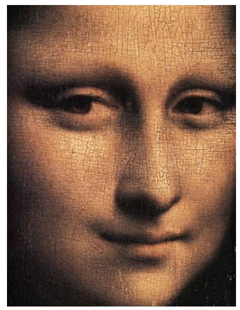
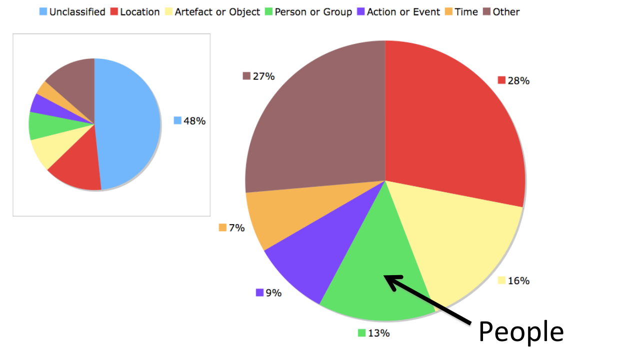
Face Detection and Recognition



Importance of Faces/People



Tag classification into broad WordNet categories

"Flickr Tag Recommendation based on Collective Knowledge", Borkur Sigurbjörnsson, Roelof van Zwol, www 2008

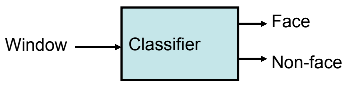
The First Selfie

from *The New Yorker*, March 16, 2015

Face Detection and Recognition

Face Detection Problem

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



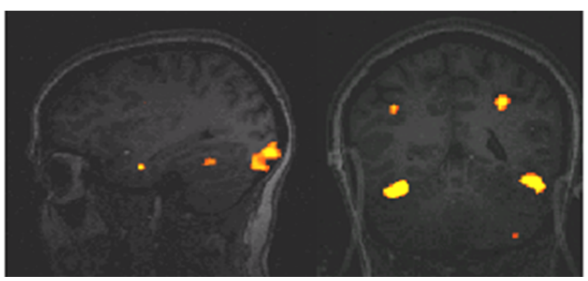
Face Detection now in most Digital Cameras

Canon Powershot

Also, smile and blink detection too in some cameras

Face Detection in Humans

There are cells that detect faces in the "Fusiform Face Area" of brain



Robust Real-Time Face Detection

Viola and Jones, 2003

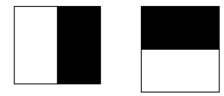
The Viola-Jones Algorithm

- Feature Extraction
  - At each position and scale, compute a feature vector based on a 24 x 24 window of pixels
- Classification
  - A type of decision tree classifier is used to decide if window contains a face or not

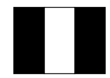
Features

- The system classifies images based on the value of simple features of 4 types:

Two-rectangle

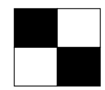


Three-rectangle



Value = ∑ (pixels in white area) − ∑ (pixels in black area)

Four-rectangle



Computing Features

- Given a window size of 24 x 24, the set of 4 types of rectangular features is scaled and shifted over image, resulting in a feature vector of length ~160,000
- Want to rapidly compute these features

Integral Image

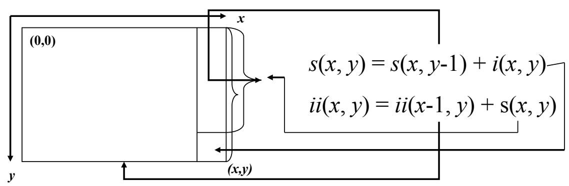
- Aka "Summed Area Table" (1984)
- Intermediate representation of the image
  - Sum of all pixels above and to left of (x, y) in image i:

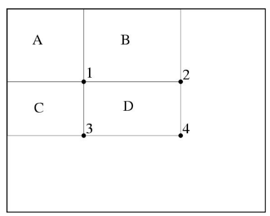
$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y')$$

- Computed in one pass over the image:

$$ii(x, y) = i(x, y) + ii(x\text{-}1, y) + ii(x, y\text{-}1) - ii(x\text{-}1, y\text{-}1)$$

Using the Integral Image





- Using the integral image representation one can compute the value of ***any*** rectangular sum in constant time
- For example, the integral sum in rectangle D is computed as:

$$ii(4) + ii(1) - ii(2) - ii(3)$$

Building the Classifier

Boosting (Schapire 1989)

- Randomly select $n_1 < n$ samples from training set D without replacement to obtain D_1
  - Train weak classifier C_1
- Select $n_2 < n$ samples from D with half of the samples misclassified by C_1 to obtain D_2
  - Train weak classifier C_2
- Select all samples from D that C_1 *and* C_2 disagree on
  - Train weak classifier C_3
- Final classifier is the majority vote of the 3 weak classifiers

Terminology

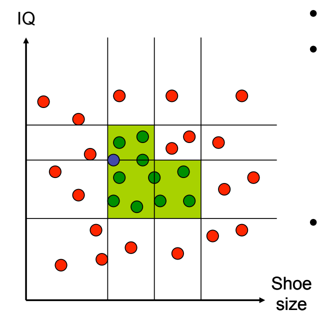
- **Weak Classifier**: < 50% error over any distribution
- **Strong Classifier**: thresholded linear combination of weak classifier outputs

AdaBoost - Adaptive Boosting

- Instead of sampling, ***re-weight***
- Learn a single simple classifier
- Classify the data
- Look at where it makes errors
- Re-weight the data so that the inputs where errors were made get higher weight in the learning process
- Now learn a 2nd simple classifier using the weighted data
- Combine the 1st and 2nd classifiers and weight the data according to where they make errors
- Learn a 3rd classifier based on the weighted data
- Continue until T simple classifiers are learned
- Final classifier is the weighted combination of all T classifiers

AdaBoost

AdaBoost combines many weak classifiers into one strong classifier



- Initialize sample weights
- For each cycle:
  - Find a classifier that performs well on the weighted samples
  - Increase weights of ***misclassified*** examples
- Return a weighted list of classifiers

AdaBoost Main Ideas

- **Individual features will be used as weak classifiers**
- Concatenate several detectors serially into a cascade
- Boost (using a version of AdaBoost) a number of features to get 'good enough' detectors

Weak Classifiers

- Defined by a feature, *f*, computed on a window in the image, a threshold, θ, and a parity, *p*:

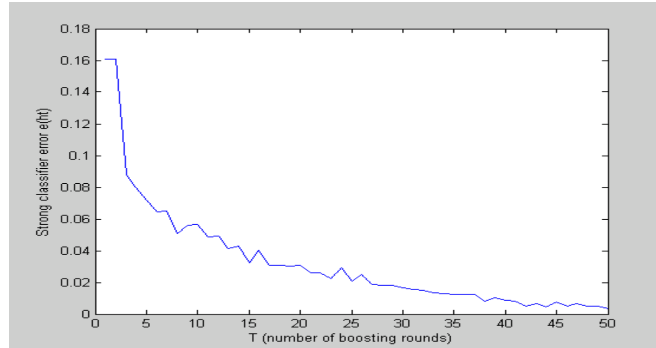
$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

- Thresholds are obtained by the mean value for the feature on both classes and then averaging the two values
- Parity defines the direction of the inequality

AdaBoost

- Given a training set of images
  - For t = 1, …,T (rounds of boosting)
    - A weak classifier is trained using a *single* feature
    - The error of the classifier is calculated
  - The classifier (i.e., single feature) with the lowest error is selected, and combined with the previous ones to make a strong classifier
- After a *T* rounds a *T*-strong classifier is created
  - It is the weighted linear combination of the selected weak classifiers

Classifier Error is Driven Down



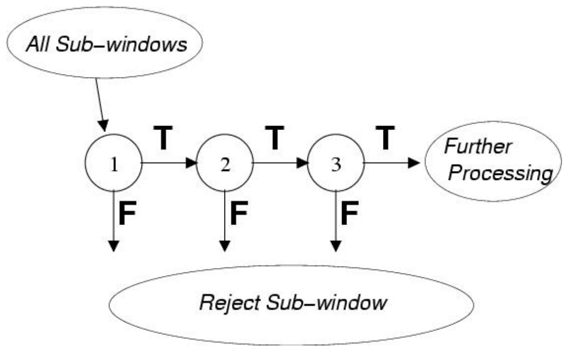
Main Ideas

- **Individual features will be used as weak classifiers**
- **Concatenate several detectors serially into a cascade**
- Boost (using a version of AdaBoost) a number of features to get 'good enough' detectors

Cascading

- Start with a simple classifier that rejects many of the negative windows while detecting almost all positive windows plus some negative windows
- Positive results from the first classifier triggers the evaluation of a second (more complex) classifier, and so on
  - **Each classifier is trained with the false positives from the previous classifier**
- A negative outcome at any point leads to the immediate rejection of the window (i.e., the window does ***not*** contain a face)

Cascading



Results

Detection in Real Images

- Basic classifier operates on 24 x 24 windows
- Scaling
  - Features evaluated at multiple scales
  - Scale by factors of 1.25
  - Scale the detector (rather than the images)
- Location
  - Move detector around the image (e.g., 1 pixel increments)
- Final Detection
  - A real face may result in multiple nearby detections
  - Postprocess detected windows to combine overlapping detections into a single detection

Training Data Set

4,916 hand-labeled face images (24 x 24)

9,500 non-face images

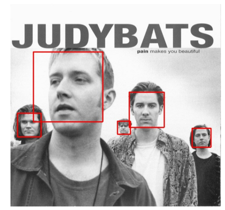
Structure of the Detector

- 38 layer cascade
- 6,060 features

Layer number	1	2	3 to 4	5 to 38
Number of feautures	2	10	50	-
Detection rate	100%	100%	-	-
Rejection rate	50%	80%	-	-

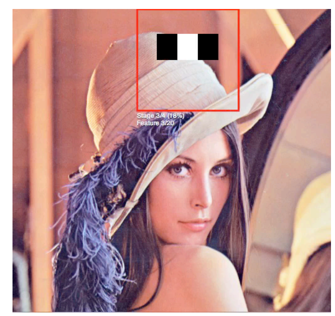
Results

- Notice detection at multiple scales



Results

Viola-Jones Algorithm Visualization



http://vimeo.com/12774628#

Real-Time Viola-Jones Demo

http://flashfacedetection.com/

http://flashfacedetection.com/camdemo2.html

Face Detection and Recognition

Face Recognition Problem

database

Query face

Face Verification Problem

- Face Authentication/Verification (1:1 matching)

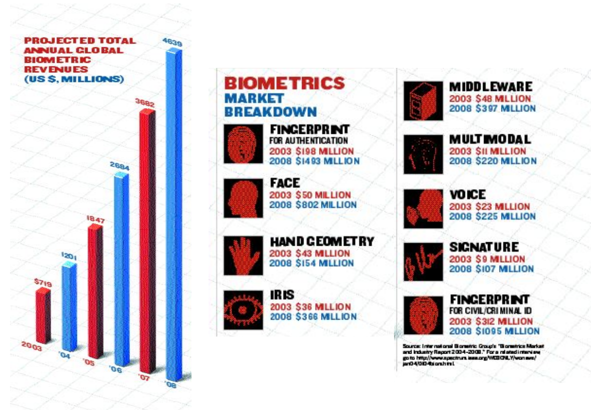
- Face Identification/Recognition (1:N matching)

Application: Access Control

www.viisage.com

www.visionics.com

Biometric Authentication



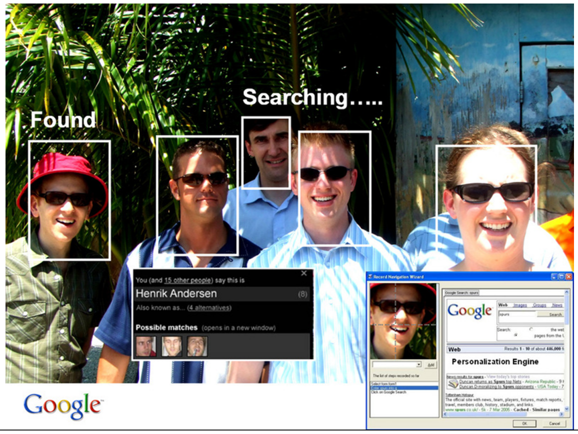
Application: Surveillance

- Video Surveillance (online or offline)

Face Scan at Airports

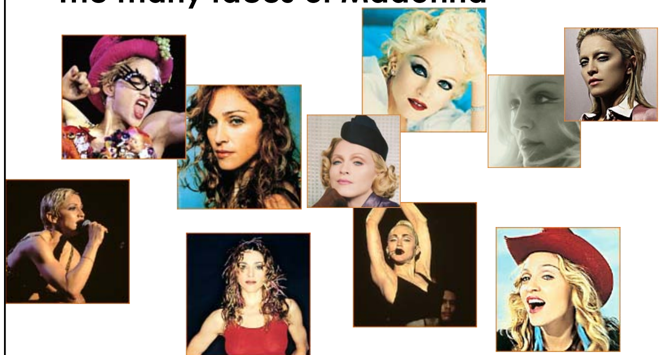
www.facesnap.de

Application: Autotagging Photos in Facebook, Flickr, Picasa, iPhoto, …



Why is Face Recognition Hard?

The many faces of Madonna

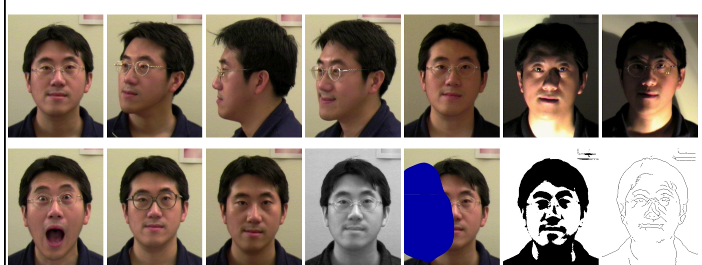


Recognition should be Invariant to

- Lighting variation
- Head pose variation
- Different expressions
- Beards, disguises
- Glasses, occlusion
- Aging, weight gain
- …

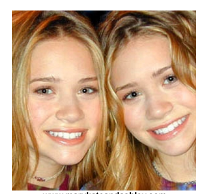
Intra-class Variability

- Faces with intra-subject variations in pose, illumination, expression, accessories, color, occlusions, and brightness



Inter-class Similarity

- Different people may have very similar appearance


www.marykateandashley.com

Twins


news.bbc.co.uk/hi/english/in_depth/americas/2000/us_elections

Father and son

Blurred Faces *are* Recognizable

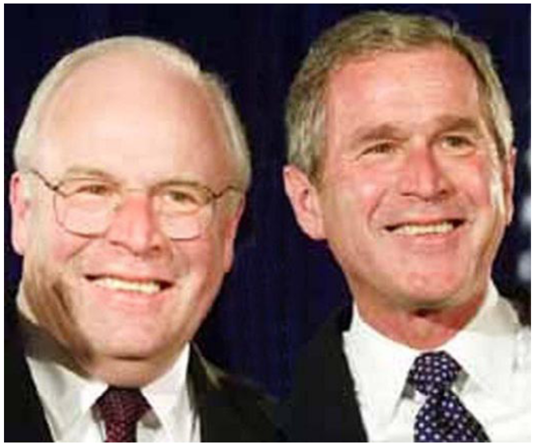
Blurred Faces *are* Recognizable

Michael Jordan, Woody Allen, Goldie Hawn, Bill Clinton, Tom Hanks, Saddam Hussein, Elvis Presley, Jay Leno, Dustin Hoffman, Prince Charles, Cher, and Richard Nixon. The average recognition rate at this resolution is one-half.

Illumination and Shading Affect Interpretation

Context is Important

P. Sinha and T. Poggio, I think I know that face, *Nature* **384**, 1996, 404.



P. Sinha and T. Poggio, Last but not least, *Perception* **31**, 2002, 133.

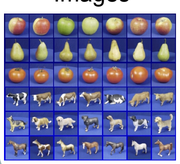
Face Recognition Architecture

Face Detection → (Image window) → Feature Extraction → (Feature Vector) → Classification → Face Identity

Components of a Recognition System

Training Phase

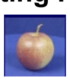
Training Images



→ Image Features → Training → Learned model

Training Labels → Training

Testing Phase



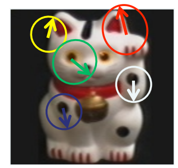
Test Image

→ Image Features → Classifier → "apple"

Learned model → Classifier

What Features? Let's Start with Interest Points

Query

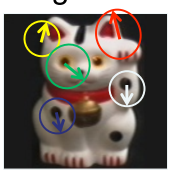


Database

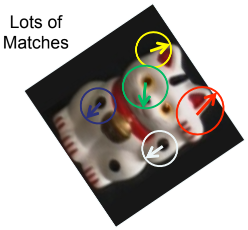
Compute interest points (keypoints) for every image in the database and the query

Slide by D. Hoiem

See how many keypoints are close to keypoints in each other image



Lots of Matches



Few or No Matches

But this will be really, really slow!

Slide by D. Hoiem

Key Idea: "Visual Words"

- Cluster the keypoint descriptors

Slide by D. Hoiem

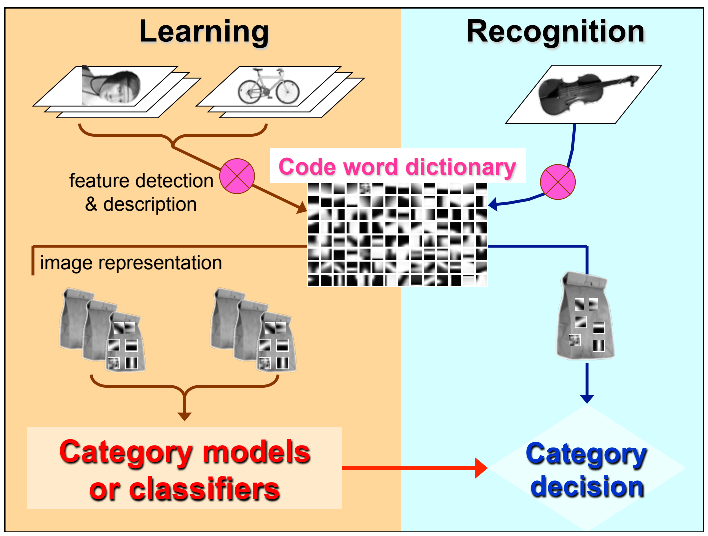
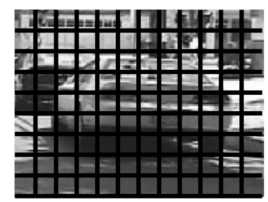
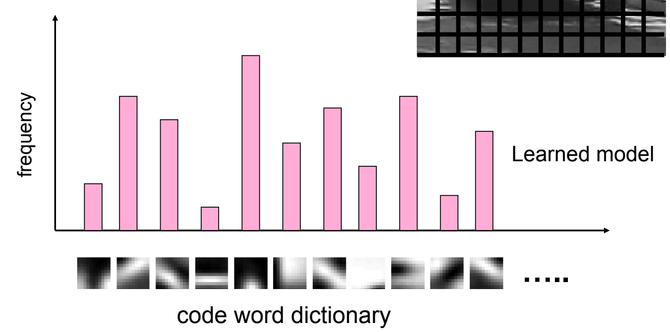
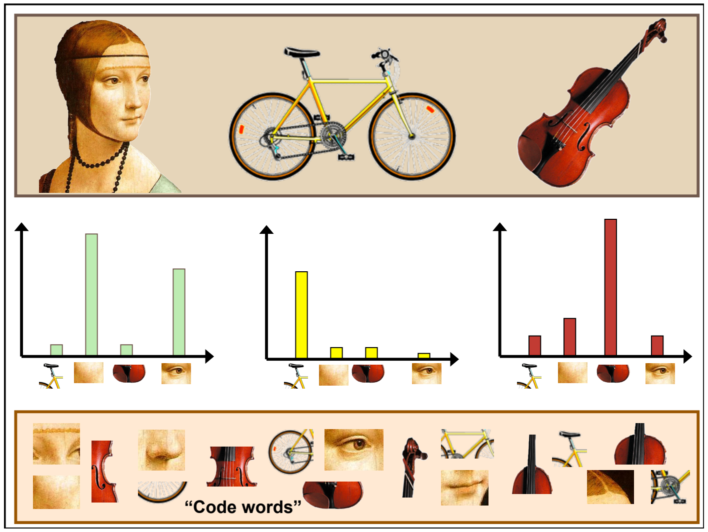


Image Representation







Key Idea: "Visual Words"

K-Means algorithm

1. Randomly select K centers

2. Assign each point to nearest center

3. Compute new center (mean) for each cluster

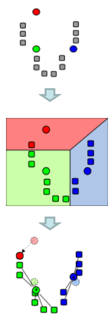


Illustration: http://en.wikipedia.org/wiki/K-means_clustering

Slide by D. Hoiem

Key Idea: "Visual Words"

K-Means algorithm

1. Randomly select K centers

2. Assign each point to nearest center

3. Compute new center (mean) for each cluster

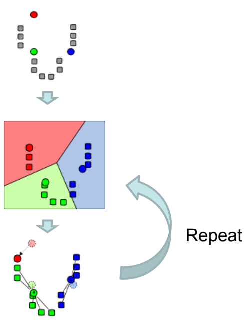
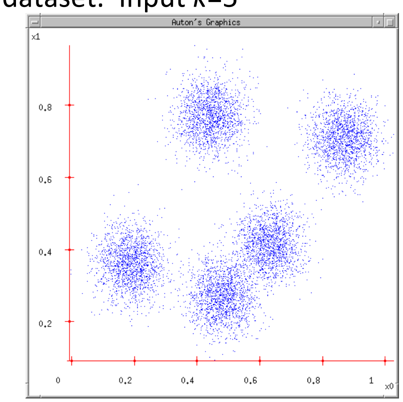


Illustration: http://en.wikipedia.org/wiki/K-means_clustering

Slide by D. Hoiem

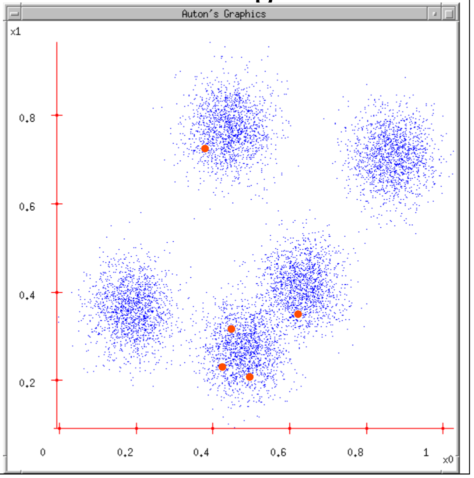
K-Means Clustering

- The dataset. Input *k*=5



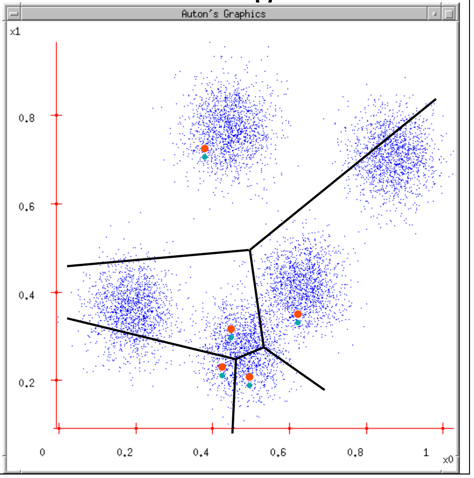
K-Means Clustering

- Randomly pick 5 positions as initial cluster centers (not necessarily data points)



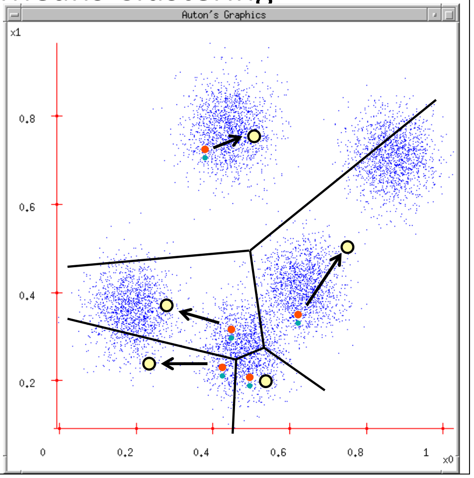
K-Means Clustering

- Each point finds which cluster center it is closest to; the point belongs to that cluster



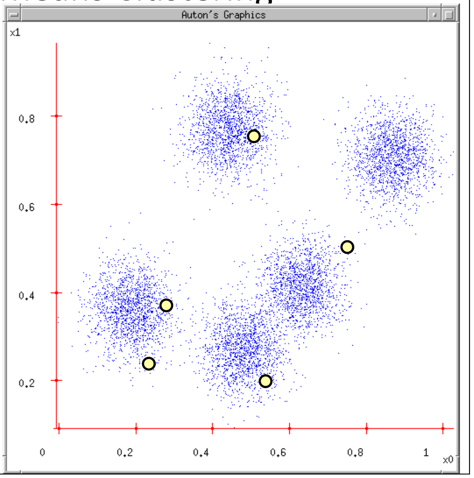
K-Means Clustering

- Each cluster computes its new centroid based on which points belong to it



K-Means Clustering

- Each cluster computes its new centroid, based on which points belong to it
- Repeat until convergence (i.e., no cluster center moves)



Key Idea: "Visual Words"

- Cluster the keypoint descriptors
- Assign each descriptor to a cluster number
  - What does this buy us?
  - Each descriptor was 128-dimensional, now is 1 integer (easy to match!)
  - Is there a catch?
    - Need **a lot** of clusters (e.g., 1 million) if we want points in the same cluster to be very similar

Slide by D. Hoiem

Key Idea: "Visual Words"

- Cluster the keypoint descriptors
- Assign each descriptor to a cluster number
- Represent an image region with a count of these "visual words"

Slide by D. Hoiem

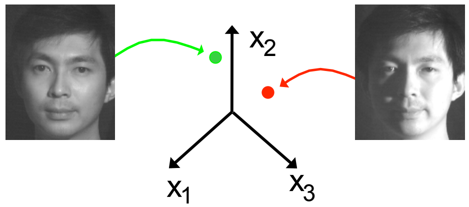
Key Idea: "Visual Words"

- Cluster the keypoint descriptors
- Assign each descriptor to a cluster number
- Represent an image region with a count of these "visual words"
- An image is a good match if it has a lot of the same the

Slide by D. Hoiem

Face Recognition using Eigenfaces

Image as a Feature Vector

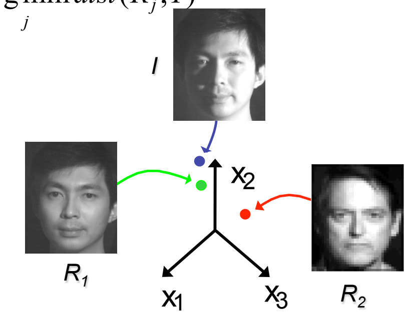


- Consider an n-pixel image to be a point in an n-dimensional space, $\boldsymbol{x} \in \mathbf{R}^n$
- Each pixel value is a coordinate of $\boldsymbol{x}$

Nearest Neighbor Classifier

$\{ R_j \}$ is the set of training images

$$ID = \arg\min_j dist(R_j, I)$$

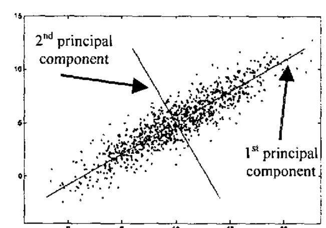


The Space of All Face Images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100 x 100 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***

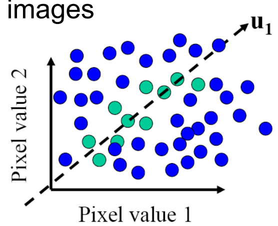
Eigenfaces (Turk and Pentland, 1991)

- Use Principle Component Analysis (PCA) to reduce the dimensionality



The 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



Principal Component Analysis (PCA)

- Pattern recognition in high-dimensional spaces
  - Problems arise when performing recognition in a high-dimensional space ("curse of dimensionality")
  - Significant improvements can be achieved by first mapping the data into a *lower-dimensional sub*-space

$$x = \begin{bmatrix} a_1 \\ a_2 \\ \dots \\ a_N \end{bmatrix} --> reduce\ dimensionality --> y = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_K \end{bmatrix} \quad (K << N)$$

  - The goal of PCA is to reduce the dimensionality of the data while retaining as much as possible of the variation present in the original dataset

Principal Component Analysis

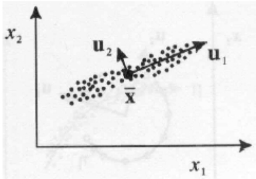
- Given: N data points (images) $\mathbf{x}_1, \dots, \mathbf{x}_N$ in R^d
- We want to find a new set of features, u, that are linear combinations of original ones:

$$u(\mathbf{x}_i) = \mathbf{u}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

  ($\boldsymbol{\mu}$: mean of data points)
- What vector $\mathbf{u}$ in R^d captures the *most variance* of the data?

Principal Component Analysis (PCA)

- Geometric interpretation
  - PCA projects the data along the directions where the data varies the most
  - These directions are determined by the eigenvectors of the covariance matrix corresponding to the largest eigenvalues
  - The magnitude of an eigenvalue corresponds to the variance of the data along the eigenvector direction



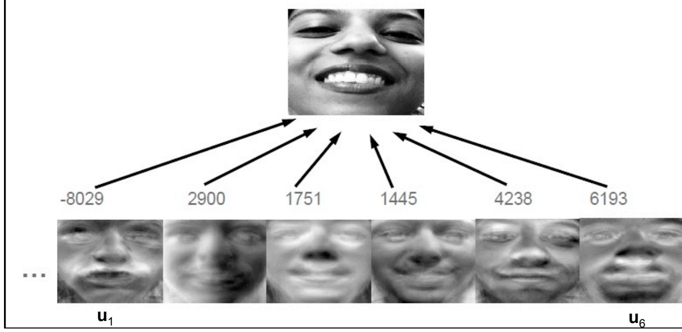
Eigenfaces: Key Idea

- Assume that most face images lie in a low-dimensional subspace determined by the first k ($k < d$) directions of maximum variance
- Use PCA to determine the eigenvectors or "eigenfaces," $\mathbf{u}_1, \ldots, \mathbf{u}_k$ with largest eigenvalues
- Represent all face images in the training set as linear combinations of these eigenfaces

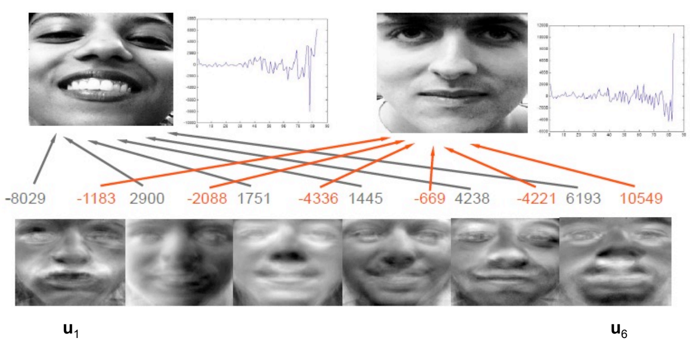
M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Eigenface Representation

Each face image is represented by a weighted combination of a small number of "component" or "basis" faces, $\mathbf{u}_i$



Eigenface Representation



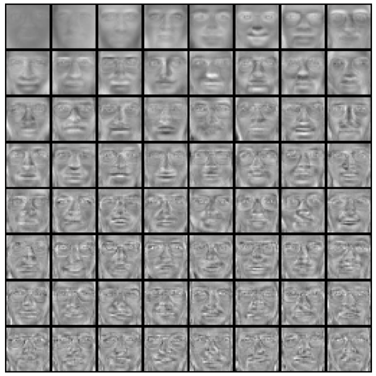
Eigenfaces Example

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

Eigenfaces Example

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

Mean: μ



Eigenfaces Example

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

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$
$$= \quad w_1, \ldots, w_k$$

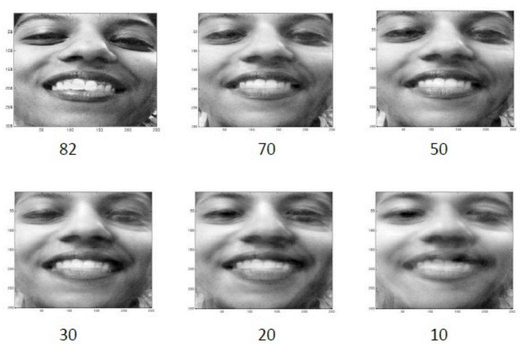
- Reconstruction:

 =  + 

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

Reconstruction

The more eigenfaces you use, the better the reconstruction, but even a small number gives good quality for matching



Recognition with Eigenfaces

- TRAINING: Process labeled training images:
  - Find mean **μ** and covariance matrix **Σ**
  - Find *k* principal components (eigenvectors of **Σ**): $\mathbf{u}_1, \ldots \mathbf{u}_k$
  - Project each training image $\mathbf{x}_i$ onto subspace spanned by principal components:
    $(w_{i1}, \ldots, w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x}_i - \boldsymbol{\mu}))$

- RECOGNITION: Given query image, **x**:
  - Project onto subspace:
    $(w_1, \ldots, w_k) = (\mathbf{u}_1^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu}))$
  - Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face
  - Classify as closest (nearest neighbor) training face in *k*-dimensional subspace

Experimental Results

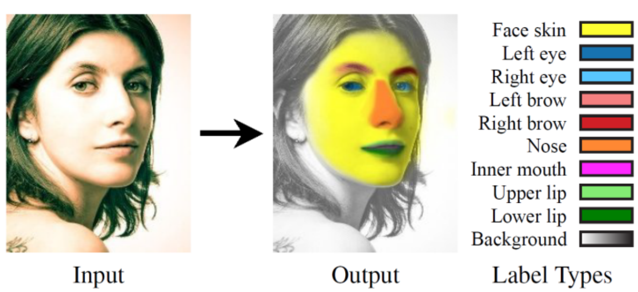
- Training set: 7,562 images of approximately 3,000 people
- k = 20 eigenfaces computed from a sample of 128 images
- Test set accuracy on 200 faces was 95%

Limitations

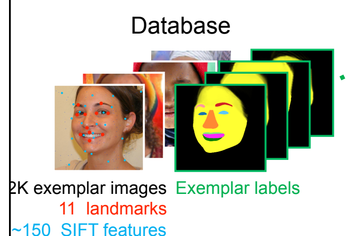
- The same person may appear differently due to
  - Beard, moustache
  - Glasses
  - Makeup
  - Facial expression
- These have to be represented separately

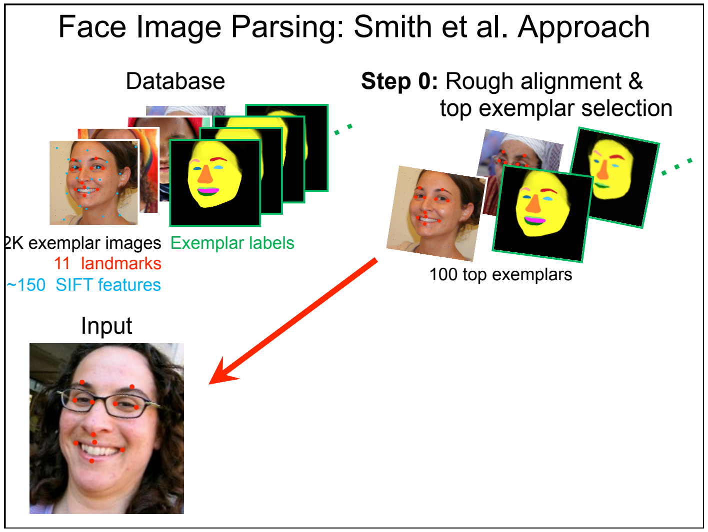
Face Image Parsing

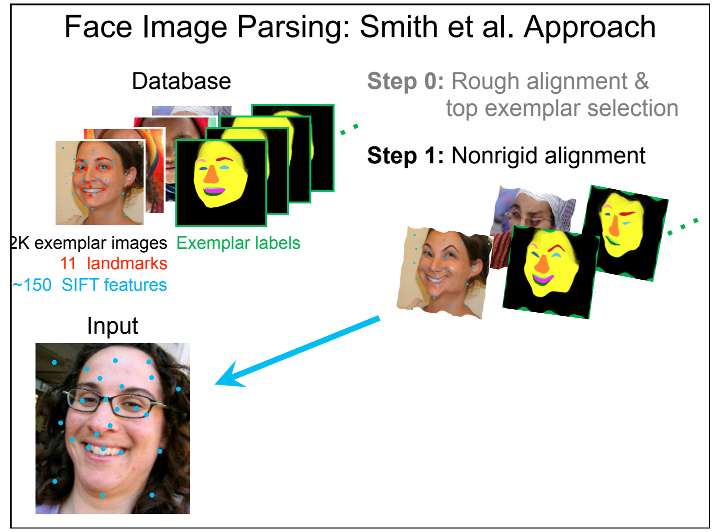
Goal: Given an input face image, automatically segment the face into its constituent parts

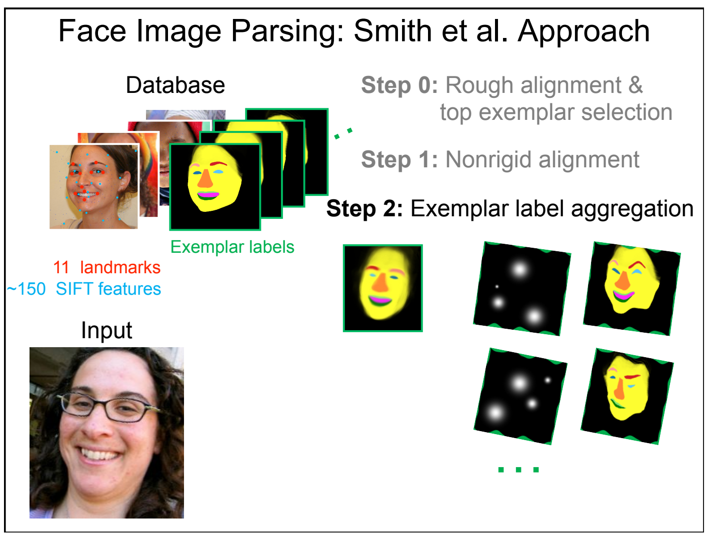


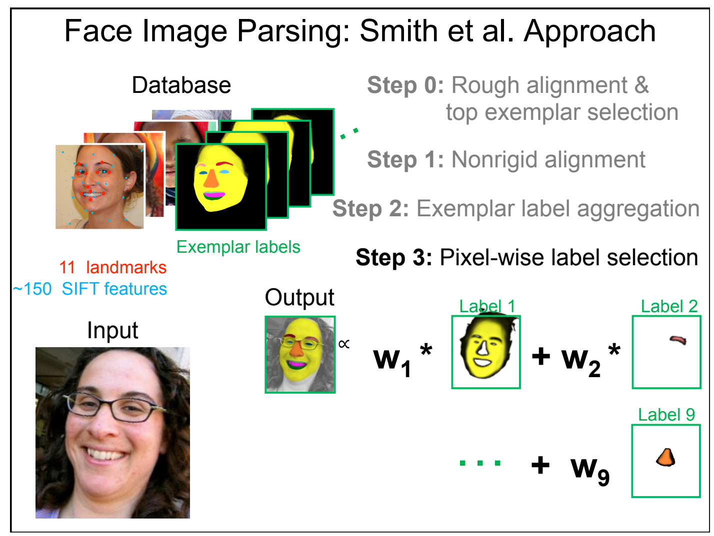
Face Image Parsing: Smith et al. Approach




Face Image Parsing: Smith et al. Approach
Database
Step 0: Rough alignment &
top exemplar selection
2K exemplar images
Exemplar labels
11 landmarks
~150 SIFT features
100 top exemplars
Input


Face Image Parsing: Smith et al. Approach
Database
Step 0: Rough alignment &
top exemplar selection
Step 1: Nonrigid alignment
2K exemplar images
Exemplar labels
11 landmarks
~150 SIFT features
Input

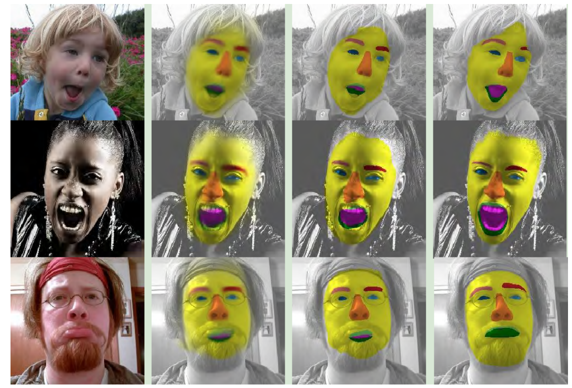

Face Image Parsing: Smith et al. Approach
Database
Step 0: Rough alignment &
top exemplar selection
Step 1: Nonrigid alignment
Step 2: Exemplar label aggregation
Exemplar labels
11 landmarks
~150 SIFT features
Input
. . .


Face Image Parsing: Smith et al. Approach
Database
Step 0: Rough alignment &
top exemplar selection
Step 1: Nonrigid alignment
Step 2: Exemplar label aggregation
Step 3: Pixel-wise label selection
Exemplar labels
11 landmarks
~150 SIFT features
Output
Input
Label 1
Label 2
Label 9
∝ w1 * + w2 *
. . . + w9

Face Image Parsing: Results

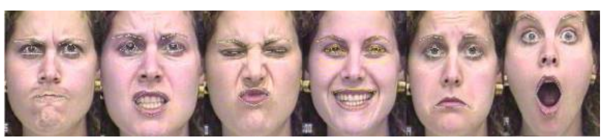
Input | *Soft* segments | *Hard* segments | Ground truth

Face Image Parsing: Results



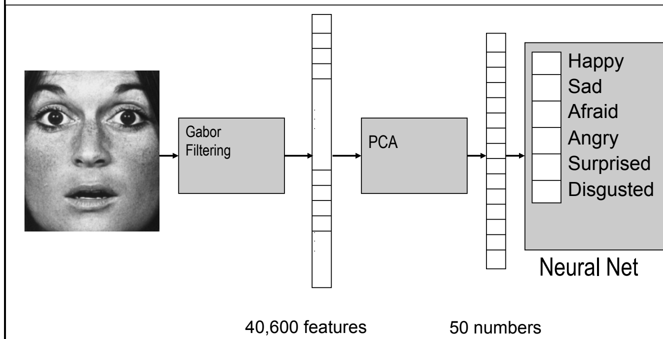
Input | *Soft* segments | *Hard* segments | Ground truth

Facial Expression Recognition



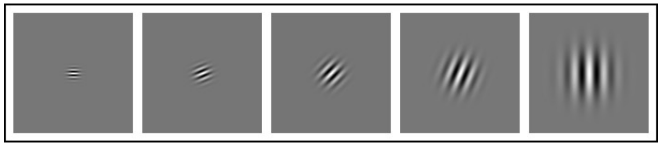
Anger | Fear | Disgust | Happy | Sad | Surprise

Facial Expression Recognition

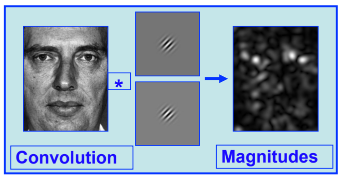


Features from a Gabor Filter Bank

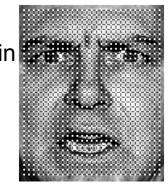
- Basic feature: the 2D Gabor wavelet filter (Daugman, 1985):



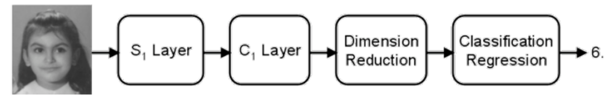
- These model the processing in early visual areas



Subsample in a 29 x 36 grid



Human Age Estimation



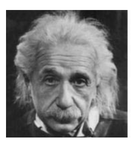
Feature extraction (Gabor filters)

PCA

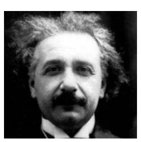
SVM/SVR

Results

32



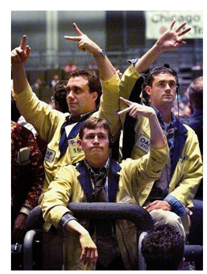
76

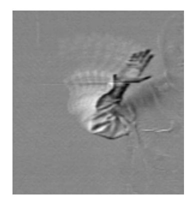


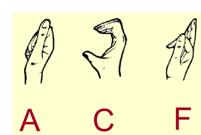
52

Human Gesture Recognition

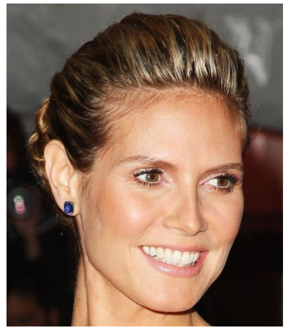
- Communicative human movement







Face Verification Problem: Are these Images of the Same Person?



Recognition using Visual Attributes

4-Legged	White	Male
Orange	Symmetric	Asian
Striped	Ionic columns	Beard
Furry	Classical	Smiling

Attributes can Define Categories

Female Eyeglasses Middle-aged Dark hair

Attributes can Define Categories

Caucasian Teeth showing Outside Tilted head

Prior Approaches

Images | Low-level features → Verification

RGB
HOG
LBP
SIFT
…

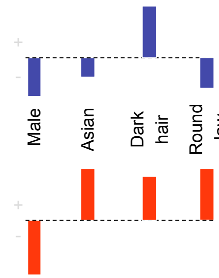
Different

RGB
HOG
LBP
SIFT
…

Attributes Approach

Images | Low-level features → Attributes → Verification

RGB
HOG
LBP
SIFT
…

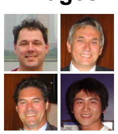


Different

RGB
HOG
LBP
SIFT
…

Method 1: Learning an Attribute Classifier

Training images → **Low-level features** → **Feature selection** → **Train Classifier (SVM)**



Males

RGB
HoG
HSV
…

RGB, Nose

HoG, Eyes

HSV, Hair

Edges, Mouth

…

↓

Gender classifier

↓

Male

0.87

(or binary)

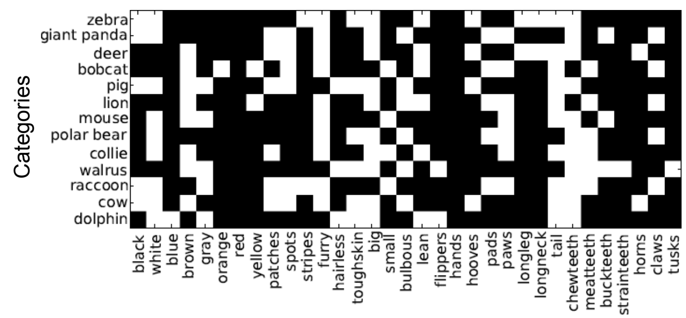
Females

RGB
HoG
HSV
…

Positive and negative examples

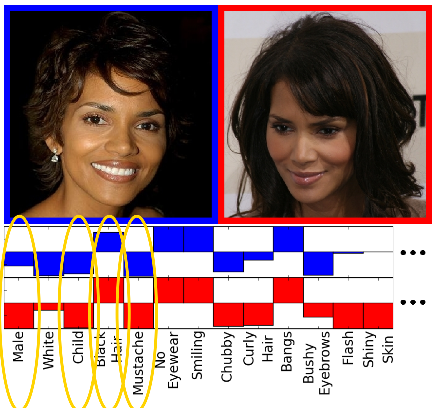
Animals with Attributes

Training Phase: Train binary attribute classifiers using all images from all categories that have that attribute



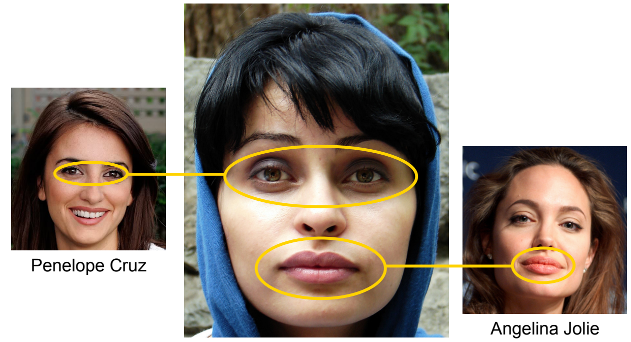
[Osherson et al., 91; Kemp et al., 06

Using Attributes to Perform Verification



Verification Classifier (SVM)

Method 2: Describe Faces using Similes

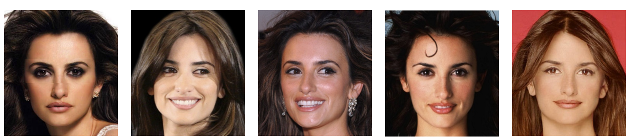


Penelope Cruz

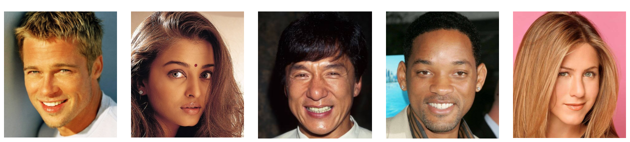
Angelina Jolie

Describe a person's appearance in terms of the similarity of different parts of their face to a limited set of "reference" people

Training Simile Classifiers

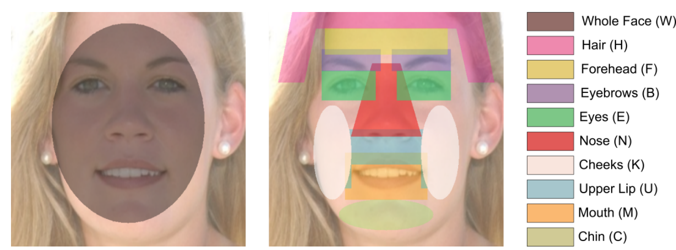


Images of Penelope Cruz's eyes



Images of other people's eyes

Face Regions



(After Alignment)

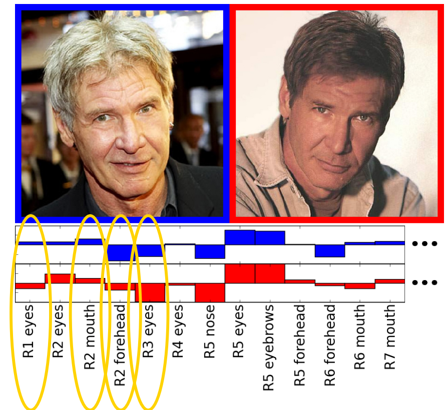
Reference People

Reference Person R1

Reference Person R2

60 reference faces used

Using Simile Classifiers for Verification



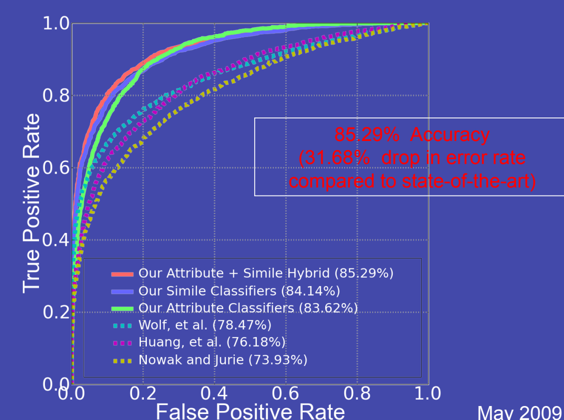
Verification Classifier (SVM)

Experimental Evaluation

Labeled Faces in the Wild Dataset

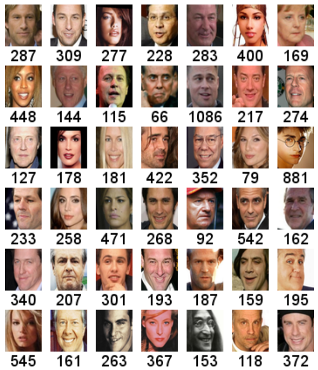
- 6,000 face pairs (3,000 same; 3,000 different)
- 10-fold cross-validation

Attributes-based Performance on LFW



May 2009

PubFig Dataset & Benchmark



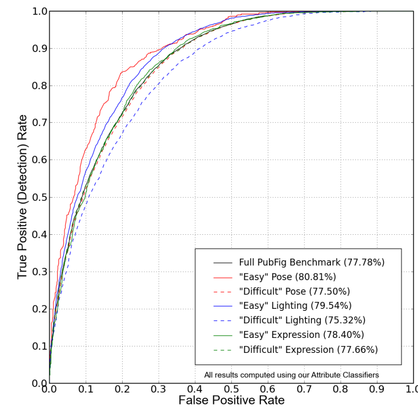
Public figures:
- Politicians
- Celebrities

Larger & deeper:
- 60,000 Images
- 200 People
- 300 Images per person

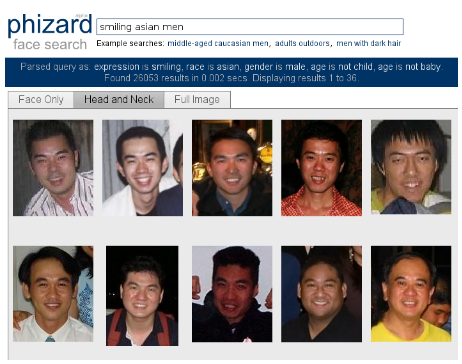
Subsets:
- Pose
- Illumination
- Expression

http://www.cs.columbia.edu/CAVE/databases/pubfig/

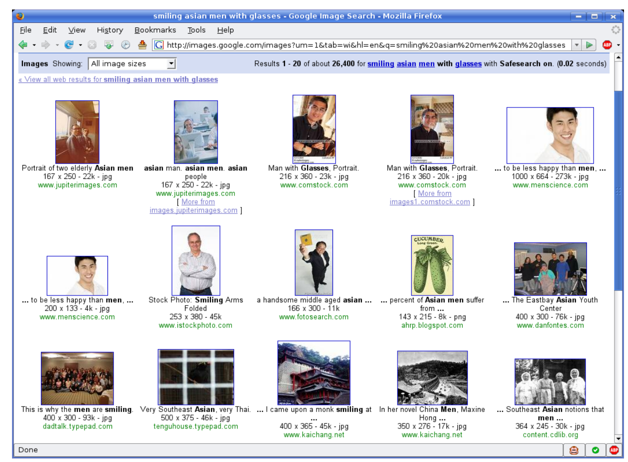
Results on PubFig



FaceTracer: A Face Search Engine



[N. Kumar et al., "FaceTracer: A Search Engine for Large Collections of Images with Faces", ECCV 2008]



Google: "smiling asian men with glasses" 7/08

Face Tracer:

A Search Engine for Large Collections of Images with Faces

Online Submission ID : 654