Announcements

- PS3 out tonight; due 6/4, 11:59 pm

Outline

- Last time: window-based generic object detection
  - basic pipeline
  - face detection with boosting as case study
Window-based models
Building an object model

Given the representation, train a binary classifier

Car/non-car Classifier

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

Window-based models
Generating and scoring candidates

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

\[ h(x) = \begin{cases} +1 & \text{if } f(x) > \theta \text{ otherwise} \end{cases} \]


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


Weights Increased

Weak Classifier 3

Final classifier is a combination of weak classifiers
Discriminative classifier construction

- Nearest neighbor
  - 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, Seme, Poggio, 2001,...
- Boosting
  - Viola, Jones 2001, Torralba et al. 2004, Opelt et al. 2006,...
- Conditional Random Fields
  - McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003...

Nearest Neighbor classification

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

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

Black = negative
Red = positive

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

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
6+ million geotagged photos by 109,788 photographers

Annotated by Flickr users

Quantitative Evaluation Test Set

Which scene properties are relevant?
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.

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
Im2GPS: Scene Matches
The Importance of Data

[Image: A graph showing the percentage of correct locations within 200km vs. database size, with curves labeled as 'First Nearest Neighbor Scene Match' and 'Chance - Random Scenes'.]


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 (slow during testing)
  - Storage of data
  - Must have a meaningful distance function

Outline

- Discriminative classifiers
  - Boosting (last time)
  - Nearest neighbors
    - Support vector machines
Linear classifiers

Lines in $\mathbb{R}^2$

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

$ax + cy + b = 0$

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

$ax + cy + b = 0$

$w \cdot x + b = 0$
Let \( w = [a, c] \) and \( x = [x', y'] \)

\[ ax + cy + b = 0 \]

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

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

Distance from point to line.
Linear classifiers
• Find linear function to separate positive and negative examples

\[ x, \text{positive}: \quad x \cdot w + b \geq 0 \]
\[ x, \text{negative}: \quad x \cdot w + b < 0 \]

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.

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

For support vectors, \[ x \cdot w + b = \pm 1 \]

Support vector machines

- Want line that maximizes the margin.

For support vectors, \( \mathbf{x} \cdot \mathbf{w} + b = \pm 1 \)

Distance between points and line:

\[
|\frac{\mathbf{x} \cdot \mathbf{w} + b}{||w||}| = \frac{|1 - 1|}{||w||} = \frac{2}{||w||}
\]

Margin \( M \)

Finding the maximum margin line

1. Maximize margin \( 2/||w|| \)
2. Correctly classify all training data points:
   \[
   \mathbf{x}, \text{ positive (} y_i = 1) : \mathbf{x} \cdot \mathbf{w} + b \geq 1 \\
   \mathbf{x}, \text{ negative (} y_i = -1) : \mathbf{x} \cdot \mathbf{w} + b \leq -1
   \]

Quadratic optimization problem:

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

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

- Solution: \( w = \sum \alpha_i y_i x_i \)

Support vector learned weight

Finding the maximum margin line

- Solution:
  \[ b = y_j - w \cdot x_j \text{ (for any support vector)} \]
  \[ w \cdot x + b = \sum \alpha_i y_i x_i \cdot x + b \]

- Classification function:
  \[ f(x) = \text{sign} (w \cdot x + b) \]
  \[ = \text{sign} \left( \sum \alpha_i y_i x_i \cdot x + b \right) \]

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

- 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 linearly separable:

\[ \Phi: x \rightarrow \phi(x) \]

---

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: 
  \[ w = \sum \alpha_i y_i x_i \]
  \[ b = y_i - w \cdot x_i \] (for any support vector)
  \[ w \cdot x + b = \sum \alpha_i y_i x_i \cdot x + b \]
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 a similarity function that corresponds to an inner product in some expanded feature space.

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 $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$:
$$K(x_i, x_j) = (1 + x_i^T x_j)^2 = 1 + x_i^2 x_j^2 + 2 x_i x_j x_i x_j^2 + x_i^2 x_j^2 + 2 x_i x_j + 2 x_i x_j^2$$
$$[1 \ x_i^2 \sqrt{2} x_i x_j \ x_j^2 \sqrt{2} x_i \sqrt{2} x_j]\cdot [1 \ x_j^2 \sqrt{2} x_i x_j \ x_j^2 \sqrt{2} x_i \sqrt{2} x_j] \Rightarrow \varphi(x_i)^T \varphi(x_j)$$
where $\varphi(x) = [1 \ x_i^2 \sqrt{2} x_i x_j \ x_j^2 \sqrt{2} x_i \sqrt{2} x_j]$.

Nonlinear SVMs
- The kernel trick: instead of explicitly computing the lifting transformation $\varphi(x)$, define a kernel function $K$ such that
  $$K(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$$
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)

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

Nonlinear SVMs

- The kernel trick: instead of explicitly computing the lifting transformation \( \phi(x) \), define a kernel function \( K \) such that
  
  \[ K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \]

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

\[ \sum \alpha_i y_i K(x_i, x) + b \]

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 (i.e., training data).
4. Use this “kernel matrix” to solve for SVM support vectors & weights.
5. To classify a new test 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.
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>3.88%</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>10.6%</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>58.93%</td>
</tr>
</tbody>
</table>

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

• Task:
  – Classify as male or female, forced choice
  – No time limit

Moghaddam and Yang, Face & Gesture 2000.

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>N</td>
<td>4032</td>
<td>252</td>
</tr>
<tr>
<td>Error</td>
<td>6.54%</td>
<td>30.7%</td>
</tr>
</tbody>
</table>

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

Moghaddam and Yang, Face & Gesture 2000.
Hardest examples for humans

Top five human misclassifications

True classification:

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 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**
  - 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

Questions?

See you Tuesday!