Outline

• **Last time:** window-based generic object detection
  – basic pipeline
  – face detection with boosting as case study
Generic category recognition: representation choice

Window-based

Part-based
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
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
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


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


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


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
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
Outline

• Last time: window-based generic object detection
  – basic pipeline
  – face detection with boosting as case study

• Today: discriminative classifiers for image recognition
  – nearest neighbors (+ scene match app)
  – support vector machines (+ gender, person app)
Nearest Neighbor classification

- Assign label of nearest training data point to each test data point

Black = negative
Red = positive

Voronoi partitioning of feature space for 2-category 2D data

Novel test example
Closest to a positive example from the training set, so classify it as positive.

Voronoi partitioning of feature space for 2-category 2D data
K-Nearest Neighbors classification

- For a new point, find the \( k \) closest points from training data
- Labels of the \( k \) points “vote” to classify

If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

Black = negative
Red = positive

Source: D. Lowe
A nearest neighbor recognition example
Where in the World?

Where in the World?
Where in the World?
6+ million geotagged photos by 109,788 photographers

Annotated by Flickr users

Slides: James Hays
6+ million geotagged photos by 109,788 photographers

Annotated by Flickr users

Slides: James Hays
Which scene properties are relevant?
Spatial Envelope Theory of Scene Representation
Oliva & Torralba (2001)

A scene is a single surface that can be represented by global (statistical) descriptors
Global texture: capturing the “Gist” of the scene

Capture global image properties while keeping some spatial information

\[ V = \{ \text{energy at each orientation and scale} \} = 6 \times 4 \text{ dimensions} \]

\[ | V_t | \rightarrow \text{PCA} \rightarrow G \]

Gist descriptor

Oliva & Torralba IJCV 2001, Torralba et al. CVPR 2003
Which scene properties are relevant?

- Gist scene descriptor
- Color Histograms - L*A*B* 4x14x14 histograms
- Texton Histograms – 512 entry, filter bank based
- Line Features – Histograms of straight line stats
Scene Matches

Scene Matches

Scene Matches

The Importance of Data

Feature Performance

- First Nearest Neighbor Scene Match
- Mean Shift Mode, Largest Cluster
- Chance—Random Scenes

Percentage of Estimates Within 200km

Feature Used to Estimate Geolocation:
- Color
- Geometry
- Gist
- Lines
- 16x16
- Textons
- 5x5
- All features
Nearest neighbors: pros and cons

• Pros:
  – Simple to implement
  – Flexible to feature / distance choices
  – Naturally handles multi-class cases
  – Can do well in practice with enough representative data

• Cons:
  – Large search problem to find nearest neighbors
  – Storage of data
  – Must know we have a meaningful distance function
Outline

• Discriminative classifiers
  – Boosting (last time)
  – Nearest neighbors
  – Support vector machines
Linear classifiers
Lines in $\mathbb{R}^2$

Let \( w = \begin{bmatrix} a \\ c \end{bmatrix} \) and \( x = \begin{bmatrix} x \\ y \end{bmatrix} \)

\[
ax + cy + b = 0
\]
Lines in $\mathbb{R}^2$

Let \( w = \begin{bmatrix} a \\ c \end{bmatrix} \quad x = \begin{bmatrix} x \\ y \end{bmatrix} \)

\[ ax + cy + b = 0 \]

\[ w \cdot x + b = 0 \]
Lines in $\mathbb{R}^2$

Let $w = \begin{bmatrix} a \\ c \end{bmatrix}$, $x = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$w \cdot x + b = 0$$
Let $w = \begin{bmatrix} a \\ c \end{bmatrix}$, $x = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$w \cdot x + b = 0$$

Distance from point to line $D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}}$
Let \( w = \begin{bmatrix} a \\ c \end{bmatrix} \) \( x = \begin{bmatrix} x \\ y \end{bmatrix} \)

\[
ax + cy + b = 0
\]

\[
w \cdot x + b = 0
\]

Distance from point to line:

\[
D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{w^T x_0 + b}{\|w\|}
\]

\( (x_0, y_0) \)
Linear classifiers

• Find linear function to separate positive and negative examples

\[
\begin{align*}
x_i \text{ positive: } & \quad x_i \cdot w + b \geq 0 \\
x_i \text{ negative: } & \quad x_i \cdot w + b < 0
\end{align*}
\]

Which line is best?
Support Vector Machines (SVMs)

- Discriminative classifier based on optimal separating line (for 2d case)
- Maximize the margin between the positive and negative training examples
Support vector machines

- Want line that maximizes the margin.

\[
\begin{align*}
\text{x}_i \text{ positive } (y_i = 1) : & \quad \text{x}_i \cdot \text{w} + b \geq 1 \\
\text{x}_i \text{ negative } (y_i = -1) : & \quad \text{x}_i \cdot \text{w} + b \leq -1 \\
\text{For support vectors,} & \quad \text{x}_i \cdot \text{w} + b = \pm 1
\end{align*}
\]

Support vector machines

- Want line that maximizes the margin.

\[ wx + b = 1 \]
\[ wx + b = 0 \]
\[ wx + b = -1 \]

**For support vectors:**
\[ x_i \cdot w + b = \pm 1 \]

**Distance between point and line:**
\[ \frac{|x_i \cdot w + b|}{||w||} \]

**Margin M:**
\[ M = \left| \frac{1}{||w||} - \frac{-1}{||w||} \right| = \frac{2}{||w||} \]
Support vector machines

- Want line that maximizes the margin.

\[ w \cdot x + b = \pm 1 \]

\[ x_i \text{ positive } (y_i = 1): \quad x_i \cdot w + b \geq 1 \]
\[ x_i \text{ negative } (y_i = -1): \quad x_i \cdot w + b \leq -1 \]

For support vectors, \( x_i \cdot w + b = \pm 1 \)

Distance between point and line:

\[ \frac{|x_i \cdot w + b|}{\|w\|} \]

Therefore, the margin is \( \frac{2}{\|w\|} \)
Finding the maximum margin line

1. Maximize margin \( \frac{2}{||w||} \)

2. Correctly classify all training data points:

- \( x_i \) positive \((y_i = 1)\): \( x_i \cdot w + b \geq 1 \)
- \( x_i \) negative \((y_i = -1)\): \( x_i \cdot w + b \leq -1 \)

**Quadratic optimization problem:**

Minimize \( \frac{1}{2} w^T w \)

Subject to \( y_i(w \cdot x_i + b) \geq 1 \)
Finding the maximum margin line

- Solution: \( \mathbf{w} = \sum_{i} \alpha_i y_i \mathbf{x}_i \)

- Learned weight
- Support vector
Finding the maximum margin line

• Solution: \[ w = \sum \alpha_i y_i x_i \]
  \[ b = y_i - w \cdot x_i \] (for any support vector)

\[ w \cdot x + b = \sum \alpha_i y_i x_i \cdot x + b \]

• Classification function:

\[ f(x) = \text{sign} \left( w \cdot x + b \right) \]

\[ = \text{sign} \left( \sum \alpha_i y_i x_i \cdot x + b \right) \]

If \( f(x) < 0 \), classify as negative, if \( f(x) > 0 \), classify as positive
Questions

• What if the features are not 2d?
• What if the data is not linearly separable?
• What if we have more than just two categories?
Questions

• **What if the features are not 2d?**
  – Generalizes to d-dimensions – replace line with “hyperplane”

• **What if the data is not linearly separable?**

• **What if we have more than just two categories?**
Person detection with HoG’s & linear SVM’s

• Map each grid cell in the input window to a histogram counting the gradients per orientation.

• Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

Code available: http://pascal.inrialpes.fr/soft/olt/
Person detection with HoG’s & linear SVM’s

- Histograms of Oriented Gradients for Human Detection, Navneet Dalal, Bill Triggs, International Conference on Computer Vision & Pattern Recognition - June 2005
Questions

• What if the features are not 2d?
• **What if the data is not linearly separable?**
• What if we have more than just two categories?
Non-linear SVMs

- Datasets that are linearly separable with some noise work out great:

- But what are we going to do if the dataset is just too hard?

- How about… mapping data to a higher-dimensional space:
Non-linear SVMs: feature spaces

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:

\[ \Phi: x \rightarrow \varphi(x) \]
The “Kernel Trick”

- The linear classifier relies on dot product between vectors $K(x_i, x_j) = x_i^T x_j$
Finding the maximum margin line

- Solution: \( \mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \)
  \[ b = y_i - \mathbf{w} \cdot \mathbf{x}_i \] (for any support vector)

\[ \mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b \]
The “Kernel Trick”

- The linear classifier relies on dot product between vectors $K(x_i, x_j) = x_i^T x_j$

- If every data point is mapped into high-dimensional space via some transformation $\Phi: x \rightarrow \varphi(x)$, the dot product becomes:

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

- A kernel function is similarity function that corresponds to an inner product in some expanded feature space.

Slide from Andrew Moore’s tutorial: http://www.autonlab.org/tutorials/svm.html
Example

2-dimensional vectors $x = [x_1 \ x_2]$;

let $K(x_i, x_j) = (1 + x_i^T x_j)^2$

Need to show that $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$:

$$K(x_i, x_j) = (1 + x_i^T x_j)^2,$$

$$= 1 + x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{j1} x_{i2} x_{j2} + x_{i2}^2 x_{j2}^2 + 2 x_{i1} x_{j1} + 2 x_{i2} x_{j2}$$

$$= \begin{bmatrix} 1 & x_{i1}^2 & \sqrt{2} x_{i1} x_{i2} & x_{i2}^2 & \sqrt{2} x_{i1} & \sqrt{2} x_{i2} \end{bmatrix}^T$$

$$\begin{bmatrix} 1 & x_{j1}^2 & \sqrt{2} x_{j1} x_{j2} & x_{j2}^2 & \sqrt{2} x_{j1} & \sqrt{2} x_{j2} \end{bmatrix}$$

$$= \phi(x_i)^T \phi(x_j),$$

where $\phi(x) = \begin{bmatrix} 1 & x_1^2 & \sqrt{2} x_1 x_2 & x_2^2 & \sqrt{2} x_1 & \sqrt{2} x_2 \end{bmatrix}$

from Andrew Moore’s tutorial: http://www.autonlab.org/tutorials/svm.html
Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(x)$, define a kernel function $K$ such that

$$K(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i K(x_i, x) + b$$
Examples of kernel functions

- **Linear:**
  \[ K(x_i, x_j) = x_i^T x_j \]

- **Gaussian RBF:**
  \[ K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \]

- **Histogram intersection:**
  \[ K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k)) \]
SVMs for recognition

1. Define your representation for each example.

2. Select a kernel function.

3. Compute pairwise kernel values between labeled examples.

4. Use this “kernel matrix” to solve for SVM support vectors & weights.

5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.
Example: learning gender with SVMs

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Moghaddam and Yang, Face & Gesture 2000.
Face alignment processing

- Multiscale Head Search
- Feature Search

Scale → Warp → Mask

Processed faces

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.
Learning gender with SVMs

• Training examples:
  – 1044 males
  – 713 females

• Experiment with various kernels, select Gaussian RBF

\[ K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \]
Support Faces

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.
## Classifier Performance

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Overall</td>
</tr>
<tr>
<td>SVM with RBF kernel</td>
<td>3.38%</td>
</tr>
<tr>
<td>SVM with cubic polynomial kernel</td>
<td>4.88%</td>
</tr>
<tr>
<td>Large Ensemble of RBF</td>
<td>5.54%</td>
</tr>
<tr>
<td>Classical RBF</td>
<td>7.79%</td>
</tr>
<tr>
<td>Quadratic classifier</td>
<td>10.63%</td>
</tr>
<tr>
<td>Fisher linear discriminant</td>
<td>13.03%</td>
</tr>
<tr>
<td>Nearest neighbor</td>
<td>27.16%</td>
</tr>
<tr>
<td>Linear classifier</td>
<td>58.95%</td>
</tr>
</tbody>
</table>
Gender perception experiment: How well can humans do?

• Subjects:
  – 30 people (22 male, 8 female)
  – Ages mid-20’s to mid-40’s

• Test data:
  – 254 face images
  – Low res (6 males, 4 females)
  – High res versions

• Task:
  – Classify as male or female, forced choice
  – No time limit
Gender perception experiment: How well can humans do?

Stimuli

Results

<table>
<thead>
<tr>
<th></th>
<th>High-Res</th>
<th>Low-Res</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>6.54%</td>
<td>30.7%</td>
</tr>
</tbody>
</table>

$\sigma = 3.7\%$

Moghaddam and Yang, Face & Gesture 2000.
Human vs. Machine

- SVMs performed better than any single human test subject, at either resolution

Figure 6. SVM vs. Human performance
Hardest examples for humans

Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.
Questions

• What if the features are not 2d?
• What if the data is not linearly separable?
• What if we have more than just two categories?
Multi-class SVMs

• Achieve multi-class classifier by combining a number of binary classifiers

• **One vs. all**
  – Training: learn an SVM for each class vs. the rest
  – Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value

• **One vs. one**
  – Training: learn an SVM for each pair of classes
  – Testing: each learned SVM “votes” for a class to assign to the test example
SVMs: Pros and cons

• Pros
  • Many publicly available SVM packages:
    http://www.kernel-machines.org/software
  • http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  • Kernel-based framework is very powerful, flexible
  • Often a sparse set of support vectors – compact at test time
  • Work very well in practice, even with very small training sample sizes

• Cons
  • No “direct” multi-class SVM, must combine two-class SVMs
  • Can be tricky to select best kernel function for a problem
  • Computation, memory
    – During training time, must compute matrix of kernel values for every pair of examples
    – Learning can take a very long time for large-scale problems

Adapted from Lana Lazebnik
Questions?

See you Thursday!