd}(A)\), and you can verify that: \( A = U \Sigma V^T \)
- \( r = \text{Rank}(A) = \# \) of non-zero singular values.
- \( U, V \) give us orthonormal bases for the subspaces of \( A \):
  - 1st \( r \) columns of \( U \): Column space of \( A \)
  - Last \( n-r \) columns of \( U \): Left nullspace of \( A \)
  - 1st \( r \) columns of \( V \): Row space of \( A \)
  - Last \( n-r \) columns of \( V \): Nullspace of \( A \)
- For \( d < r \), the first \( d \) column of \( U \) provide the best \( d \)-dimensional basis for columns of \( A \) in least squares sense.

Singular Value Decomposition

Excellent ref: ‘Matrix Computations,” Golub, Van Loan

- Any \( m \) by \( n \) matrix \( A \) may be factored such that \( A = U \Sigma V^T \)
- \( U: m \) by \( m \), orthogonal matrix
  - Columns of \( U \) are the eigenvectors of \( AA^T \)
- \( V: n \) by \( n \), orthogonal matrix,
  - Columns are the eigenvectors of \( A^TA \)
- \( \Sigma: m \) by \( n \), diagonal with non-negative entries (\( \sigma_1, \sigma_2, \ldots, \sigma_s \)) with \( s = \min(m,n) \) are called the singular values
  - Singular values are the square roots of eigenvalues of both \( AA^T \) and \( A^TA \)
  - Result of SVD algorithm: \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_s \) (sorted by significance)

Thin SVD

- Any \( m \) by \( n \) matrix \( A \) may be factored such that \( A = U \Sigma V^T \)
- If \( m > n \), then one can view \( \Sigma \) as:
  - \( \Sigma = \begin{bmatrix} \Sigma \ 0 \end{bmatrix} \)
  - Where \( \Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_s) \) with \( s = \min(m,n) \), and lower matrix is \( (n-m) \times m \) of zeros.
- Alternatively, you can write: \( A = U \Sigma V^T \)
- In Matlab, thin SVD is \([U \ \Sigma \ V] = \text{svds}(A)\)
Performing PCA with SVD

- Singular values of $A$ are the square roots of eigenvalues of both $AA^T$ and $A^TA$ & Columns of $U$ are corresponding Eigenvectors
- Given a collection of $n$ vectors $a_1, \ldots, a_n$
  \[
  \sum_{i=1}^{n} a_i a_i^T = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^T = AA^T
  \]
- Covariance matrix is:
  \[
  \Sigma = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T
  \]
- So, ignoring $1/n$ subtract mean image $\mu$ from each input image, create data matrix, and perform thin SVD on the data matrix and take top $k$ Columns of $U$.

Eigenfaces

- **Modeling**
  1. Given a collection of $n$ labeled training images,
  2. Compute mean image and covariance matrix.
  3. Compute $k$ Eigenvectors (note that these are images) of covariance matrix corresponding to $k$ largest Eigenvalues.
  4. Project the training images to the $k$-dimensional Eigenspace.
- **Recognition**
  1. Given a test image, project to Eigenspace.
  2. Perform classification to the projected training images.

Eigenfaces: Training Images

- [Turk, Pentland 91]
Projection, and reconstruction

- An $n$-pixel image $x \in \mathbb{R}^n$ can be projected to a low-dimensional feature space $y \in \mathbb{R}^m$ by $y = Wx$
- From $y \in \mathbb{R}^m$, the reconstruction of the point is $W^Ty$
- The error of the reconstruction is: $||x - W^TWx||$

Reconstruction using Eigenfaces

- Given image on left, project to Eigenspace, then reconstruct an image (right).

Underlying assumptions

- Background is not cluttered (or else only looking at interior of object)
- Lighting in test image is similar to that in training image.
- No occlusion
- Size of training image (window) same as window in test image.

Face detection using “distance to face space”

- Scan a window $\omega$ across the image, and classify the window as face/not face as follows:
  - Project window to subspace, and reconstruct as described earlier.
  - Compute distance between $\omega$ and reconstruction.
  - Local minima of distance over all image locations less than some threshold are taken as locations of faces.
  - Repeat at different scales.
  - Possibly normalize windows intensity so that $||\omega|| = 1$. 