COMPUTATIONAL VISION:

FACE DETECTION AND RECOGNITION

Class 5: Computational Vision

2021-2022

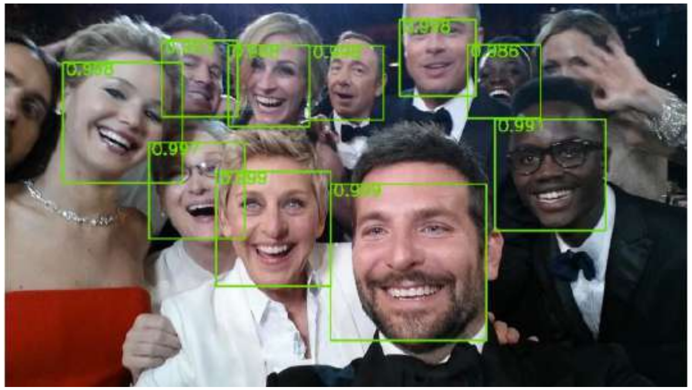
Index

1) The problem of detection and object recognition

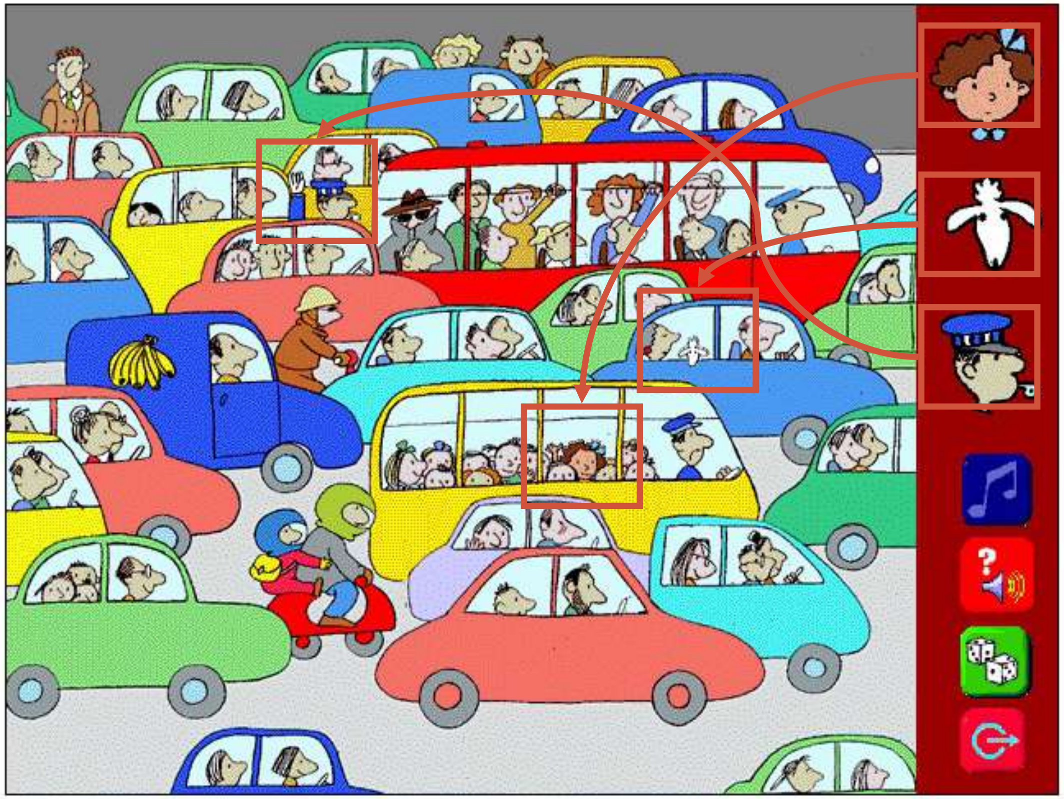
    1. Stimulus equivalence
    2. Detection vs. Classification vs Identification

2) Feature-based methods for face detection

    1. Integral image
    2. Adaboost
    3. Cascade of classifiers



What is the problem of object detection and recognition?

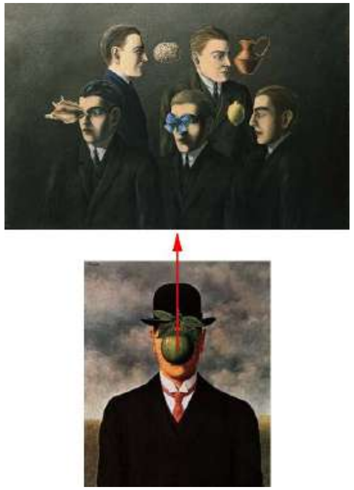


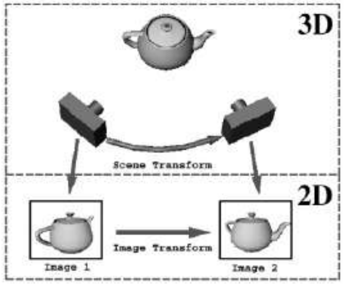
Object detection: Identify and locate human faces in an image regardless of their position, scale, in plane rotation, orientation, pose and illumination.

- A very difficult problem

What's the difficulty of object detection and recognition?

Problem: stimulus equivalence – an infinite number of views correspond to the same object.



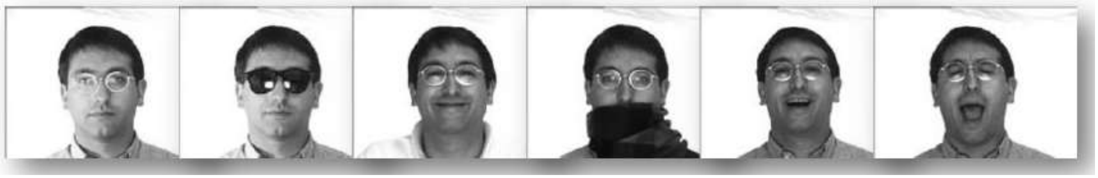


We need invariant features!

1 object$\leftrightarrow$ ∞ images

Reasons: scale, orientation, etc.

Intra-class variability



Inter-class confusion

Paul Newman/Marlon Brando

What's the problem of object detection, categorization and identification?

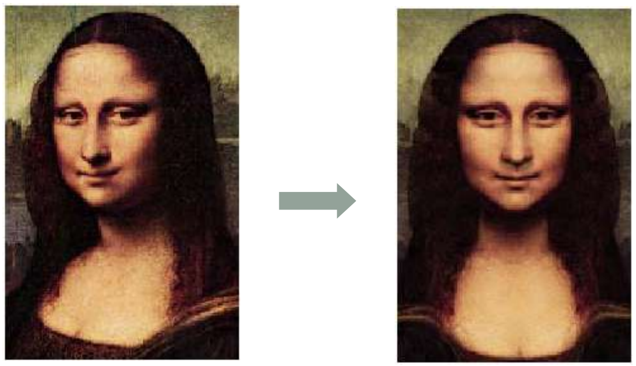
Difference between classify (categorize) and identify.

What’s the problem of object detection and recognition?

Why is it useful to recognize classes?

Recognition of new objects within a class:

- We can infer the properties (uses, dangers, ...) of things we've never seen! (Imagine the first European who saw a tiger!);
- Restricts the number of models for identifying (indexing);
- Allows the use of specific information to identify the class (eg neutralizing facial expressions);
- It enables generalization from very partial information (e.g. Mona Lisa!).



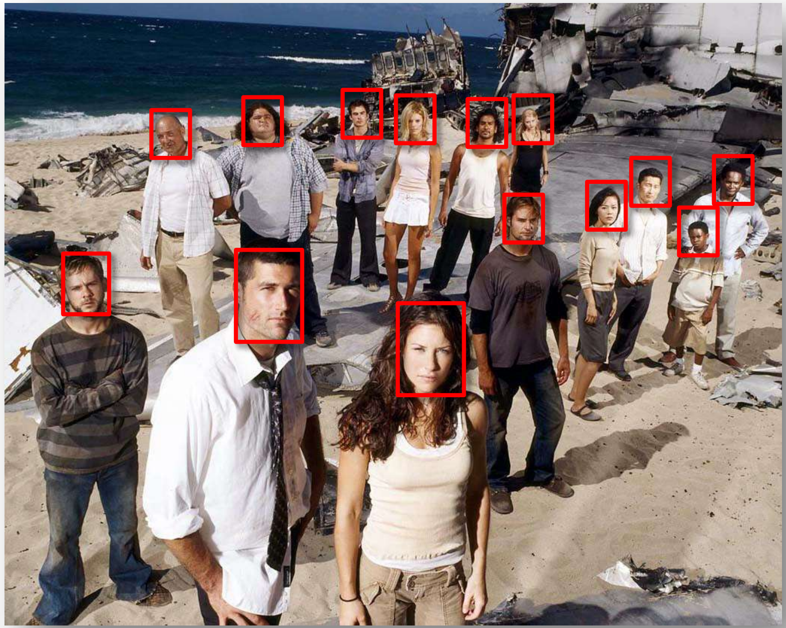
Face detection vs. recognition

THE PROBLEM

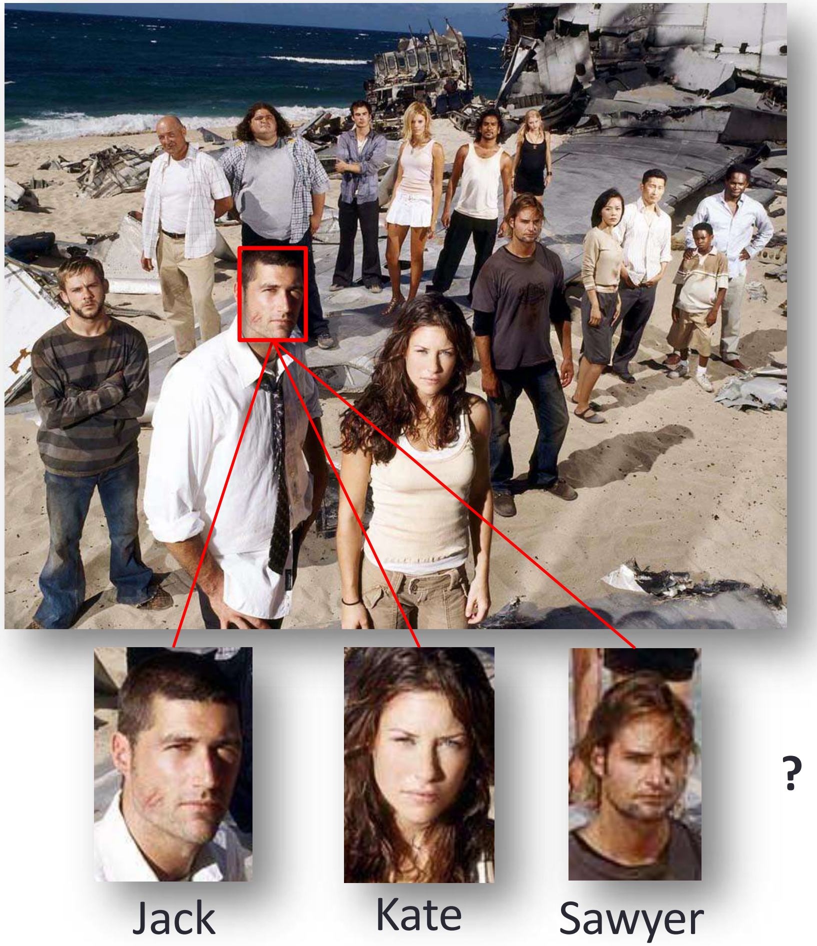
Given an image, automatically detect faces – i.e. detect the location of faces.

Detecion vs. Recognition

Detection



Recognition



Supervised classification method

Classification of image features

Goal: Define a space of image features allowing to represent objects based on their appearance (or through a set of local features) in the image.

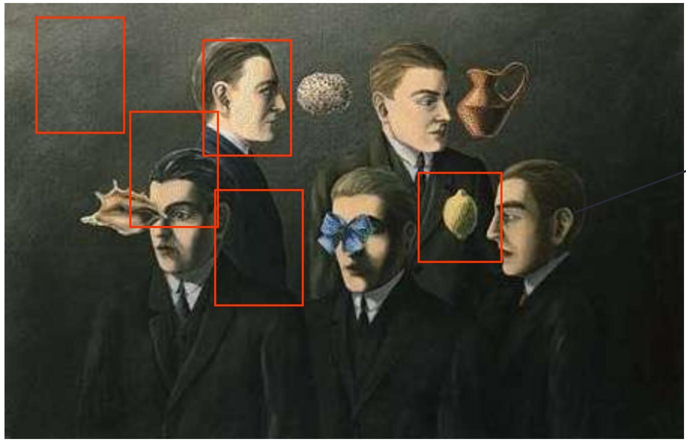
Three principal aspects:

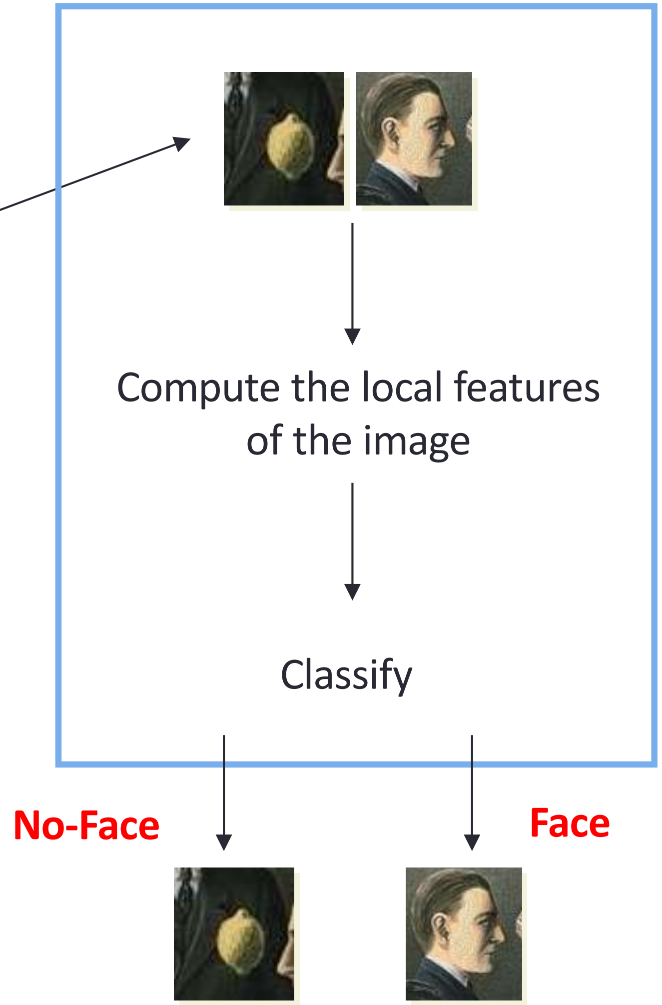
- **Adequate representation** (build descriptors of objects).
  - Normally, we try to reduce the size of the data so that the invariance is kept and the other dimensions are removed.

- **Training,** from a set of objects examples with their descriptors**.**

- **Detection or recognition** of a new object instance by using its descriptor and the learned model.

Face detection



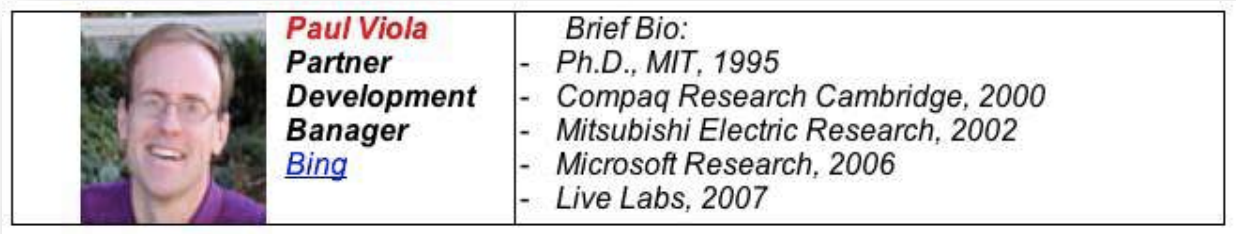


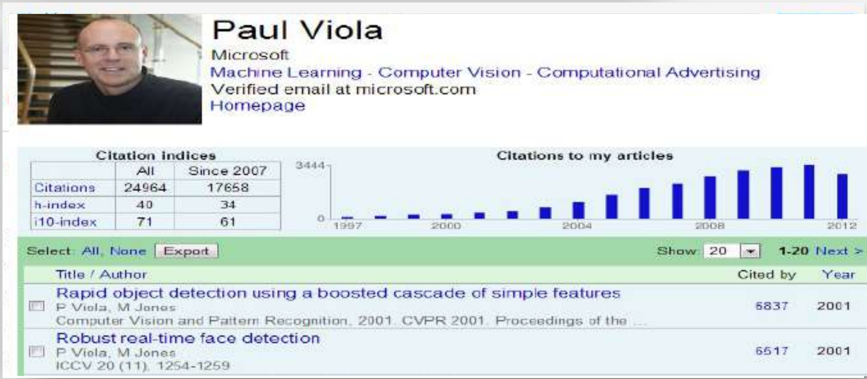
The image is divided into a large set of windows (overlapped). Each window is classified as ”face" or ”no face", based on a set of local measures.

Face detection: Viola & Jones

The main approach for face detection:

Robust Real-Time Face Detection
Paul Viola & Michael Jones
International Journal of Computer Vision, 2004.





Viola & Jones

Goals: of the face detector of Viola & Jones:

- Accurate detection of faces
- Fast algorithm!
- Real-time detection (video processing)

With a camera we can not wait long to take the photo!

Viola & Jones

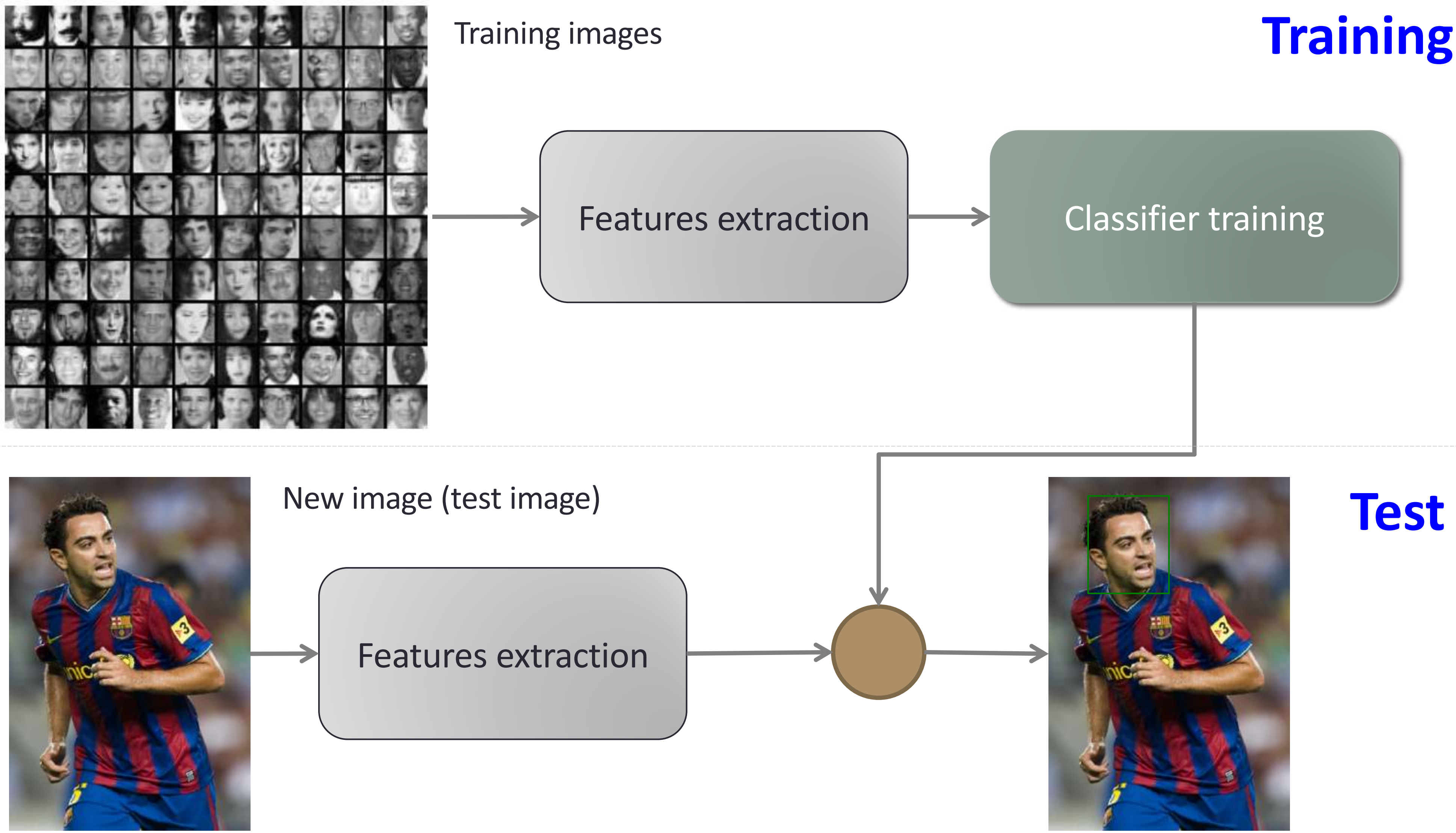
Main focus

The Viola & Jones' method is formulated as a standard supervised classification problem.

The major steps are basically three:

- Extraction of image features
- Training a decision rule, called classifier
- Test for new images using the trained classifier.

Supervised classification scheme



Viola & Jones

PRINCIPAL CONCEPTS

In this class, the main concepts to be learned are:

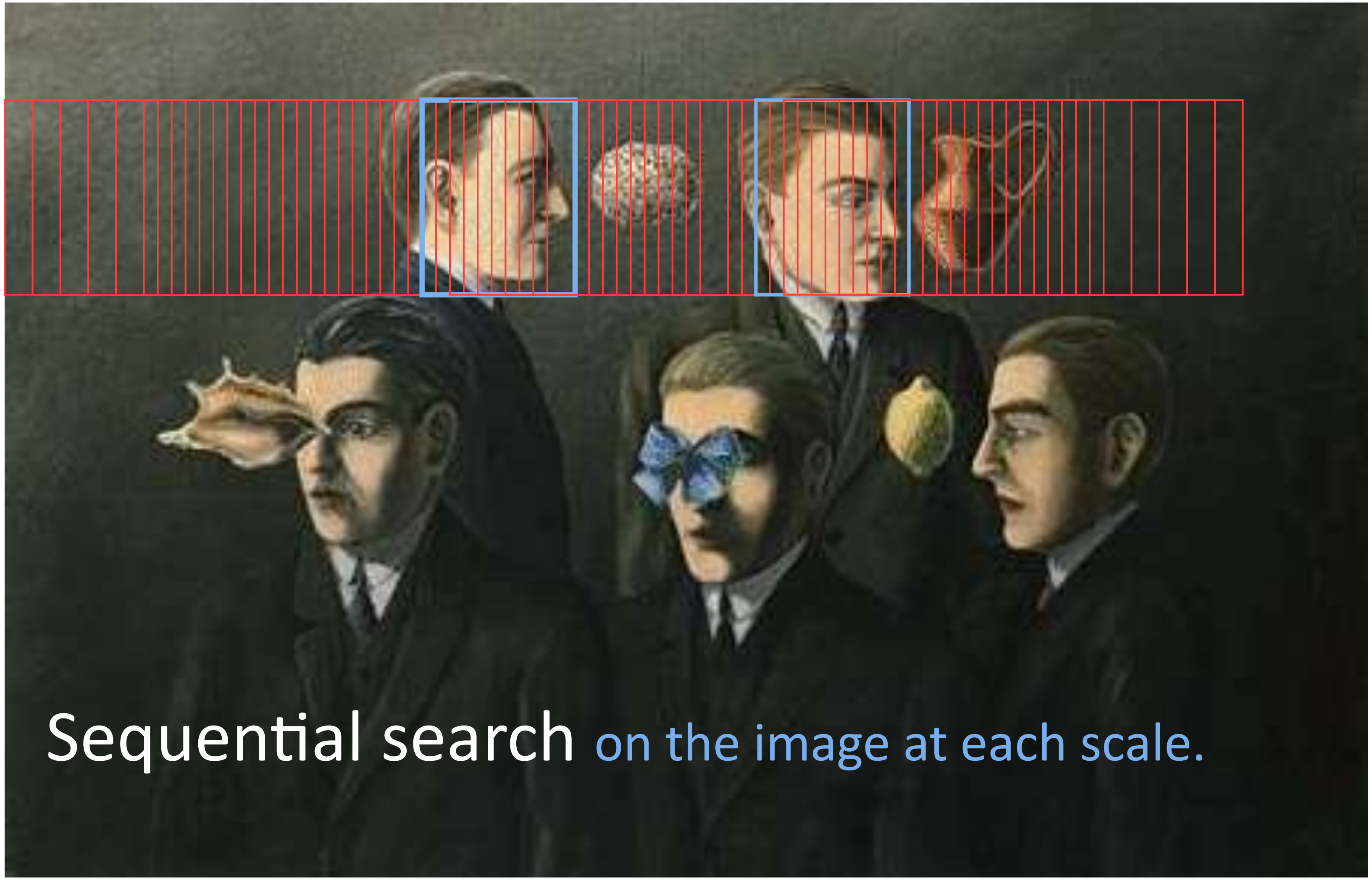
A) Features
- “Rectangular” features (Haar features)
- Integral images

B) Classifier
- AdaBoost Classifier
- Multiple Classifier (Cascade classifiers)

These concepts are sufficient to understand the face detector of Viola & Jones!

Finding faces are always done through a **sliding window** across the image, so we are talking about **sequential search**.

Face detection



Given the sliding window for each pixel, the question is: is there a face?

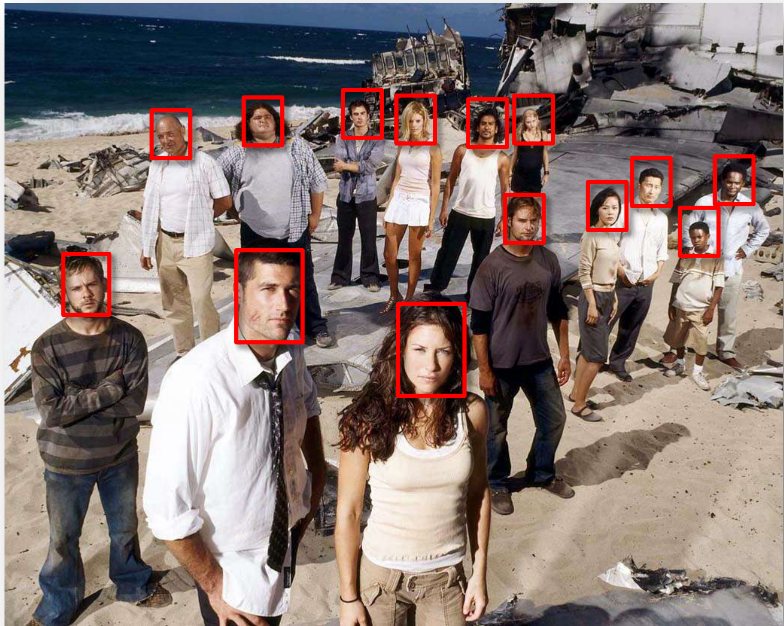
Face detection

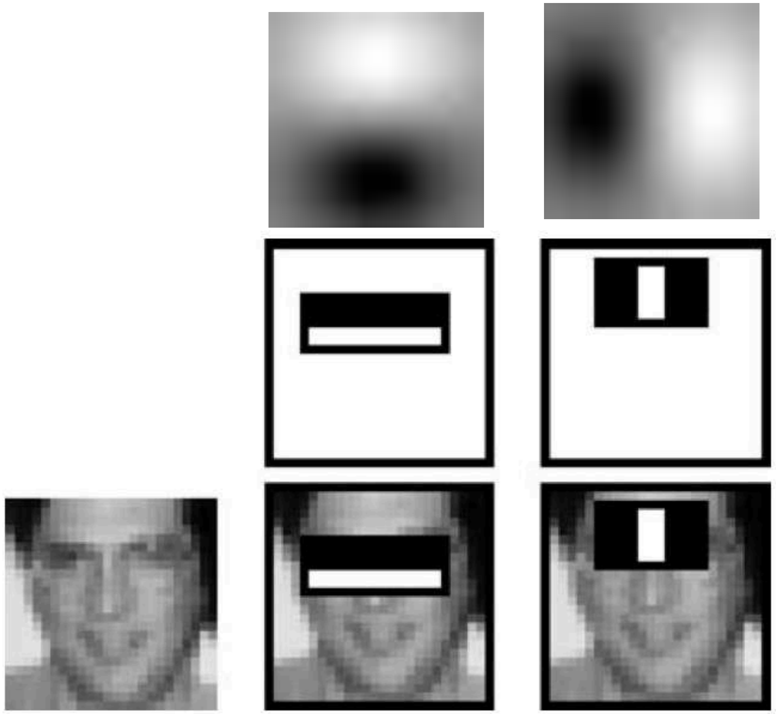
1) **Haar image features**
2) Integral images
3) AdaBoost
4) Classifiers cascade

Features

HAAR Features

What is the information they provide, when applied to face images?

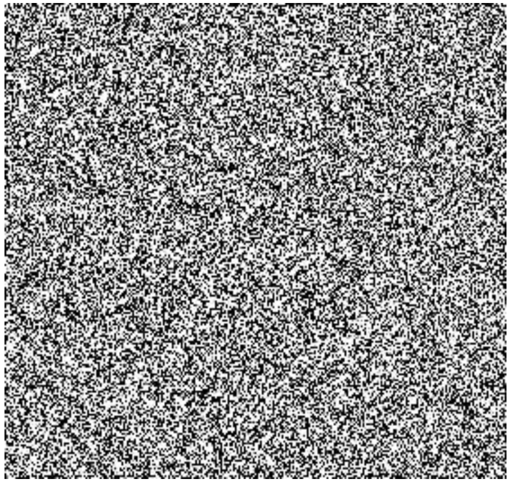




Difference in intensity between eyes and cheeks

Difference in intensity between eyes and nose

Example



Source

Result

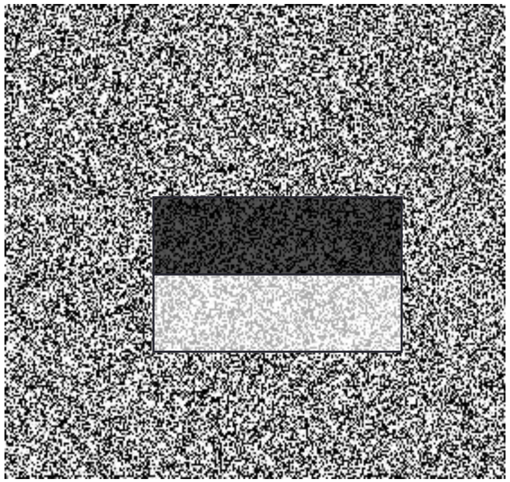


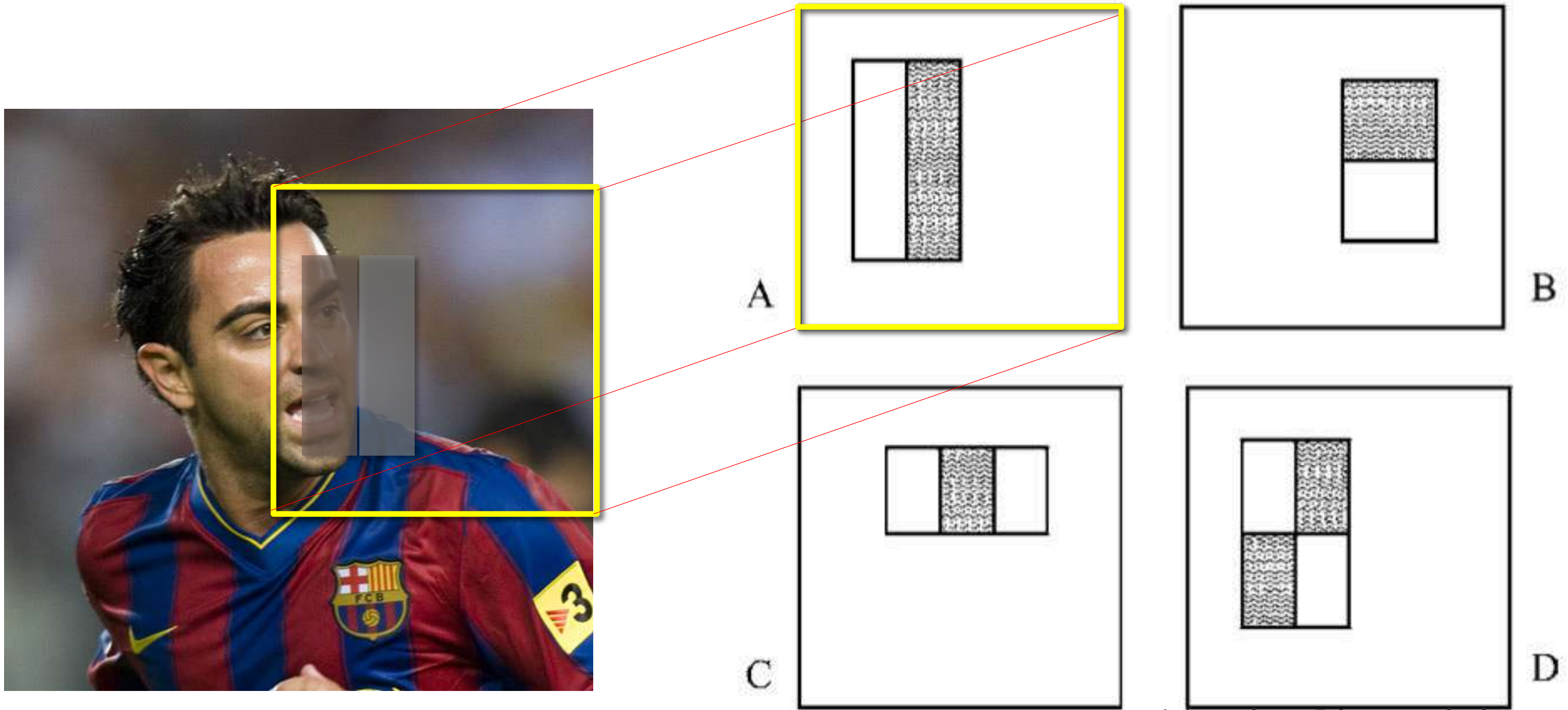
Image features

Haar features

Considering an image, we consider a square window in a given position: this window apply the Haar-type features. Repeat for each posible position.

- How Haar features are interpreted?
- What image structures do they show?

1st order: Vertical and horitzontal changes



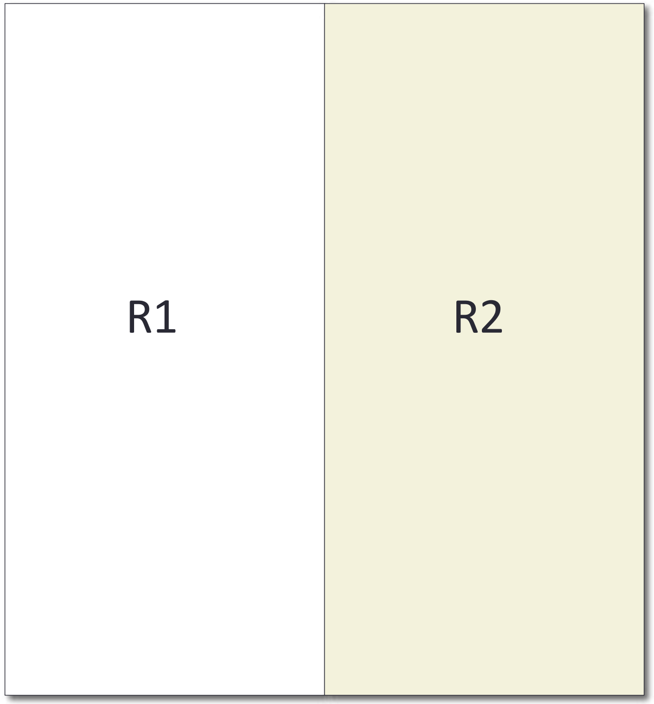
2nd order

1st order: Diagonal changes

Features

Haar features effect

Each feature mask, applied to the image I (x, y) indicates two regions:



The F value of the image feature corresponding to the mask that is applied in the particular pixel is the sum of the pixels in the white region minus the sum of the pixels in the dark region:

$$F_k = \sum_{(i,j)\in R1} I(i,j) - \sum_{(i,j)\in R2} I(i,j)$$

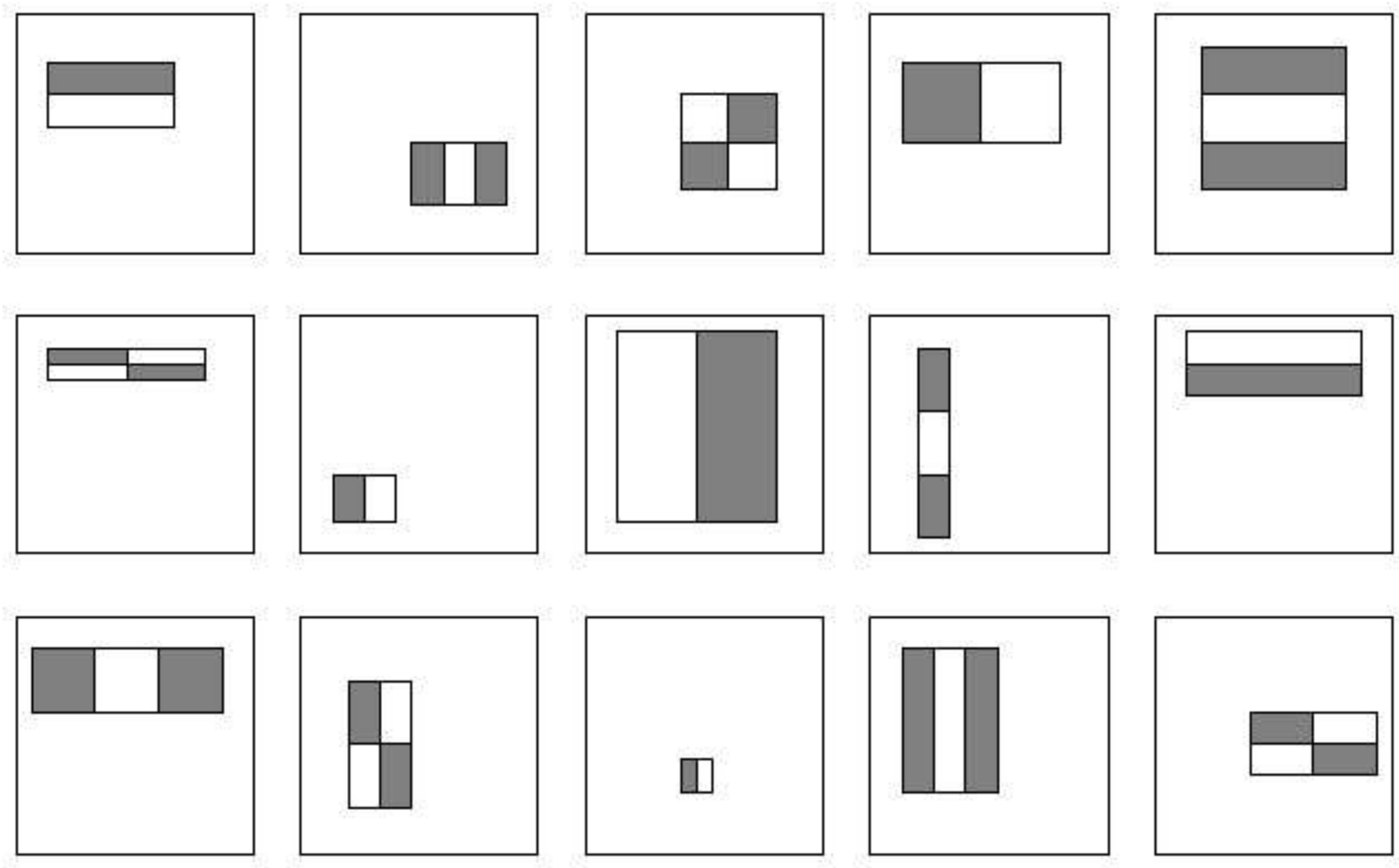
Typically, a set of K masks is used:

- F_k is the feature that corresponds to the k^{th} mask.

Features

Feature set

The masks have different size, shape, and position with regard to the square window



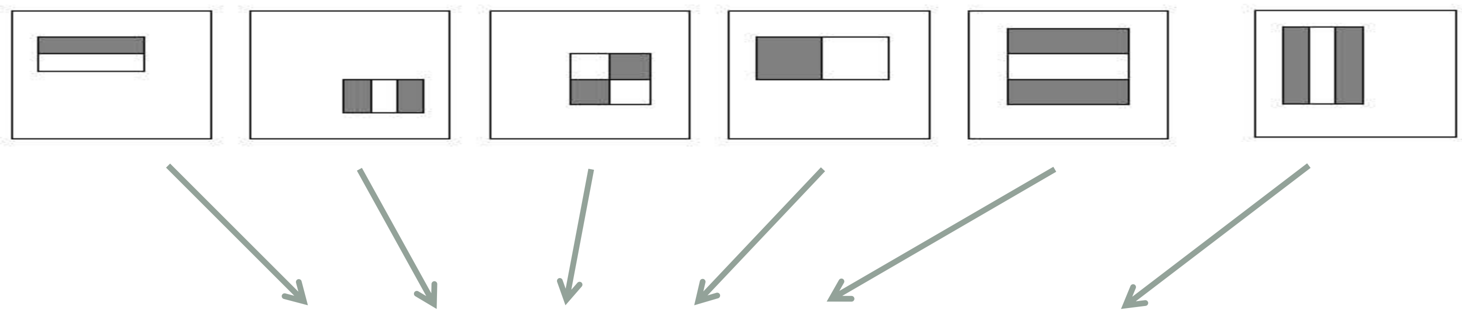
You can define up to **16 million masks** using:

- Different shape
- Different scale
- Different location of the squares.

Image features

Features extraction

Given the set of masks, how is the feature vector of each window constructed?



$$X = [\ F_1,..\ F_i,..\ F_j,..\ F_k,...\ F_l,\ ...\ ,\ F_n]$$

The feature vector describes the content of the window, and is used to train the classifier to detect the face and later, given the trained classifier to detect a face in new images.

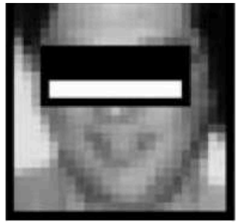
Face detection

1) Haar image features
2) **Integral images**
3) AdaBoost
4) Classifiers cascade

Integral Images

The main requirement of Viola & Jones detector is SPEED!

How can we compute Haar features quickly?



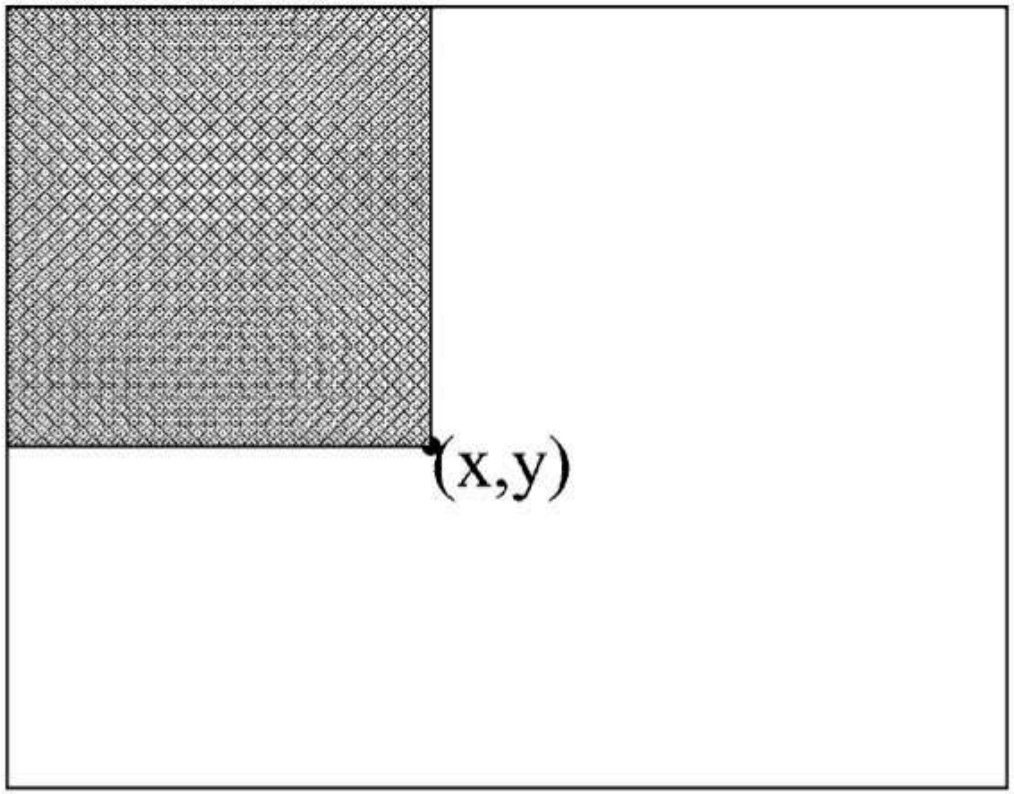
~~Convolution?~~

$$\cancel{I(x,y)\otimes K(x,y)=\iint I(x,y)K(x-\tau_x,y-\tau_y)d\tau_x d\tau_y}$$

INTEGRAL IMAGES

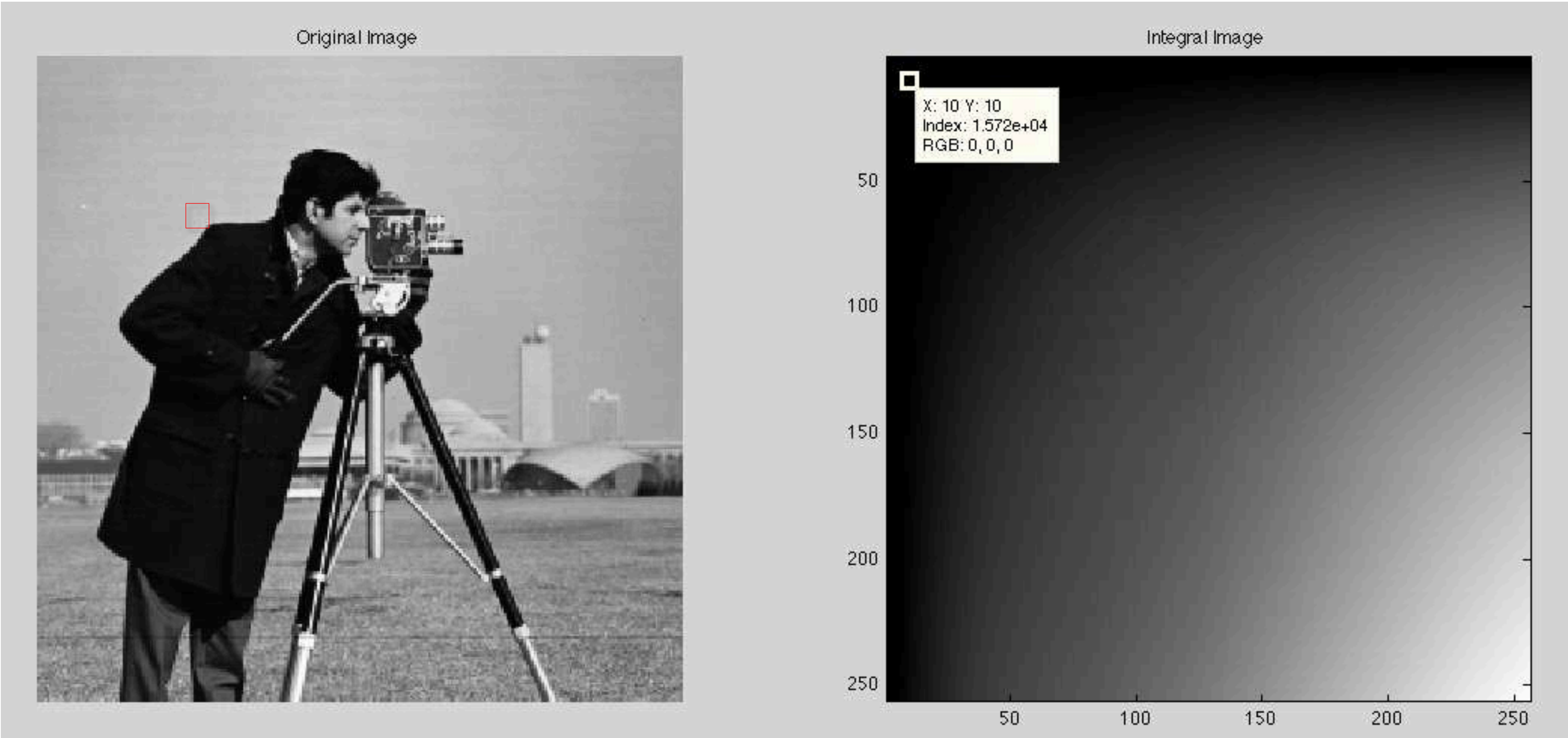
Integral Images

Df: The integral image S is constructed as follows: the value at position (x, y) is the sum of all pixels in the image I above and left of point (x, y):



$$S(x,y) = \sum_{i=1}^{x} \sum_{j=1}^{y} I(i,j)$$

Integral Images

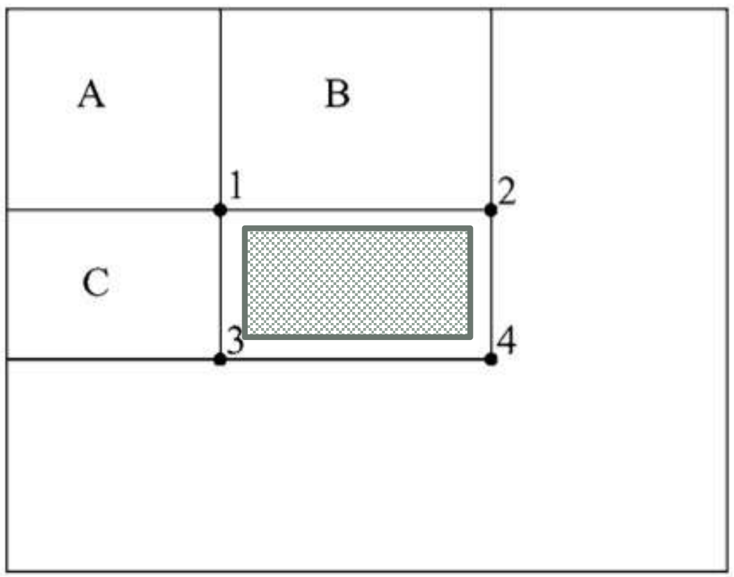


Integral Images

Why are we interested in the integral images?

The Haar features are based on sums and subtractions of image rectangles.

>> How to compute the area of a rectangle by an integral image?



1 = area of A

2 = area of A + B

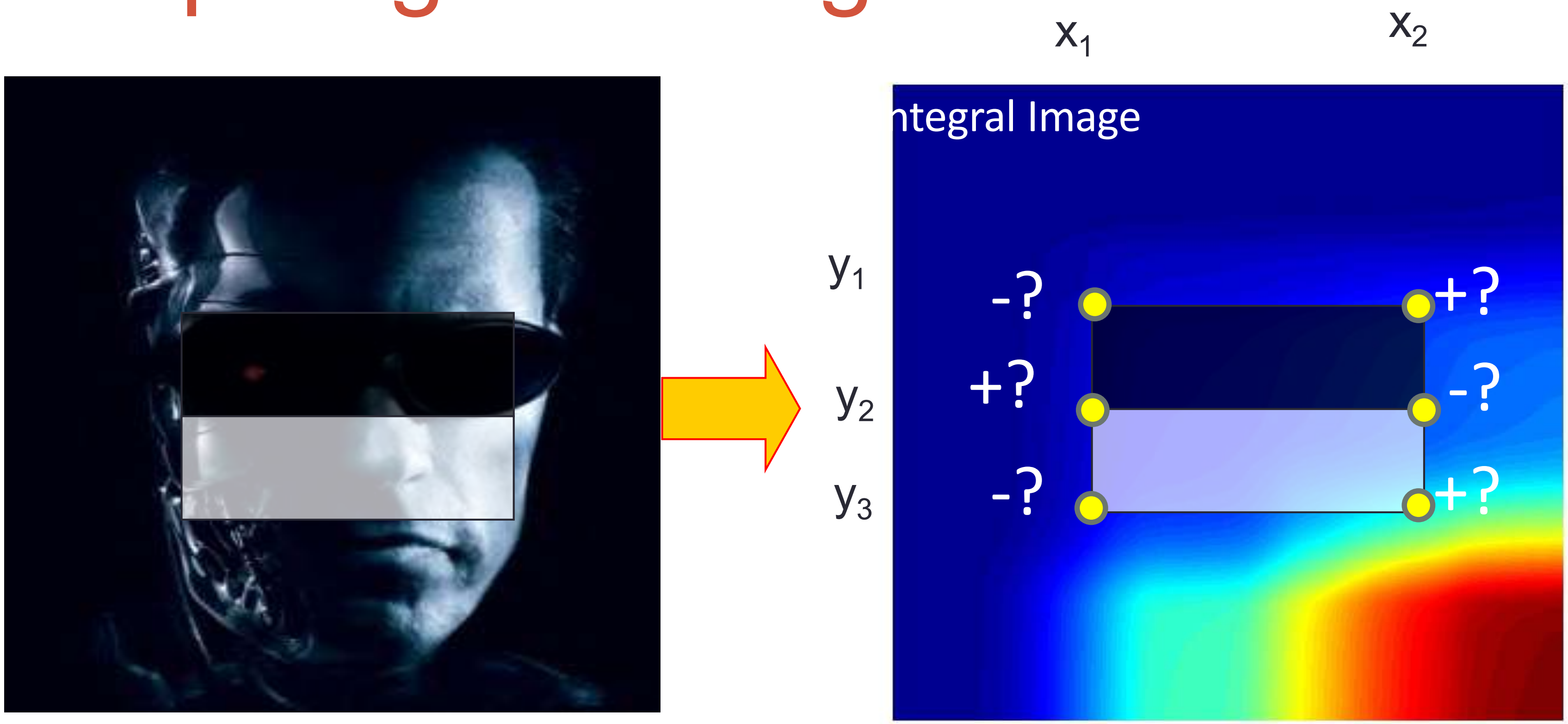
3 = area of A + C

4 = area of A+B+C+D

Area of **D = 4-2-(3-1)**

The computation of the features through integral images, is reduced to computing a set of 3 **sums and subtractions on the integral image points**!!! VERY FAST!

Computing a rectangle feature



Convolution with a mask is substituted by a subtraction of pixels of the integral image.

- What is each pixel of the integral image representing?
- If I is the integral image, what would be the result of the convolution:

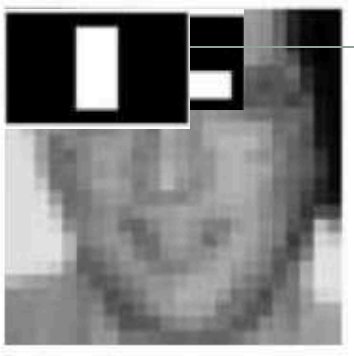
$$I(x2,y3) - I(x1,y3) - I(x2,y2) + I(x1,y2) - (I(x2,y2) - I(x1,y2) - I(x2,y1) + I(x1,y1))$$

$$I(x2,y3) - I(x1,y3) - 2 * I(x2,y2) + 2 * I(x1,y2) + I(x2,y1) - I(x1,y1)?$$

Features

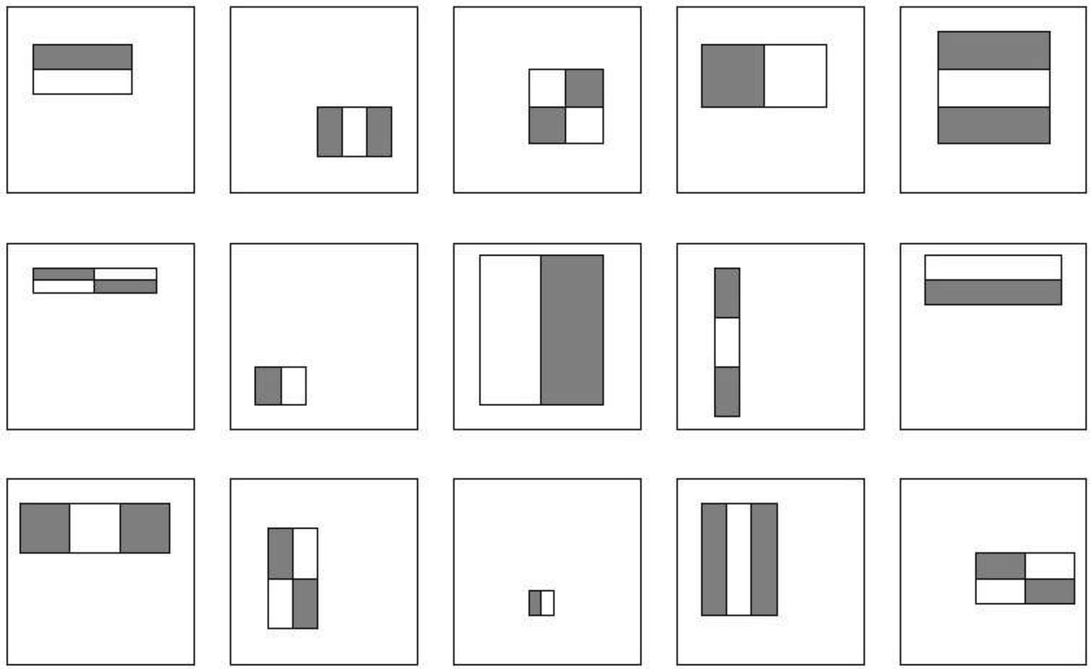
FEATURES FOR FACE DETECTION:

Computing values of the I and II order Haar features at different scale and at different points of the image of the face => thousands of features.



$x_1, \quad x_2, \quad x_3 \quad ,\ldots.x_n \qquad X_{n+1} \quad \ldots. X_m$

Viola-Jones detector: features



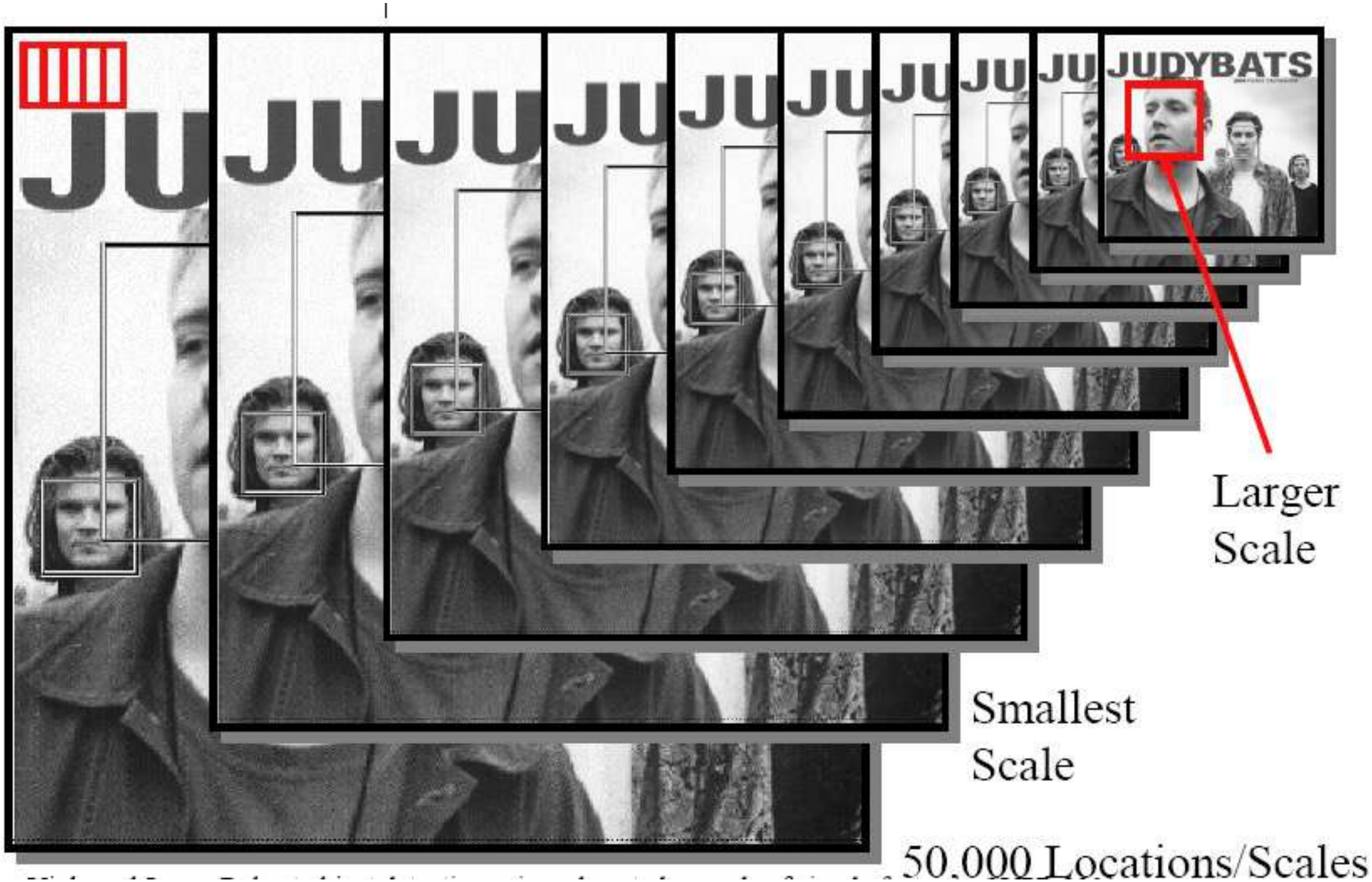
Considering all possible filter parameters: position, scale, and type:

180.000+ possible features associated with each 24 x 24 window

Which subset of these features should we use to determine if a window has a face?

Kristen Grauman

The scale



Re-scaling of the image instead of generating filters of different window.

Face detection

1) Haar image features

2) Integral images

3) **AdaBoost**

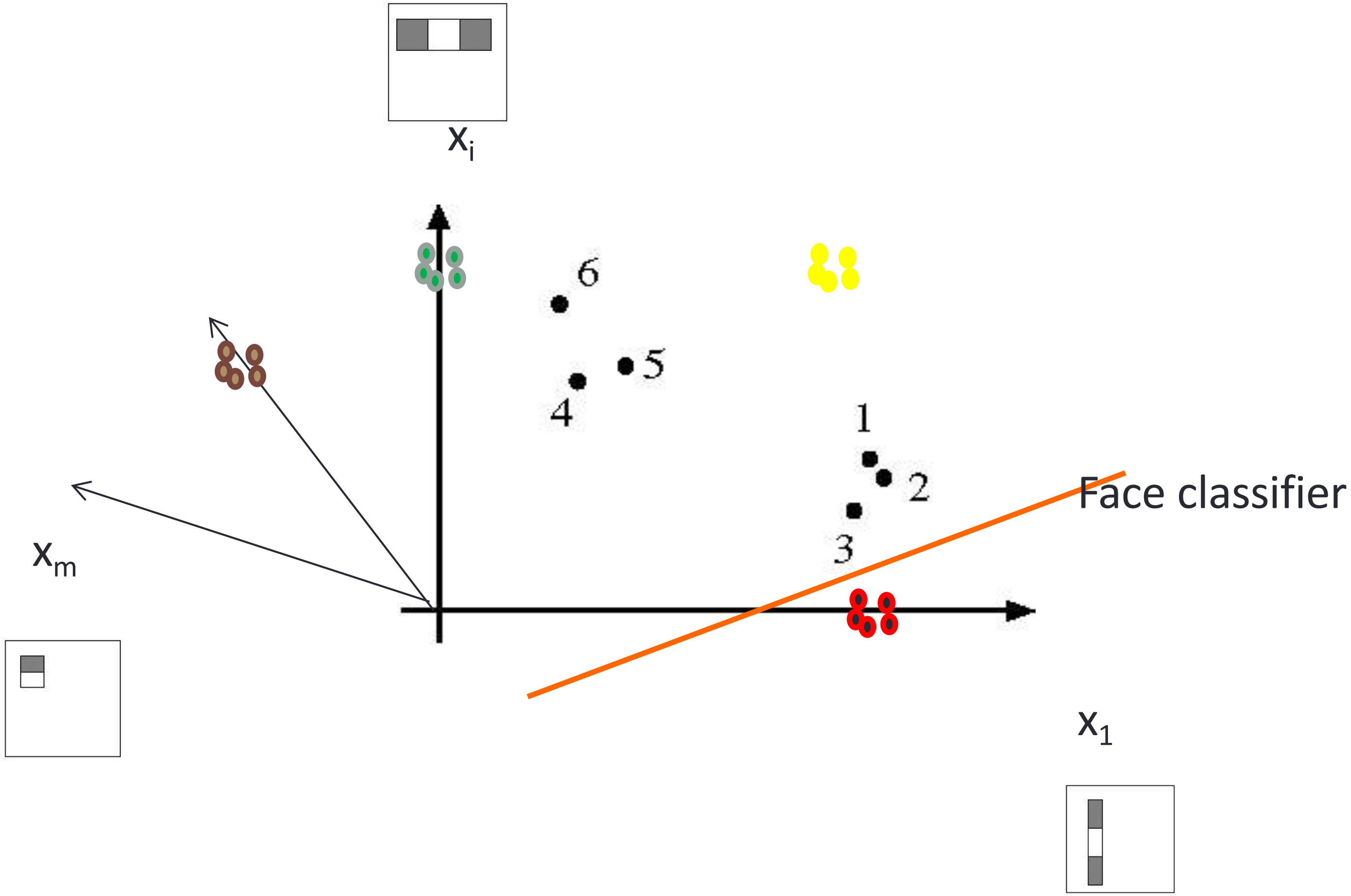
4) Classifiers cascade

Face classification

In the feature space, we need to separate windows containing faces from those that do not contain them.

>> We need a CLASSIFIER!

Face classification



AdaBoost

Viola & Jones uses AdaBoost classifier to detect faces.

- AdaBoost: Adaptive Boosting
  Introduced by Freund & Schapire in 1999

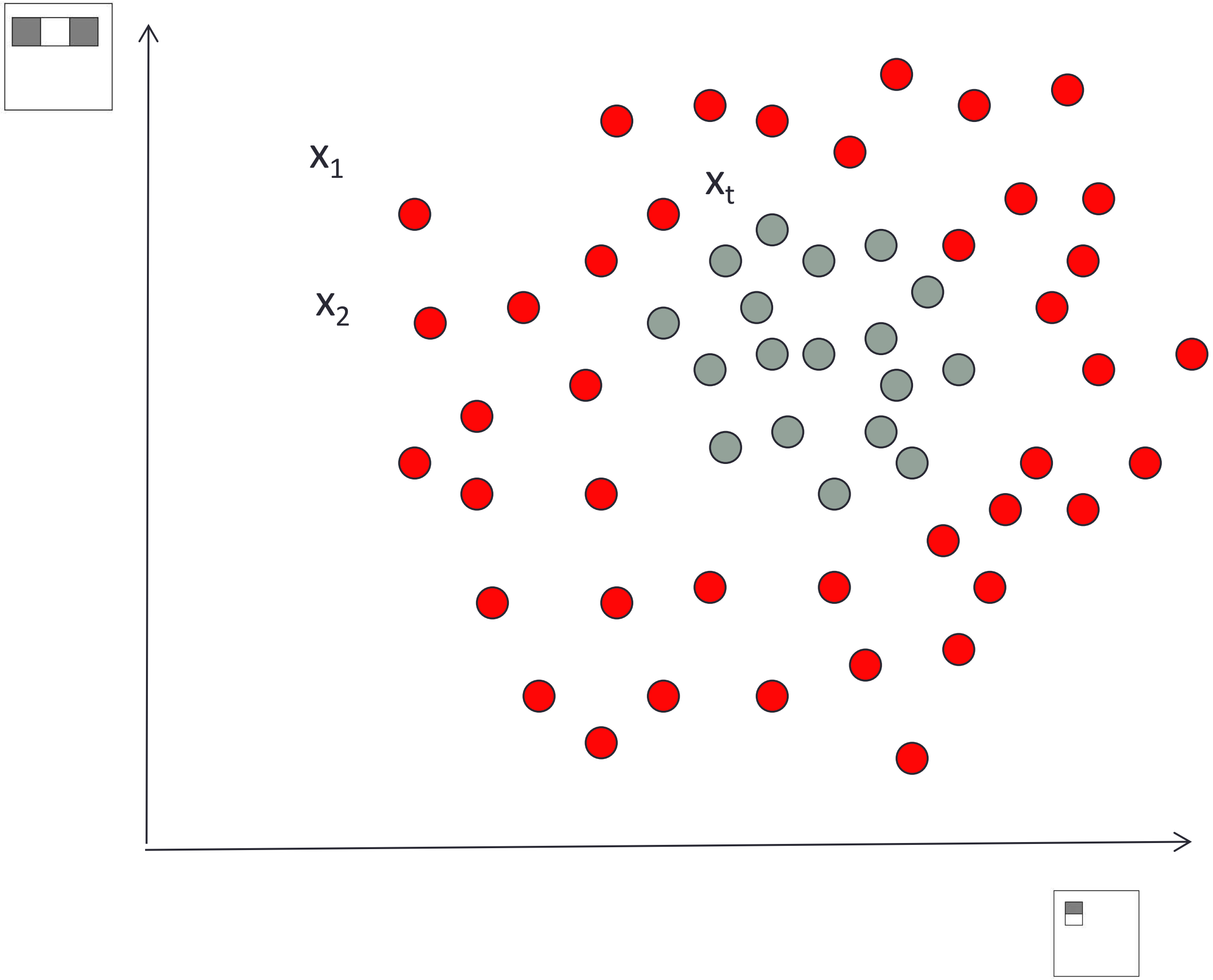
- It is a classification algorithm that joins several weak classifiers (weak) to form a single strong classifier (strong)
  BOOSTING >>

- The weak classifiers are defined at each iteration, and each is devoted to examples that are misclassified in the previous classifier:
  ADAPTIVE >>

From the Oxford Dictionary:
boost |bo͞ost|
verb [with obj.]
help or encourage (something) to increase or improve: *a range of measures to boost tourism*.

AdaBoost

ADABOOST EXAMPLE



Each point (image) is labeled with a class label:

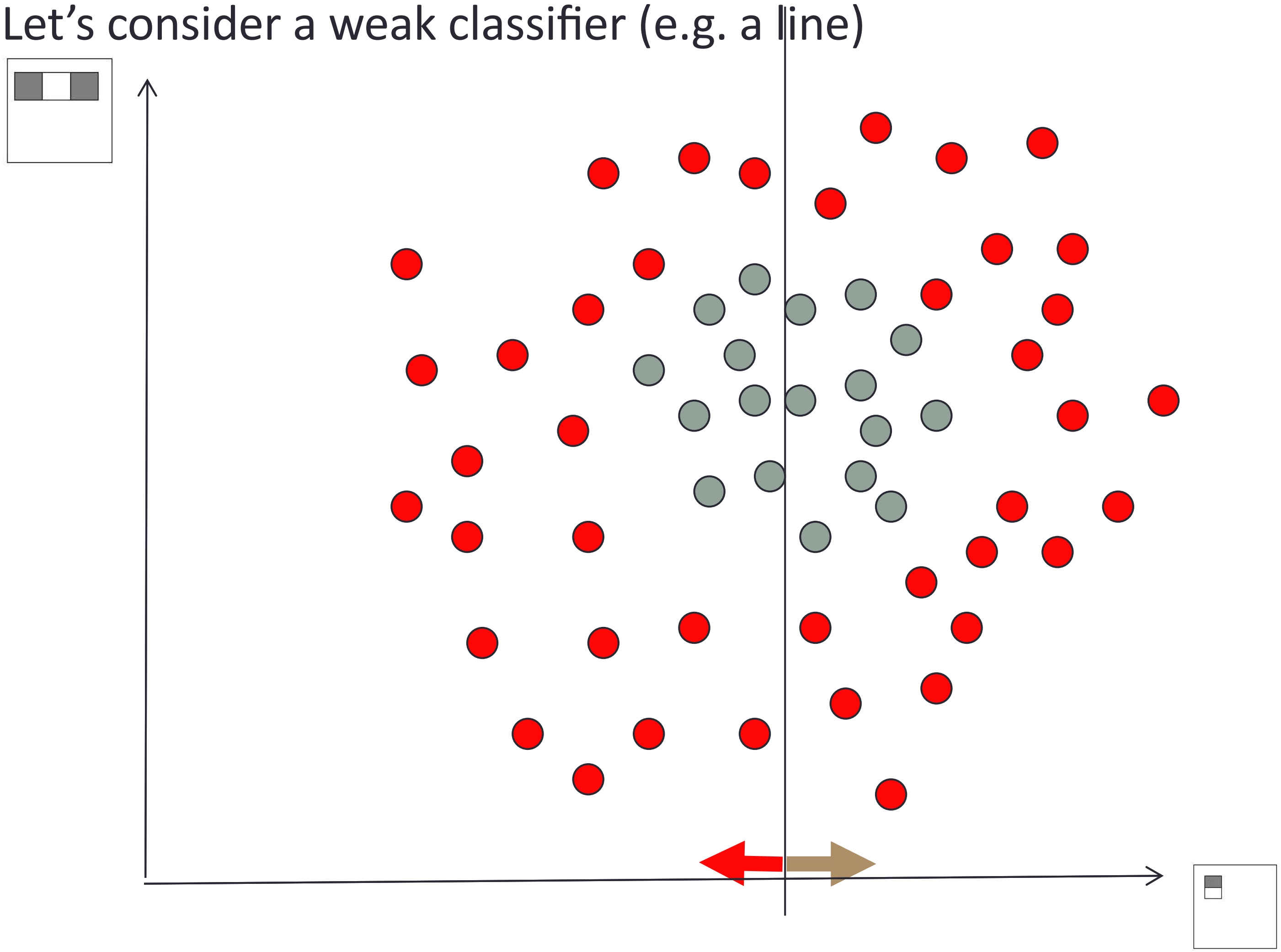
$$y_t = \begin{cases} +1 \; (\;) \\ -1 \; (\;) \end{cases}$$

and weight $w_t = 1/N$

N = points number.

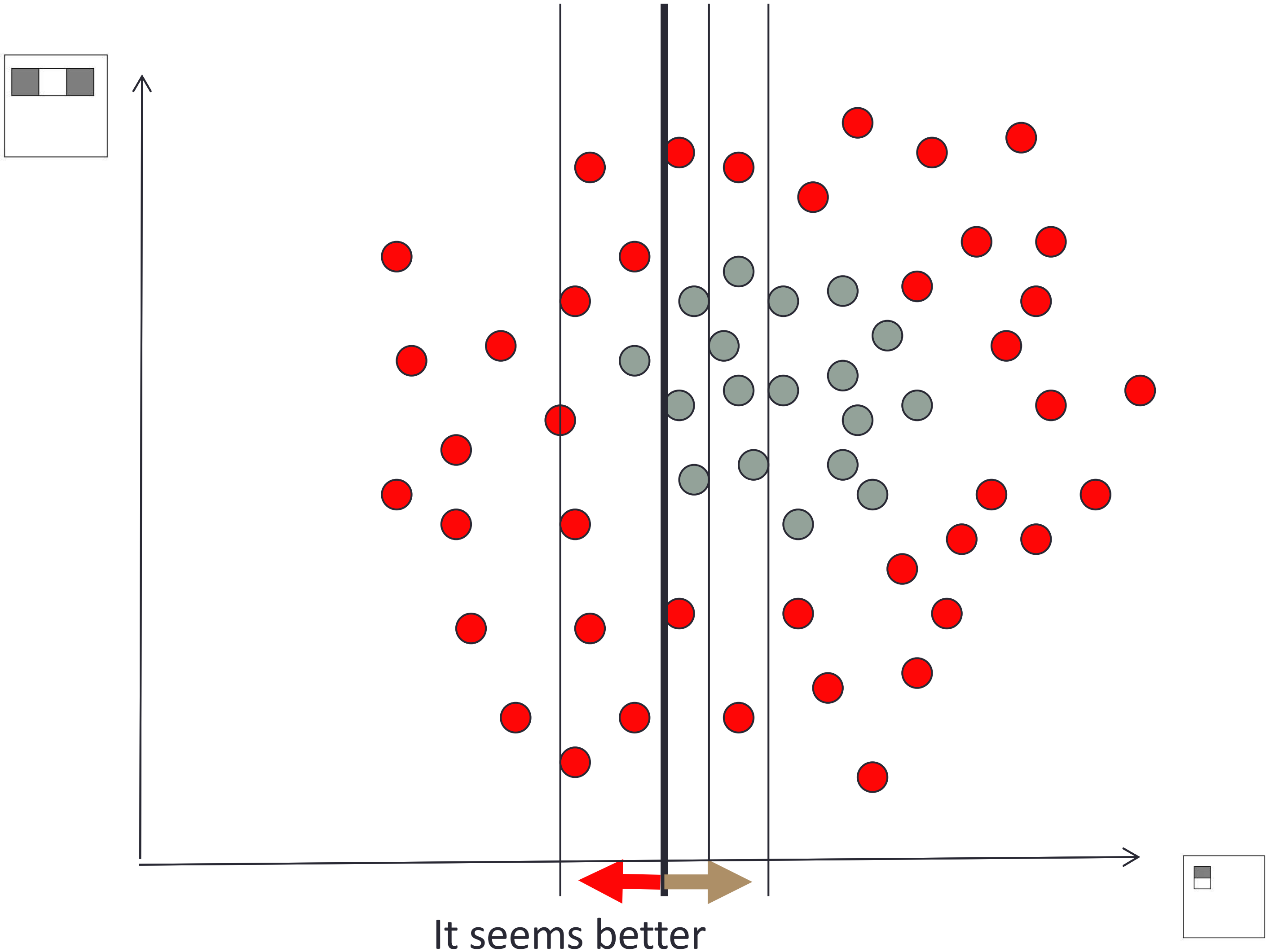
AdaBoost

Let's consider a weak classifier (e.g. a line)



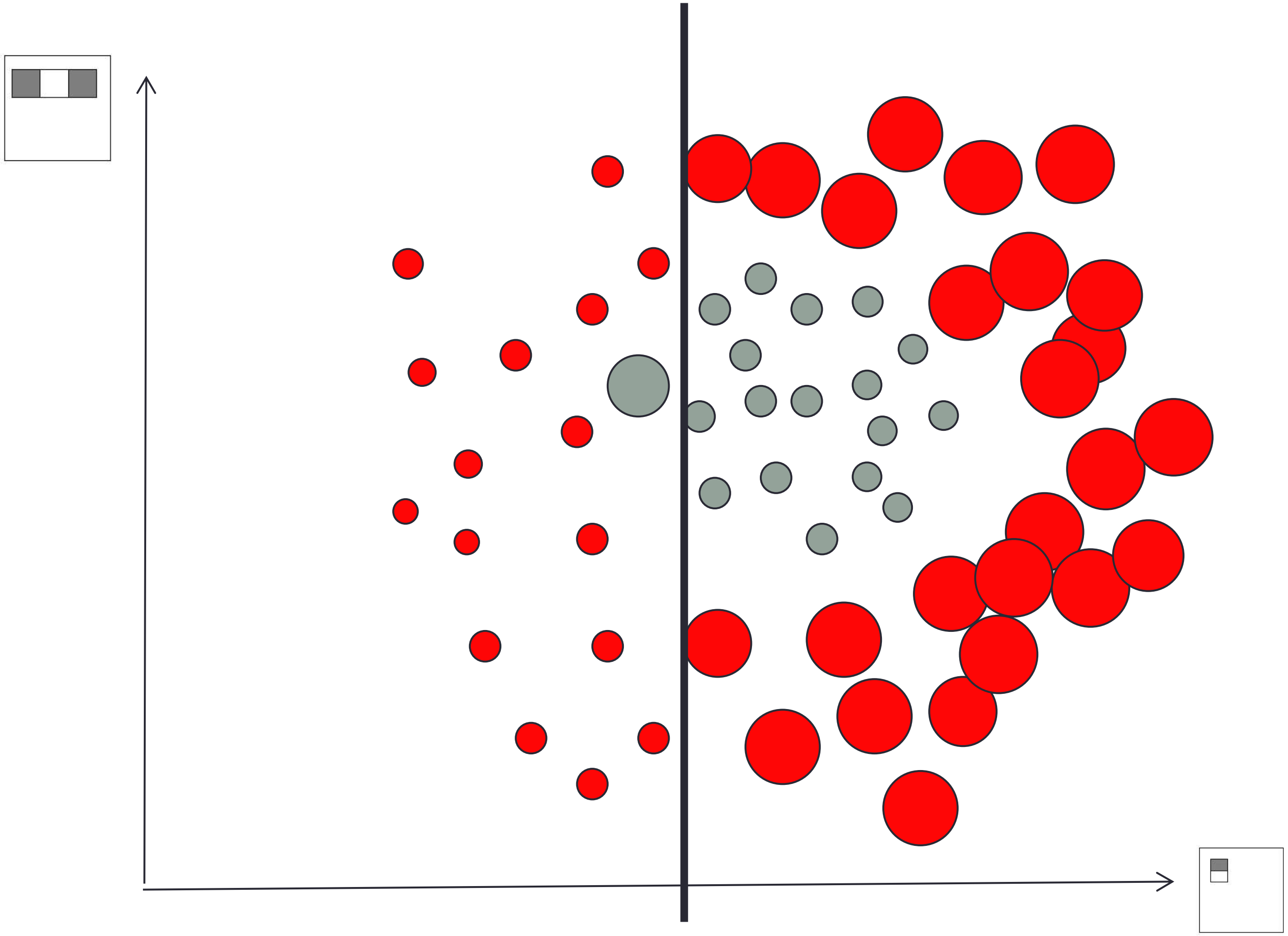
h => p(error) = 0.5 random

AdaBoost



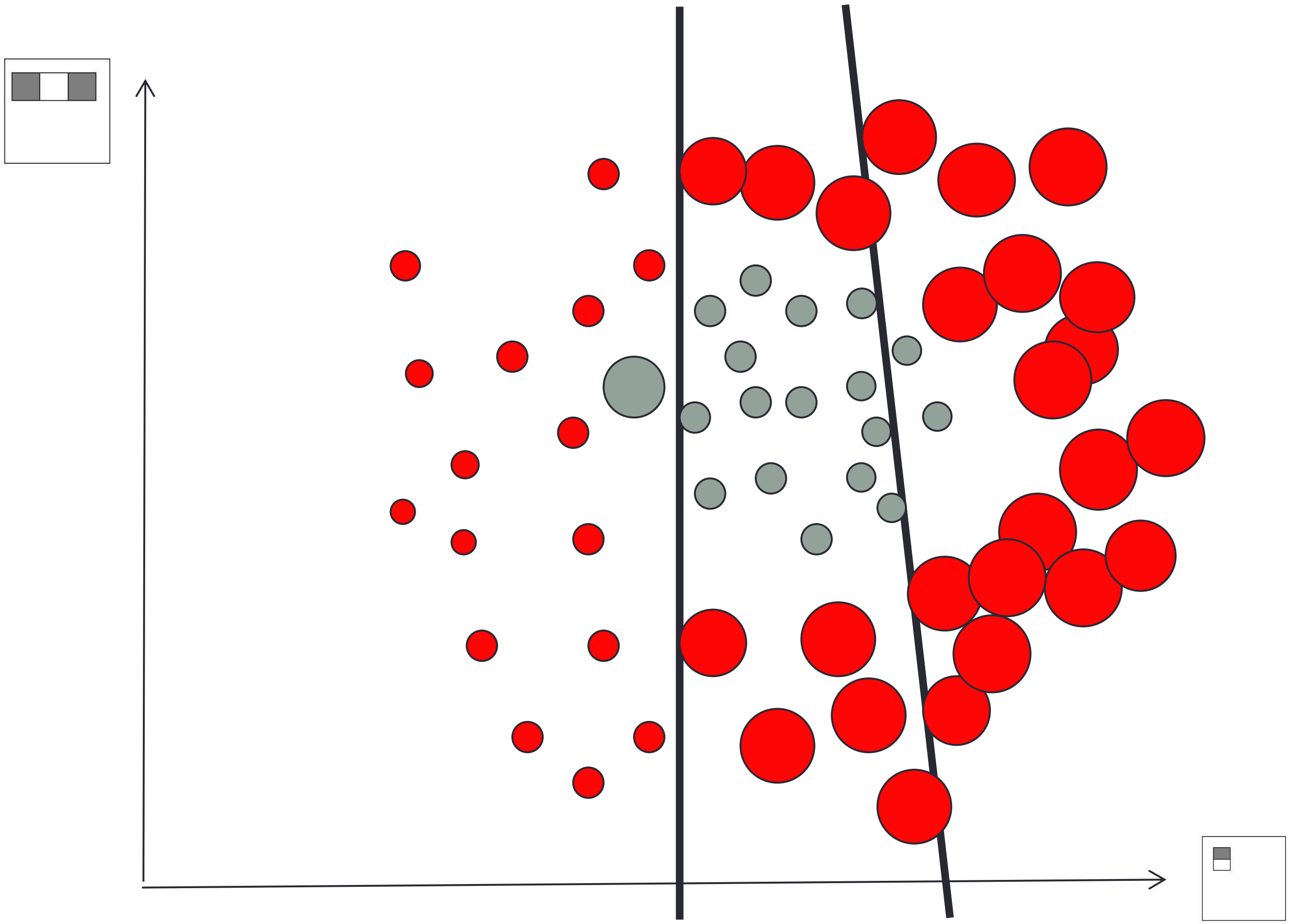
It seems better

AdaBoost



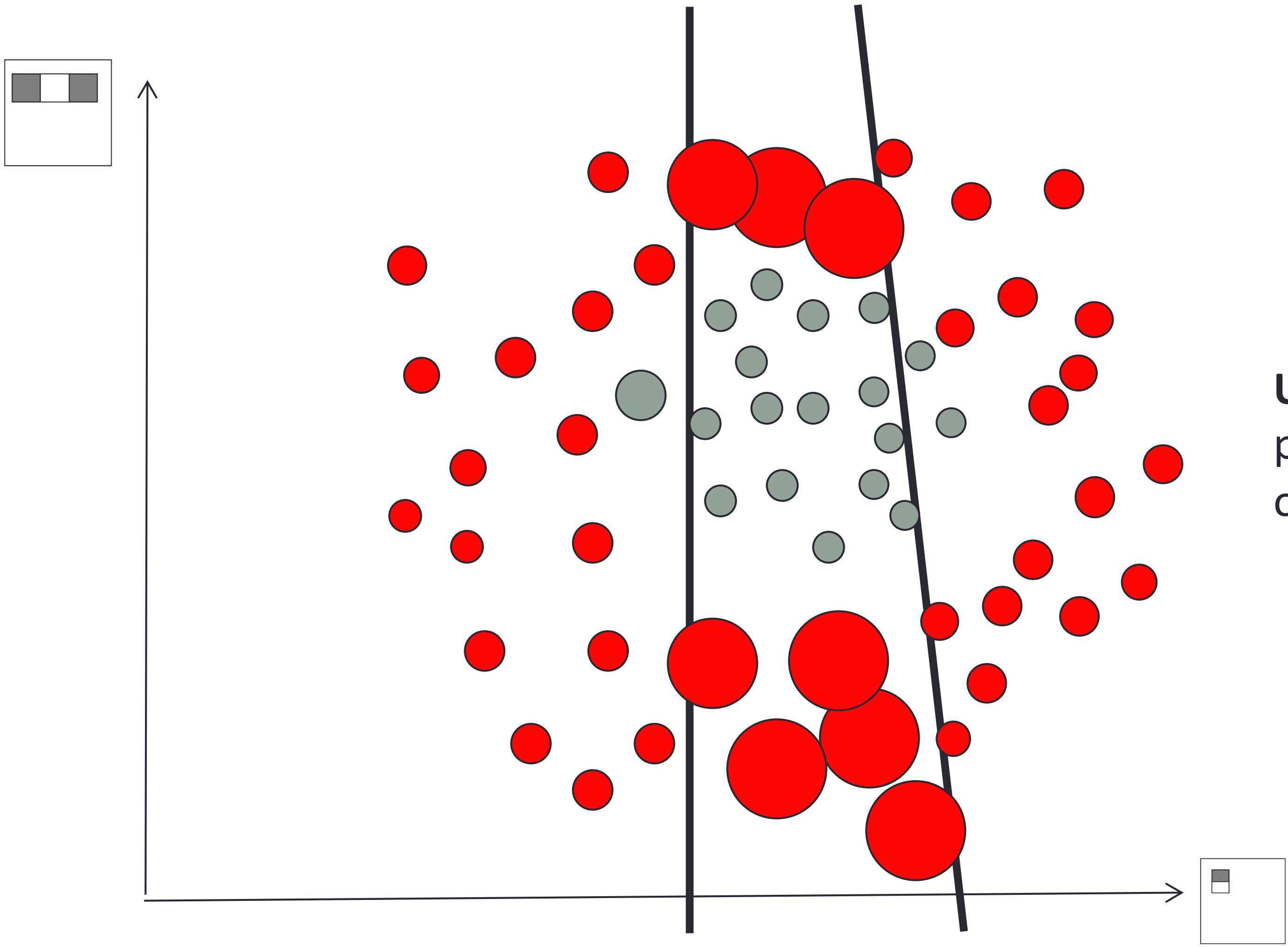
We consider the new problem and seek the best weak classifier.

AdaBoost



We consider the new problem and seek the best weak classifier.

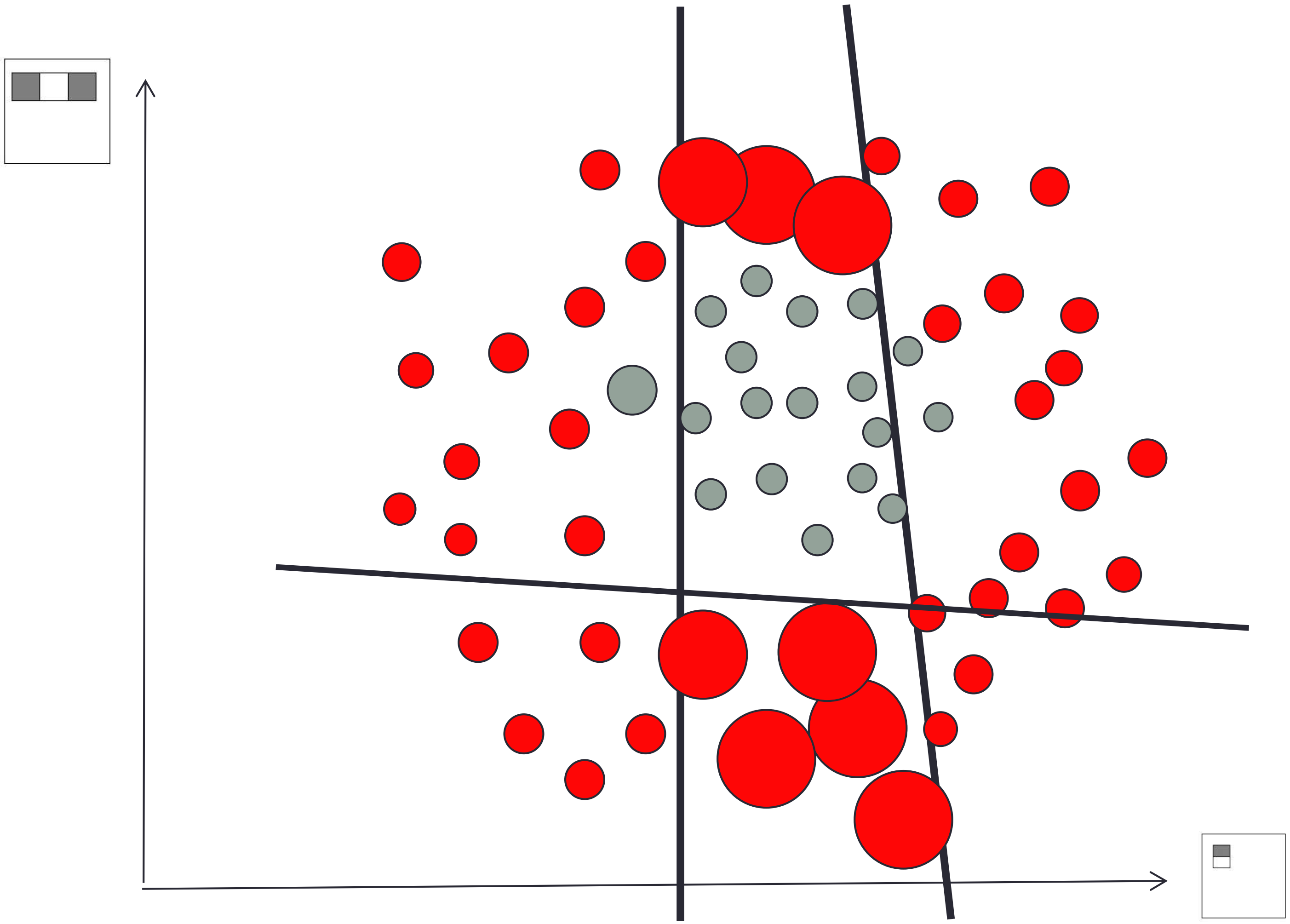
AdaBoost



Update the weights: giving priority to wrongly classified ones.

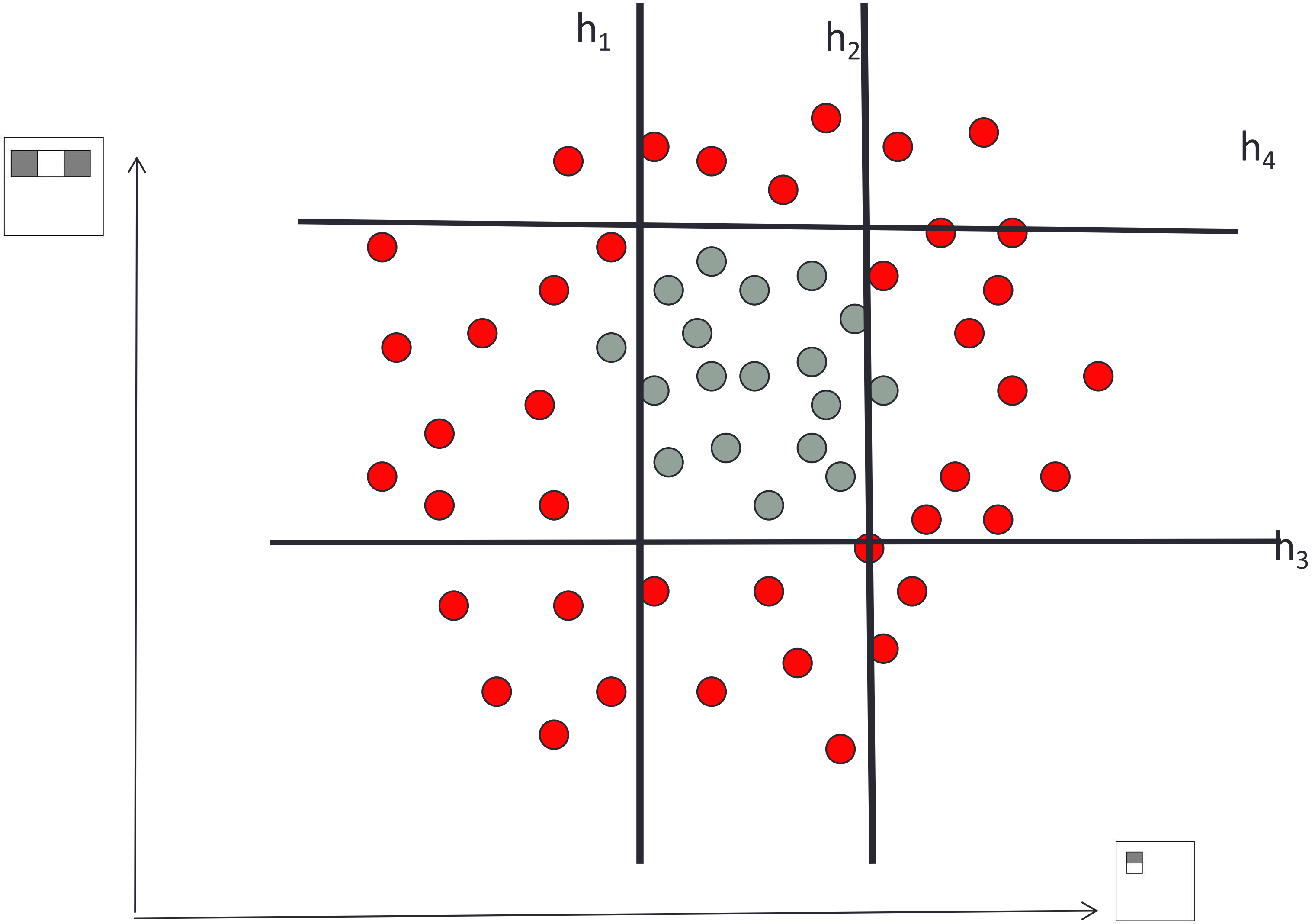
We consider the new problem and seek the best weak classifier.

AdaBoost



We consider the new problem and seek the best weak classifier.

AdaBoost

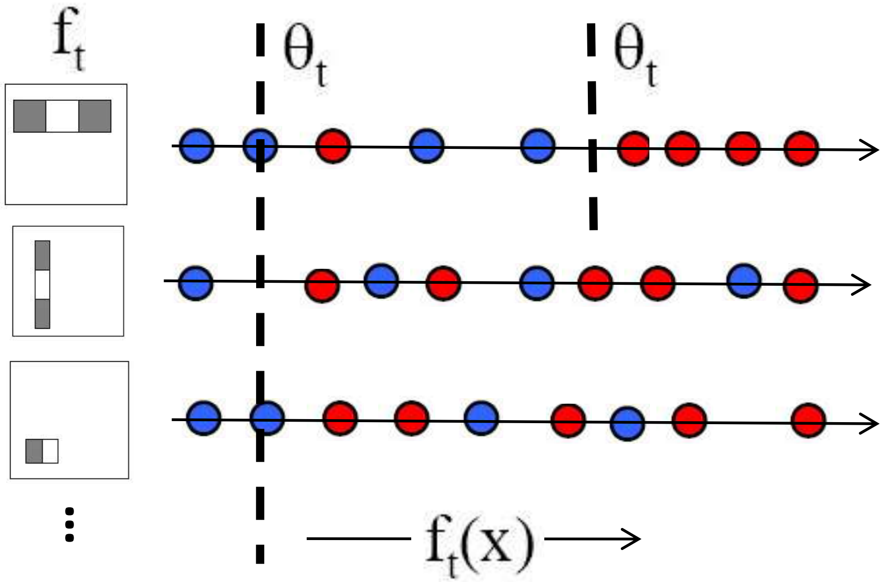


The strong (final) classifier C is composed as a sum of all weak classifiers. Once trained C(x), and given a new instance x', we compute where C(x') "falls”.

Viola-Jones detector: AdaBoost

WEAK CLASSIFIER

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



Outputs of a possible rectangle feature on faces and non-faces.

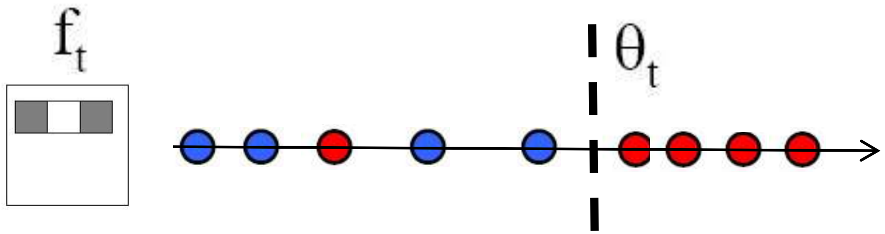
Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

From: Kristen Grauman

The Adaboost algorithm

- **Step 1**: Initialize i=1.
- **Step 2**: If i==T (max number of iterations), go to Step 6, else go to Step 3:
- **Step 3**: Given the descriptors f(x) = (f_1, f_2, ... f_t, ...) of a set of training objects (for which we know their class labels – face or not face), for each feature i look for the threshold Θ that separates the two classes with minimum error.



$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

- **Step 4**: We choose the feature f_t (x) that separates with the smallest error the training examples.
  - The chosen classifier h_t (x) is called weak classifier, t is the index of the feature.

- **Step 5**: Sum the new single classifier h_t (x) which corrects the error of the sum of the previous classifiers, to the previous ones. Give more priority to the wrongly classified samples (faces and no faces).

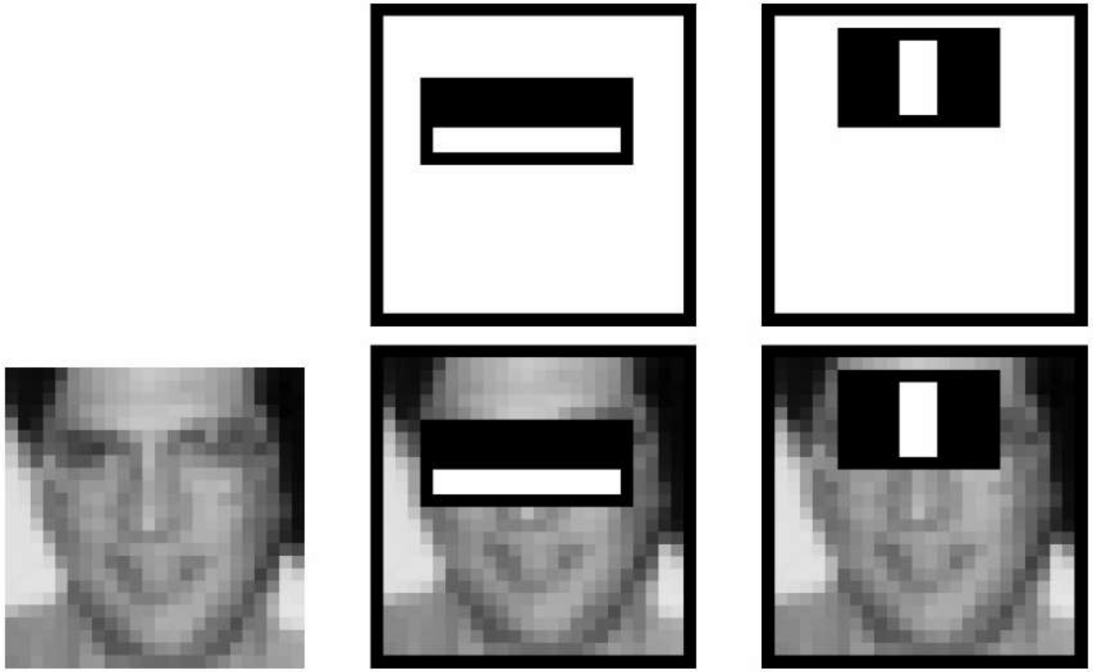
- **Step 6:** The ultimate classifier is the sum of the weak classifiers added at all iterations

$$F_t(x)=F_{t-1}(x)+h_t(x),\ t=1,2,\ldots T$$


Stage 10/15 (68%)
Feature 1/71

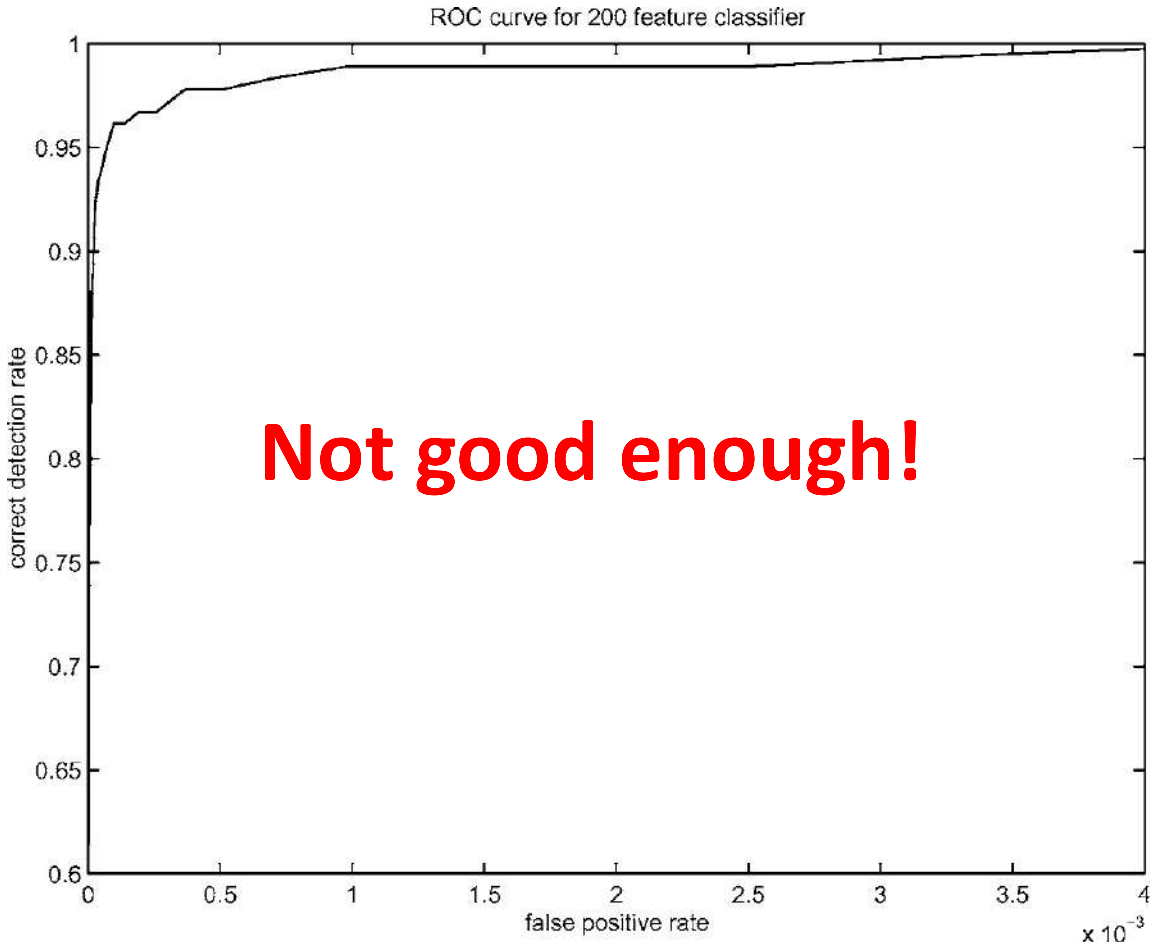
Boosting for face detection

- First two features selected by boosting:



This feature combination can yield 100% detection rate and 50% false positive rate

- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084



- Even if the filters are fast to compute, each new image has a lots of possible windows to search.

- How to make the detection more efficient?

Face detection

1) Haar image features
2) Integral images
3) AdaBoost
4) **Cascade of classifiers**

Cascade of classifiers

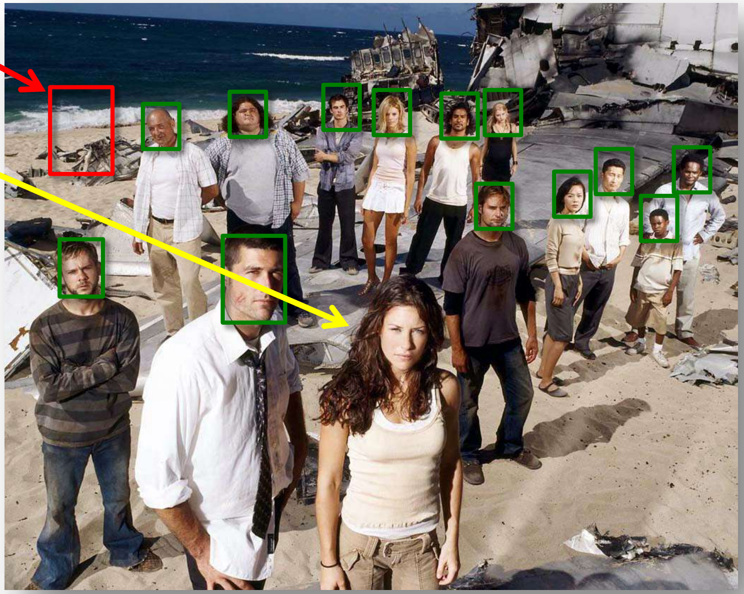
Now we have to design an appropriate classifier for the problem of face detection.

We recall that the main objective is to obtain an algorithm:

- Fast and
- Reliable.

In general, we need to accurately estimate the error detection by:

- *False Positives (FP)*
- *False Negatives (FN)*



True Positive Rate = TPr

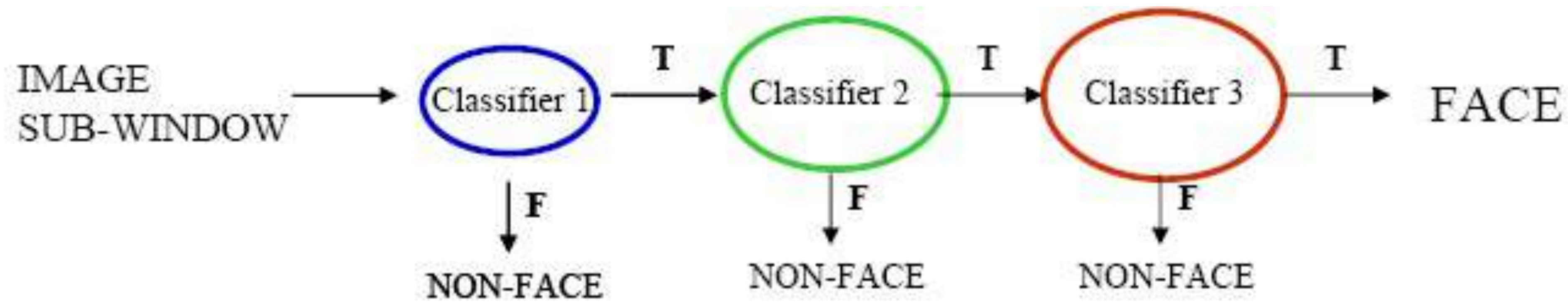
False Positive Rate = FPr

Cascade of classifiers

The idea of Viola & Jones is that each iteration of the cascade can accept a False Positive, but no False Negative, that is not to be missed any face!!!

What about the False Positives?

- The idea is that you can refine them with another classifier!
- CASCADE of classifiers



Cascade of classifiers

CRITERIA TO DESIGN A CASCADE

1) Each classifier is an AdaBoost classifier.

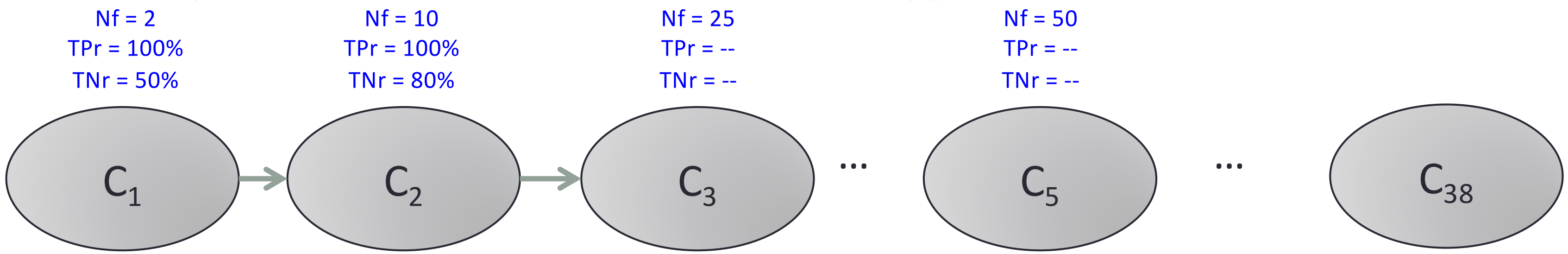
2) The first classifier C_1 is the simplest of all, and only classifies based on two features.

3) The following classifiers are more complex and use more features to refine the results of previous classifiers.

4) Each subsequent classifier C_i is **trained with the error** of the classifier $C_{i\text{-}1}$.

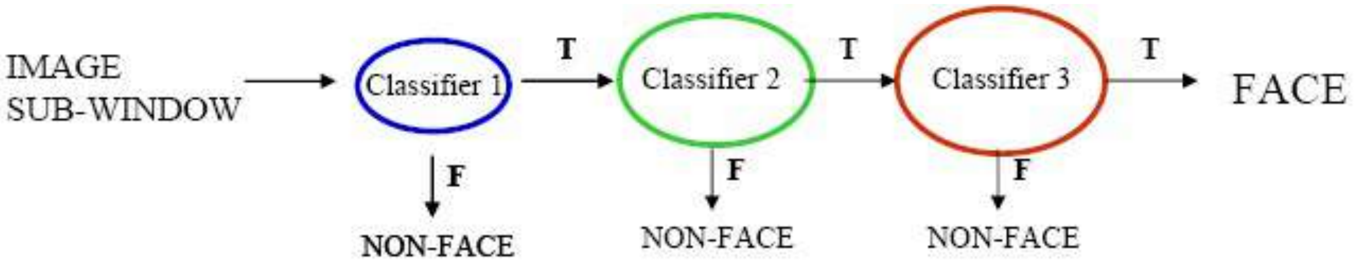
5) For each classifier C_i (during the training phase), we decide the FPr value we want to obtain, and add features until the decided value is not achieved! (N_f = number of features).

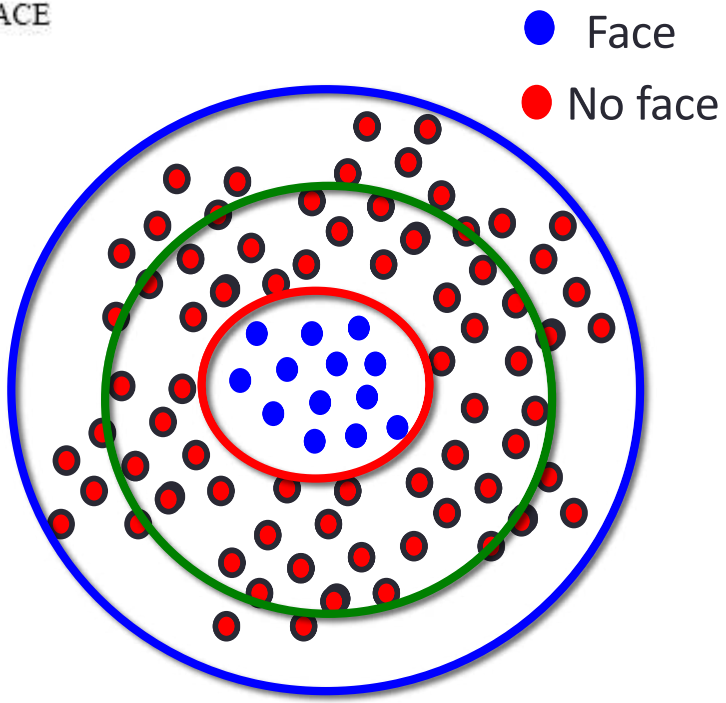
6) If FPr of C_i is not enough, we add another classifier, C_{i+1}.



Cascade of classifiers

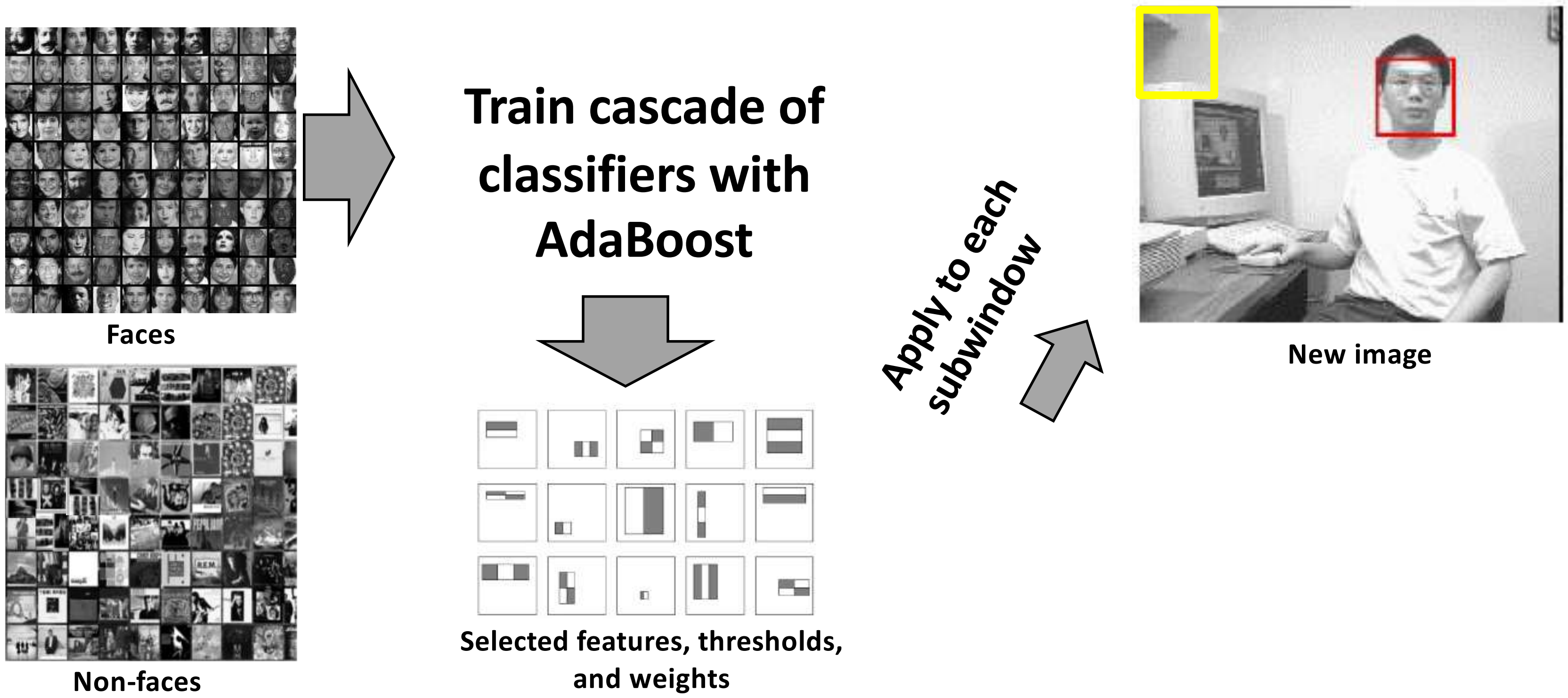
¿How to interpret the cascade in the feature space?





Each classifier in the cascade reduces the number of false negatives, keeping the number of successes (TP)!

Viola-Jones detector: summary

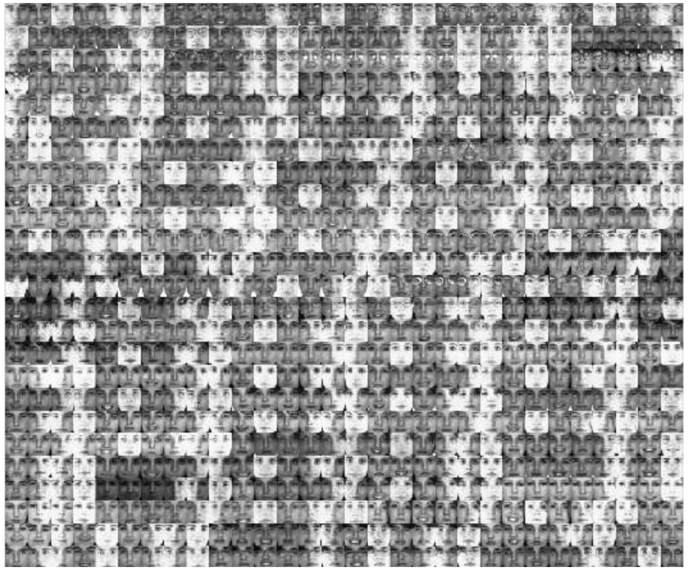


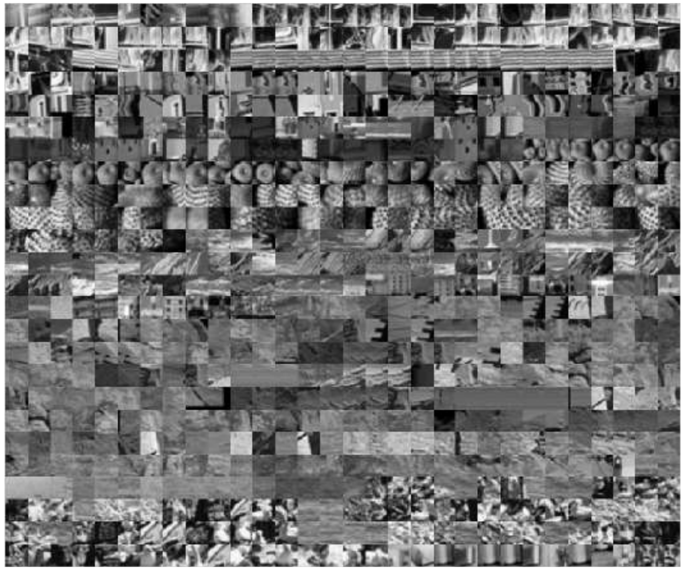
Kristen Grauman

The classifier "learns" from the data

Training data

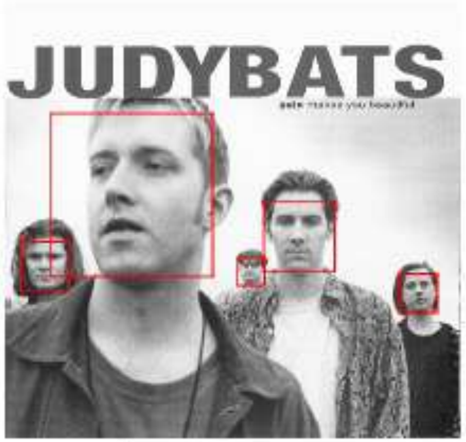
- 5000 faces (front) samples
- 108 no faces samples
- Many variations among individuals
- Different lighting, position, rotation
- Number of classifiers: 38.
- Number of features: 6060.
- Base Detector Resolution: 24x24 pixels
- Number of scales: 12.
- Scale factor for each scale: 1.25 (re-scaled features, not the image).
- Detection time: real-time

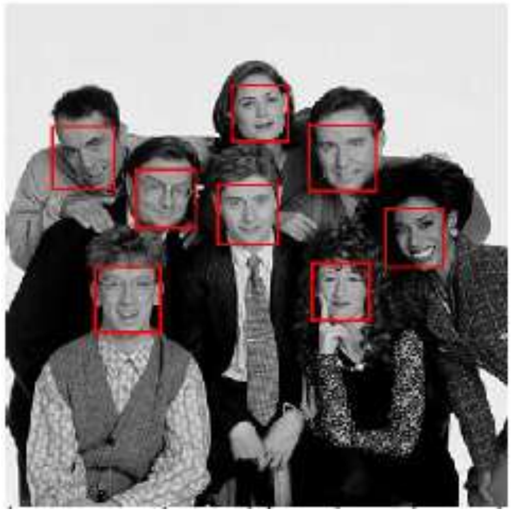


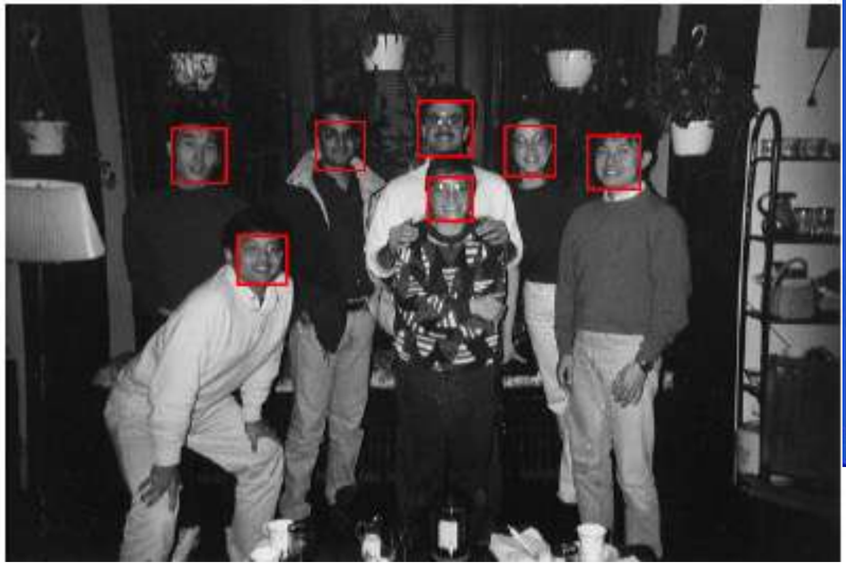


Face detection

RESULTS





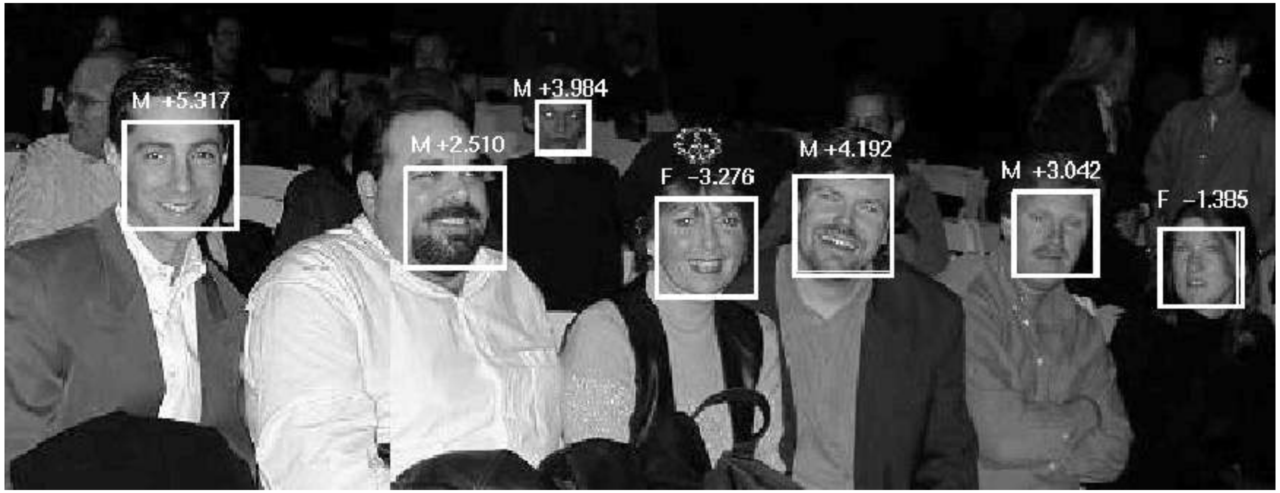


Other detection tasks

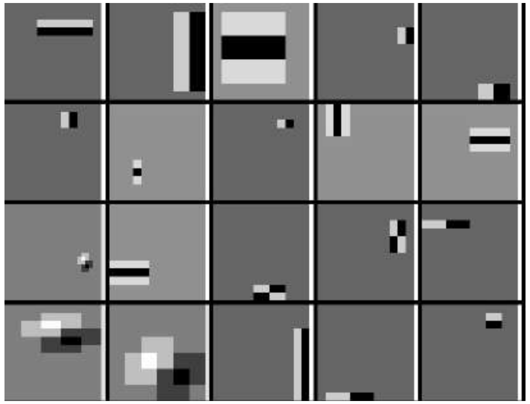
Facial Feature Localization

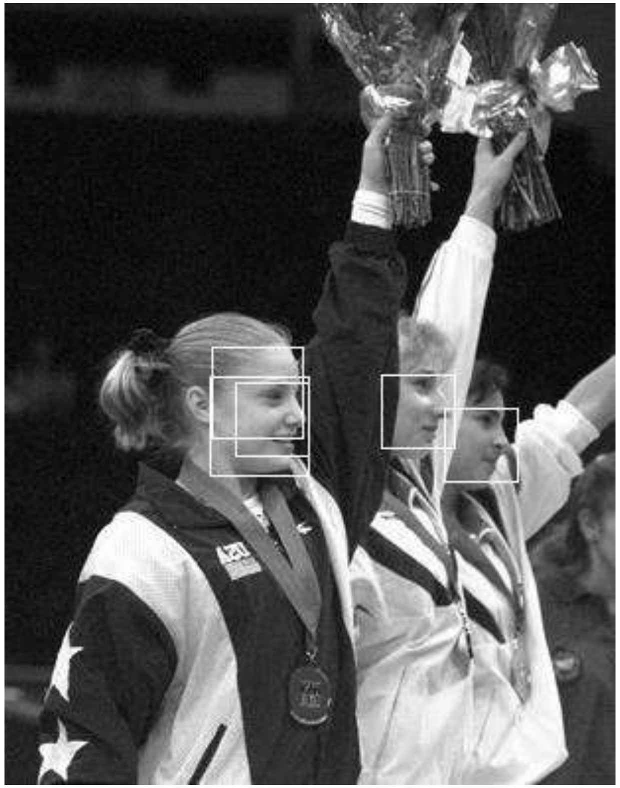
Profile Detection

Male vs. female



Profile Features



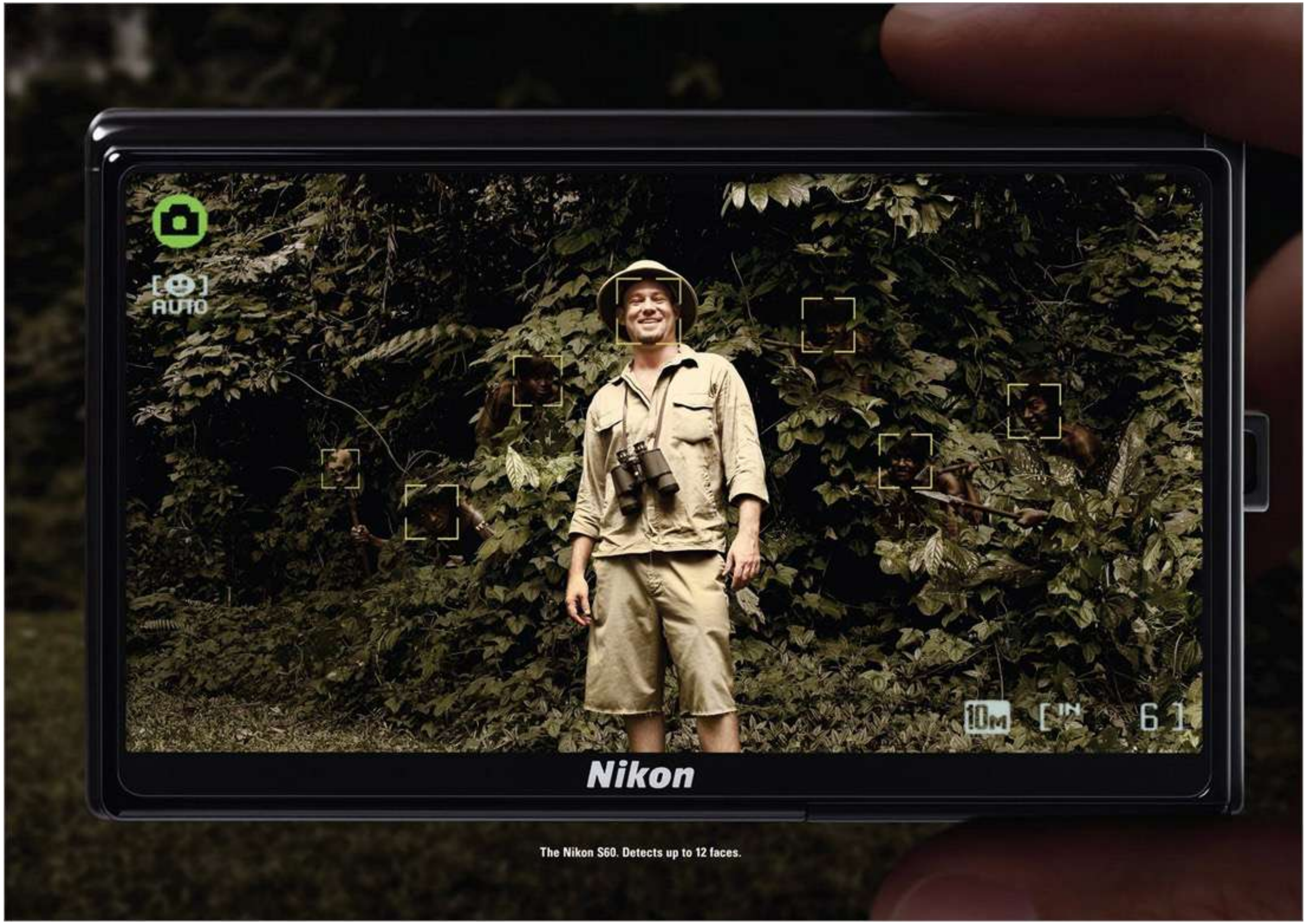


Face detection

APPLICATIONS

Auto-focus in digital cameras (+ smile detection).
Detection / Recognition of people (Video surveillance).
Data collection for cataloging / tagging photos (Facebook, iPhoto, etc.).

Funny Nikon ads



"The Nikon S60 detects up to 12 faces."

Consumer application: Apple iPhoto

[Photos for Mac]

Applications

Lexus LS600 Driver Monitor System


ZDNet Asia
Where Technology Means Business

Home | News | Insight | Reviews | TechGuides | Jobs | Blogs | Videos | Community | Downloads | IT Library

Software | Hardware | Security | Communications | Business | Internet | Photos

Search ZDNet Asia

News > Internet

Google now erases faces, license plates on Map Street View

By Elinor Mills, CNET News.com
Friday, August 24, 2007 01:37 PM

Google has gotten a lot of flack from privacy advocates for photographing faces and license plate numbers and displaying them on the Street View in Google Maps. Originally, the company said only people who identified themselves could ask the company to remove their image.

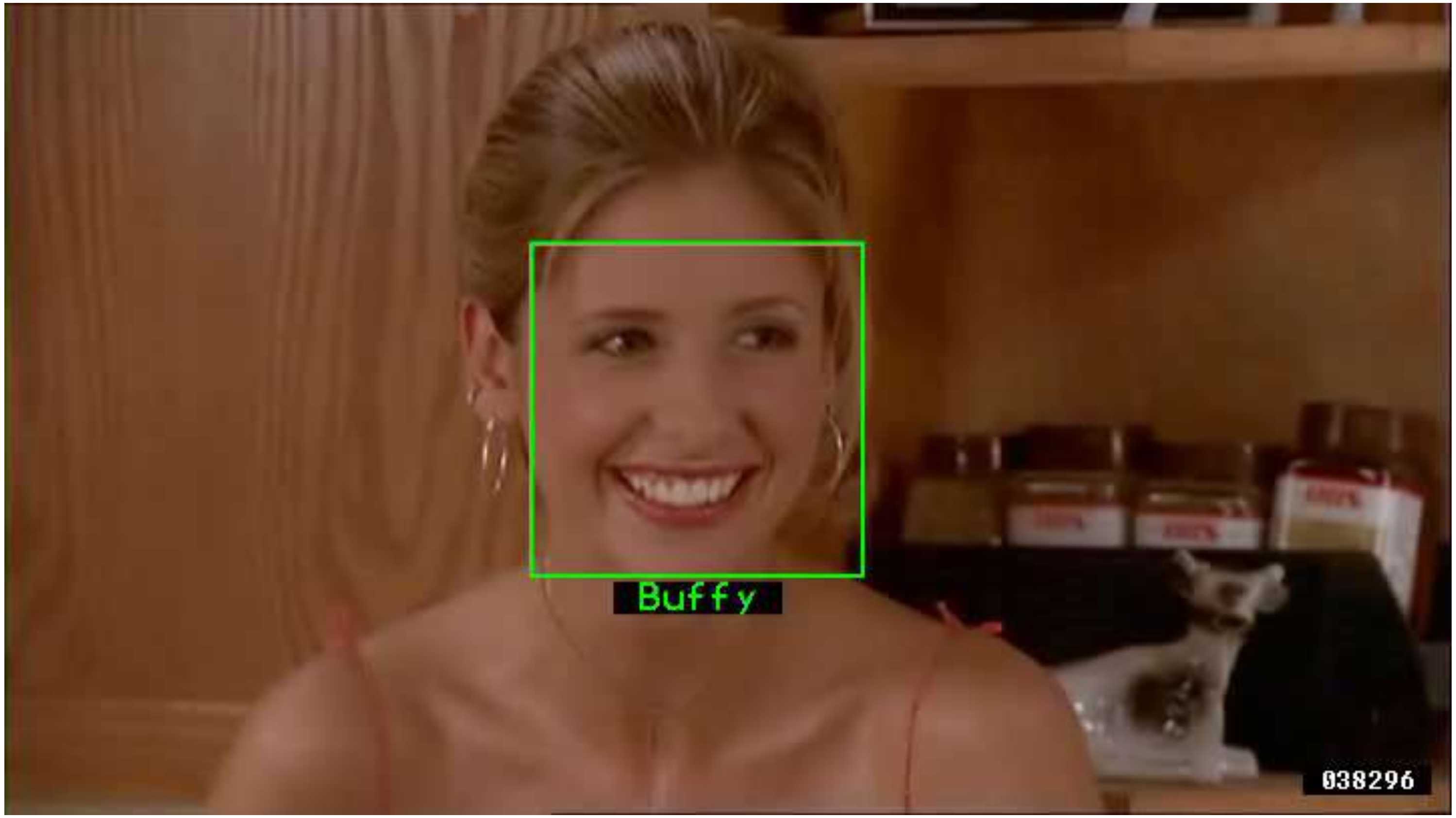
But Google has quietly changed that policy, partly in response to criticism, and now anyone can alert the company and have an image of a license plate or a recognizable face removed, not just the owner of the face or car, says Marissa Mayer, vice president of search products and user experience at Google.

"It's a good policy for users and also clarifies the intent of the product," she said in an interview following her keynote at the Search Engine Strategies conference in San Jose, Calif., Wednesday.

The policy change was made about 10 days after the launch of the product in late May, but was not publicly announced, according to Mayer. The company is removing images only when someone notifies them and not proactively, she said. "It was definitely a big policy change inside."

News from Countries/Region
» Singapore » India » China/HK/T
» Malaysia » Philippines » ASEAN
» Thailand » Indonesia » Asia Pacifi

What's Hot | Latest News
- Is eBay facing seller revolt?
- Report: Amazon may again be mulling Netflix bu
- Mozilla maps out Jetpack add-on transition plan
- Google begins search for Middle East lobbyist
- Google still thinks it can change China



Frontal faces detected and then tracked, character names inferred with alignment of script and subtitles.

Everingham, M., Sivic, J. and Zisserman, A., "Hello! My name is... Buffy" - Automatic naming of characters in TV video, BMVC 2006. http://www.robots.ox.ac.uk/~vgg/research/nface/index.html

Conclusions

- The Viola & Jones method is a method for automatic detection of faces in images.

- The Haar features provide an optimal description of the features of an image window that approximate the first and second derivatives of the image.
  - A huge set of features (... approx. 16M Features)

- Using integral images, the Haar features can be computed very quickly!
  - Efficient representation of the whole image with images

- The cascade of classifiers enables a very low rate of false negatives by detecting faces in real time!

  - Efficient selection of the features due to Adaboost.

- Fastest detector faces the literature.

Appearance-based Recognition

Let’s consider the problem of face recognition

The task is to recognize the identity of a person (from a given set of people images - training set).

Goal of the automatic analysis

The problem of face recognition:

Goal: *To define a space of image features that allow to represent objects based on their appearance (or a set of local features) in the image.*

There are several parts to consider:

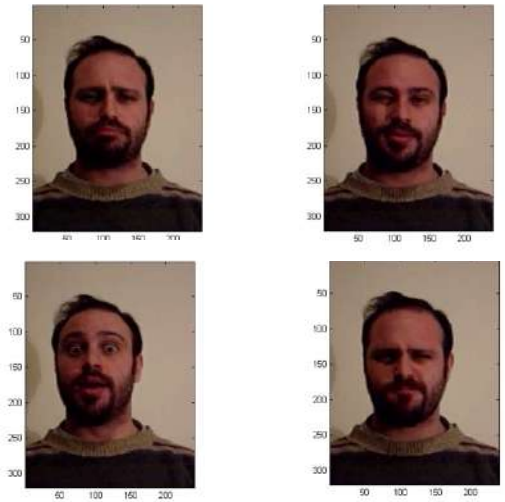
- **Define** an appropriate **representation** (descriptors of objects)
  - Normally, **reduce** the size of the data preserving the invariance and removing redundant dimensions.
- **Train a classifier** from a set of examples with their descriptors.
- **Recognize** a new face example using the learned model.

Problems and difficulties of facial analysis

Problem: Equivalence between stimuli.

In some cases the object recognition can find simple mechanisms of recognition associated with simple characteristics of images that are unambiguous signs of the presence of the object.

But in most cases there is no other way that learning complex descriptions.



Dimensionality and redundancy

The appearance ...

Instead of storing features or 3D models, directly store a collection of many views of the object.

How can we store this information in a compact and efficient way?

How do we encode this information as interesting visual representation?

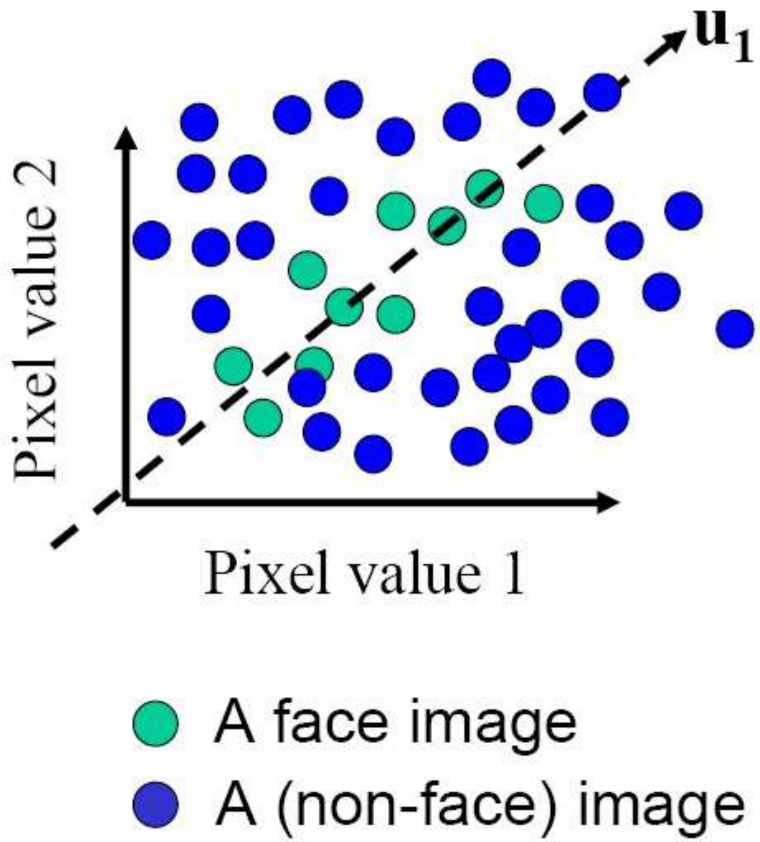


The space of all face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
- However, relatively few 10,000-dimensional vectors correspond to valid face images
- We want to effectively model the subspace of face images

The feature space of all face images

- We want to construct a low-dimensional linear subspace that best explains the variation in the set of face images



Index

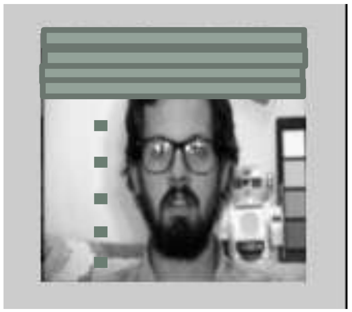
1) Method of face recognition – eigenfaces
   1) Feature space of eigenfaces
   2) Algorithm for dimensionality reduction
   3) Eigenface trick

2) Examples and applications

Many slides adapted from K. Grauman, S. Lazebnik, J. Vitrià and D. Lowe

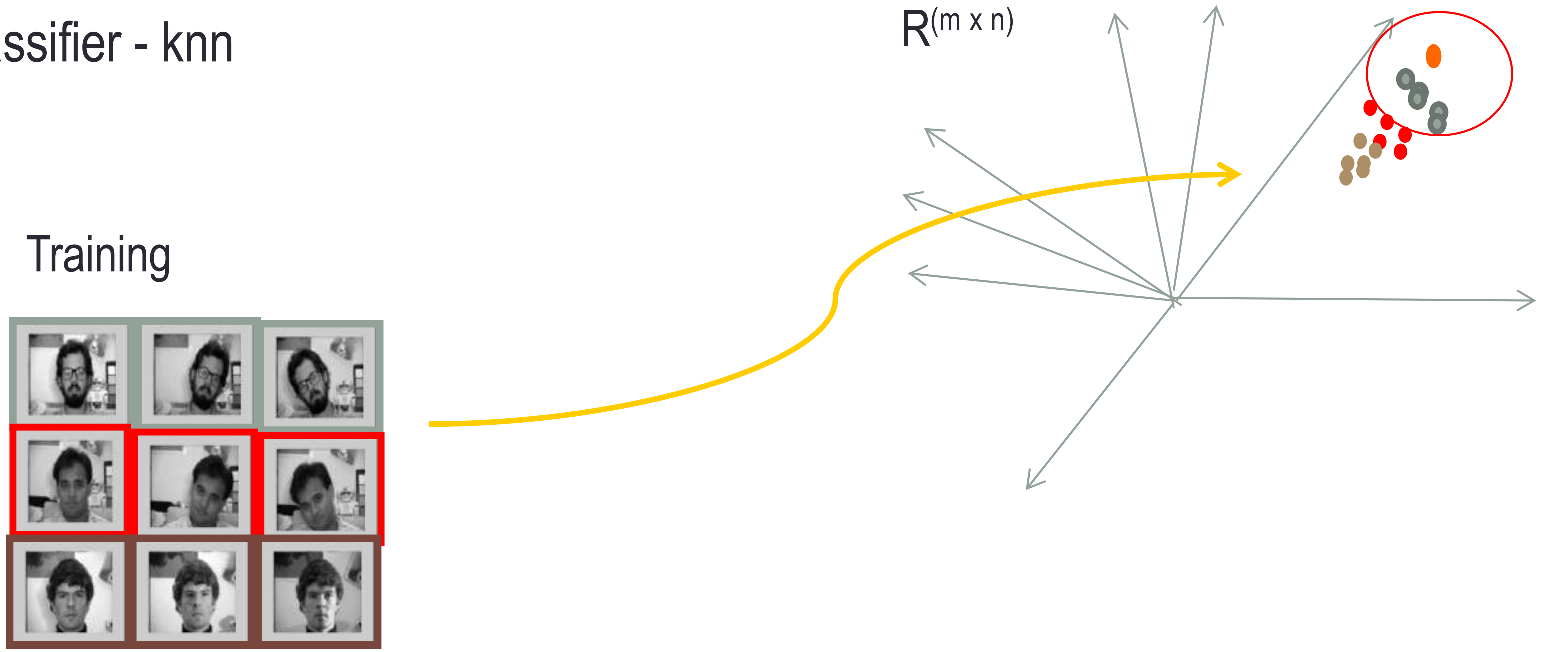
The feature space of faces and eigenfaces recognition

1. Features



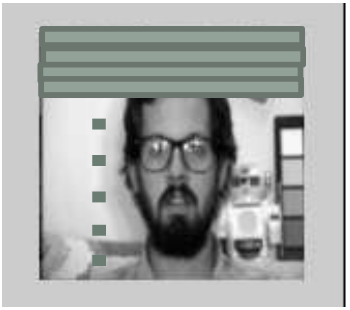
$(x_1, x_2, \ldots\ldots\ldots\ldots x_{(m \times n)})$

2. Classifier - knn



The feature space of faces and eigenfaces recognition

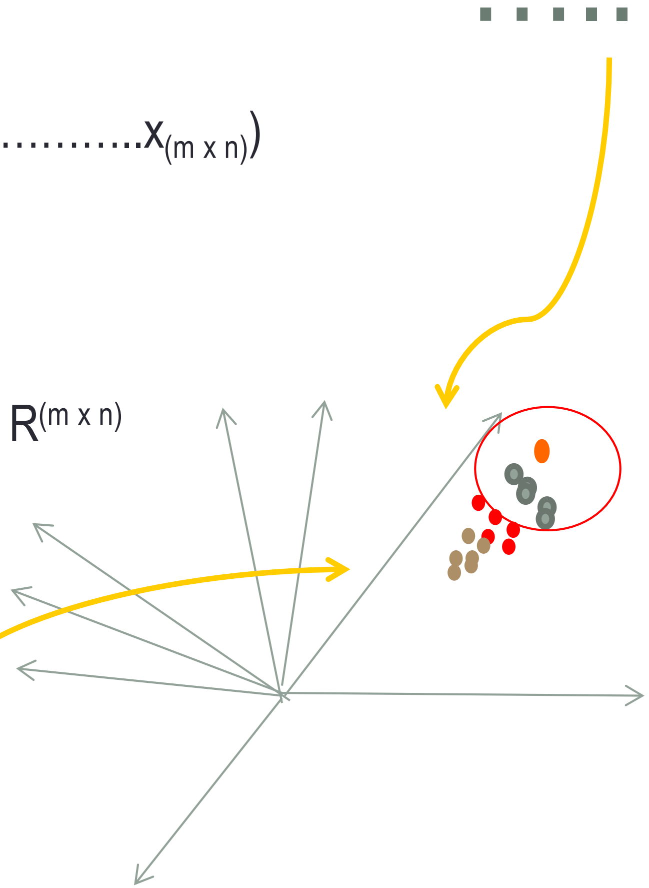
Representation of faces as points in high dimensional space



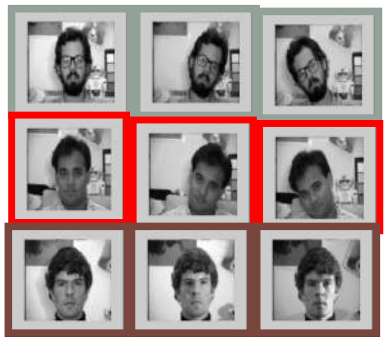
$(x_1, x_2, \ldots\ldots\ldots\ldots x_{(m \times n)})$

Each image has m rows and n columns -> defines a vector of (mxn) elements.

$R^{(m \times n)}$



Training



Recognition:

Classifier KNN – assigns the majority label of the k closest neighbors of the training set.

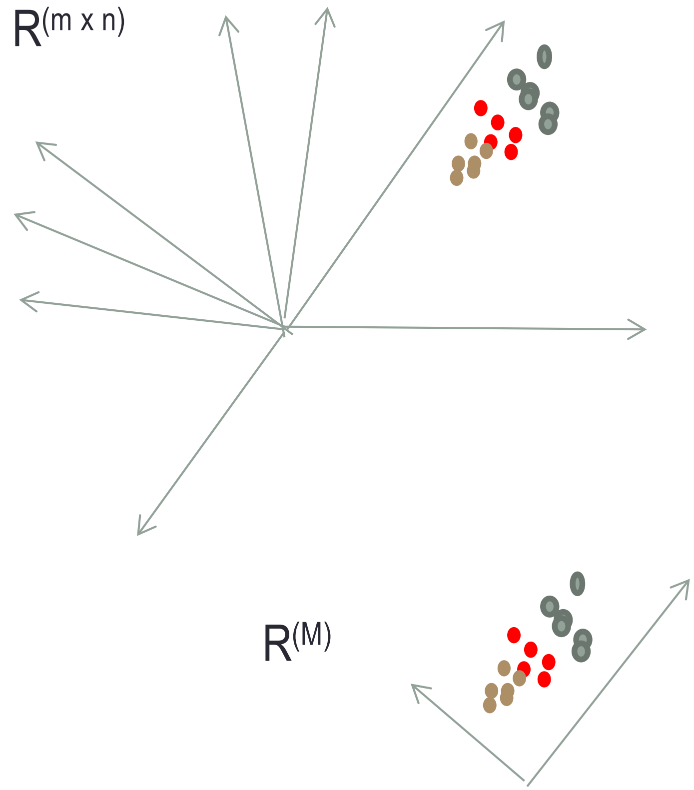
Eigenfaces

Representation of faces as points in high dimension space

•We use KNN in the R^{mxn} space -> costly and slow (m x n = 256 * 256 = 65536).

•Can we find a more compact representation of images where each face is represented via a small set of parameters?

•We look for a transformation of the original space to a smaller space (M << (m x n)), where faces are represented with their coordinates in this new space R^M?



What is the space of the faces? - Build from the training set of face images!

Eigenfaces method: Dimensionality and redundancy

Tool: dimensional reduction of the original data.

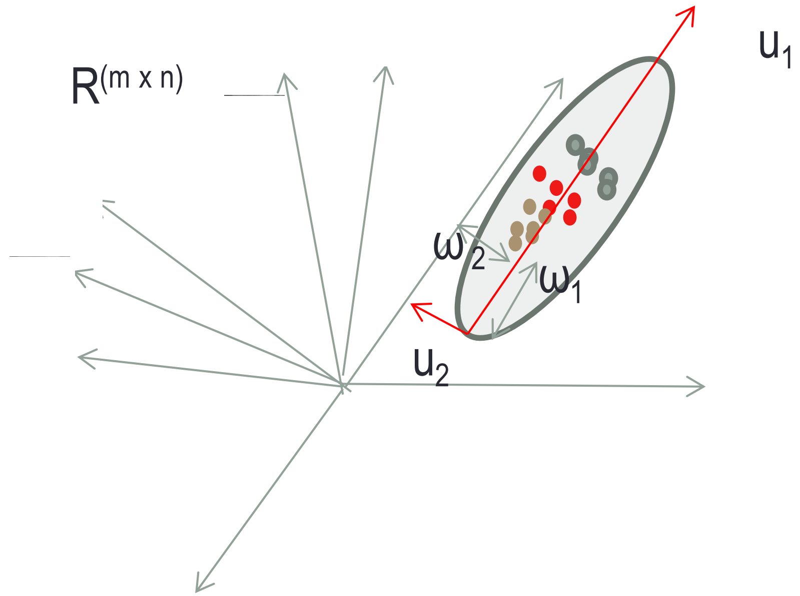
$$T:(x_1, x_2, \ldots, x_n) \rightarrow (y_1, y_2, \ldots, y_m), m << n;$$

... Retaining the information necessary to classify, recognize, etc!

And removing - or minimizing - information that is not relevant (lighting, small variations ..).

Eigenfaces

How to classify the faces?



If you have the reduced space and want to classify a face X_i:
Project it into the new (reduced) space:

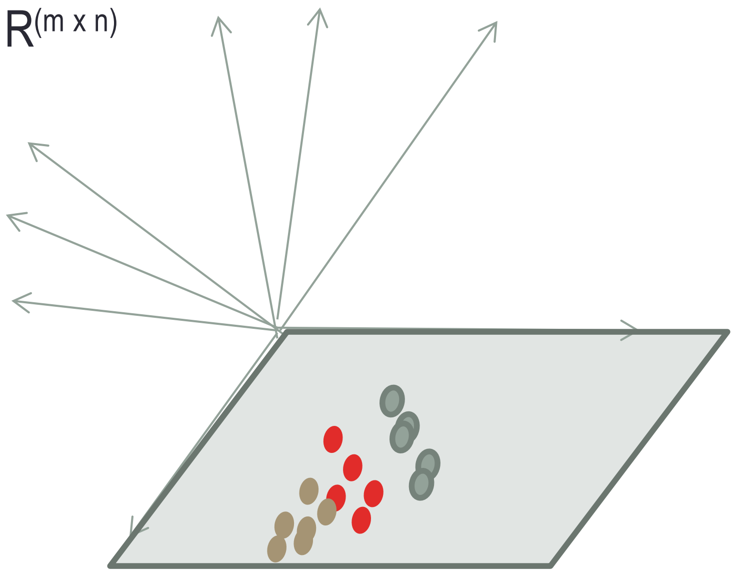
$$Y_i = (X_i - \bar{X})^T * (u_1, u_2, \ldots u_M) \longrightarrow (\omega_1, \omega_2, \ldots \omega_M)$$

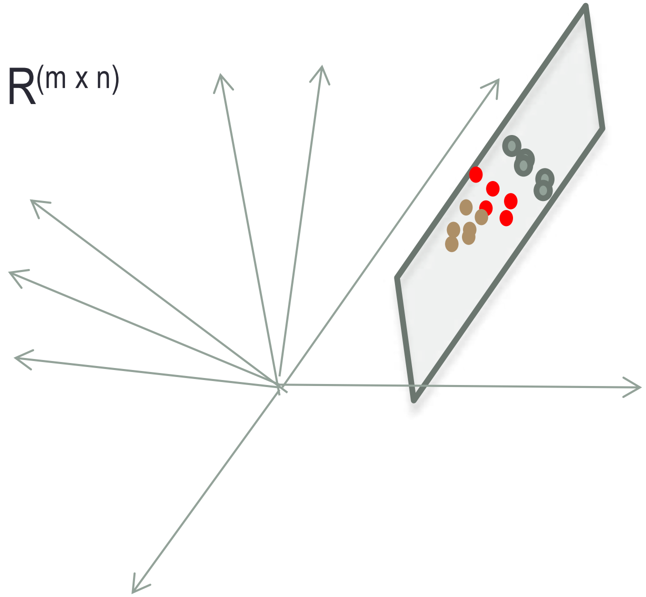
and apply the classifier e.g. knn considering the k projected neighbor faces of the training set.

How to build a reduced space?

Often the data "live" in a smaller space R^M (M << (m x n)).
The data are not always uniformly distributed.

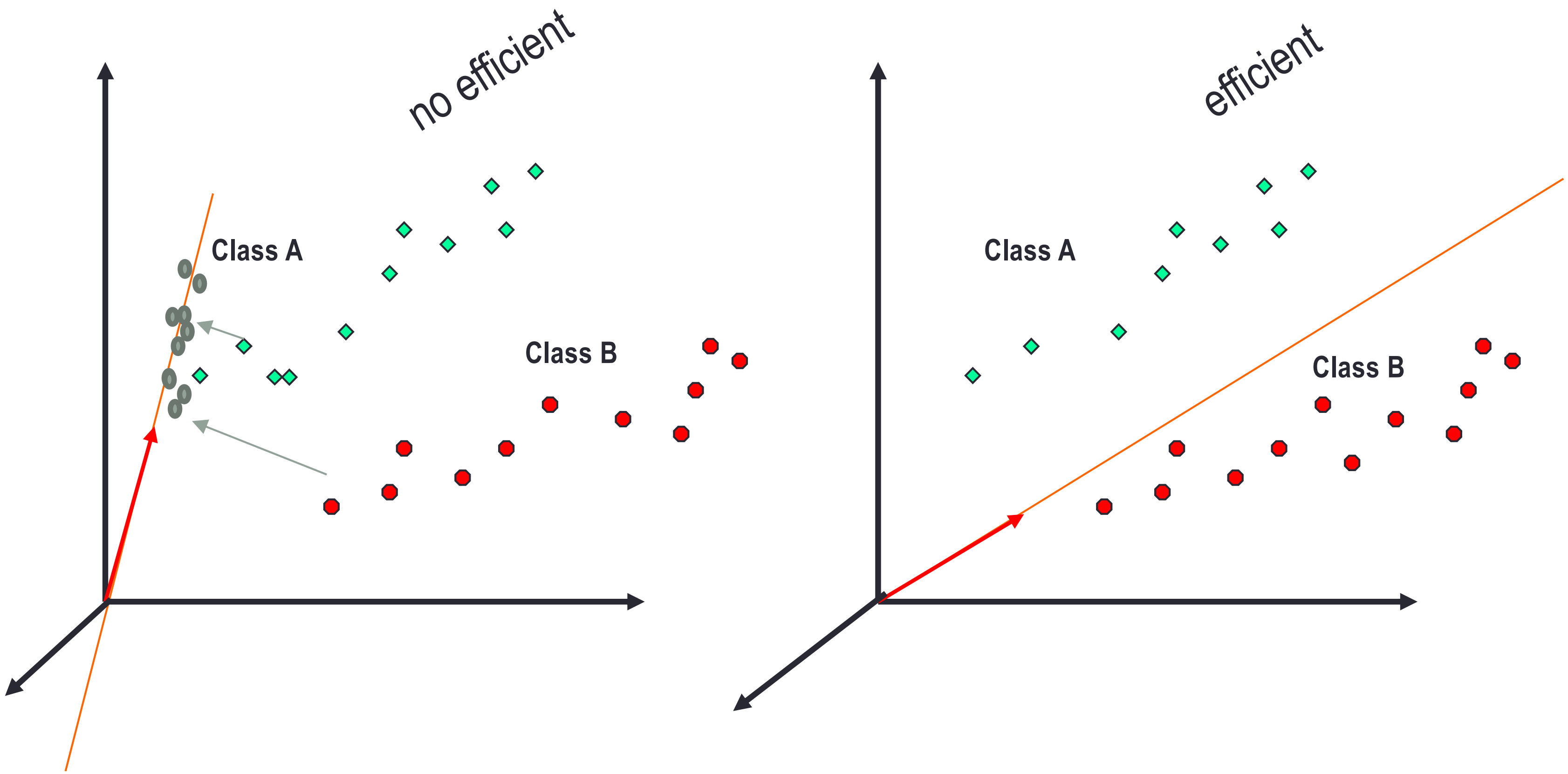
The technique that allows to find the reduced subspace where data live, is called: Principal Component Analysis (PCA).





Properties of Principal Component Analysis

- seeks directions efficient for representing data in all its variance
- reduces the dimensions of data
- accelerates the execution time of the algorithms



Index

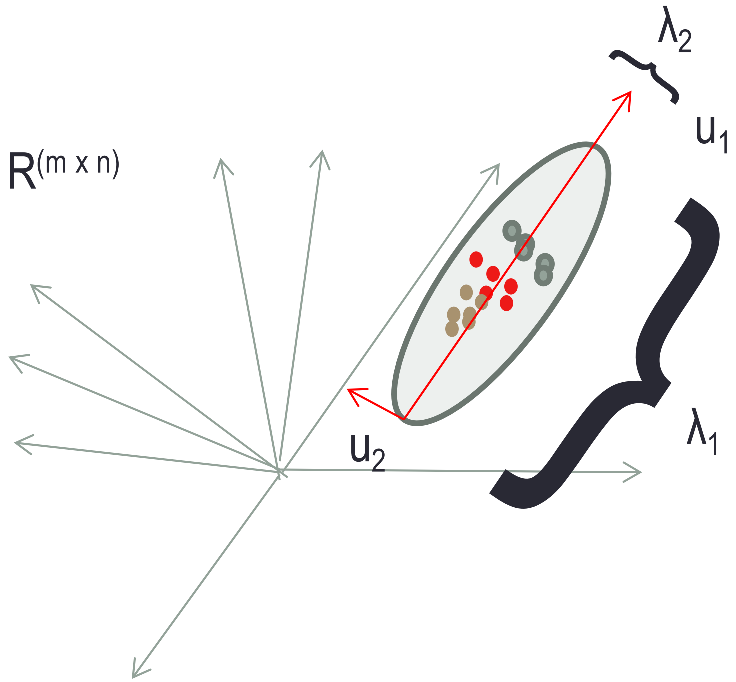
1) Method of face recognition – eigenfaces
   1) Feature space of eigenfaces
   2) Algorithm for dimensionality reduction
   3) Eigenface trick

2) Examples and applications

Many slides adapted from K. Grauman, S. Lazebnik, J. Vitrià and D. Lowe

Principal Component Analysis: the algorithm

According to the PCA, if we compute:



- the **eigenvectors** (e1, e2, ...) of the **covariance matrix** define the axis of maximum variance,
- and the **eigenvalues** give a measure of the **variance of** the data.

$$\Sigma = \frac{1}{M}\sum_{i=1}^{M}(X_i - \overline{X})(X_i - \overline{X})^T = AA^T;$$

Eigenfaces

How to find the best sub-space to represent the face family?

- Given M images of size (mxn) -> let's construct vector X_i, i = 1 ... M in the space $R^{m \times n}$.

- Given the images of training: X_i, i = 1 ... M, let's compute the mean image: $\bar{X} = \frac{1}{M}\sum_{i=1}^{M} X_i$

- Construct the covariance matrix Σ:

$$\Sigma = \frac{1}{M}\sum_{i=1}^{M}(X_i - \bar{X})(X_i - \bar{X})^T = AA^T; \quad A = [X_1 - \bar{X}, X_2 - \bar{X}, \ldots X_M - \bar{X}]$$

- The eigenvectors of the covariance matrix are called eigenfaces.

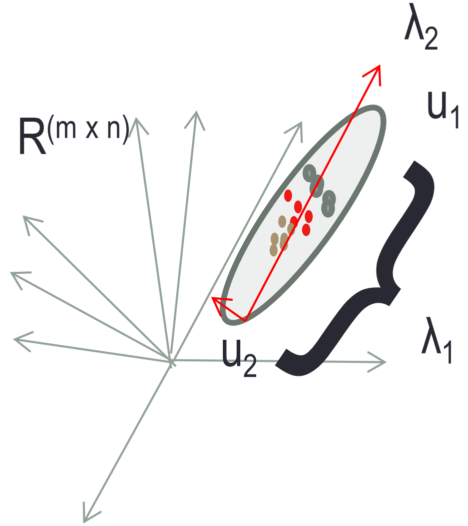
- => A is of size (mxn) x M, A^*A^T is of size (mxn) x (mxn)! En nuestro caso $A_{(65000x200)}$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Eigenfaces

How to find the best sub-space to represent the face family?

•Eigenvectors (u_1, u_2, ...) of the covariance matrix Σ define the subspace that represents the data (faces) distribution.



•Remember: given any matrix *B*: a vector u is called eigenvector, iff:

$$Bu = \lambda u$$

•λ is the eigenvalue of the matrix.

•Eigenvectors (u_1, u_2, ...) are orthogonal and form a basis.

•The original image once centered is projected into the eigenspace: X-> Y:

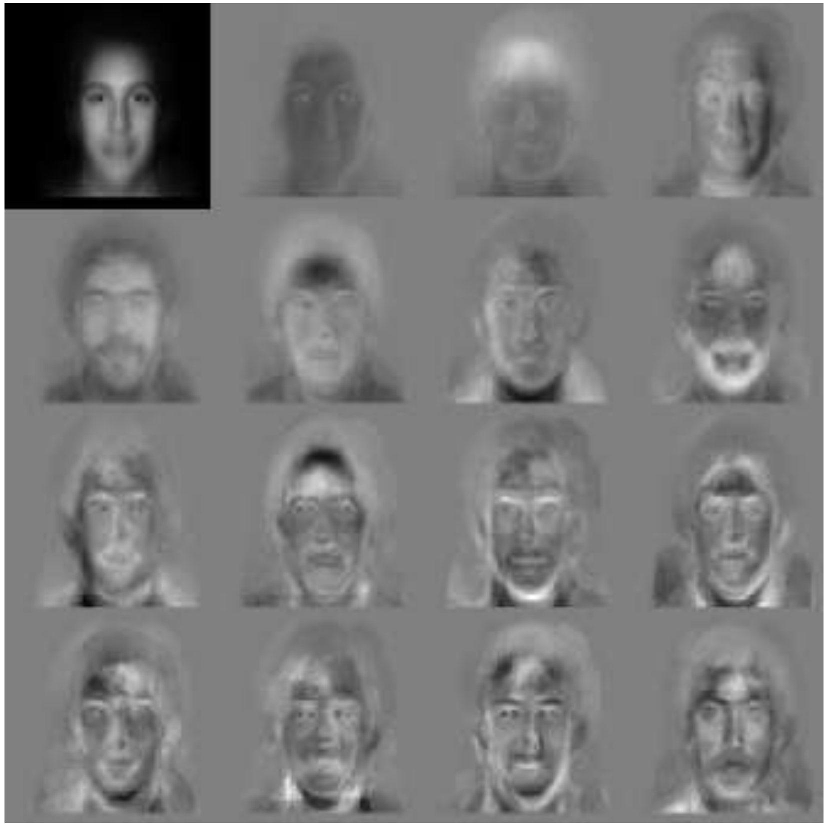
$$Y_i = (X_i - \bar{X})^T * (u_1, u_2, \ldots u_M)$$

Eigenfaces illustration

Example of eigenvectors faces.

The mean face and the eigenfaces of the covariance matrix of the faces (shown as pictures):

Mean face $\overline{X}$

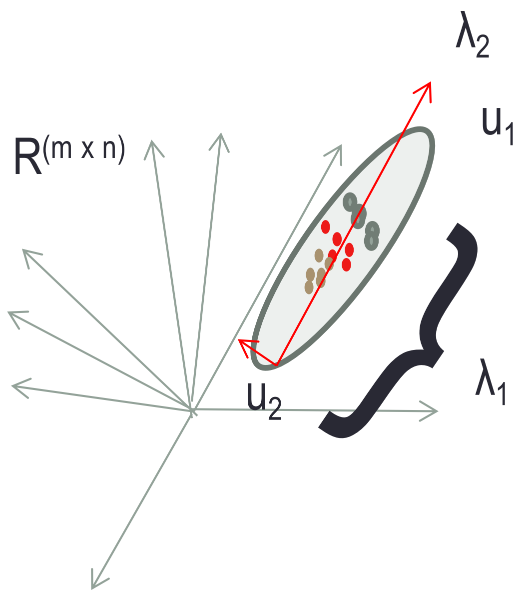


Eigenfaces (U)

Eigenfaces

What is the eigenvalue representing?

•The eigenvalues λ (eigenvalues) measure the variance of the data in the direction of the eigenvector =>



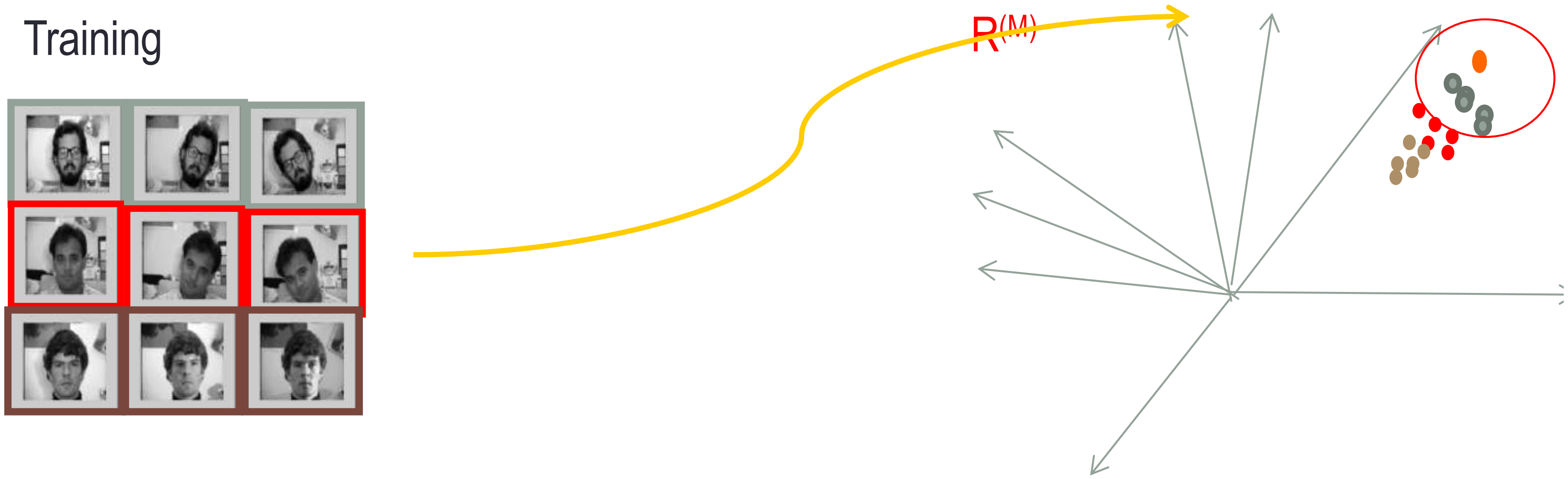
•The larger the λ_i, there is more variance in the data vector in the direction of the eigenvector u_i.

•if $\lambda_i = 0$ => we can avoid eigenvector u_i! Why?

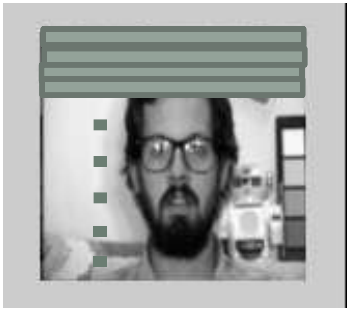
•=> We only are interested in the first k eigenvectors of the largest eigenvalues (k = 1,2, ... M-1).

The procedure for recognition with Eigenfaces

1. **Training:** Given the training set, we compute the eigenfaces U = (u_1, u_2, ... u_m), U - eigenvectors of $A*A^T$



2. **Recognition:** Given a new face, we center it and project it into the reduced space:



$(x_1,x_2,\ldots\ldots\ldots\ldots x_{(m \times n)})$->$Y=(\omega_1,\omega_2,\omega_3\ldots\omega_M)$

Project X:

$$Y = (X - \bar{X})^T * U$$

3. *Classify* – using knn in R^M

Index

1) Method of face recognition – eigenfaces
   1) Feature space of eigenfaces
   2) Algorithm for dimensionality reduction
   3) Eigenface trick

2) Examples and applications

Many slides adapted from K. Grauman, S. Lazebnik, J. Vitrià and D. Lowe

Eigenfaces

How to obtain the eigenface sub-space?

- **Objective:** To find the eigenvectors and eigenvalues of AA^T.

- **Problem**: The covariance matrix AA^T is size (mxn) x (mxn) -> find the eigenvalues is untreatable!
  - $A_{(65000x200)}$,
  - $AA^T_{(65000x65000)}$
  - $A^T A_{(200x200)}$

- **Tip**: Instead of $AA^T V= \Lambda V$ consider $(A^TA)V= \Lambda V$,
  - V is a matrix with columns - the eigenvectors of A^TA,
  - Λ is a diagonal matrix with elements - the eigenvalues,
  - A^TA is of size MxM (M << mxn)!

- But how to get the eigenvalues and eigenvectors of AA^T?

- **Trick**: Multiplying both sides with A:
  - $A(A^T AV)=A (\Lambda V) \Rightarrow AA^T (AV)= \Lambda (AV) \Rightarrow AA^T U= \Lambda U$ where $U=AV$

- => U is the matrix with columns the eigenvectors of the matrix AA^T, and U = AV.

Eigenfaces

How to obtain the eigenface sub-space?

Advantage: the matrix A^TA is of size M x M, where M is the number of images for training!

Note: If M << (m x n), it can be shown that there will be only M-1 values different from 0!

If the images are of size 256 x 256 = 65536 -> A: 65536xM, AA^T: 65536x65536, A^TA is of MxM.

The eigenvectors of the matrix A^TA V => eigenvectors of the covariance matrix Σ: $u_i=A\,v_i$.

A^TA and AA^T have the same eigenvalues Λ!

Eigenfaces

Algorithm to obtain the eigenfaces

1. Construct the matrix $A^T A$ of size M x M where the columns are centered faces:

$$A = [X_1 - \bar{X}, X_2 - \bar{X}, \ldots X_M - \bar{X}]$$

2. Compute the eigenvectors $V = [v_1, v_2, \ldots v_m]$ matrix of $A^T A$.

3. Sort by magnitude of their corresponding eigenvalues and keep the most important eigenvectors (with higher eigenvalues).

4. The M eigenfaces are obtained by multiplying the matrix A with v_i:

$$u_l = \sum_{k=1}^{M} v_{lk} A_k \,, \; l = 1, \ldots M$$

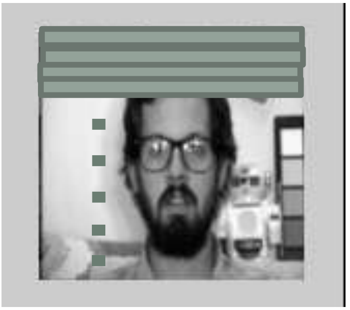
where A_k are the columns of the matrix A.

Notice the large reduction of the problem!

Eigenfaces: the Algorithm for Face Recognition

1. Given the training set, we compute the eigenfaces $U = (u_1, u_2, \ldots u_m)$, $U = AV$, V - eigenvectors of A^TA

2. Center and project the new face to recognize it:

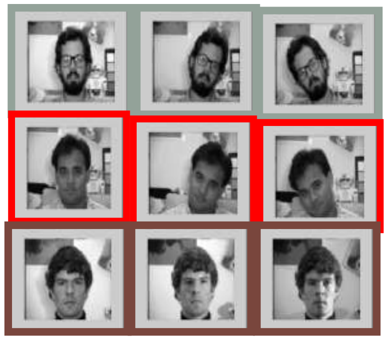


$(x_1, x_2, \ldots\ldots\ldots\ldots x_{(m \times n)}) \rightarrow Y = (\omega_1, \omega_2, \omega_3 \ldots \omega_M)$

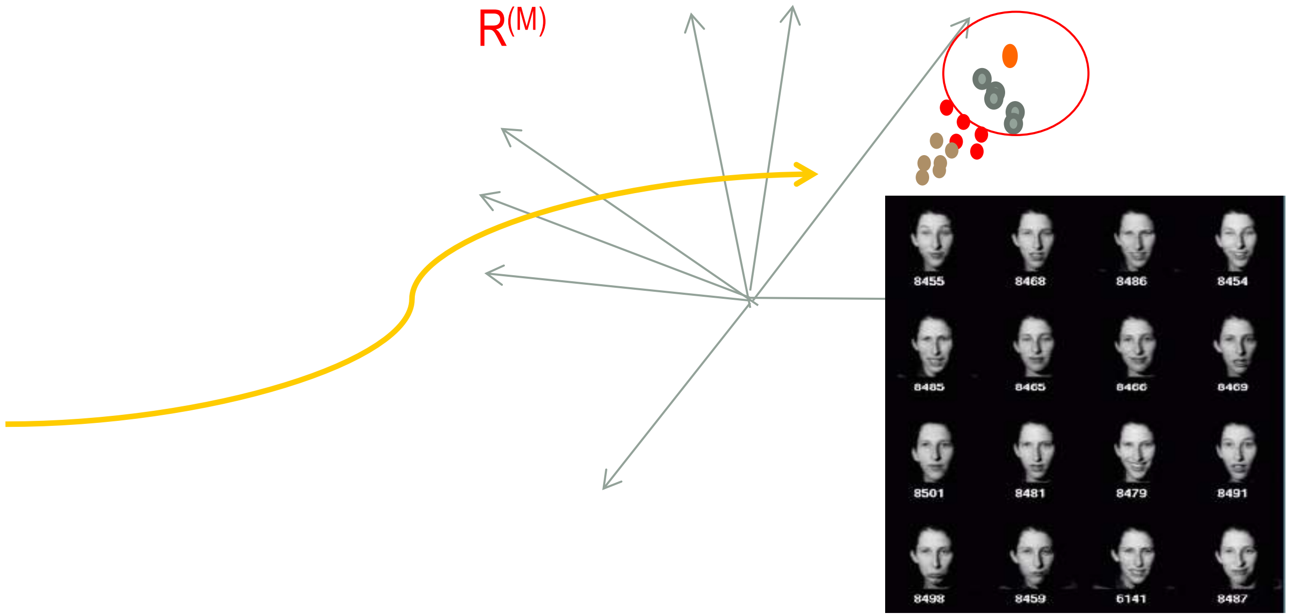
Project X:

$$Y = (X - \overline{X})^T * U$$

3. Classify– by knn in R^M



Training set



Index

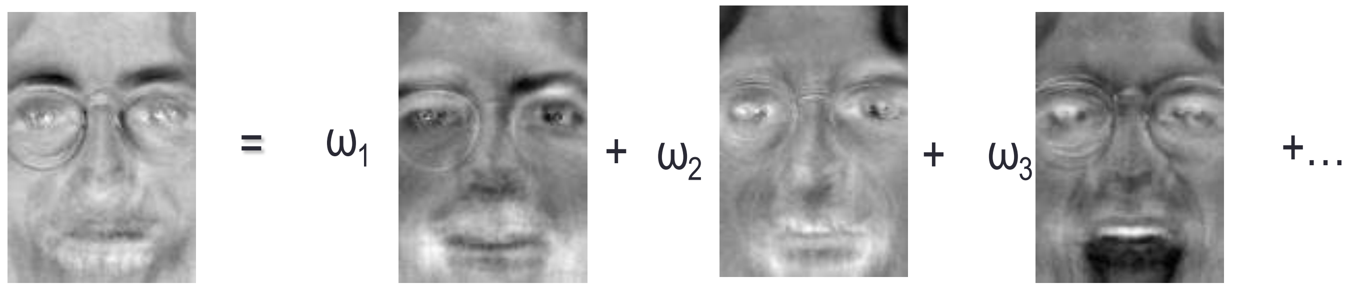
1) Method of face recognition – eigenfaces
   1) Feature space of eigenfaces
   2) Algorithm for dimensionality reduction
   3) Eigenface trick
   4) Illustration

2) Examples and applications

Many slides adapted from K. Grauman, S. Lazebnik, J. Vitrià and D. Lowe

Eigenfaces

Interpretation: Since eigenfaces are forming a base, we can express a face as a linear combination of eigenfaces.



Exercise: Given 2 images and 2 eigenfaces, express how each of the right two images is represented by the eigenfaces left images) as base (what would be the weights)?

Note: The method of eigenfaces recommends that the faces are aligned (usually, eyes-centered)!

Eigenfaces

What would be the representation of the original faces in the reduced space?

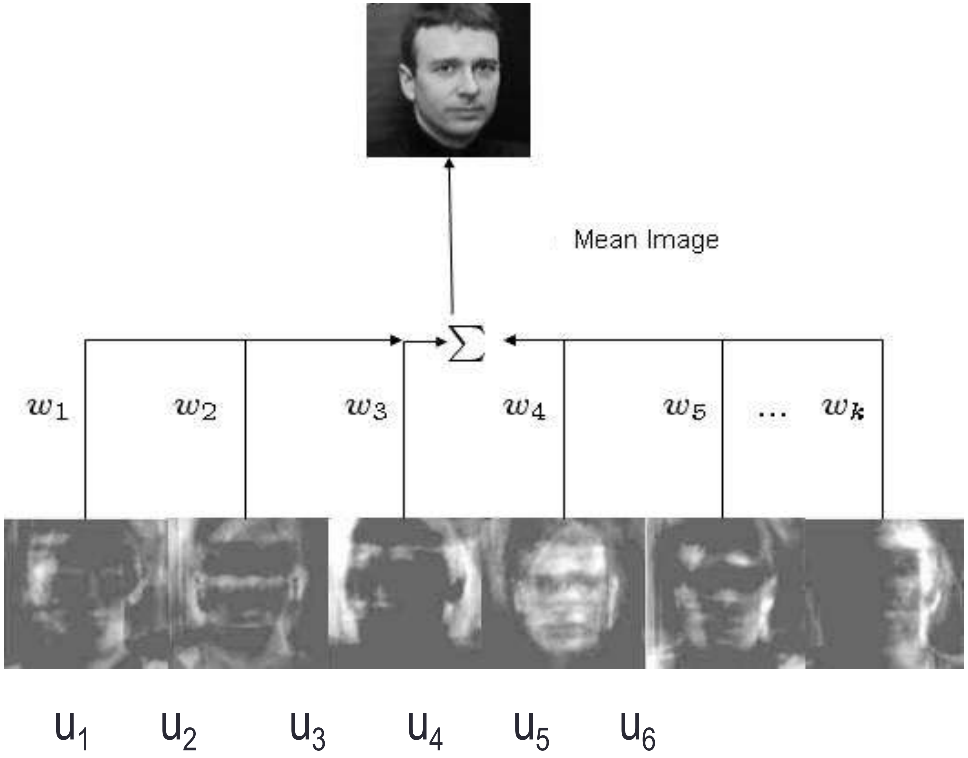
•The original image X is projected in the reduced space : X->Y:

$$\vec{Y}_i = (X_i - \overline{X})^T * (u_1, u_2, \ldots u_k)$$

where $Y_i = (\omega_1, \omega_2, \ldots \omega_k)$ is the vector representation in the reduced space (k = 1,2, ... M).

Hence:

$$X_i = \overline{X} + \sum_{i=1}^{k} \omega_i u_i$$



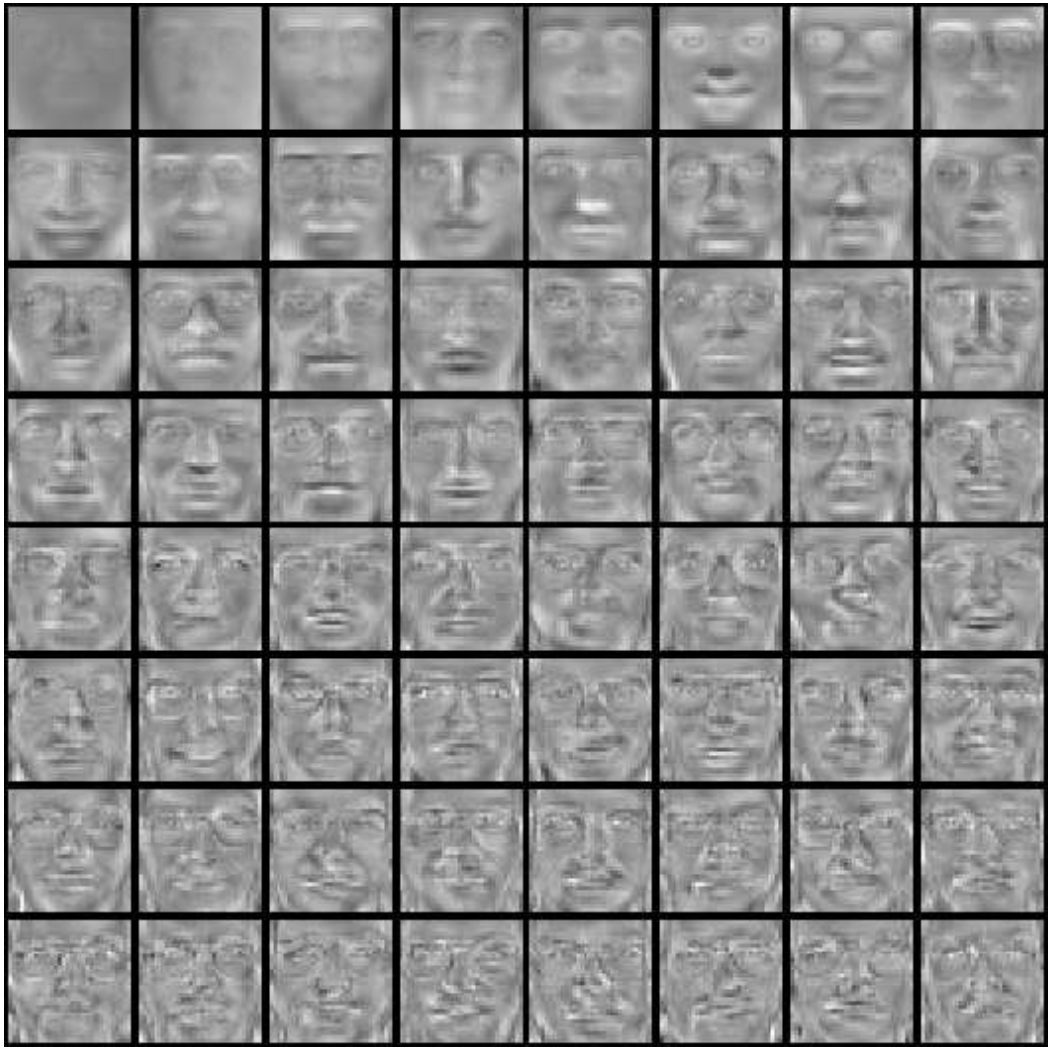
Eigenfaces example

- Training images
  - $\mathbf{x}_1,\ldots,\mathbf{x}_N$

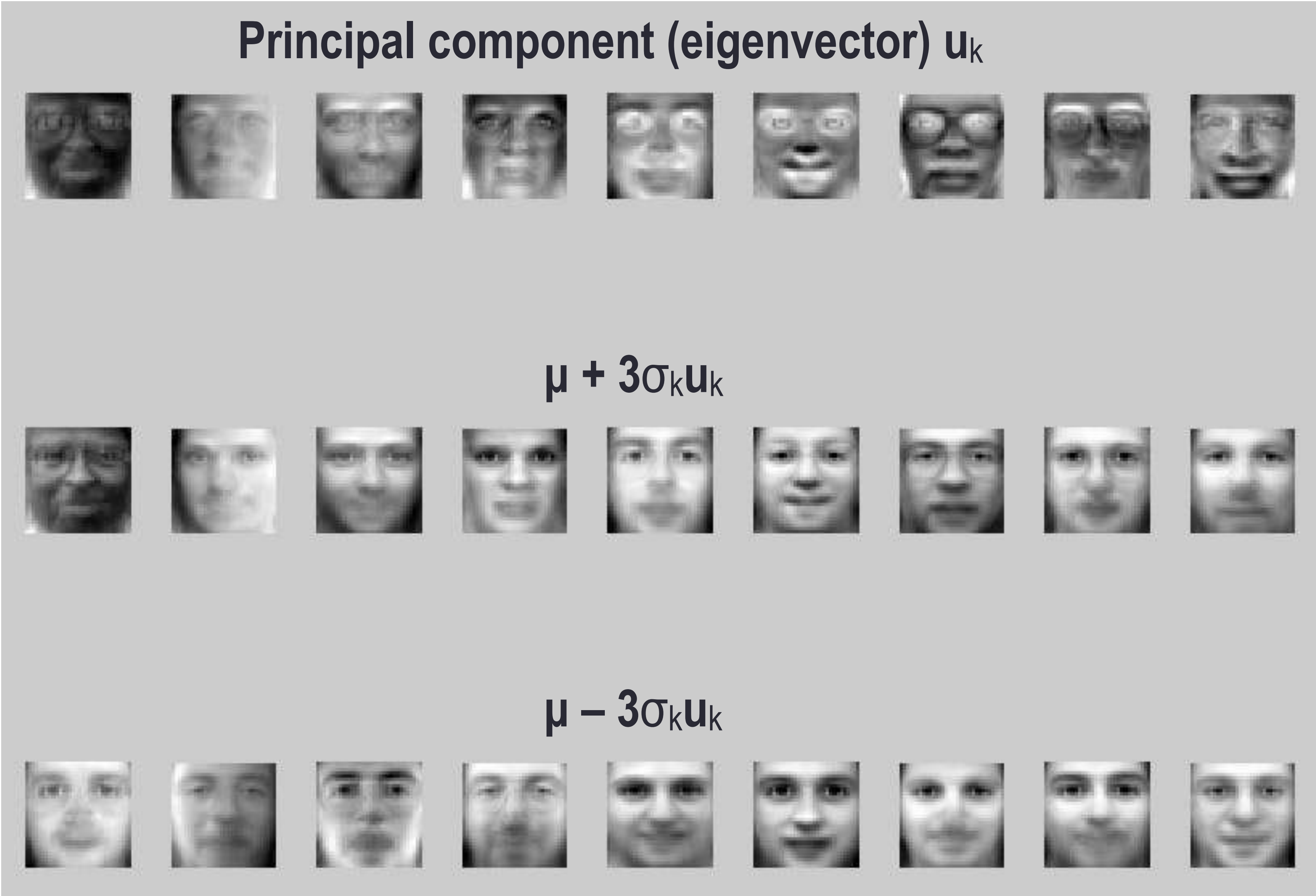
Eigenfaces example

Mean: $\boldsymbol{\mu}$

Top eigenvectors: $\mathbf{u}_1,\ldots\mathbf{u}_k$



Eigenfaces example



Eigenfaces example

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

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

$$= \quad w_1, \ldots, w_k$$

Eigenfaces example

- 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:

 =  + 

$$\hat{\mathbf{x}} = \boldsymbol{\mu} + w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + w_3\mathbf{u}_3 + w_4\mathbf{u}_4 + \ldots$$

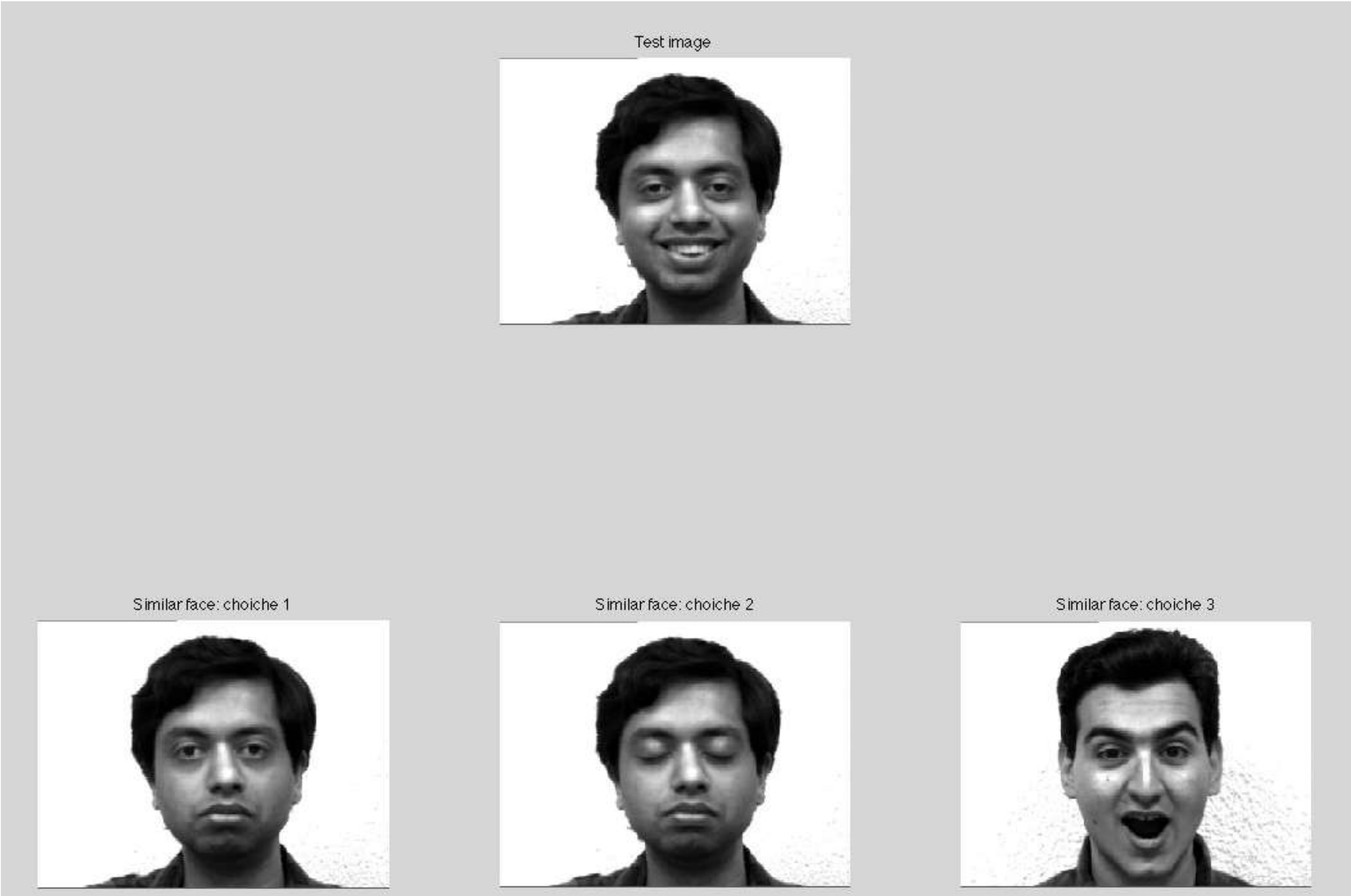
Index

1) Method of face recognition – eigenfaces
   1) Feature space of eigenfaces
   2) Algorithm for dimensionality reduction
   3) Eigenface trick
   4) Illustration

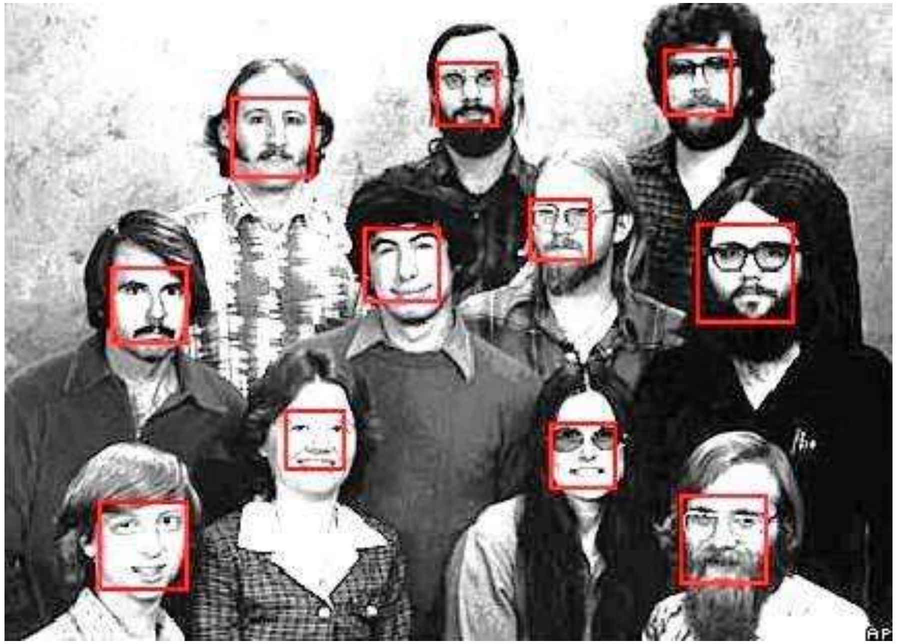
2) Examples and applications

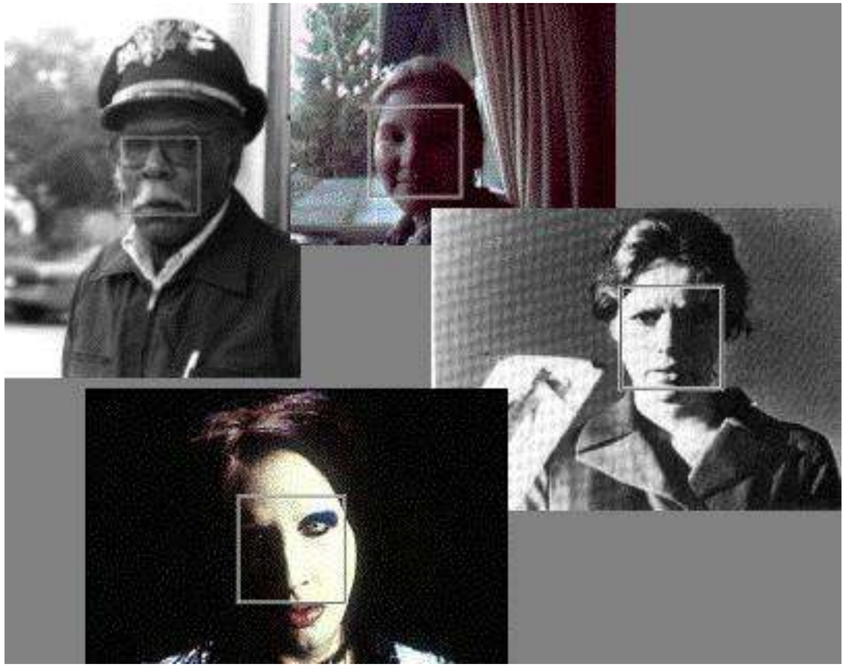
Many slides adapted from K. Grauman, S. Lazebnik, J. Vitrià and D. Lowe

Eigenfaces: Example



Face detection and recognition





Limitations

- Global appearance method: not robust to misalignment,

Eigenfaces

Exemple: Eigenfaces for face detection.

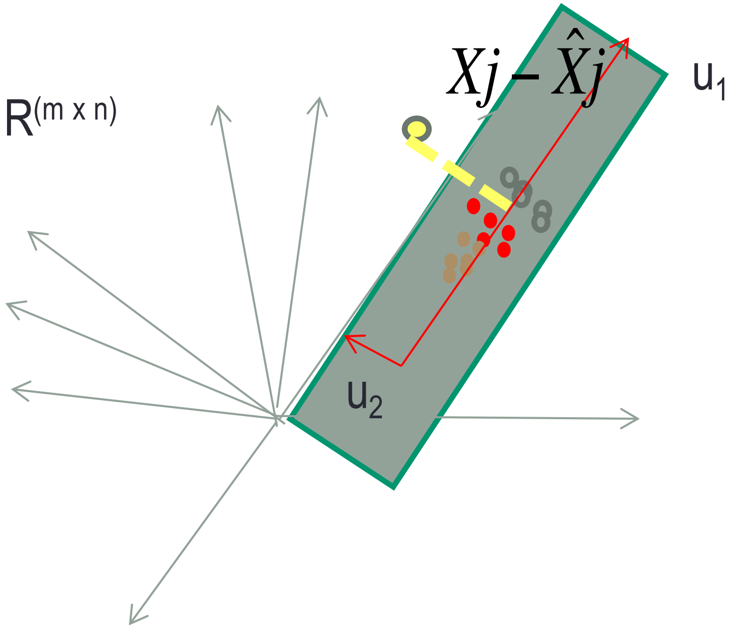
The space defined by the eigenfaces can be viewed as a subspace of the space of images.

We can always calculate the distance between any image and the face subspace!

And use it as a criterion for deciding whether an image is a face or not.

Eigenfaces

How do we detect a face?



If you want to know if it is a face or not,
a) Project it
b) Retroproject it in the original space
c) Obtain the difference.

$$Yj = (Xj - \bar{X})^T * (u_1, u_2, \ldots u_l)$$

$$\hat{X}j = \bar{X} + \sum_{i=1}^{k} \omega_i u_i$$

$$Xj - \hat{X}j$$

Eigenfaces

How do we detect a face?

Conclusions

- The problem of detection, localization, identification and recognition of objects / faces is a big challenge with very interesting results.
- Eigenfaces (& eigenspaces) is a robust technique to represent the different appearances of objects in a small and compact way.
- The technique eigenfaces is based on computing the eigenvalues and eigenvectors of the covariance matrix of the training data.
- It has the advantage of being fast and robust.
- It has the limitation that it works only when the faces are aligned and have the same scale, and pose angle.
- Numerous applications to face recognition, gender, facial expressions, monitoring, detection of anatomical organs, etc:
- http://www.youtube.com/watch?v=QZmPEI-YU8g