Visual Recognition

P J Narayanan

CS5765. Computer Vision. Spring 2013

CVIT, IIIT, Hyderabad

Visual Object Recognition

- Recognize an **“object”** using visual cues alone
  - Abdul Kalam, Person
  - Taj Mahal, Building
  - TATA Nano, Car
  - Sukhoi Su30MKI, Airplane
  - iPhone 4S, Mobile Phone
- **Object Instance** vs **Object Category**
  - Question: **Is Manmohan Singh in this image?**
  - Question: **Is there a person in this image?**
- **Object Detection** vs **Object Recognition**
  - **Detection**: Is there a person? If yes, where?
  - **Recognition**: Who is that person?

Recognition: Challenges

Visual recognition is challenging because:

- Variation in appearance: Lighting, Clothing, Aging
- Variation in viewpoints: geometric distortions
- Occlusion: Recognizing 3D objects from 2D projections
- Intra-class variations: What is a chair?
- Semantic gap: Algorithms have no **“understanding”**
- Bulk of data to be processed: Computational difficulty
- And many other issues ....

Also because human vision is **very** good at recognition!

Variations on a Theme

Machine Vision

Recognition under highly controlled environments

- Clean, binarized images. Good available examples.
- Template matching: Compare image to image
  - Translation, rotation, and some scale invariance
- Boundary/area descriptors
  - Length, width, elongatedness, moments, etc.
  - Invariant properties
- Hough Transforms and Geometric Hashing
  - Match in a space where similarity implies locality
  - Parameter space in Hough transforms

Brush up basic image processing now!!

Recognition: Basic Steps

Two steps for everything to be done on a computer:

- **Representation**
  - Pixels and RGB values: Too low level
  - Higher levels: edges, lines, regions, parts, etc.
  - Generic vs specific
  - Abstraction using feature vectors
- **Processing**
  - Does the current example qualify?
  - Classification in a feature vector space

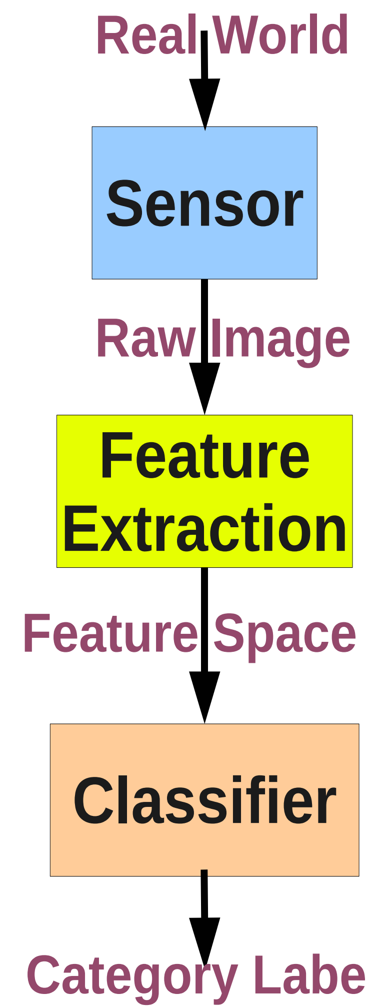
Strategies for Recognition

- Expert systems: Early attempts
  - Capture the knowledge of human experts!
    A set of rules that act on the input data
  - Highly limited in scope and possibilities.
    Is all knowledge captured?
- Syntactic recognition, graph matching
  - Describe objects as relations on symbols
  - Sensor imperfections or noise!
    Vision is analog; language processing is digital!
- Hashing: Find a function that maps identical objects to same bin and different objects to different bins!
- Data driven: Classification in a space of feature vectors

Brush up basic pattern recognition now!!

A Generic Recognition Pipeline

- Collect lots of representative, labelled samples, covering all variability in the data
- Extract **invariant** features that don't change under different observation conditions.
- Represent examples using **feature vectors**.
- Select a suitable **classifier**. Should be powerful enough to handle the problem.
- Train the classifier using examples
- Evaluate performance. Training/Testing examples. k-fold cross validation, leave one out cross validation.
- If satisfied, apply it to other examples. Hope the **generalization** is good!



Feature Vectors

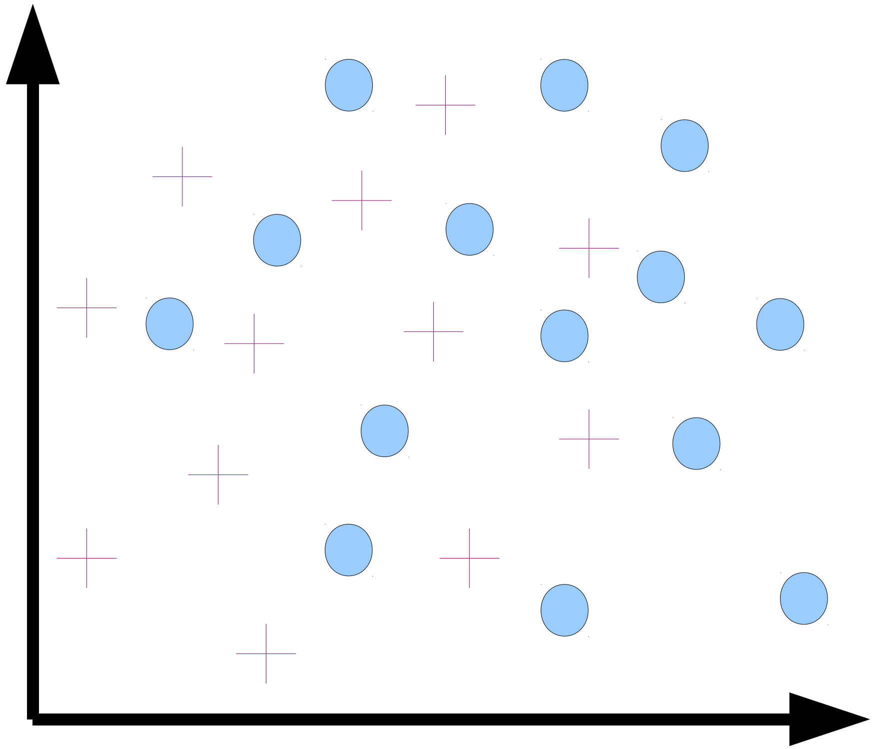
- Measure relevant properties, arrange as a vector
- **Invariance** is the key to recognition.
  Find properties that do not change significantly
  across the expected variations in the input
- Global features: Computed from the whole image or object. Histograms, color variations, etc.
- Local features: Computed from small neighbourhoods. Local histograms, local descriptors, etc.
- Feature dimensionality and issues. Classification boundaries can be complex in high dimensions. They may need more samples to estimate.

Classification

- Assign *classes* or *labels* to vectors
- Optimum classification: Bayesian classifier
  - Select class with maximum posterior probability
- Several classifiers or tools have appeared recently
  - Linear classifiers: Find lines between classes
    - Binary classifiers: **Yes/No** decisions
  - Artificial Neural Networks: some non-linearity
  - Support Vector Machines: Linear in a high dimensional space
  - Decision Trees: Branching based on feature values
  - K-Nearest Neighbour: Directly from samples without parameters

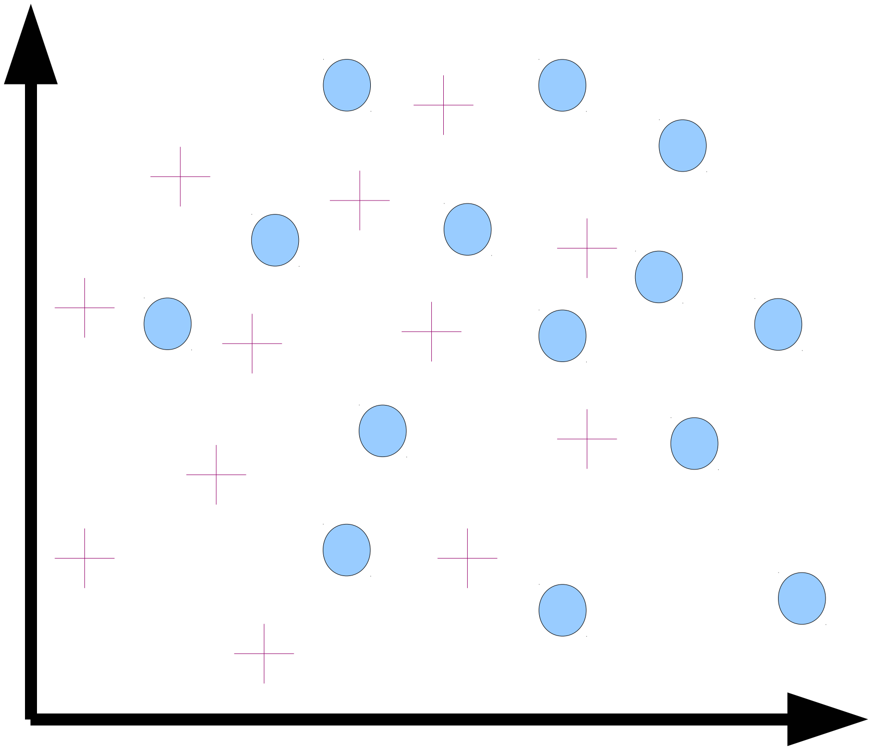
Generalization: Classifying unseen samples correctly!

Undesirable Situation

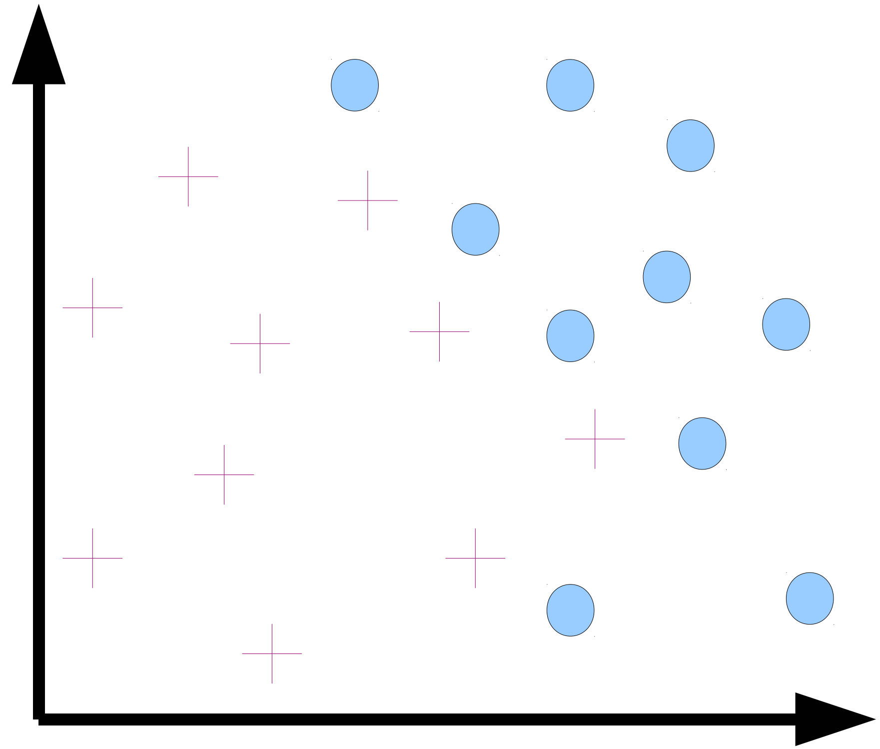


Bad feature space.

Better Situation

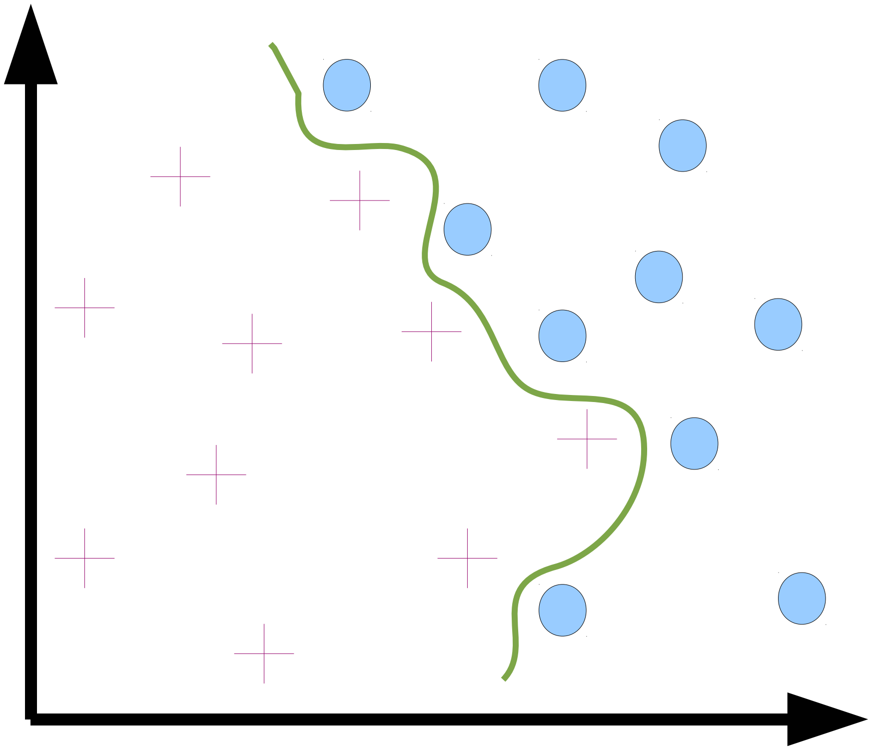


Bad feature space.



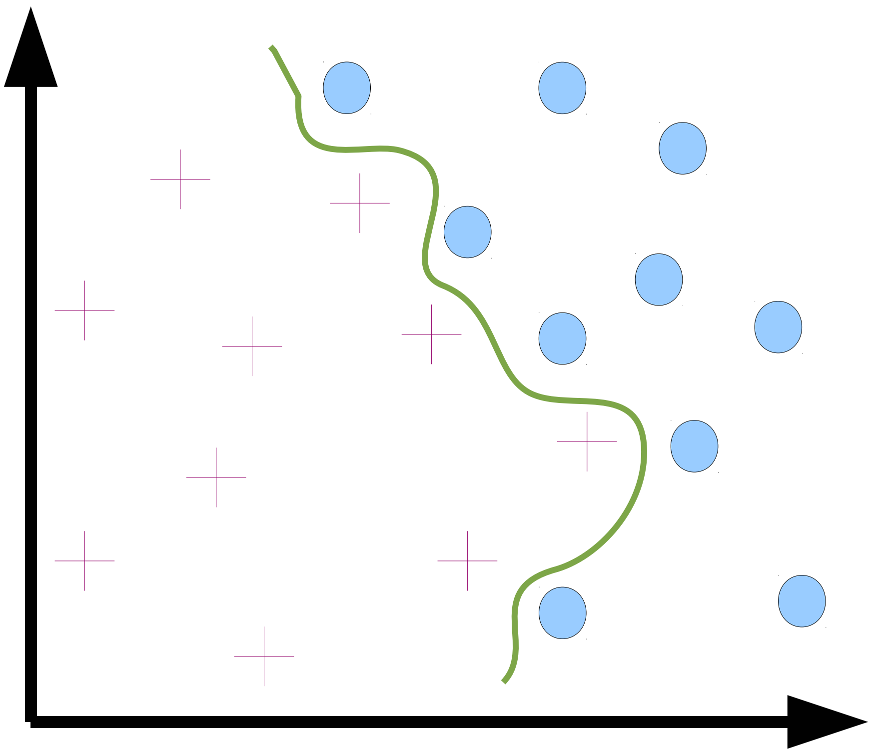
Better feature space

Undesirable Boundary

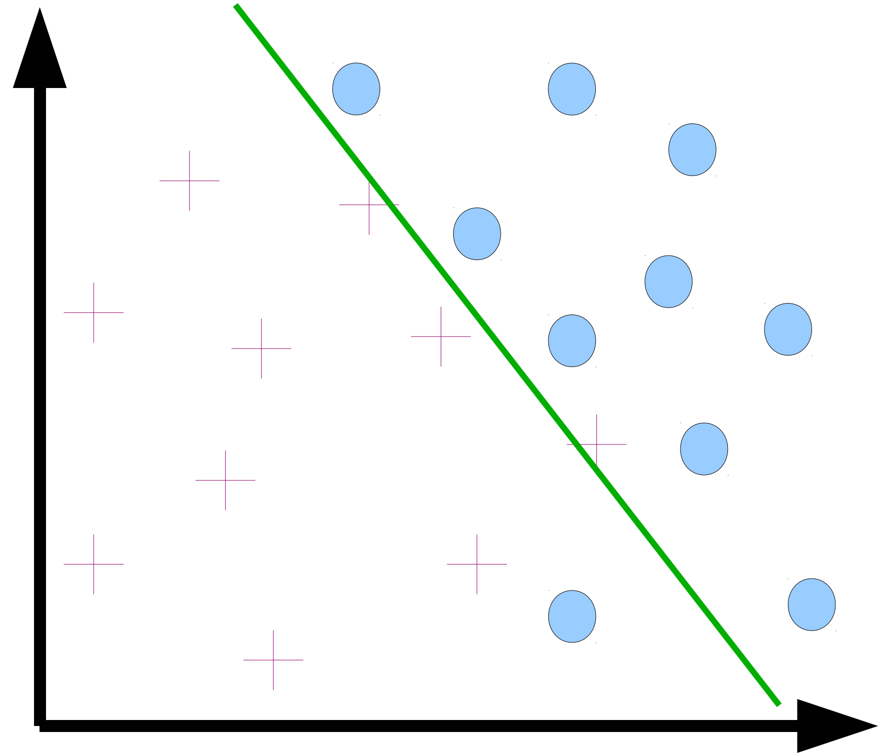


Bad classification boundary

Better Boundary



Bad classification boundary



Better boundary!

Better even with some mis-classifications!

Validation Schemes

- Training Error and Testing Error
  - Divide samples into training and testing sets. Train on the training set.
  - Training Error: Error on the training set
  - Testing Error: Error on the testing set
- k-fold cross validation
  - Divide examples into k “folds” or partitions
  - Choose each fold as testing set and train on rest
  - Evaluate error on testing fold.
  - Evaluate average error across the k iterations
- Leave One Out Cross Validation
  - Same as above, with $k = n$, the number of samples.
  - Very expensive computationally

Eigenfaces: Turk & Pentland 1991

- An important milestone in face recognition from MIT.
- Standard PCA-based technique or eigenanalysis.
- PCA for compact representations of faces in a **face space**. Recognition is (perhaps) easier in that space.
- Represent a face using all image pixels!!
  PCA will discover relevant “features” automatically!!
- Strength: Provides good recognition on a decent dataset.
- Weakness: Difficult to handle variation in lighting, pose, alignment.

Eigenfaces: Procedure

- Collect N face images with similar appearance. Front-facing, even lighting, neutral pose, etc.
- Crop and resize images to a common size and to have eyes, nose, and other features matching.
- Convert each image to an M-dimensional column vector of pixel values where M is the number of pixels.
- Subtract the mean vector for all images from each. Let a $\mathbf{A}$ be the $M \times N$ matrix of such vectors
- Let $\mathbf{S} = \mathbf{A}\mathbf{A}^{\mathbf{T}}$ be the covariance matrix of the vectors
- Find the eigenvalues of $\mathbf{S}$ and keep the top K of them. Corresponding eigenvectors are the **eigenfaces**.

Finding Eigenvalues

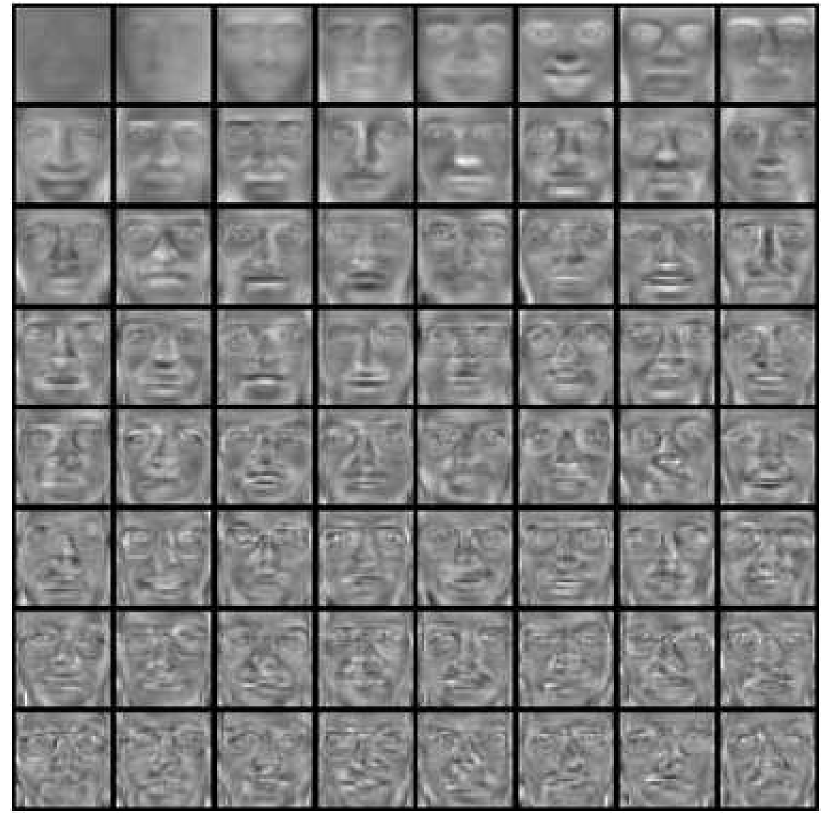
- Typically, N is 100 to 200 and M is around 10000
- $\mathbf{AA^T}$ has dimensions $M \times M$. Too large!
- $\mathbf{S\ e} = \lambda\ \mathbf{e}$ for eigenvectors. Or, $\mathbf{AA^T\ e} = \lambda\ \mathbf{e}$.
- Consider eigenvectors of $\mathbf{A^TA\ u} = \lambda\ \mathbf{u}$. Clearly, $\mathbf{AA^T\ Au} = \lambda\ \mathbf{Au}$.
- Comparing, we see $\mathbf{e} = \mathbf{Au}$ with the same eigenvalues.
- $\mathbf{A^TA}$ has dimensions $N \times N$ and is much easier to process!

Selecting K Eigenvectors

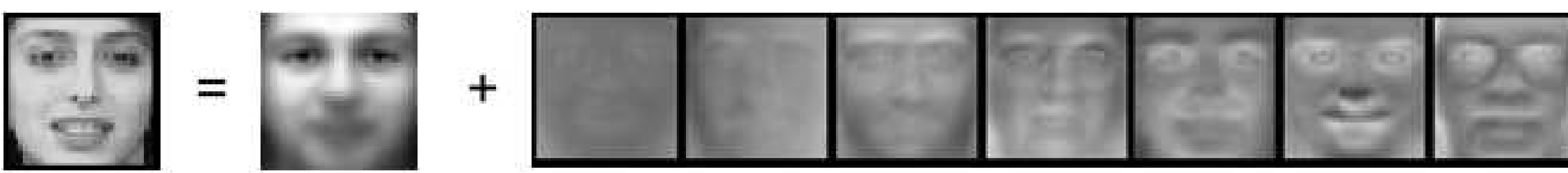
- $\mathbf{v}^{\mathbf{T}}\mathbf{e}_i$ gives the projection of a (mean subtracted) face vector $\mathbf{v}$ to eigenvector i.
- Can show that $\displaystyle\sum_{i=1}^{M} \lambda_i \, \mathbf{v}^{\mathbf{T}}\mathbf{e}_i = \mathbf{v}$
- Choose K to include some top eigenvalues. Represent each face $\mathbf{v}$ using its projections to K eigenvectors.
- Error: $\displaystyle\sum_{i=K+1}^{M} \lambda_i \, \mathbf{v}^{\mathbf{T}}\mathbf{e}_i$, due only to later eigenvectors.
- PCA: Essentially a dimensionality reduction too. It may be sufficient to use 100 or fewer dimensions (instead of the original 10K).

Example: Input Images

Meanface and Eigenfaces



Reconstruction using 7 eigenfaces:



Recognition Using Eigenfaces

- Given a face, project to the K-dimensional face-space
- Compare this vector to the vectors corresponding to all known individuals in the face space.
- If the closest example is "close enough", label the new face as that person's (Nearest Neighbour Classifier)
- If distance is large, declare it as an unknown face.
- Rejecting non-faces: Non-faces also map to the K-dimensional face space. How can they be distinguished?
- The error between reconstructed and original images will be **much higher** for non-faces. Can be used to reject them. Work as a face-detector?

Eigenfaces: Discussion

- Works reasonably well to identify faces trained on.
- Not robust to change in illumination, pose, etc.
- Eigenspace is best for compression: find **big** commonalities, remove **small** differences
- Recognition may depend on differences that lie in a later eigenvector! (like between letters **O** and **Q**)
- A good tool for compact representation and not discrimination
- Linear (or Fisher) Discriminant Analysis: focus on projections that separate classes.
- Minimize intra-class variance and maximize inter-class variance. Fisherfaces by Belhumeur/Kriegman 1997

Evaluating Recognition Systems

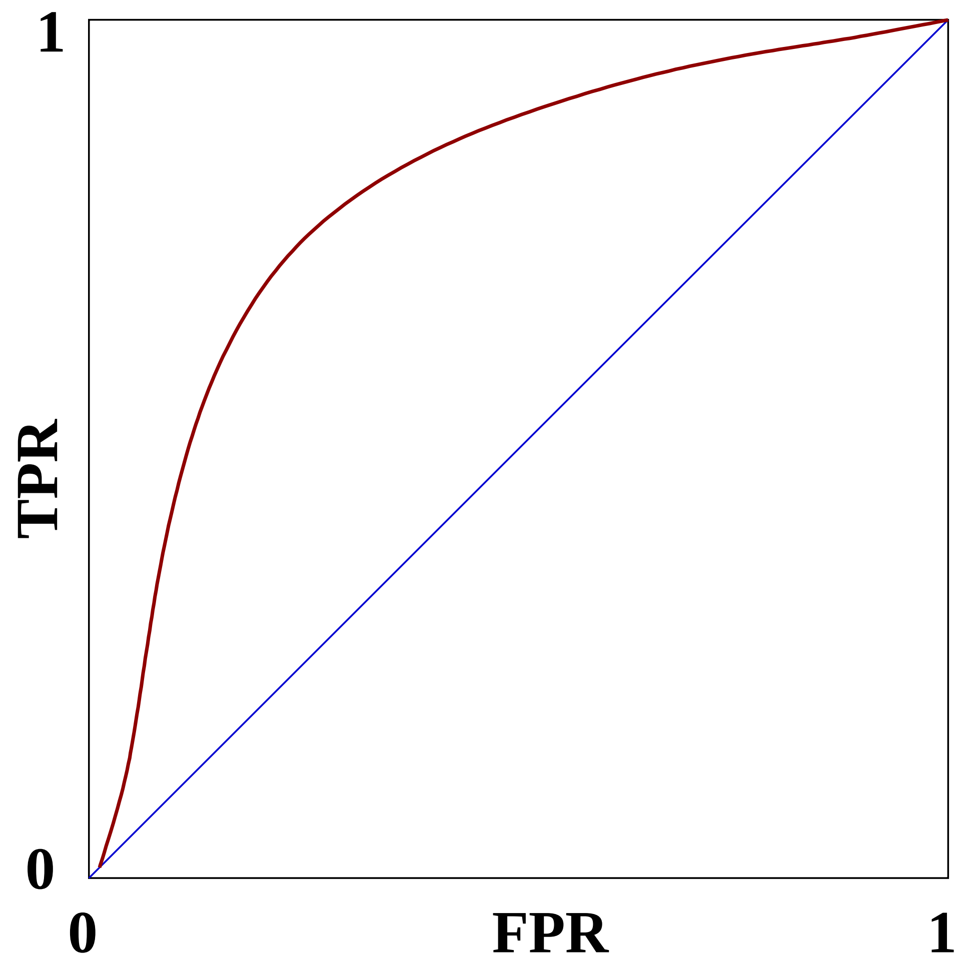
P positive and N negative examples. Recognition system classified some as positive and some as negative.

- True Positive: TP. True Negative: TN
- False Positive: FP. False Negative: FN
- All positives: P (= TP + FN)
- All negatives: N (= TN + FP)
- True Positive Rate, Sensitivity, **Recall**: TPR = TP / P
- False Positive Rate: FPR = FP / N
- True Negative Rate or Specificity: TNR = TN / N
- **Precision**: TP / (TP + FP)

ROC Curves

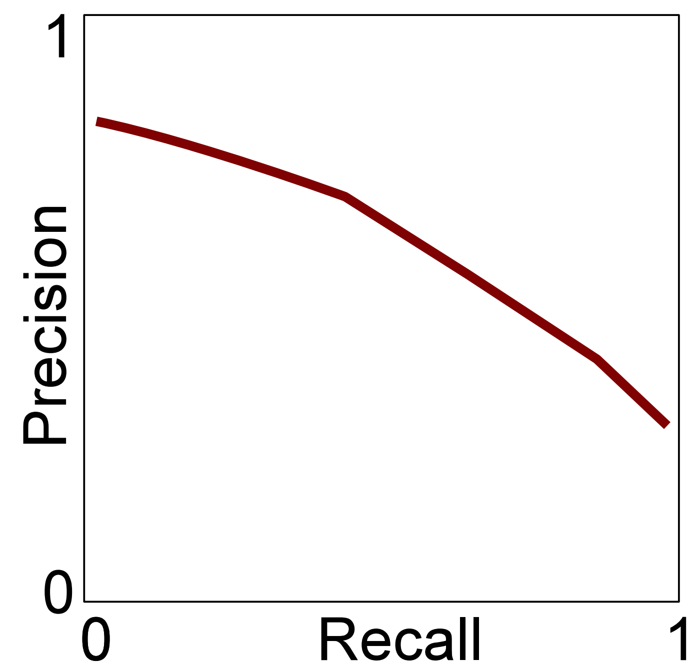
Receiver Operating Characteristic

True Positive Rate vs False Positive Rate



Precision, Recall, F-Measure

- Precision: Probability that a random positively labelled sample is positive = TP / (TP + FP)
- Recall: Probability that a random positive sample is labelled so = TP / P
- Average Precision: Area under precision-recall curve



F-meaure: Harmonic mean of the two: $2\,\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$

Use β to give more importance to recall $F_\beta = \frac{(1+\beta^2)}{\beta^2}\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$

Visual Recognition

Thank You!

P J Narayanan

Many figures are from different web sources