Object detection

Image source
Challenges of object detection

- Detector must evaluate tens of thousands of location/scale combinations
- Positive instances are rare: 0–10 per image
  - A megapixel image has ~$10^6$ pixels and a comparable number of candidate object locations
  - For computational efficiency, we should try to spend as little time as possible on the negative windows
  - To avoid having a false positive in every image, our false positive rate has to be less than $10^{-6}$
Let’s start with face detection
Let’s start with face detection

Source: Boris Babenko
Let’s start with face detection

Things iPhoto thinks are faces
Sliding window framework
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


Image Features

“Rectangle filters”

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

Source

Result

[Image of a noisy source image with a black box and a checkmark on the right side, indicating a successful 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: \( ii = \text{cumsum}(\text{cumsum}(	ext{double}(i)), 2); \)
Computing sum within a rectangle

- Let A, B, C, D be the values of the integral image at the corners of a rectangle.
- What is the sum of pixel values within the rectangle?
  \[ \text{sum} = A - B - C + D \]
- Only 3 additions are required for any size of rectangle!
Computing a rectangle feature
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 \( \sim 160,000 \)!
- 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?
Boosting

- Boosting is a classification scheme that combines weak learners into a more accurate ensemble classifier.
- Weak learners based on rectangle filters:

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

- Ensemble classification function:

  \[
  C(x) = \begin{cases} 
  1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\
  0 & \text{otherwise}
  \end{cases}
  \]
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 for face detection

• First two features selected by boosting:

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

• **Pros:**
  - Integrates classifier training with feature selection
  - Complexity of training is linear in the number of training examples
  - Flexibility in the choice of weak learners, boosting scheme
  - Testing is fast
  - Easy to implement

• **Cons:**
  - Needs many training examples
  - Training is slow
  - Often doesn’t work as well as SVM or a deep neural network (especially for many-class problems)
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

• Chaining together classifiers is a good way to drive down the false positive rate
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 it meets the target rates on the validation set
  • Need to lower AdaBoost threshold to maximize detection (as opposed to minimizing total classification error)

• 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)
Output of Face Detector on Test Images
Related detection tasks

Facial Feature Localization

Profile Detection

Gender classification
Summary: Viola/Jones detector

• Rectangle features
• Integral images for fast computation
• Boosting for feature selection
• Attentional cascade for fast rejection of negative windows
Next step: Generic object detection
Histograms of oriented gradients (HOG)

- Partition image into blocks and compute histogram of gradient orientations in each block

Pedestrian detection with HOG

- Train a pedestrian template using a linear support vector machine

positive training examples

negative training examples

N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
Pedestrian detection with HOG

• Train a pedestrian template using a linear support vector machine
• At test time, convolve feature map with template
• Find local maxima of response
• For multi-scale detection, repeat over multiple levels of a HOG pyramid

Example detections

[Dalal and Triggs, CVPR 2005]
Discriminative part-based models

- Single rigid template usually not enough to represent a category
  - Many objects (e.g. humans) are articulated, or have parts that can vary in configuration

- Many object categories look very different from different viewpoints, or from instance to instance
Discriminative part-based models

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan,
Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Discriminative part-based models

Multiple components

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan,
Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Discriminative part-based models

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan,
Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Object detection progress

PASCAL VOC

Before CNNs

Using CNNs

Source: R. Girshick