Classifiers for Recognition
Reading: Chapter 22 (skip 22.3)

- Examine each window of an image
- Classify object class within each window based on a training set images

Example: A Classification Problem

- Categorize images of fish—say, “Atlantic salmon” vs. “Pacific salmon”
- Use features such as length, width, lightness, fin shape & number, mouth position, etc.
- Steps
  1. Preprocessing (e.g., background subtraction)
  2. Feature extraction
  3. Classification

Bayes Risk

Some errors may be inevitable: the minimum risk (shaded area) is called the Bayes risk

Discriminative vs Generative Models

Finding a decision boundary is not the same as modeling a conditional density.

Loss functions in classifiers

- Loss
  - some errors may be more expensive than others
    - e.g. a fatal disease that is easily cured by a cheap medicine with no side-effects -> false positives in diagnosis are better than false negatives
    - We discuss two class classification: L(1→2) is the loss caused by calling 1 a 2
  - Total risk of using classifier s

\[ R(s) = Pr(1 \to 2 | \text{using } s) \cdot L(1 \to 2) + Pr(2 \to 1 | \text{using } s) \cdot L(2 \to 1) \]

Histogram based classifiers

- Use a histogram to represent the class-conditional densities
  - (i.e. p(x|1), p(x|2), etc)
- Advantage: Estimates converge towards correct values with enough data
- Disadvantage: Histogram becomes big with high dimension so requires too much data
  - but maybe we can assume feature independence?
Example Histograms

Kernel Density Estimation

- Parzen windows: Approximate probability density by estimating local density of points (same idea as a histogram)
  - Convolve points with window/kernel function (e.g., Gaussian) using scale parameter (e.g., sigma)

Density Estimation at Different Scales

- Example: Density estimates for 5 data points with differently-scaled kernels
- Scale influences accuracy vs. generality (overfitting)

Example: Kernel Density Estimation
Decision Boundaries

Application: Skin Colour Histograms

- Skin has a very small range of (intensity independent) colours, and little texture
  - Compute colour measure, check if colour is in this range, check if there is little texture (median filter)
  - Get class conditional densities (histograms), priors from data (counting)
- Classifier is
  - if $p(\text{skin} \mid x) > \theta$, classify as skin
  - if $p(\text{skin} \mid x) < \theta$, classify as not skin

Skin Colour Models
Skin Colour Classification

For every pixel $p$ in $I_{test}$

- Determine the chrominance values $(a, b)$ of $I_{test}(p)$
- Lookup the skin likelihood for $(a, b)$ using the skin chrominance model
- Assign this likelihood to $I_{test}(p)$

Results

Figure from "Statistical color models with application to skin detection," M.J. Jones and J. Rehg. Proc. Computer Vision and Pattern Recognition, 1999 copyright 1999, IEEE

ROC Curves

(Receiver operating characteristics)

Plots trade-off between false positives and false negatives for different values of a threshold

Figure from "Statistical color models with application to skin detection," M.J. Jones and J. Rehg. Proc. Computer Vision and Pattern Recognition, 1999 copyright 1999, IEEE

Nearest Neighbor Classifier

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

K-Nearest Neighbors

- For a new point, find the k closest points from training data
- Labels of the k points “vote” to classify
- Avoids fixed scale choice—uses data itself (can be very important in practice)
- Simple method that works well if the distance measure correctly weights the various dimensions

Neural networks

- Compose layered classifiers
  - Use a weighted sum of elements at the previous layer to compute results at next layer
  - Apply a smooth threshold function from each layer to the next (introduces non-linearity)
  - Initialize the network with small random weights
  - Learn all the weights by performing gradient descent (i.e., perform small adjustments to improve results)
Training

- Adjust parameters to minimize error on training set
- Perform gradient descent, making small changes in the direction of the derivative of error with respect to each parameter
- Stop when error is low, and hasn’t changed much
- Network itself is designed by hand to suit the problem, so only the weights are learned

Face Finder: Training

- Positive examples:
  - Preprocess ~1,000 example face images into 20 x 20 inputs
  - Generate 15 “clones” of each with small random rotations, scalings, translations, reflections
- Negative examples
  - Test net on 120 known “no-face” images

Face Finder: Results

- 79.6% of true faces detected with few false positives over complex test set

Architecture of the complete system: they use another neural net to estimate orientation of the face, then rectify it. They search over scales to find bigger/smaller faces.

Face Finder Results: Examples of Misses

Find the face!

- The human visual system needs to apply serial attention to detect faces (context often helps to predict where to look)

Convolutional neural networks

- Template matching using NN classifiers seems to work
- Low-level features are linear filters
  - why not learn the filter kernels, too?

A convolutional neural network, LeNet: the layers filter, subsample, filter, subsample, and finally classify based on outputs of this process.

Support Vector Machines

- Try to obtain the decision boundary directly
  - potentially easier, because we need to encode only the geometry of the boundary, not any irrelevant wiggles in the posterior.
  - Not all points affect the decision boundary