Audio-Visual Speech Processing

COM 4110 / COM 6070

Jon Barker

j.barker@dcs.shef.ac.uk

http://www.dcs.shef.ac.uk/~jon

Department of Computer Science

University of Sheffield

Objectives

- To examine more sophisticated face detection techniques.

Topics

- Pattern Classification Background Material
- Linear Discriminant Analysis - Fisherfaces
- Principal Component Analysis - Eigenfaces

Reading

- *Eigenfaces for recognition*, Turk and Pentland, 1991
- *Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection*, Belhumeur, Hespanha and Kriegman, 1997

Overview

- Pattern Classification Problems - Background.
  - Decision Boundaries.
  - Linear versus Non-Linear Classifiers.
  - Classification of N-Dimensional Data.
- Nearest Neighbour Classification - Why Not?
- Components of the IBM face detection system:
  - Chromaticity-based classification, (Lecture 4)
  - **Fisherfaces - Linear Discriminant Analysis,**
  - **Eigenfaces - Principal Component Analysis.**

Pattern Classification Problems

Many things in life can be categorised as belonging to one of a number of **discrete classes**.

The pattern classification problem: Given an object we must decide which class it belongs to (i.e. **labelling an unlabelled object**).

Examples:

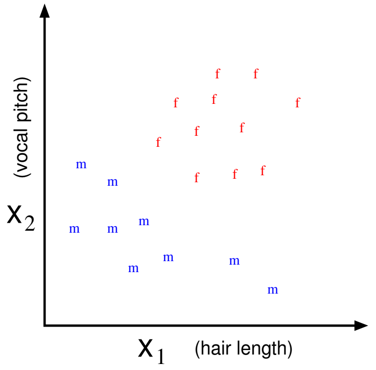
- We may have a handwritten character and we want to decide which letter of the alphabet it is.
- We may have an audio recording of a person speaking a digit, and want to decide whether its 0,1,2 ... or 9.
- We may have a video recording of an unknown person speaking and want to decide whether the person is male or female.

Pattern Classification Problems

We will base the classification on measurements of a set of **features**.

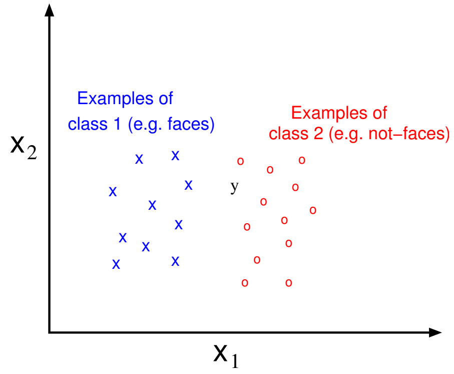
For example, for the **male versus female** classification problem we might take:

- a measurement of their **hair length** (x_1),
- and a measurement of their **average vocal pitch** (x_2).



Pattern Classification Problems

We are supplied with some **labelled training data** giving examples of each class. e.g. below we have 2-D data points belonging to either class 1 or class 2.

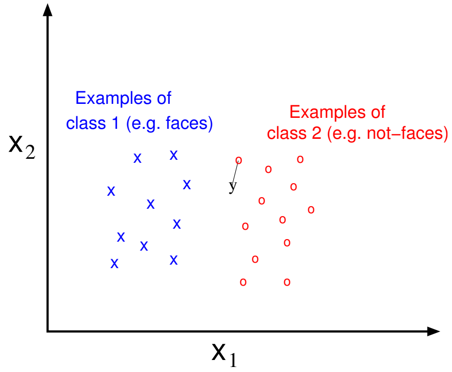


The task is to assign **unlabelled** examples to the correct class.

e.g does the point *y* belong to class 1 or to class 2.

Example: The Nearest Neighbour Classifier

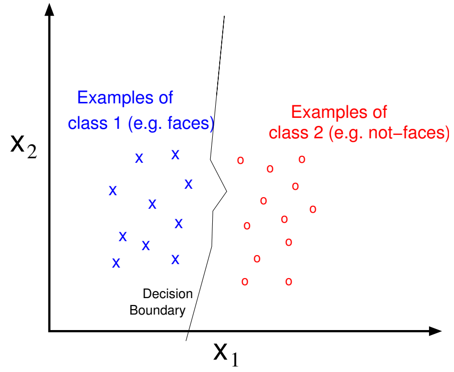
A **nearest neighbour** classifier assigns the unlabelled point to the class of the nearest point in the training data.



In our example, the point y would be classified as belonging to class 1 because its nearest neighbour in the training data is a member of class 2.

The Decision Boundary

The **decision boundary** is the line which separates the region that the classifier would assign to class 1 from the region it would assign to class 2.



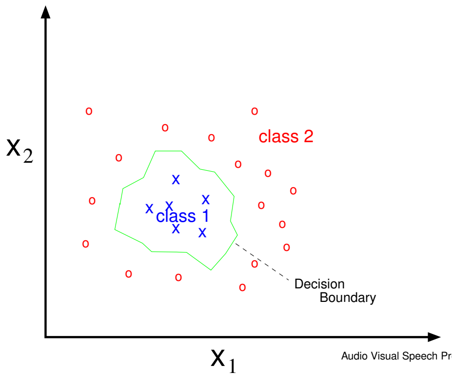
The figure shows the decision boundary for the nearest neighbour classifier. All points on the left of the boundary are nearer to an x and those on the right are nearer to an o.

Linear Versus Non-Linear Classifiers

The form of the decision boundary depends on the type of classifier:

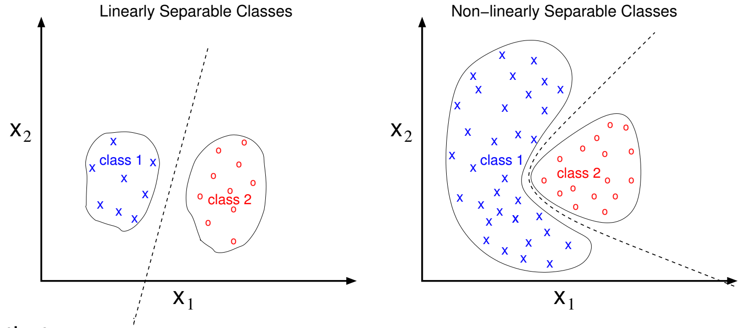
- **Linear classifiers** (e.g. the Fisher Linear discriminant) *can only* form straight line decision boundaries.
- **Non-linear classifiers** (e.g. neural networks) *can possibly* form decision boundaries that are not straight.

Nearest neighbour classifiers are non-linear. The decision boundary is **piecewise linear** (i.e. it is composed of a number of straight segments), but it can be of arbitrary shape.



Linear Separability

Classes which can be separated by a straight line are known as **linearly separable**.



It follows that:

- Data from **linearly separable classes** can be classified without error using a **linear classifier**.
- Error-free classification of data from **non-linearly separable classes** requires a **non-linear classifier**.

Classification of N-Dimensional Data

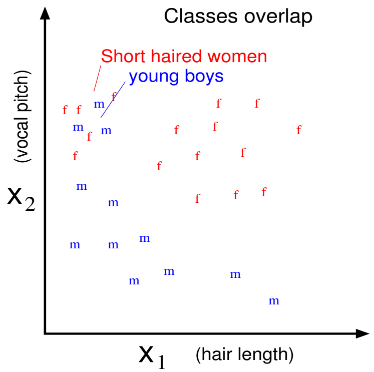
Our examples so far have considered the classification of 2-d data points, (x_1, x_2).

e.g. We might try to classify someone's sex based on just two features:

- a measurement of their hair length (x_1),
- and a measurement of their average vocal pitch (x_2).

However, many women have short hair, and small boys have high voices.

The classes **overlap**. So no classifier can work reliably.



Classification of N-Dimensional Data

In general classification can be made more reliable by observing **many** features - e.g. hair length, voice pitch, height, weight, mouth width, etc.

- These observations can be expressed in an N-dimensional **feature vector**, $\mathbf{x} = (x_1, x_2, x_3, ..., x_N)$.
- Classification now occurs in an N-dimensional **feature space**.
- Classes are less likely to overlap in a **higher dimensional space**.

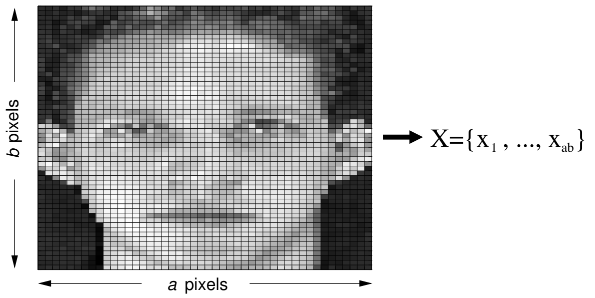
The principals remain the same, but the problems become harder to visualise. e.g.

- with 3 dimensions decision boundaries become **surfaces** rather than lines.
- 3-d linear classifiers have decision boundaries described by 2-d planes (i.e. flat surfaces).
- N-d linear classifiers have decision boundaries described by (N-1) dimensional **hyperplanes**.

Classification of Face Data

The **grey-level of each pixel** in the image is taken as a separate feature.

So, an image that is a pixels wide and b pixels high, will be represented by an $a \times b$ dimensional feature vector, $\mathbf{x}$.



Typical values for a and b are around 40 and 50, i.e. 2,000 pixels.

So classification of faces versus not-faces will occur in a **roughly 2,000-dimensional feature space** (very hard to visualise!).

Nearest Neighbour Classification - Why Not?

Nearest neighbour classification looks like an attractive approach:

- It is conceptually very **simple**,
- It can generate **decision boundaries of any shape**.

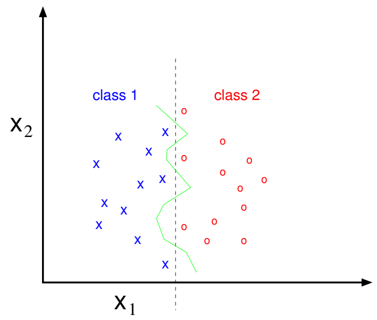
However, it has some **major drawbacks** that make it unsuitable for the face classification task:

- **Computationally expensive** - to find the nearest neighbour we must calculate the distance between the test image and *every* training image.
- **Large storage requirements** - every example image in the training set must be stored.
- May **overfit** the training data and not **generalise** well.

Overfitting

If there is insufficient training data the nearest neighbour technique may **overfit**.

In the example below the labelled points are sampled from two linearly separable classes.



However, the nearest neighbour decision boundary is a poor fit to the true class boundary. Its shape is very **sensitive** to the where the training examples happen to be.

Note, this problem can be reduced by using the k-nearest neighbours, rather than just the 1-nearest, i.e. the point is assigned to the class to which the majority of the k-nearest neighbouring training example belong (for some $k > 1$).

Lecture 5: Face Detection (Part 2)

Overview

- Classification Problems - Background.
  - Decision Boundaries.
  - Linear versus Non-Linear Classifiers.
  - Classification of N-Dimensional Data.
- Nearest Neighbour Classification - Why Not?
- **Components of the IBM face detection system:**
  - Chromaticity-based classification, (Lecture 4)
  - **Fisherfaces - Linear Discriminant Analysis,**
  - **Eigenfaces - Principal Component Analysis.**

Linear Discriminant Analysis

An image with dimensions, a pixels wide by b pixels high, can be represented by an $a \times b$ dimensional vector $\mathbf{x}$.

Linear discriminant analysis (LDA) attempts to find a linear combination of the dimensions of $\mathbf{x}$ that can discriminate the target classes (e.g. faces versus not-faces).

i.e. attempt to find $\mathbf{w}$ and w_0 such that, if:

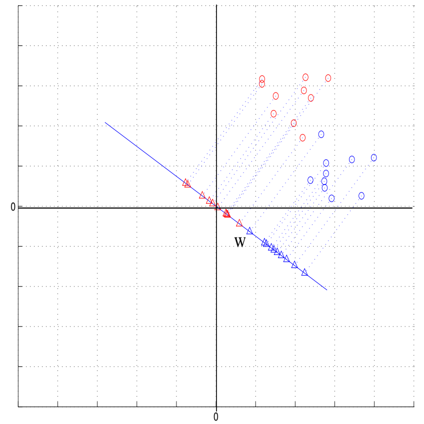
$$y = \mathbf{w}^t\mathbf{x}$$

then:

$$\begin{aligned} y \geq w_0 \quad &\Rightarrow \quad x \in faces \\ y < w_0 \quad &\Rightarrow \quad x \in notfaces \end{aligned}$$

The Fisher Linear Discriminant

Points are projected onto a 1-dimensional line:



Need to find the line direction that best separates the classes.

Duda and Hart (1973)

The Fisher Linear Discriminant

Consider n d-dimensional samples, $\mathbf{x}_1, ..., \mathbf{x}_n$, with

- n_1 samples in subset $\mathcal{X}_1$ and,
- n_2 samples in subset $\mathcal{X}_2$.

Let $\mathbf{w}$ be a vector such that $||\mathbf{w}|| = 1$.

Projecting samples onto an axis in the direction $\mathbf{w}$ produces:

$$y = \mathbf{w}^t\mathbf{x}$$

We now have n samples, $y_1, ..., y_n$ divided into subsets $\mathcal{Y}_1$ and $\mathcal{Y}_2$.

We want subsets $\mathcal{Y}_1$ and $\mathcal{Y}_2$ to be **well separated**.

The Fisher Linear Discriminant

Measure of separation given by **difference in sample means**:

$$\tilde{m}_i = 1/n_i \sum_{y \in \mathcal{Y}_i} y = 1/n_i \sum_{x \in \mathcal{X}_i} \mathbf{w}^t\mathbf{x} = \mathbf{w}^t\mathbf{m}_i$$

We want the separation, $|\tilde{m}_1 - \tilde{m}_2|$, to be great compared to the scatter within each class:

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m}_i)^2$$

So we want to maximise (with respect to $\mathbf{w}$):

$$J(\mathbf{w}) = |\tilde{m}_1 - \tilde{m}_2|^2 / \tilde{s}_1^2 + \tilde{s}_2^2$$

The Fisher Linear Discriminant

$J(\mathbf{w})$ can be rewritten in terms of the original data and the transformation matrix $\mathbf{w}$ as:

$$J(\mathbf{w}) = \frac{\mathbf{w}^t S_B \mathbf{w}}{\mathbf{w}^t S_W \mathbf{w}}$$

where,

- S_B is called the **between-class scatter matrix**:

$$S_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^t$$

- S_W is called the **within-class scatter matrix**:

$$S_W = \sum_{x \in \mathcal{X}_1} (\mathbf{x} - \mathbf{m}_1)(\mathbf{x} - \mathbf{m}_1)^t + \sum_{x \in \mathcal{X}_2} (\mathbf{x} - \mathbf{m}_2)(\mathbf{x} - \mathbf{m}_2)^t$$

The Fisher Linear Discriminant

$$J(\mathbf{w}) = \frac{\mathbf{w}^t S_B \mathbf{w}}{\mathbf{w}^t S_W \mathbf{w}}$$

This is the **generalised Rayleigh quotient**, and it can be shown that $\mathbf{w}$ which maximises J must satisfy the generalised eigenvalue problem:

$$S_B \mathbf{w} = \lambda S_W \mathbf{w}$$

Alternatively:

$$S_W^{-1} S_B \mathbf{w} = \lambda \mathbf{w}$$

The Fisher Linear Discriminant

We require the $\mathbf{w}$ which solves:

$$S_W^{-1} S_B \mathbf{w} = \lambda \mathbf{w}$$

We can avoid solving the eigenvalues and eigenvectors of $S_W^{-1} S_B$ by noting that $S_B \mathbf{w}$ is always in the direction of $\mathbf{m}_1 - \mathbf{m}_2$.

(This is because S_B is the outer product $(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^t$)

And since the scale of w is immaterial we can write the **solution**:

$$\mathbf{w} = S_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$$

Applying The Fisher Linear Discriminant

Once we have calculated the vector $\mathbf{w}$ and the scalar threshold w_0 **applying the Fisher Linear Discriminant is straightforward**.

1. Convert the image into a feature vector $\mathbf{x}$.
2. Compute the vector inner-product, $y = \mathbf{w}^t \mathbf{x}$.
3. Then if $y \geq w_0$ classify the image as a **face**, else classify it as **not a face**.

This is computationally inexpensive.

- If there are p pixels in the image, then classification requires just p multiplications, $p - 1$ additions and 1 comparison.
- The cost scales linearly with the number of pixels.
- Unlike the nearest neighbour classifier the cost does not depend on the number of training examples.

Interim Summary

- Face detection is performed by performing **face/non-face classification** on each sub-image in the visual scene.
- A **(k-)nearest neighbour classifier** is simple but computationally expensive, and may require lots of training data.
- The **Fisher linear discriminant** (FLD) is easy to compute, and very cheap to employ
- Note, even if the face and not-face classes are not linearly separable, the threshold can be tuned to safely reject non-face examples.

Lecture 5: Face Detection (Part 2)

Overview

- Classification Problems - Background.
  - Decision Boundaries.
  - Linear versus Non-Linear Classifiers.
  - Classification of N-Dimensional Data.
- Nearest Neighbour Classification - Why Not?
- **Components of the IBM face detection system:**
  - Chromaticity-based classification, (Lecture 4)
  - Fisherfaces - Linear Discriminant Analysis,
  - **Eigenfaces - Principal Component Analysis.**

Eigenfaces and Distance From Face Space

The Eigenface technique is an application of **Principle Component Analysis (PCA)**.

Again we represent images as vectors of dimensionality *width* $\times$ *height* pixels e.g. for an image of size 40 $\times$ 50 pixels we have a 2,000 dimensional vector.

Images of faces have a similar overall configuration and will not therefore be randomly distributed in this huge space.

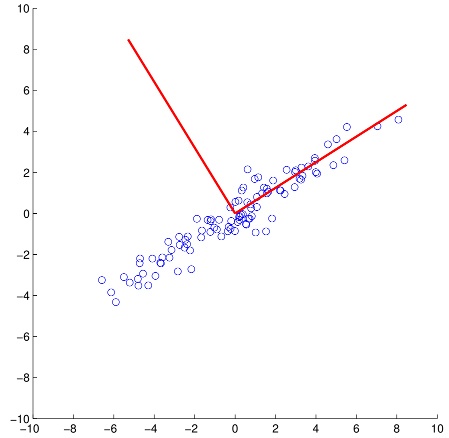
They will in general be **approximately described** by a relatively **low dimensional subspace**.

PCA can be employed to find the axes of this low dimension subspace. We call this subspace, "**face space**"'.

We can classify images using their **Distance From Face Space (DFFS)** i.e. faces are near to face space, non-faces are further from face space.

Principle Component Analysis (PCA)

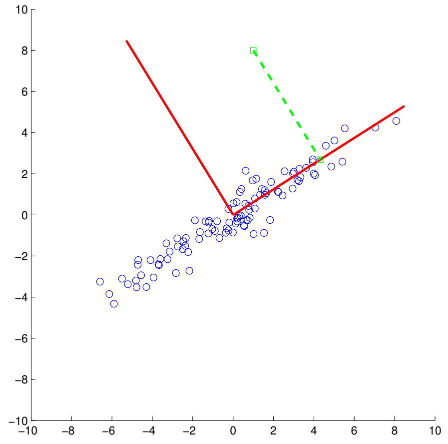
Consider the following highly correlated data.



Points may be well approximated in terms of the distance along a new 1-dimensional axis

Principle Component Analysis (PCA)

The green point belongs to another class - far from the blue cluster.



The green point is not well represented by the new axis.

It is far from the 1-d space in which the blue points lie.

Principal Component Analysis

How do we find the set of axes which best describe the 'face space'?

Consider n d-dimensional samples, $\mathbf{x}_1, ..., \mathbf{x}_n$,

Consider, y_i, the projection of these points onto a new axis $\mathbf{p}$

$$y_1 = \mathbf{p}_1^t \mathbf{x}$$

The first principal component is defined as the linear combination $y_1 = \mathbf{p}_1^t \mathbf{x}$ that has the largest possible variance given $\mathbf{p}^t \mathbf{p} = 1$.

i.e. we want to find the $\mathbf{p}$ which maximises the variance of the projected points y.

Principal Component Analysis

The variance of the projected points y, is given by:

$$S_y = \frac{1}{n-1} \sum_{i=1}^{n} (y_{1i} - \tilde{y}_1)^2$$

But given , $y_1 = \mathbf{p}_1^t \mathbf{x}$,

$$(y_{1i} - \tilde{y}_1)^2 = \mathbf{p}_1^t (\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t \mathbf{p}_2$$

and

$$\sum_{i=1}^{n} (y_{1i} - \tilde{y}_1)^2 = \sum_{i=1}^{n} \mathbf{p}_1^t (\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t \mathbf{p}_2 = \mathbf{p}_1^t [\sum_{i=1}^{n} (\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t] \mathbf{p}_2$$

Principal Component Analysis

So, putting this together we have,

$$S_y = \mathbf{p}_1^t [\frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t] \mathbf{p}_2$$

the bit inside [] is just S_x, i.e. the covariance of the original points, so:

$$S_y = \mathbf{p}_1^t S_x \mathbf{p}_1$$

And we want to maximise S_y with respect to p_1 and subject to the constraint:

- $\mathbf{p}_1^t \mathbf{p}_1 = 1$.

Principal Component Analysis

The p_1 that maximises $S_y^t = \mathbf{p}_1 S_x \mathbf{p}_1$ must satisfy:

$$S_x \mathbf{p}_1 = \lambda \mathbf{p}_1$$

But this has many solutions (eigenvalues) which one do we pick?

Note, premultiplying both sides by $\mathbf{p}_1^t$, that:

$$\mathbf{p}_1^t S_x \mathbf{p}_1 = \lambda \mathbf{p}_1^t \mathbf{p}_1$$

We want to maximise $\mathbf{p}_1^t S_x \mathbf{p}_1$, so we choose the eigenvalue, λ, which has the largest value.

Principal Component Analysis

For the second axis, p_1, we again want to maximise $S_y^t = \mathbf{p}_2 S_x \mathbf{p}_2$ but now subject:to the two constraints:

- $\mathbf{p}_2^t \mathbf{p}_2 = 1$ and
- $\mathbf{p}_2^t \mathbf{p}_1 = 0$ (i.e. 2nd axis is orthogonal to 1st).

With these constraints it can be shown that $\mathbf{p}_2$ is in fact the eigenvector of S_x associated with the 2nd largest eigenvalue.

We continue the process, so that each new axis maximised S_y^t while being constrained to be orthogonal to all the others found so far. It turns out, the new axes are simply the eigenvectors of S_x, ordered by their respective eigenvalues.

Applying PCA to Face Data

We can apply PCA to vectors defined by a training set of face images.

Each eigenvector of the correlation matrix can be displayed as an image.

These images are known as **eigenfaces**.

Any face-like image should be well approximated by a linear combination of the first few eigenfaces.

A Computational Efficiency

Problem:

A typical face image may have a resolution as high as 256 by 256 pixels, and will therefore by represented by a 65,736 element vector.

The correlation matrix is therefore a 65,536 by 65,536 matrix.

Solving the eigenvalue problem with such large matrices is not practical.

But there is a clever trick!

A Computational Efficiency

Lets express the covariance matrix S_x as a product:

$$S_x = AA^t$$

where $A = [\mathbf{\Phi}_1, \mathbf{\Phi}_2, ..., \mathbf{\Phi}_M]$ and $\mathbf{\Phi}_i = \mathbf{x}_i - \tilde{\mathbf{x}}$.

The eigenvalue problem we need to solve is now written as:

$$S_x \mathbf{p}_1 = AA^t \mathbf{p}_1 = \lambda \mathbf{p}_1$$

AA^t has dimensionality determined by the image size (very large).

A Computational Efficiency

Trick: Rather than considering the eigenvalues of AA^t consider the related problem:

$$A^t A \mathbf{v} = \lambda \mathbf{v}$$

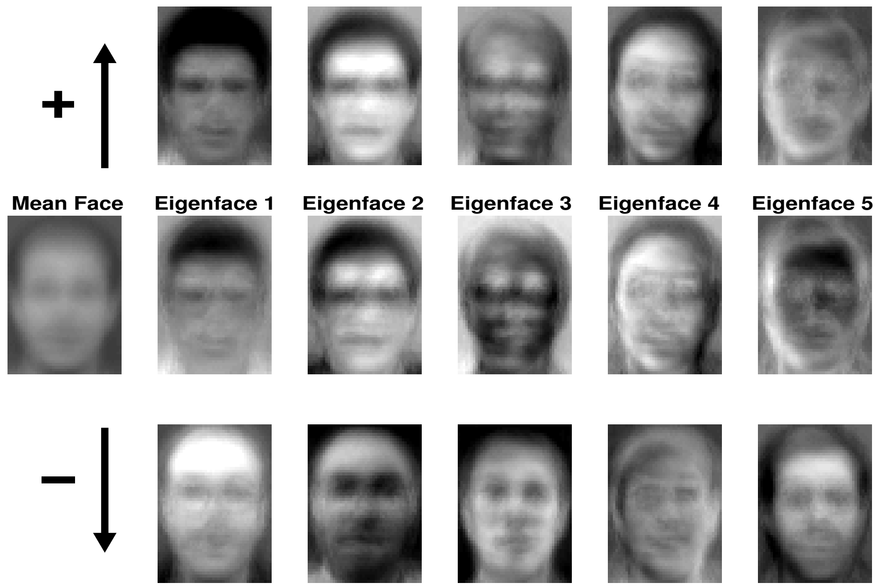
The dimensionality of $A^t A$ is determined by the number of images in the training set. This is typically much smaller.

Premultiplying both sides by A:

$$AA^t A \mathbf{v} = \lambda A \mathbf{v}$$

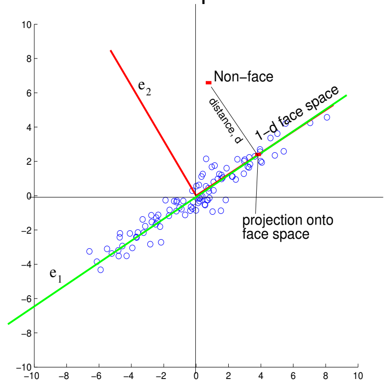
we see that the eigenvectors of AA^t are given by $A\mathbf{v}$ where, $\mathbf{v}$, are the (directly obtainable) eigenvectors of $A^t A$.

1st 5 Eigenfaces



Face Space

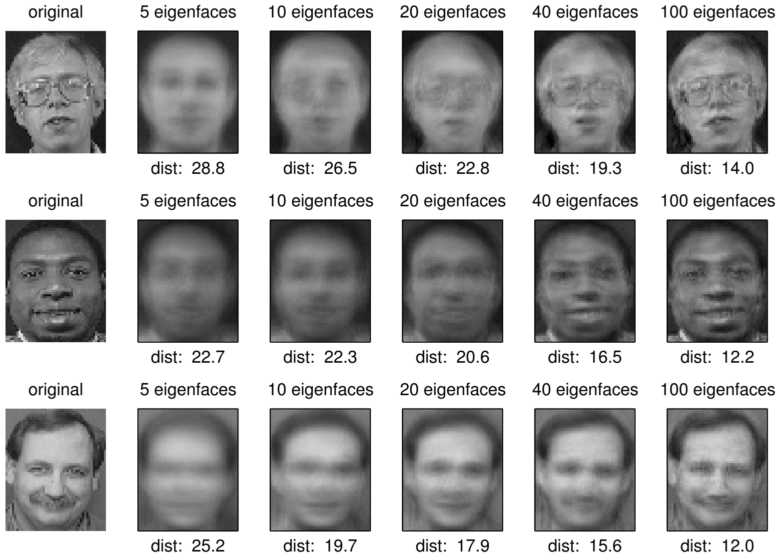
All face-like can be well approximated by a linear combination of the first few eigenfaces. i.e. they lie close to the ‘face space’ that is defined by the eigenface axes.



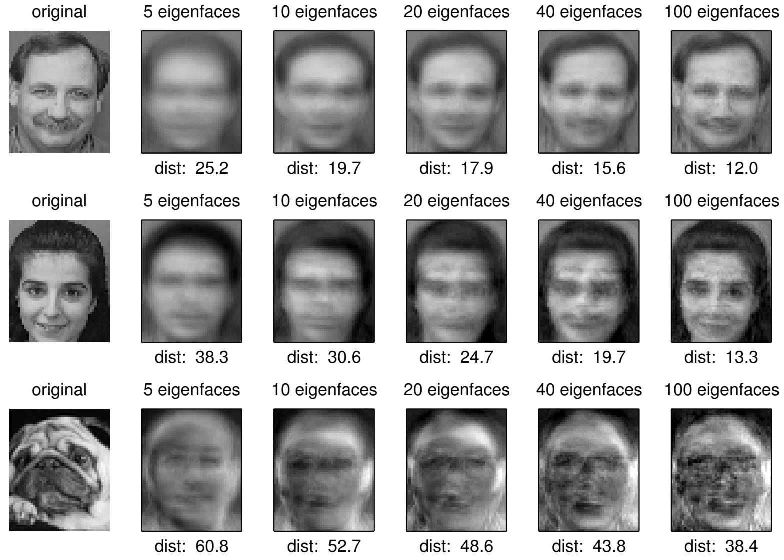
Non face-like images will lie further from these axes.

Therefore we can classify an image as face-like or non face-like by measuring the distance, d, between the image and its projection onto the ’face space’.

Distance From Face Space

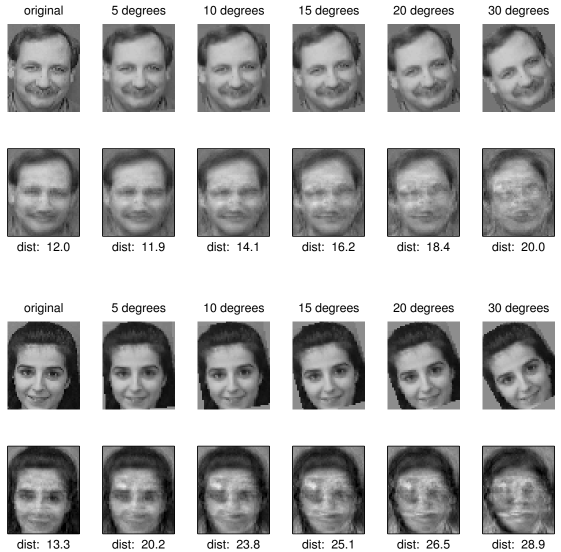


Distance From Face Space



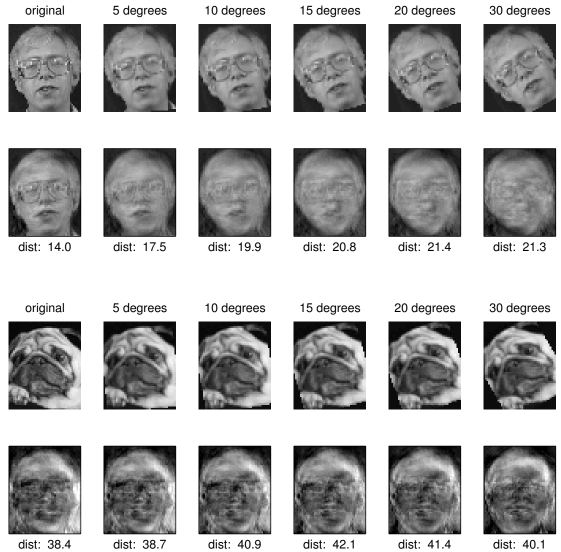
Sensitivity to Rotation

Distances from original to projection in 100-eigenface face space:



Sensitivity to Rotation

Distances from original to projection in 100-eigenface face space:

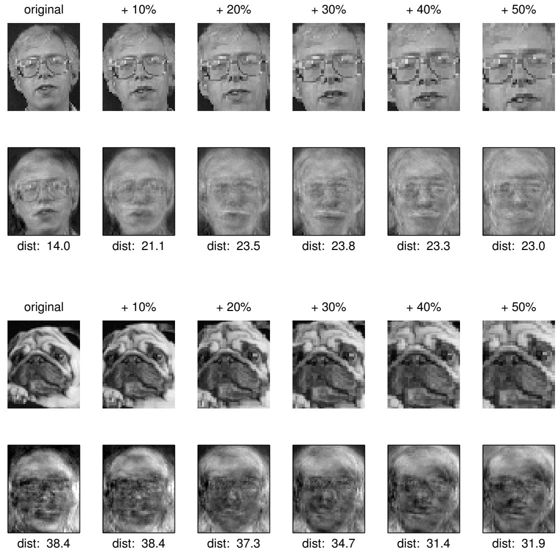


Sensitivity to Scaling

Distances from original to projection in 100-eigenface face space:

Sensitivity to Scaling

Distances from original to projection in 100-eigenface face space:



Some Advantages and Disadvantages of the DFFS Technique

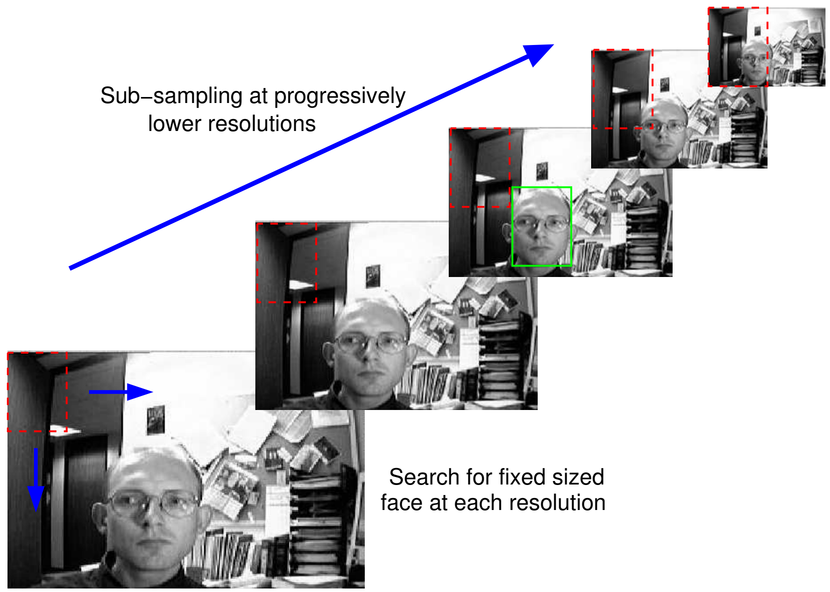
Some advantages:

- Relatively insensitive to small face rotations.
- Relatively insensitive to small scale changes.

Some disadvantages:

- More computationally expensive than the Fisher Linear Discriminant
- Evidence that it doesn't handle lighting variations as well as the FLD - e.g see *Belhumeur et al (1996)*.

Searching for Faces at Multiple Resolutions



IBM Face-Detection System Summary

The IBM face detector proceeds (roughly) as follows:

```
For each image resolution {
  For each sub-image location {
    If  skin-tone ratio > T1,
      If fisher discriminant > T2,
        If  Distance From Face Space < T3
           ACCEPT
        else REJECT
      else REJECT
    else REJECT
  } (end for each location)
} (end for each resolution)
```


Summary

- Face detection is performed by performing **face/non-face classification** on each sub-image in the visual scene.
- A **(k-)nearest neighbour classifier** is simple but computationally expensive, and may require lots of training data.
- The **Fisher linear discriminant** (FLD) is easy to compute, and very cheap to employ. Although the face and not-face classes are not likely to be linearly separable, the technique can be used to filter out images that do not resemble faces.
- The **Distance From Face Space** (DFFS) classification method is an aspect of the classic **Eigenface** technique of Turk and Pentland (1991).
- The **IBM** system uses a combination of the FLD and the DFFS to achieve fast and reliable classification.

Lecture 6&7 Preview: Visual Feature Parameterisation

The lecture will describe the three different type of visual feature that are employed in audio-visual speech processing:

- Low-level video pixel based feature (such as image transform features),
- High-level lip-model based features,
- hybrid features, based on the combination of high and low level features

The processing techniques that are employed to extract such features will be studied.

References

- Belhumeur, Hespanha and Kreigman, (1997) Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection, In *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 19(7), 711–720
- Duda and Hart (1973) *Pattern classification and scene analysis*, John Wiley and Sons, New York.
- A.W.Senior (1999) Face and feature finding for a face recognition system, In *Proc. Second International Conference on Audio- and Video-based Biometric Person Authentication*, 154–159, Washington, 1999.
- Turk and Pentland (1991) Eigenfaces for recognition, In *Journal of Cognitive Neuroscience*, 3(1), 71–83