Discriminative classifiers for object classification

Thursday, Nov 12
Kristen Grauman
UT-Austin

Last time

– Supervised classification
  • Loss and risk, Bayes rule
  • Skin color detection example
– Sliding window detection
  • Classifiers, boosting algorithm, cascades
  • Face detection example

– Limitations of a global appearance description
– Limitations of sliding window detectors
Example: learning skin colors

- We can represent a class-conditional density using a histogram (a "non-parametric" distribution)

Now we get a new image, and want to label each pixel as skin or non-skin.

Bayes rule

\[
P(skin \mid x) = \frac{P(x \mid skin)P(skin)}{P(x)}
\]

\[
P(skin \mid x) \propto P(x \mid skin)P(skin)
\]
Detection via classification: Main idea

Fleshing out this pipeline a bit more, we need to:
1. Obtain training data
2. Define features
3. Define classifier

Training examples

Feature extraction

Car/non-car Classifier

Adaboost: Intuition

Weak Classifier 1

Weights Increased

Weak Classifier 2

Weak classifier 3

Final classifier is combination of the weak classifiers
Feature extraction: rectangular filters

“Rectangular” filters

Feature output is difference between adjacent regions

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

Avoid scaling images → scale features directly for same cost

Viola & Jones, CVPR 2001

Feature extraction: filter library

Considering all possible filter parameters: position, scale, and type:

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

Use AdaBoost both to select the informative features and to form the classifier
AdaBoost for feature+classifier selection

- 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.

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.

Viola-Jones Face Detector: Results

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

• Discriminative classifiers
  – Boosting (last time)
  – Nearest neighbors
  – Support vector machines
    • Application to pedestrian detection
    • Application to gender classification

Nearest Neighbor classification

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

Black = negative
Red = positive

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

Black = negative
Red = positive

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

Example: nearest neighbor classification

- We could identify the penguin in the new view based on the distance between its chest spot pattern and all the stored penguins' patterns.

Labeled database of known penguin examples
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
    • Application to pedestrian detection
    • Application to gender classification
Linear classifiers

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

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

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

$$ax + cy + b = 0$$

$$w \cdot x + b = 0$$
Let \( \mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix} \) and \( \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \)

\[
ax + cy + b = 0
\]

\[
\mathbf{w} \cdot \mathbf{x} + b = 0
\]

distance from point to line

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

\[
= \frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|}
\]

distance from point to line
Linear classifiers
- Find linear function to separate positive and negative examples

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

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.

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

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

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

For support vectors:
\[
\frac{w^T x + b}{\|w\|} = \pm 1 \quad M = \frac{1}{\|w\|} - \frac{-1}{\|w\|} = 2
\]

Support vector machines

- Want line that maximizes the margin.

\[ w^T x + b > 0 \]
\[ w^T x + b < 0 \]

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

**Quadratic optimization problem:**

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

Subject to \( y_i (w \cdot x_i + b) \geq 1 \)

One constraint for each training point.

Note sign trick.

Finding the maximum margin line

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

- Support vector learned weight

Finding the maximum margin line

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

- Classification function:
  \[
  f(x) = \text{sign} \left( \mathbf{w} \cdot \mathbf{x} + b \right)
  \]
  \[
  = \text{sign} \left( \sum_i \alpha_i \mathbf{x}_i \cdot \mathbf{x} + b \right)
  \]
  \[
  \text{If } f(x) < 0, \text{ classify as negative, if } f(x) > 0, \text{ classify as positive}
  \]

- Notice that it relies on an inner product between the test point \( \mathbf{x} \) and the support vectors \( \mathbf{x}_i \)

- (Solving the optimization problem also involves computing the inner products \( \mathbf{x}_i \cdot \mathbf{x}_j \) between all pairs of training points)
Questions

• How is the SVM objective different from the boosting objective?
• 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

• How is the SVM objective different from the boosting objective?
• 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?
Planes in $\mathbb{R}^3$

Let $\mathbf{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ and $\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$a x + b y + c z + d = 0$$

$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|a x_0 + b y_0 + c z_0 + d|}{\sqrt{a^2 + b^2 + c^2}} = \frac{\mathbf{w}^T \mathbf{x} + d}{\|\mathbf{w}\|}$$

distance from point to plane

Hyperplanes in $\mathbb{R}^n$

Hyperplane $H$ is set of all vectors $\mathbf{x} \in \mathbb{R}^n$ which satisfy:

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b = 0$$

$$\mathbf{w}^T \mathbf{x} + b = 0$$

$$D(H, \mathbf{x}) = \frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|}$$

distance from point to hyperplane
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:

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

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:

$$\sum_i \alpha_i y_i K(x_i, x) + b$$
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:**  
  \[ K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k)) \]

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 for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Give this “kernel matrix” to SVM optimization software to identify support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.
Pedestrian detection

- Detecting upright, walking humans also possible using sliding window’s appearance/texture; e.g.,

Example: pedestrian detection with HoG’s and 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/
Pedestrian detection with HoG’s & SVM’s

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

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>3.38%</td>
</tr>
<tr>
<td>SVM with cubic polynomial kernel</td>
<td>4.88%</td>
</tr>
<tr>
<td>Large Ensemble of RBF&lt;sup&gt;2&lt;/sup&gt;</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>

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 (6 males, 4 females)
  – Low res and high res versions
• 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 →

<table>
<thead>
<tr>
<th>Stimuli</th>
<th>High-Res</th>
<th>Low-Res</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 4032</td>
<td>6.54%</td>
<td>30.7%</td>
</tr>
<tr>
<td>N = 252</td>
<td>Error</td>
<td>Error</td>
</tr>
</tbody>
</table>

σ = 3.7%

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

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

Hardest examples for humans

Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.
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

Summary

• Discriminative classifiers applied to object detection / categorization problems.
  – Boosting (last time)
  – Nearest neighbors
  – Support vector machines
    • Application to pedestrian detection
    • Application to gender classification

Adapted from L. Lezehnik