

High-Level Vision

Face Recognition II



CS332 Visual Processing
Department of Computer Science
Wellesley College

Face Detection: Viola & Jones

Multiple view-based classifiers based on simple features that best discriminate face vs. non-face

Most discriminating features *learned* from thousands of samples of face and non-face image windows

Attentional mechanism:
cascade of increasingly discriminating classifiers improves performance

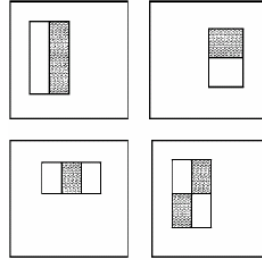


Viola & Jones

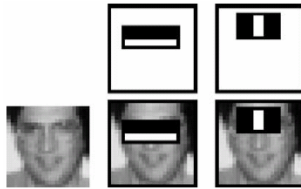
Use simple *rectangle features*:

ΣI in gray area(s) - ΣI in white area(s)
within 24 x 24 image sub-windows

- Initially consider 160,000 potential features per sub-window!
- features computed very efficiently



Which features best distinguish face vs. non-face?



Learn most discriminating features from thousands of samples of face and non-face image windows

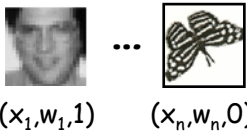
1-3

Learning the Best Features

weak classifier using one feature:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$$

x = image window
 f = feature
 p = +1 or -1
 θ = threshold



n training samples,
equal weights,
known classes

$$C(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

normalize weights, 0-1

find next best weak classifier

$$\epsilon_t = \min_{f,p,\theta} \sum_i w_i |h(x_i, f, p, \theta) - y_i|$$

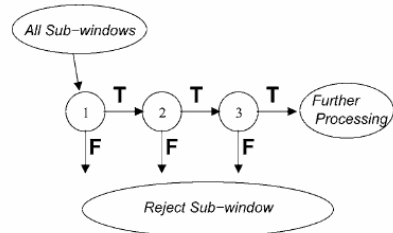
final classifier

use classification errors to adjust weights

~ 200 features yields good results for "monolithic" classifier

1-4

"Attentional Cascade" of Increasingly Discriminating Classifiers



First classifiers use a few highly discriminating features, low threshold

- 1st classifier uses 2 features, removes 50% non-face windows
- later classifiers distinguish harder examples

- Increases efficiency
 - Allows use of many more features
- Cascade of 38 classifiers, using ~6000 features

1-5

Viola & Jones Results



With additional diagonal features, classifiers were created to handle image rotations and profile views



1-6

Eigenfaces for recognition (Turk & Pentland)

$\Psi(x,y)$



Perform *principal components analysis* (PCA) on a large set of training images, to create a set of *eigenfaces* that span the data set

First components capture most of the variation across the database, later components capture more subtle variations

$\Psi(x,y)$: average face (common across face samples)

Each face image $F(x,y)$ can be expressed as a weighted combination of the eigenfaces $E_i(x,y)$:

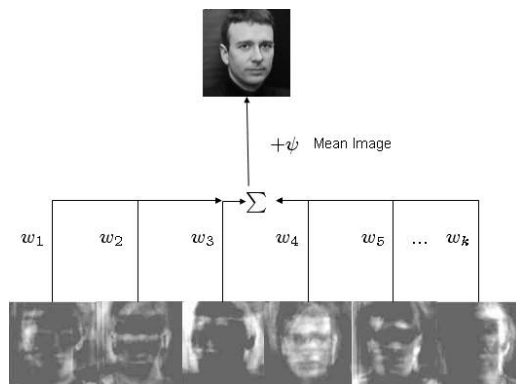
$$F(x,y) = \Psi(x,y) + \sum_i w_i * E_i(x,y)$$

1-7

Representation of Individual Faces

Each face image $F(x,y)$ can be expressed as a weighted combination of the eigenfaces $E_i(x,y)$:

$$F(x,y) = \Psi(x,y) + \sum_i w_i * E_i(x,y)$$



Recognition process:

- (1) Compute weights w_i for novel face image
- (2) Find image in face database with most similar weights

Sample Results of Eigenface Approach



Website (with MATLAB code):

<http://www.cs.princeton.edu/~cdecoro/eigenfaces/>

Christopher DeCoro, Princeton Univ., 2004

Wang & Tang: Recognizing Photo Sketches



Application: Automatic retrieval of photos of suspects from a police mug-shot database, given a sketch artist's rendering from an eye-witness description

Options:

- (1) Create photo from new sketch and match to photos in database
- (2) Create sketches from photos in database and match new sketch to database sketches

Note: Witness and artist could modify sketch interactively based on retrieved photos of potential suspects

Basic assumptions and approach

- Assumes photos have frontal pose, normal lighting, neutral expression, no occlusion (OK for mug-shots)
- Synthesizes *local face structures* at different scales using an iterative network model based on Markov Random Fields

- Sketch synthesis from photo:

(1) Training set of photo-sketch pairs used to select candidate sketch patches for each photo patch

(2) Local interactions in network select sketch patches that match photo and stitch together smoothly

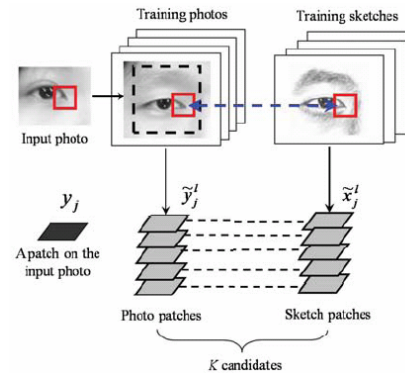
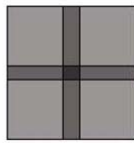


- Similar approach can be used to synthesize sketches from photos or synthesize photos from sketches (evaluate both options)

1-11

Synthesizing a Sketch From a Photo

Step 1: overlapping photo patches from aligned/transformed photo



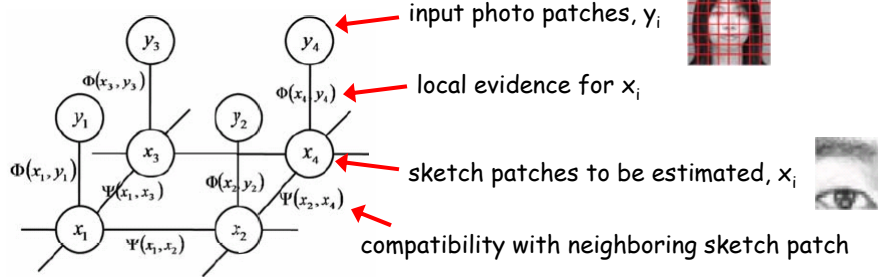
Step 2: for each photo patch, find K most similar photo patches in training set and collect corresponding sketches

1-12

Sketch Synthesis with MRF Network

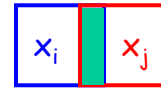
Step 3: build multiscale Markov network to estimate sketch patches for input photo

Step 4: stitch patches together



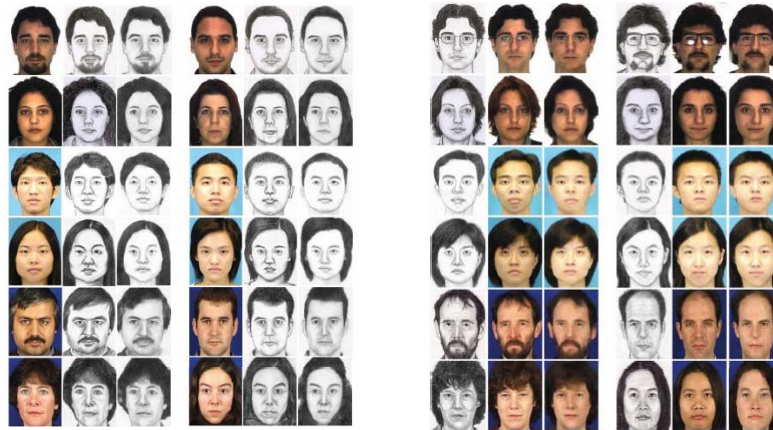
Maximize:

$$p(x_1, \dots, x_N, y_1, \dots, y_N) = \prod_{j_1 j_2} \Psi(x_{j_1}, x_{j_2}) \prod_j \Phi(x_j, y_j)$$



1-13

Wang & Tang Results



sketch synthesis

photo synthesis

1-14