Face Detection and Alignment

Prof. Xin Yang
HUST
Face detection

Many slides adapted from P. Viola
Face detection

• Basic idea: slide a window across image and evaluate a face model at every location
Challenges of face detection

- Sliding window detector must evaluate tens of thousands of location/scale combinations
- Faces are rare: 0–10 per image
  - For computational efficiency, we should try to spend as little time as possible on the non-face windows
  - A megapixel image has \( \sim 10^6 \) pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image, our false positive rate has to be less than \( 10^{-6} \)
The Viola/Jones Face Detector

- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - Integral images for fast feature evaluation
  - Boosting for feature selection
  - Attentional cascade for fast rejection of non-face windows


"Rectangle filters"

Value = \[ \sum (\text{pixels in white area}) - \sum (\text{pixels in black area}) \]
Example

Source

Result
Fast computation with integral images

• The *integral image* computes a value at each pixel \((x,y)\) that is the sum of the pixel values above and to the left of \((x,y)\), inclusive.

• This can quickly be computed in one pass through the image.
Computing the integral image
Computing the integral image

- Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$
- Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$

MATLAB: $ii = 	ext{cumsum}(	ext{cumsum}(	ext{double}(i)), 2);$
Computing sum within a rectangle

- Let $A, B, C, D$ be the values of the integral image at the corners of a rectangle.

- Then the sum of original image values within the rectangle can be computed as:
  \[ \text{sum} = A - B - C + D \]

- Only 3 additions are required for any size of rectangle!
Example
Feature selection

• For a 24x24 detection region, the number of possible rectangle features is \( \sim 160,000! \)
Feature selection

• For a 24x24 detection region, the number of possible rectangle features is ~160,000!
• At test time, it is impractical to evaluate the entire feature set
• Can we create a good classifier using just a small subset of all possible features?
• How to select such a subset?
• Boosting is a classification scheme that works by combining weak learners into a more accurate ensemble classifier
  – A weak learner need only do better than chance

• Training consists of multiple boosting rounds
  – During each boosting round, we select a weak learner that does well on examples that were hard for the previous weak learners
  – “Hardness” is captured by weights attached to training examples

Training procedure

• Initially, weight each training example equally

• In each boosting round:
  – Find the weak learner that achieves the lowest weighted training error
  – Raise the weights of training examples misclassified by current weak learner

• Compute final classifier as linear combination of all weak learners (weight of each learner is directly proportional to its accuracy)

• Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme (e.g., AdaBoost)

Boosting illustration

Weak Classifier 1
Boosting illustration

Weights Increased
Boosting illustration

Weak Classifier 2
Boosting illustration

Weights Increased
Boosting illustration

Weak Classifier 3
Final classifier is a combination of weak classifiers
Boosting for face detection

- Define weak learners based on rectangle features

\[ h_t(x) = \begin{cases} 
1 & \text{if } f_t(x) > \theta_t \\
0 & \text{otherwise} 
\end{cases} \]
Boosting for face detection

• Define weak learners based on rectangle features

• For each round of boosting:
  – Evaluate each rectangle filter on each example
  – Select best threshold for each filter
  – Select best filter/threshold combination
  – Reweight examples

• Computational complexity of learning: $O(MNK)$
  – $M$ rounds, $N$ examples, $K$ features
Boosting for face detection

• First two features selected by boosting:

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

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

Not good enough!

Receiver operating characteristic (ROC) curve
Attentional cascade

• We start with simple classifiers which reject many of the negative sub-windows while detecting almost all positive sub-windows

• Positive response from the first classifier triggers the evaluation of a second (more complex) classifier, and so on

• A negative outcome at any point leads to the immediate rejection of the sub-window
Attentional cascade

- Chain classifiers that are progressively more complex and have lower false positive rates:

![Diagram of Classifier Cascade]

Receiver operating characteristic

% False Pos

% Detection

0 100

0 50

IMAGE

SUB-WINDOW

Classifier 1

Classifier 2

Classifier 3

FACE

T

F

T

F

T

F

T

NON-FACE

NON-FACE

NON-FACE
Attentional cascade

• The detection rate and the false positive rate of the cascade are found by multiplying the respective rates of the individual stages.

• A detection rate of 0.9 and a false positive rate on the order of $10^{-6}$ can be achieved by a 10-stage cascade if each stage has a detection rate of 0.99 ($0.99^{10} \approx 0.9$) and a false positive rate of about 0.30 ($0.3^{10} \approx 6 \times 10^{-6}$).
Training the cascade

• Set target detection and false positive rates for each stage

• Keep adding features to the current stage until its target rates have been met
  – Need to lower AdaBoost threshold to maximize detection (as opposed to minimizing total classification error)
  – Test on a validation set

• If the overall false positive rate is not low enough, then add another stage

• Use false positives from current stage as the negative training examples for the next stage
The implemented system

• Training Data
  – 5000 faces
    • All frontal, rescaled to 24x24 pixels
  – 300 million non-faces
    • 9500 non-face images
  – Faces are normalized
    • Scale, translation

• Many variations
  – Across individuals
  – Illumination
  – Pose
System performance

- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous detector of comparable accuracy (Rowley et al., 1998)
Summary: Viola/Jones detector

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows
Output of Face Detector on Test Images
Other detection tasks

Facial Feature Localization

Profile Detection

Male vs. female
Profile Detection
Profile Features
Example using Viola-Jones detector

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
Consumer application: iPhoto 2009

http://www.apple.com/ilife/iphoto/

Slide credit: Lana Lazebnik
Consumer application: iPhoto 2009

Things iPhoto thinks are faces

Slide credit: Lana Lazebnik
Consumer application: iPhoto 2009
Can be trained to recognize pets!

What other categories are amenable to *window-based representation*?
Pedestrian detection

- Detecting upright, walking humans also possible using sliding window’s appearance/texture; e.g.,

SVM with Haar wavelets [Papageorgiou & Poggio, IJCV 2000]

Space-time rectangle features [Viola, Jones & Snow, ICCV 2003]

SVM with HoGs [Dalal & Triggs, CVPR 2005]
Window-based detection: strengths

• Sliding window detection and global appearance descriptors:
  - Simple detection protocol to implement
  - Good feature choices critical
  - Past successes for certain classes
Window-based detection: Limitations

- High computational complexity
  - For example: 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations!
  - If training binary detectors independently, means cost increases linearly with number of classes
- With so many windows, false positive rate better be low
Limitations (continued)

- Not all objects are “box” shaped
Limitations (continued)

• Non-rigid, deformable objects not captured well with representations assuming a fixed 2d structure; or must assume fixed viewpoint

• Objects with less-regular textures not captured well with holistic appearance-based descriptions
Limitations (continued)

- If considering windows in isolation, context is lost

Figure credit: Derek Hoiem
Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions