Face detection

Jana Kosecka

Many slides adapted from P. Viola, S. Lazebnik and many others

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 ~10^6 pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image 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


A totally different idea

- Use many very simple features
- Learn cascade of tests for target object
- Efficient if:
  - features easy to compute
  - cascade short

Using Many Simple Features

- Viola Jones / Haar Features

(Generalized) Haar Features:

- rectangular blocks, white or black
- 3 types of features:
  - two rectangles: horizontal/vertical
  - three rectangles
  - four rectangles
  - in 24x24 window: 180,000 possible features

Example

Source

Result
Integral Image

Def: The integral image at location \((x, y)\), is the sum of the pixel values above and to the left of \((x, y)\), inclusive. We can calculate the integral image representation of the image in a single pass.

\[
\begin{align*}
ii(x, y) &= \text{value of the integral image -- sum of all pixels above and left of \((x, y)\)} \\
s(x, y) &= \text{cumulative row sum}
\end{align*}
\]

\[
\begin{align*}
s(x, y) &= s(x, y-1) + i(x, y) \\
ii(x, y) &= ii(x-1, y) - s(x, y)
\end{align*}
\]

Efficient Computation of Rectangle Value

Using the integral image representation one can compute the value of any rectangular sum in constant time.

Example: Rectangle D

\[
ii(4) + ii(1) - ii(2) - ii(3)
\]

As a result two-, three-, and four-rectangular features can be computed with 6, 8 and 9 array references respectively.

Idea: Compute lot of simple features – outputs of convolution with the box like filters

Object detection: classification problem

Feature selection

- For a 24x24 detection region, the number of possible rectangle features is \(~160,000!\)

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

- 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


Problem

- How to avoid evaluating (all possible rectangles in 24 x 24 window)?
- 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?

- Answer: Boosting [AdaBoost, Freund/Shapire]
  - Finds small set of features that are “sufficient”
  - Generalizes very well
  - Requires positive and negative examples

AdaBoost Idea (in Viola/Jones):

- Given set of “weak” classifiers:
  - Pick best one
  - Reweight training examples, so that misclassified images have larger weight
  - Reiterate; then linearly combine resulting classifiers

Weak classifiers: Haar features

Boosting illustration

- Weak classifier is a hyperplane
Boosting illustration

Weights Increased

Weak Classifier 2

Boosting illustration

Weights Increased

Weak Classifier 3
Boosting illustration

Final classifier is a combination of weak classifiers

Boosting vs. SVM

- Advantages of boosting
  - Integrates classification with feature selection
  - Complexity of training is linear instead of quadratic in the number of training examples
  - Flexibility in the choice of weak learners, boosting scheme
  - Testing is fast
  - Easy to implement

- Disadvantages
  - Needs many training examples
  - Often doesn't work as well as SVM (especially for many-class problems)

AdaBoost learning algorithm

Discrete AdaBoost (Freund & Schapire 1996)

1. Start with weights $w_i = 1/N$, $i = 1, \ldots, N$.
2. Repeat for $m = 1, 2, \ldots, M$:
   - (a) Fit the classifier $f_m(x) \in \{-1, 1\}$ using weights $w_i$ on the training data.
   - (b) Compute $e_m = \sum_{i=1}^{N} w_i \cdot 1_{y_i \neq f_m(x)}$.
   - (c) Set $w_i = w_i \exp(w_i)$ and normalize so that $\sum w_i = 1$.
3. Output the classifier $\sum_{m=1}^{M} \alpha_m f_m(x)$

Boosting for face detection

- Define weak learners based on rectangle features

\[
 h_{\text{window}}(x) = \begin{cases} 
 1 & \text{if } p_{f_i}(x) > p_i \theta_i \\
 0 & \text{otherwise} 
\end{cases}
\]
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

Example Classifier for Face Detection

A classifier with 200 rectangle features was learned using AdaBoost

95% correct detection on test set with 1 in 14084 false positives.

Classifier are Efficient

- Given a nested set of classifier hypothesis classes

Slide credit: Frank Dellaert, Paul Viola, Forsyth&Ponce

Slide credit: Frank Dellaert, Paul Viola, Forsyth&Ponce
A 1 feature classifier achieves 100% detection rate and about 50% false positive rate.
A 5 feature classifier achieves 100% detection rate and 40% false positive rate (20% cumulative)
using data from previous stage.
A 20 feature classifier achieves 100% detection rate with 10% false positive rate (2% cumulative)

Face Localization Features
- Learned features reflect the task
Face Profile Detection

Face Profile Features

Finding Cars (DARPA Urban Challenge)
- Hand-labeled images of generic car rear-ends
- Training time: ~5 hours, offline

Generating even more examples
- Generic classifier finds all cars in recorded video.
- Compute offline and store in database
Summary Viola-Jones

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows
- Many simple features
  - Generalized Haar features (multi-rectangles)
  - Easy and efficient to compute
- Discriminative Learning:
  - finds a small subset for object recognition
  - Uses AdaBoost
- Result: Feature Cascade
  - 15fps on 700MHz Laptop (=fast!)
- Applications, Face detection, Car detection, Many others