Today

• Window-based generic object detection
  – basic pipeline
  – boosting classifiers
  – face detection as case study
Generic category recognition: basic framework

• Build/train object model
  – Choose a representation
  – Learn or fit parameters of model / classifier

• Generate candidates in new image

• Score the candidates
Generic category recognition: representation choice

Window-based

Part-based

Kristen Grauman
Window-based models
Building an object model

Simple holistic descriptions of image content
- grayscale / color histogram
- vector of pixel intensities
Window-based models
Building an object model

- Pixel-based representations sensitive to small shifts

- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation
Window-based models
Building an object model

- Consider edges, contours, and (oriented) intensity gradients
Window-based models
Building an object model

- Consider edges, contours, and (oriented) intensity gradients

- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination
Window-based models
Building an object model

Given the representation, train a binary classifier
Discriminative classifier construction

**Nearest neighbor**

10^6 examples

Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

**Neural networks**

LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
...

**Support Vector Machines**

Guyon, Vapnik
Heisele, Serre, Poggio, 2001, ...

**Boosting**

Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006, ...

**Conditional Random Fields**

McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003 ...

Slide adapted from Antonio Torralba
Window-based models
Generating and scoring candidates

Car/non-car Classifier
Window-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier

Kristen Grauman
Face detection and recognition

Detection -> Recognition

“Sally”
Challenges of face detection

• Sliding window detector must evaluate tens of thousands of location/scale combinations

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

Accepted Conference on Computer Vision and Pattern Recognition 2001

Rapid Object Detection using a Boosted Cascade of Simple Features

Paul Viola
viola@merl.com
Mitsubishi Electric Research Labs
201 Broadway, 8th FL
Cambridge, MA 02139

Michael Jones
mjones@crl.dec.com
Compaq CRL
One Cambridge Center
Cambridge, MA 02142

Abstract

This paper describes a machine learning approach for video object detection at 15 frames per second on a conventional 700 MHz Intel Pentium III. In other face detection systems, auxiliary information, such as image differences in video sequences,
Boosting intuition

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: training

- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner that achieves the lowest *weighted* training error
  - Raise 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)
Main idea:

– Represent local texture with efficiently computable “rectangular” features within window of interest
– Select discriminative features to be weak classifiers
– Use boosted combination of them as final classifier
– Form a cascade of such classifiers, rejecting clear negatives quickly
Viola-Jones detector: features

“Rectangular” filters

Feature output is difference between adjacent regions

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

Efficiently computable with integral image: any sum can be computed in constant time

Integral image

Value at \((x,y)\) is sum of pixels above and to the left of \((x,y)\)

Adapted from Kristen Grauman and Lana Lazebnik
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: \( ii(x, y) = ii(x, y-1) + s(x, y) \)
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!
Considering all possible filter parameters: position, scale, and type:

180,000+ possible features associated with each 24 x 24 window

Which subset of these features should we use to determine if a window has a face?

Use AdaBoost both to select the informative features and to form the classifier
Viola-Jones detector: AdaBoost

- Want to select the single rectangle feature and threshold that best separates **positive** (faces) and **negative** (non-faces) training examples, in terms of **weighted** error.

Outputs of a possible rectangle feature on faces and non-faces.

Resulting weak classifier:

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

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Kristen Grauman
1. Input the positive and negative training examples along with their labels \( \{(x_i, y_i)\} \), where \( y_i = 1 \) for positive (face) examples and \( y_i = -1 \) for negative examples. 

2. Initialize all the weights to \( w_{i,1} \leftarrow \frac{1}{N} \), where \( N \) is the number of training examples. (Viola and Jones (2004) use a separate \( N_1 \) and \( N_2 \) for positive and negative examples.)

3. For each training stage \( j = 1 \ldots M \):
   
   (a) Renormalize the weights so that they sum up to 1 (divide them by their sum).
   
   (b) Select the best classifier \( h_j(x; f_j, \theta_j, s_j) \) by finding the one that minimizes the weighted classification error
   
   \[
   e_j = \sum_{i=0}^{N-1} w_{i,j} e_{i,j}, \\
   e_{i,j} = 1 - \delta(y_i, h_j(x_i; f_j, \theta_j, s_j)).
   \]  

   (c) Compute the modified error rate \( \beta_j \) and classifier weight \( \alpha_j \),
   
   \[
   \beta_j = \frac{e_j}{1 - e_j} \quad \text{and} \quad \alpha_j = -\log \beta_j.
   \]  

   (d) Update the weights according to the classification errors \( e_{i,j} \)
   
   \[
   w_{i,j+1} \leftarrow w_{i,j} \beta_j^{1-e_{i,j}},
   \]  

   i.e., downweight the training samples that were correctly classified in proportion to the overall classification error.

4. Set the final classifier to
   
   \[
   h(x) = \text{sign} \left[ \sum_{j=0}^{m-1} \alpha_j h_j(x) \right].
   \]  

---


---

Slide adapted from K. Grauman, figure from Szeliski.
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

• Advantages of boosting
  – Integrates classification 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

• Disadvantages
  – Needs many training examples
  – Often found not to work as well as an alternative discriminative classifier, support vector machine (SVM)
Are we done?

• Even if the filters are fast to compute, each new image has a lot of possible windows to search.

• How to make the detection more efficient?
Cascading classifiers for detection

- Form a *cascade* with low false negative rates early on
- Apply less accurate but faster classifiers first to immediately discard windows that clearly appear to be negative

Kristen Grauman
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

- 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 ($0.3^{10} \approx 6 \times 10^{-6}$).
Decision tree classifier

Example problem: decide whether to wait for a table at a restaurant, based on the following attributes:

1. **Alternate:** is there an alternative restaurant nearby?
2. **Bar:** is there a comfortable bar area to wait in?
3. **Fri/Sat:** is today Friday or Saturday?
4. **Hungry:** are we hungry?
5. **Patrons:** number of people in the restaurant (None, Some, Full)
6. **Price:** price range ($, $$, $$$)
7. **Raining:** is it raining outside?
8. **Reservation:** have we made a reservation?
9. **Type:** kind of restaurant (French, Italian, Thai, Burger)
10. **WaitEstimate:** estimated waiting time (0-10, 10-30, 30-60, >60)
**Decision tree classifier**

<table>
<thead>
<tr>
<th>Example</th>
<th>Alt</th>
<th>Bar</th>
<th>Fri</th>
<th>Hun</th>
<th>Pat</th>
<th>Price</th>
<th>Rain</th>
<th>Res</th>
<th>Type</th>
<th>Est</th>
<th>Target</th>
<th>Wait</th>
</tr>
</thead>
<tbody>
<tr>
<td>X₁</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$$$</td>
<td>F</td>
<td>T</td>
<td>French</td>
<td>0–10</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>X₂</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Full</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>30–60</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₃</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>Some</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Burger</td>
<td>0–10</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>X₄</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Full</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>10–30</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>X₅</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>Full</td>
<td>$$$$</td>
<td>F</td>
<td>T</td>
<td>French</td>
<td>&gt;60</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₆</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$</td>
<td>T</td>
<td>T</td>
<td>Italian</td>
<td>0–10</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>X₇</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>None</td>
<td>$</td>
<td>T</td>
<td>F</td>
<td>Burger</td>
<td>0–10</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₈</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Some</td>
<td>$$</td>
<td>T</td>
<td>T</td>
<td>Thai</td>
<td>0–10</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>X₉</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Full</td>
<td>$</td>
<td>T</td>
<td>F</td>
<td>Burger</td>
<td>&gt;60</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₁₀</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Full</td>
<td>$$$$</td>
<td>F</td>
<td>T</td>
<td>Italian</td>
<td>10–30</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₁₁</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>None</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Thai</td>
<td>0–10</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>X₁₂</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Full</td>
<td>$</td>
<td>F</td>
<td>F</td>
<td>Burger</td>
<td>30–60</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>
Decision tree classifier

- **Patrons?**
  - None
  - Some
  - Full
    - WaitEstimate?
      - >60
        - Alternate?
          - No
        - 30-60
          - Fri/Sat?
            - Yes
          - 10-30
            - Fri/Sat?
              - No
          - 0-10
            - Alternate?
              - Yes
        - 0-10
          - Raining?
            - Yes

Lana Lazebnik
## Decision tree classifier

<table>
<thead>
<tr>
<th>Example</th>
<th>Attributes</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alt Bar Fri Hun Pat Price Rain Res Type Est</td>
<td>Wait</td>
</tr>
<tr>
<td>$X_1$</td>
<td>T F F T T Some $$$ F T French 0–10</td>
<td>T</td>
</tr>
<tr>
<td>$X_2$</td>
<td>T F F T T Full $ F F Thai 30–60</td>
<td>F</td>
</tr>
<tr>
<td>$X_3$</td>
<td>F T F F F Some $ F F Burger 0–10</td>
<td>T</td>
</tr>
<tr>
<td>$X_4$</td>
<td>T F T T T Full $ F F Thai 10–30</td>
<td>T</td>
</tr>
<tr>
<td>$X_5$</td>
<td>T F T T F Full $$$ F T French $60</td>
<td>F</td>
</tr>
<tr>
<td>$X_6$</td>
<td>F T F T T Some $$ T T Italian 0–10</td>
<td>T</td>
</tr>
<tr>
<td>$X_7$</td>
<td>F T F F F None $ T F Burger 0–10</td>
<td>F</td>
</tr>
<tr>
<td>$X_8$</td>
<td>F F F F T Some $$ T T Thai 0–10</td>
<td>T</td>
</tr>
<tr>
<td>$X_9$</td>
<td>F T T T F Full $ T F Burger $60</td>
<td>F</td>
</tr>
<tr>
<td>$X_{10}$</td>
<td>T T T T T Full $$$ F T Italian 10–30</td>
<td>F</td>
</tr>
<tr>
<td>$X_{11}$</td>
<td>F F F F F None $ F F Thai 0–10</td>
<td>F</td>
</tr>
<tr>
<td>$X_{12}$</td>
<td>T T T T T Full $ F F Burger 30–60</td>
<td>T</td>
</tr>
</tbody>
</table>
Decision tree classifier

- **Patrons?**
  - None: F
  - Some: T
  - Full:
    - **Wait Estimate?**
      - >60: F
      - 30-60: Alternate?
        - No: Reservation?
          - No: Bar?
            - No: F
            - Yes: T
        - Yes: Fri/Sat?
          - No: T
          - Yes: F
      - 10-30:
        - Hungry?
          - No: Alternate?
            - No: T
            - Yes: T
          - Yes: T
      - 0-10: T

Lana Lazebnik
## Decision tree classifier

<table>
<thead>
<tr>
<th>Example</th>
<th>Attr.</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1$</td>
<td>Alt: T, Bar: F, Fri: F, Hun: T</td>
<td>Some, $$$, F</td>
</tr>
<tr>
<td>$X_5$</td>
<td>Alt: T, Bar: F, Fri: T, Hun: F</td>
<td>Full, $$$, F</td>
</tr>
<tr>
<td>$X_{10}$</td>
<td>Alt: T, Bar: T, Fri: T, Hun: T</td>
<td>Full, $$$, F</td>
</tr>
<tr>
<td>$X_{11}$</td>
<td>Alt: F, Bar: F, Fri: F, Hun: F</td>
<td>None, $, F</td>
</tr>
</tbody>
</table>
Viola-Jones detector: summary

Train cascade of classifiers with AdaBoost

Selected features, thresholds, and weights

Apply to each subwindow

New image

Train with 5K positives, 350M negatives
Real-time detector using 38 layer cascade (0.067s)
6061 features in all layers

Adapted from Kristen Grauman
Viola-Jones detector: summary

- 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* of classifiers for fast rejection of non-face windows

Viola-Jones Face Detector: Results
Viola-Jones Face Detector: Results
Viola-Jones Face Detector: Results
Detecting profile faces?

Can we use the same detector?
Viola-Jones Face Detector: Results
Example using Viola-Jones detector

Frontal faces detected and then tracked, character names inferred with alignment of script and subtitles.

Google now erases faces, license plates on Map Street View

By Elinor Mills, CNET News.com
Friday, August 24, 2007 01:37 PM

Google has gotten a lot of flack from privacy advocates for photographing faces and license plate numbers and displaying them on the Street View in Google Maps. Originally, the company said only people who identified themselves could ask the company to remove their image.

But Google has quietly changed that policy, partly in response to criticism, and now anyone can alert the company and have an image of a license plate or a recognizable face removed, not just the owner of the face or car, says Marissa Mayer, vice president of search products and user experience at Google.

"It's a good policy for users and also clarifies the intent of the product," she said in an interview following her keynote at the Search Engine Strategies conference in San Jose, Calif., Wednesday.

The policy change was made about 10 days after the launch of the product in late May, but was not publicly announced, according to Mayer. The company is removing images only when someone notifies them and not proactively, she said. "It was definitely a big policy change inside."
Face detection and recognition
Consumer application: iPhoto 2009

http://www.apple.com/ilife/iphoto/
Consumer application: iPhoto 2009

Things iPhoto thinks are faces
Consumer application: iPhoto 2009
Can be trained to recognize pets!

http://gizmodo.com/5140703/iphotos-facial-recognition-feature-works-on-cats
Applications of Face Recognition

Surveillance

- Recording

Detecting....

Matching with Database
Name: Alireza,
Date: 25 My 2007 15:45
Place: Main corridor

Name: Unknown
Date: 25 My 2007 15:45
Place: Main corridor

Report