**Object Detection**

Haibin Ling

---

**What is Object Detection?**
- Locate an object in an input image

---

**Extensions**
- None or multiple objects
- Object segmentation
- Object detection in videos

---

**Detection as Classification**
- **Binary classification**
  \[ x \in \mathbb{R}^d \rightarrow f(x) \in \{+1,-1\} \]
- **Training/Learning** – find the classifier function \( f(x) \)
  \[ X = \{(x_i, y_i)\}_{i=1}^m \rightarrow f(x) \in \{+1,-1\} \]
- **Example** – face detection
  \[ f(x) = +1 \quad \text{face} \]
  \[ f(x) = -1 \quad \text{non-face} \]

---

**Face Detection using AdaBoost**
- **Viola & Jones, IJCV 2004**
- **Face detection**
  - Fastest at the time, 15 frames per second
- **Key ingredients**
  - Feature selection – weak classifiers \( \leftrightarrow \) features
  - Fast feature evaluation – integral image
  - Fast classifier evaluation – classifier cascade

---

**Principle of Boosting (Schapire’90)**
- 三个臭皮匠，顶个诸葛亮
  - Three cobblers with their wits combined, equal Zhuge Liang the master mind. (Or, two heads are better than one).
- In the language of learning: a strong classifier can be created by many weak classifiers.
  \[ f(x) = f_1(x) + f_2(x) + f_3(x) + \cdots \]
  - **Strong classifier**: our target.
  - **Weak classifier**: better than chance, e.g., 51% classification rate.

---

**Principle of Adaboost (Freund&Schapire’96)**
- Different weak classifier have different capability
  \[ f(x) = c_1 f_1(x) + c_2 f_2(x) + c_3 f_3(x) + \cdots \]
  - **Failure is the mother of success**
    - Sequentially add weak classifiers
    - Each new classifier, \( \hat{f}_n \), focus on currently mis-classified samples.
Toy Example – taken from Antonio Torralba @MIT

Each data point has a class label:
\[ y_i = \begin{cases} +1 \quad \text{if} \quad h(x_i) > 0 \\ -1 \quad \text{if} \quad h(x_i) < 0 \end{cases} \]

and a weight:
\[ w_i = 1 \]

This one seems to be the best

We set a new problem for which the previous weak classifier performs at chance again

Each data point has a class label:
\[ y_i = \begin{cases} +1 \quad \text{if} \quad h(x_i) > 0 \\ -1 \quad \text{if} \quad h(x_i) < 0 \end{cases} \]

We update the weights:
\[ w_i \leftarrow w_i \exp(-y_i f_i) \]

This one seems to be the best

We set a new problem for which the previous weak classifier performs at chance again

Each data point has a class label:
\[ y_i = \begin{cases} +1 \quad \text{if} \quad h(x_i) > 0 \\ -1 \quad \text{if} \quad h(x_i) < 0 \end{cases} \]

We update the weights:
\[ w_i \leftarrow w_i \exp(-y_i f_i) \]

This one seems to be the best

We set a new problem for which the previous weak classifier performs at chance again

Each data point has a class label:
\[ y_i = \begin{cases} +1 \quad \text{if} \quad h(x_i) > 0 \\ -1 \quad \text{if} \quad h(x_i) < 0 \end{cases} \]

We update the weights:
\[ w_i \leftarrow w_i \exp(-y_i f_i) \]

This one seems to be the best

We set a new problem for which the previous weak classifier performs at chance again

Each data point has a class label:
\[ y_i = \begin{cases} +1 \quad \text{if} \quad h(x_i) > 0 \\ -1 \quad \text{if} \quad h(x_i) < 0 \end{cases} \]

We update the weights:
\[ w_i \leftarrow w_i \exp(-y_i f_i) \]
The strong (non-linear) classifier is built as the combination of all the weak (linear) classifiers.

**Toy example**

**Haar Feature**

- Rectangular features
  - Two-rectangle features (A, B)
  - Three-rectangle feature (C)
  - Four-rectangle feature (D)

- Properties
  - Over complete
  - Computationally efficient
  - Feature computation $\rightarrow$ summing rectangle intensities

**Fast Feature Computation**

**What are the features used?**

- Haar features?
  - There are over 45k different Haar features.
  - Computationally prohibitive.
  - Feature selection needed.

- Joint feature selection and classifier learning
  - Combine a subset of discriminative features to create an effective classifier
  - An effective classifier $\rightarrow$ AdaBoost
  - Selected Haar features $\rightarrow$ Weak classifiers

**Performance of 200 feature face detector**

A reasonable detection rate of 0.95 can be achieved while maintaining an extremely low false positive rate of approximately $10^{-4}$.

- First features selected by AdaBoost are meaningful and have high discriminative power
- By varying the threshold of the final classifier one can construct a two-feature classifier which has a detection rate of 1 and a false positive rate of 0.4.

**Speed-up through Cascading**

- Simple, boosted classifiers can reject many negative sub-windows while detecting all positive instances.
- Series of such simple classifiers can achieve good detection performance while eliminating the need for further processing of negative sub-windows

*Training:* Subsequent classifiers are trained only on examples which pass through all the previous classifiers.
Experiments (dataset for training)

- 4916 positive training examples
  - Manually cropped
  - Aligned, normalized, and scaled
  - Resolution of 24x24
- 10,000 negative examples
  - Selected randomly
  - Sub-windows from 9500 images which did not contain faces

Experimental Results

- Testing on the MIT+CMU frontal face test set
  - 130 images
  - 507 labelled frontal faces

<table>
<thead>
<tr>
<th>Classifier</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
<th>M9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viola-Jones</td>
<td>76.1%</td>
<td>89.3%</td>
<td>96.9%</td>
<td>98.5%</td>
<td>96.1%</td>
<td>95.1%</td>
<td>91.2%</td>
<td>90.1%</td>
<td></td>
</tr>
<tr>
<td>Rowley-Baluja-Kanade</td>
<td>81.2%</td>
<td>90.4%</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>88.3%</td>
<td>-</td>
<td>80.1%</td>
</tr>
<tr>
<td>Schneiderman-Kanade</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>96.4%</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Roth-Yang-Ajuha</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

AdaBoost.M1

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 \( \epsilon_m = E_w \left[ \mathbb{I}_{f_m(x) 
eq y} \right] \).
   (c) Set \( w_i \leftarrow w_i \exp(\epsilon_m \mathbb{I}_{f_m(x) \neq y}) \), \( i = 1, 2, \ldots, N \), and renormalize so that \( \sum w_i = 1 \).
3. Output the classifier \( \text{sign} \left[ \sum_{m=1}^{M} w_m f_m(x) \right] \)

Additive Model

- Additive Model

\[
f(x) = \sum_{m=1}^{M} \beta_m b(x; \gamma_m)
\]

- Fitting an additive model by minimizing a loss function \( L(y, f(x)) \)

\[
\beta_{m*}, \gamma_{m*} = \arg \min_{\beta_m, \gamma_m} \sum_{i=1}^{N} L(y_i, f(x_i))
\]

\[
= \arg \min_{\beta_m, \gamma_m} \sum_{i=1}^{N} \left[ y_i \sum_{m} \beta_m b(x; \gamma_m) \right]
\]

Forward Stagewise Additive Fitting

1. Input: \( X = \{(x_i, y_i)\}_{i=1}^N, y \in \{-1, 1\} \)
2. Initialization \( f^{(0)}(x) = 0 \)
3. For \( m = 1 \) to \( M \):
   (a) Compute
   \[
   (\beta_{m*}, \gamma_{m*}) = \arg \min_{\beta_m, \gamma_m} \sum_{i} \left[ y_i f^{(m-1)}(x_i) + \beta b(x_i; \gamma) \right]
   \]
   (b) Set \( f^{(m)}(x) = f^{(m-1)}(x) + \beta_b b(x_i; \gamma) \)
4. Output \( f(x) = f^{(M)}(x) \)
Fitting with Exponential Loss

- Exponential loss function
  \[ L(y, f(x)) = \exp(-y \cdot f(x)) \]

- For each component
  \[ (\beta_m, g_m) = \arg \min_{\beta, g} \sum_{i=1}^N \exp\left(-y_i \left(f^{(m-1)}(x_i) + \beta b(x_i, \gamma)\right)\right) \]
  \[ = \arg \min_{\beta, g} \sum_{i=1}^N \exp\left(-y_i \left(f^{(m-1)}(x_i) + \beta b(x_i, \gamma)\right)\right) \]
  \[ = \arg \min_{\beta, g} \sum_{i=1}^N \exp\left(-\beta b(x_i, \gamma)\right) \]
  \[ w^{(m)} = \exp(-y_i f^{(m-1)}(x_i)) \]

This is AdaBoost!

- For classification, use weak classifiers \[ g(x) \in \{-1, 1\} \]
  \[ (\beta_m, g_m) = \arg \min_{\beta, g} \sum_{i=1}^N \exp\left(-\beta y_i g(x_i)\right) \]

Solve it, we have

- \[ g_m = \arg \min_{g} \sum_{i=1}^N \exp\left(-\beta y_i g(x_i)\right) \]
  \[ \beta_m = \frac{1}{2N} \log \frac{1 - err_m}{err_m} \]  — Coefficient for weak classifier \( g_m(x) \)
  \[ err_m = \frac{\sum_{i=1}^N \exp\left(-\beta y_i g(x_i)\right)}{\sum_{i=1}^N \exp\left(-\beta y_i g(x_i)\right)} \]  — Weighted training error

Object Detection using Histogram of Oriented Gradient (HOG)

Dalal & Triggs, CVPR 2005

- Focus on creating robust encoding of images
- Linear SVM as classifier on normalized image windows, is reliable & fast
- Moving window based detector with non-maximum suppression over scale space

Overall Architecture

Learning Phase
- Create normalised training data set
- Encode images into feature vectors
- Learn binary classifier
- Object/Non-object decision

Detection Phase
- Scan image at all scales and locations
- Run classifier to obtain object/non-object decisions
- Fuse multiple detections in 3-D position & scale space
- Object detections with bounding boxes

Descriptor Processing Chain

- Image Window
- Collect HOGs over detection window
- Contrast normalize over overlapping spatial cells
- Weighted vote in spatial & orientation cells
- Compute gradients
- Gamma compression

Multi-Scale Detection

- After dense multi-scale scan of detection window
- Map each detection to 3D \([x, y, \text{scale}]\) space
- Apply robust mode detection, like mean shift
  \[ H = \exp(\beta x \cdot \exp(\beta x \cdot \sigma)) \]
  \[ f(x) = \sum_{i=1}^n \exp(\beta x \cdot \text{mean}) \]
HOG for Human Detection

- (a) Input window
- (b) Average gradients
- (c) Weighted pos wts
- (d) Weighted neg wts
- (e) Dominant pos orientations
- (f) Dominant neg orientations

Figure 1. Our HOG detectors out naturally by different cues especially the head, shoulders and feet. The most active blocks are marked in the image background and民心 the center. The average gradient image over the training examples. (b) Each pixel shows the maximum positive HOG weight in the block central to the pixel. (c) Guidelines for the negative HOG weights, (d) A heat map and (e) a component HOG descriptor. (f) The HOG descriptor is weighted by, respectively, the positive and the negative HOG weights.

Descriptor Cues: Motorbikes

- Average gradients
- Weighted pos wts
- Weighted neg wts
- Dominant pos orientations
- Dominant neg orientations

Detection Examples