EECS 332 Digital Image Analysis

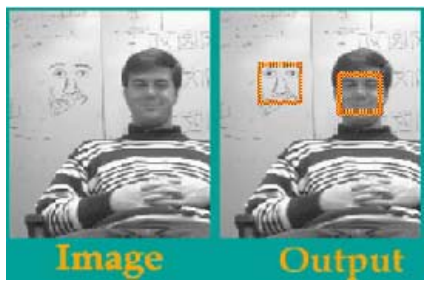
Face Detection

Ying Wu

Dept. Electrical Engineering and Computer Science.
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

Face Detection: The Problem



The Goal:

Identify and locate faces in an image

The Challenges:

- ✓ Position
- ✓ Scale
- ✓ Orientation
- ✓ Illumination
- ✓ Facial expression
- ✓ Partial occlusion

2

Outline

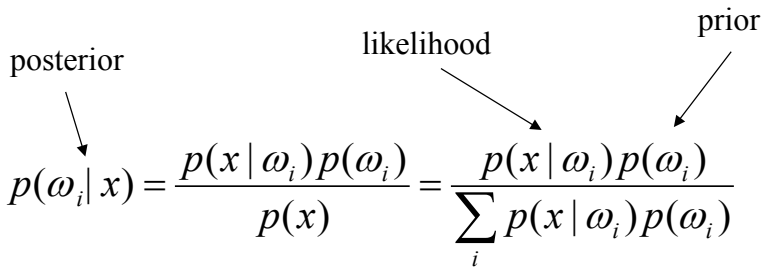
- The Basics
- Visual Detection
  - A framework
  - Pattern classification
  - Handling scales
- Viola & Jones' method
  - Feature: Integral image
  - Classifier: AdaBoosting
  - Speedup: Cascading classifiers
  - Putting things together
- Other methods
- Open Issues

3

The Basics: Detection Theory

- Bayesian decision
- Likelihood ratio detection

Bayesian Rule



5

Bayesian Decision

- Classes $\{\omega_1, \omega_2, \ldots, \omega_c\}$
- Actions $\{\alpha_1, \alpha_2, \ldots, \alpha_a\}$
- Loss: $\lambda(\alpha_k | \ \omega_i)$
- Risk: $R(\alpha_i | x) = \sum_{k=1}^{c} \lambda(\alpha_i | \omega_k) p(\omega_k | x)$
- Overall risk:
$$R = \int_x R(\alpha(x) | x) p(x) dx$$
- Bayesian decision
$$\alpha^* = \arg\min_k R(\alpha_k | x)$$

6

Minimum-Error-Rate Decision

$$\lambda(\alpha_i \mid \omega_k) = \begin{cases} 0 & i = k \\ 1 & i \neq k \end{cases}$$

$$R(\alpha_i \mid x) = \sum_{k=1}^{c} \lambda(\alpha_i \mid \omega_k) p(\omega_k) = \sum_{k \neq i} p(\omega_k \mid x) = 1 - p(\omega_i \mid x)$$

$$\text{decide} \quad \omega_i \quad \text{if} \quad p(\omega_i \mid x) > p(\omega_k \mid x) \quad \forall k \neq i$$

7

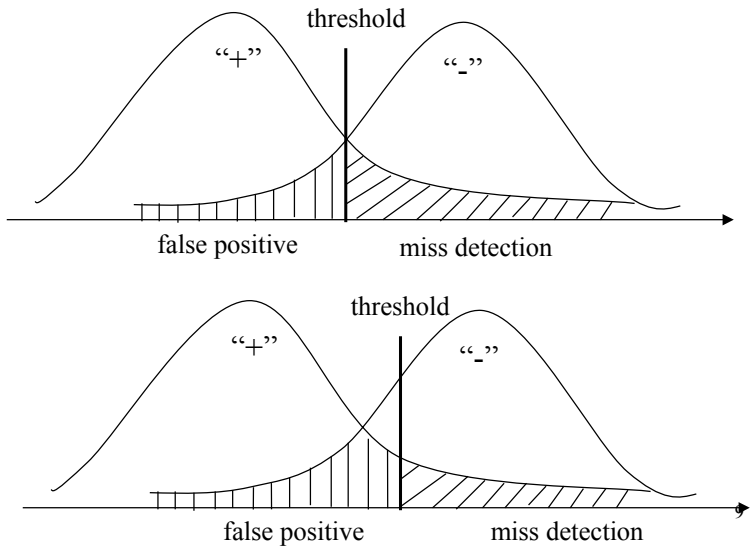
Likelihood Ratio Detection

- x – the data
- H – hypothesis
  - H0: the data does not contain the target
  - H1: the data contains the target
- Detection: p(x | H1) > p(x | H0)
- Likelihood ratio

$$\frac{p(x \mid H1)}{p(x \mid H0)} > t$$

8

Detection vs. False Positive



Visual Detection

- A Framework
- Three key issues
  - target representation
  - pattern classification
  - effective search

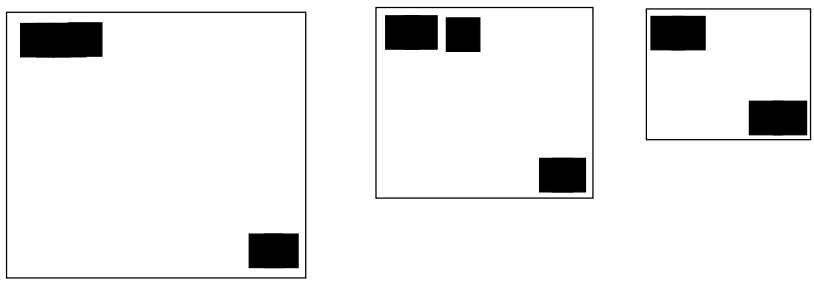
Visual Detection

- Detecting an "object" in an image
  - output: location and size
- Challenges
  - how to describe the "object"?
  - how likely is an image patch the image of the target?
  - how to handle rotation?
  - how to handle the scale?
  - how to handle illumination?

11

A Framework

- Detection window
- Scan all locations and scales



12

Three Key Issues

- Target Representation
- Pattern Classification
  - classifier
  - training
- Effective Search

13

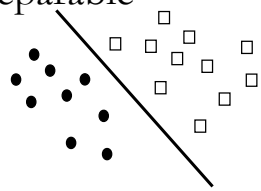
Target Representation

- Rule-based
  - e.g. "the nose is underneath two eyes", etc.
- Shape Template-based
  - deformable shape
- Image Appearance-based
  - vectorize the pixels of an image patch
- Visual Feature-based
  - descriptive features

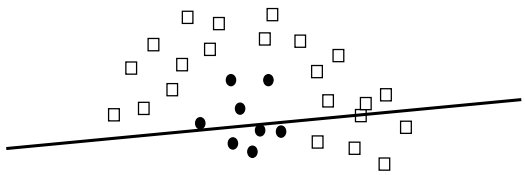
14

Pattern Classification

- Linear separable



- Linear non-separable



15

Effective Search

- Location
  - scan pixel by pixel
- Scale
  - solution I
    - ✓ keep the size of detection window the same
    - ✓ use multiple resolution images
  - solution II:
    - ✓ change the size of detection window
- Efficiency???

16

Viola & Jones' detector

- Feature → integral image
- Classifier → AdaBoosting
- Speedup → Cascading classifiers
- Putting things together

An Overview

- Feature-based face representation
- AdaBoosting as the classifier
- Cascading classifier to speedup

18

Harr-like features

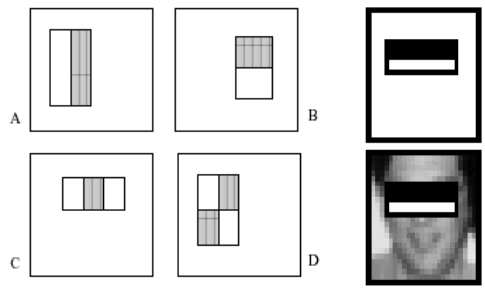


Figure 1: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

Q1: how many features can be calculated within a detection window?
Q2: how to calculate these features rapidly?

19

Integral Image

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

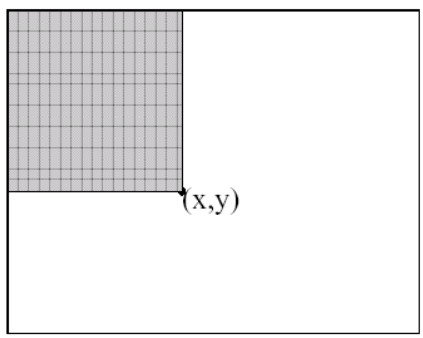


Figure 2: The value of the integral image at point (x, y) is the sum of all the pixels above and to the left.

20

The Smartness

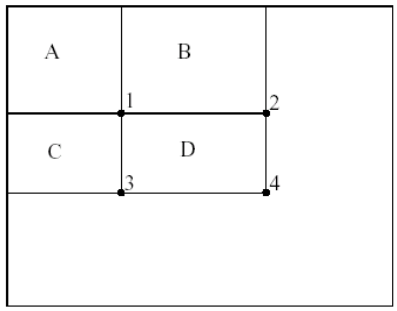


Figure 3: The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is $A+B$, at location 3 is $A+C$, and at location 4 is $A+B+C+D$. The sum within D can be computed as $4+1-(2+3)$.

$$s(x,y) = s(x,y-1)+i(x,y) \qquad ii(x,y) = ii(x-1,y)+s(x,y)$$

:1

Training and Classification

- Training
  - why?
  - An optimization problem
  - The most difficult part
- Classification
  - basic: two-class (0/1) classification
  - classifier
  - online computation

22

Weak Classifier

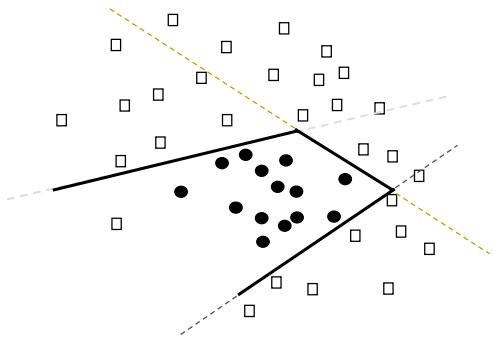
- Weak?
  - using only one feature for classification
  - classifier: ← thresholding

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

  - a weak classifier: (f_j, θ_j, p_j)
- Why not combining multiple weak classifiers?
- How???

23

Training: AdaBoosting



✓ Idea 1: combining weak classifiers

✓ Idea 2: feature selection

24

Feature Selection

- How many features do we have?
- What is the best strategy?

25

Training Algorithm

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

26

The Final Classifier

The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

- This is a linear combination of a selected set of weak classifiers

27

Learning Results

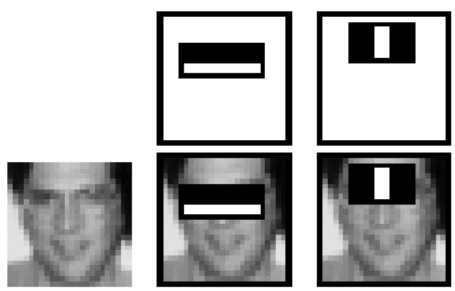


Figure 5: The first and second features selected by AdaBoost. The two features are shown in the top row and then overlayed on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

Attentional Cascade

- Motivation
  - most detection windows contain non-faces
  - thus, most computation is wasted
- Idea?
  - can we save some computation on non-faces?
  - can we reject the majority of the non-faces very quickly?
  - using simple classifiers for screening!

29

Cascading classifiers

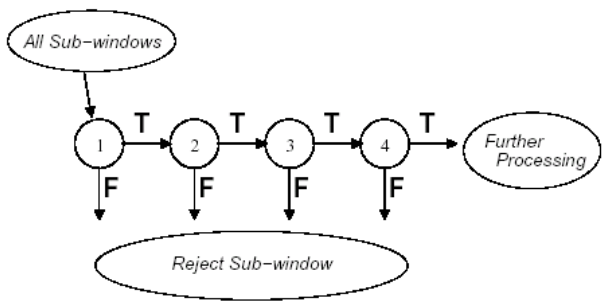


Figure 6: Schematic depiction of a the detection cascade. A series of classifiers are applied to every sub-window. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing the number of sub-windows have been reduced radically. Further processing can take any form such as additional stages of the cascade (as in our detection system) or an alternative detection system.

Designing Cascade

- Design parameters
  - # of cascade stages
  - # of features for each stage
  - parameters of each stage
- Example: a 32-stage classifier
  - S1: 2-feature, detect 100% faces and reject 60% non-faces
  - S2: 5-feature, detect 100% faces and reject 80% non-faces
  - S3-5: 20-feature
  - S6-7: 50-feature
  - S8-12: 100-feature
  - S13-32: 200-feature

31

Comparison

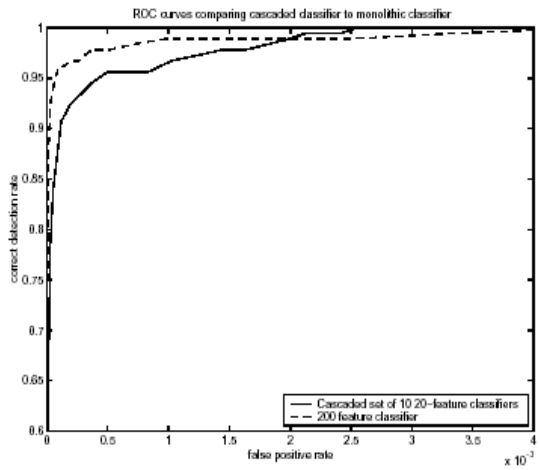


Figure 7: ROC curves comparing a 200-feature classifier with a cascaded classifier containing ten 20-feature classifiers. Accuracy is not significantly different, but the speed of the cascaded classifier is almost 10 times faster.

32

Comments

- It is quite difficult to train the cascading classifiers

33

Handling scales

- Scaling the detector itself, rather than using multiple resolution images
- Why?
  - const computation
- Practice
  - Use a set of scales a factor of 1.25 apart

34

Integrating multiple detection

- Why multiple detection?
  - detector is insensitive to small changes in translation and scale
- Post-processing
  - connect component labeling
  - the center of the component

35

Putting things together

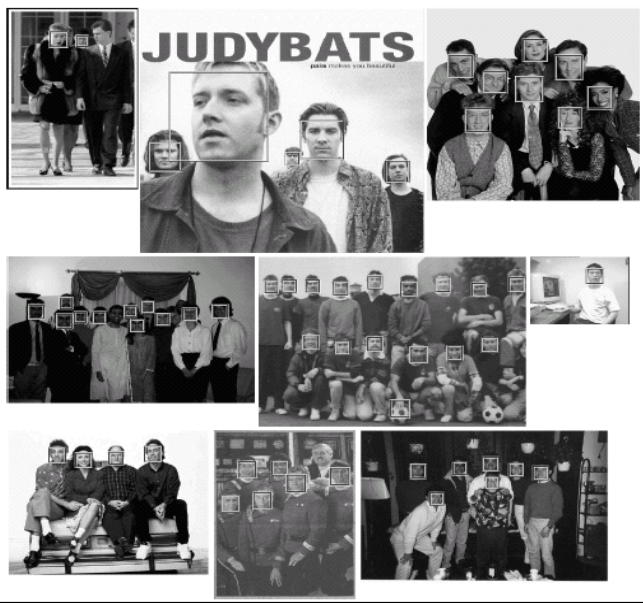
- Training: off-line
  - Data collection
    - ✓ positive data
    - ✓ negative data
  - Validation set
  - Cascade AdaBoosting
- Detection: on-line
  - Scanning the image

36

Training Data

37

Results



38

ROC

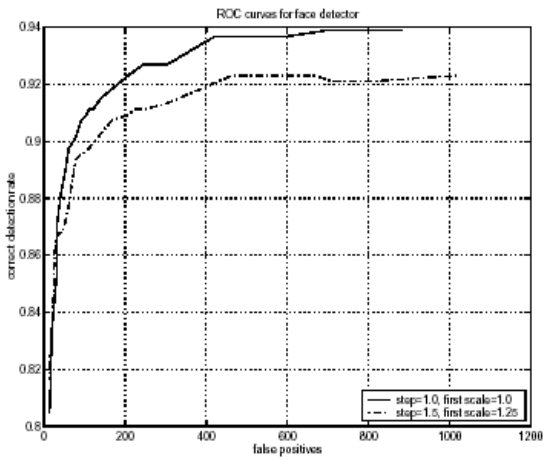


Figure 9: ROC curves for our face detector on the MIT+CMU test set. The detector was run once using a step size of 1.0 and starting scale of 1.0 (75,081,800 sub-windows scanned) and then again using a step size of 1.5 and starting scale of 1.25 (18,901,947 sub-windows scanned). In both cases a scale factor of 1.25 was used.

39

Summary

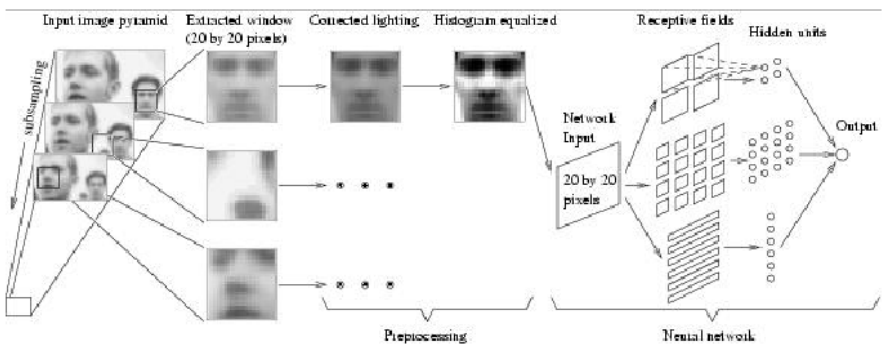
- Advantages
  - Simple → easy to implement
  - Rapid → real-time system
- Disadvantages
  - Training is quite time-consuming (may take days)
  - May need enormous engineering efforts for fine tuning

40

Other Methods

- Rowley-Baluja-Kanade

Rowley-Baluja-Kanade



Train a set of multilayer perceptrons and arbitrate a decision among all the inputs, and search among different scales, [Rowley, Baluja and Kanade, 1998]

42

RBK: Some Results

Courtesy of Rowley et al., 1998

Open Issues

- Out-of-plane rotation
- Occlusion
- Illumination