CSE 555 Spring 2009 Homework 3: Non-Parametric and Algorithm Independent Methods

Jason J. Corso
Computer Science and Engineering
University at Buffalo SUNY
jcorso@cse.buffalo.edu
Date Assigned 3 Mar 2009
Date Due 26 Mar 2009
Homework must be submitted in class. No late work will be accepted.

This homework contains both written and computer questions. You must turn in the written questions (which may be parts of the computer questions) in class and you must submit the computer code via the CSE submit script. You are free to choose your language (Matlab, C/C++ or Java).

Problem 1: No Free Lunch Theorem (25%)

This is problem 1 from chapter 9 DHS.

One of the "conservation laws" for generalization states that the positive generalization performance of an algorithm in some learning situations must be offset by the negative performance elsewhere. Consider a very simple learning algorithm that seems to contradict this law. For each test pattern, the prediction of the *majority learning algorithm* is merely the category most prevalent in the training data.

1. Show that averaged over all two-category problems of a given number of features, the off-training set error is 0.5.
2. Repeat (1) by for the *minority learning algorithm*, which always predicts the label of the category *least* prevalent in the training data.
3. Use your answers from (1) and (2) to illustrate part 2 of the No Free Lunch Theorem.

Problem 2: Jackknife Statistics (25%)

These are problems 23 and 26 from chapter 9 DHS.

Show that the jackknife estimates of the mean and the variance,

$$\mu_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n}\mu_{(i)}$$
$$\mathrm{Var}[\hat{\mu}] = \frac{(n-1)}{n}\sum_{i=1}^{n}(\mu_{(i)} - \mu_{(\cdot)}) \ ,$$

respectively, where each $\mu_{(i)}$ is a leave-one-out mean,

$$\mu_{(i)} = \frac{1}{n-1}\sum_{j\neq i} x_j$$

are equiavelent to the traditional estimates of the mean and the variance (sample mean, unbiased sampled variance).

Problem 3: Figure-Ground Segmentation (50%)

Download the dataset from the website:
`http://www.cse.buffalo.edu/~jcorso/t/CSE555/homework3-data.tar.gz`. This is just a set of a few images on which to proceed.

1. Implement the kernel-density estimation-based method for dynamically learning the pixel-color distribution for the foreground and background. The method is described in the Zhao and Davis paper we discussed

in class (`http://www.cse.buffalo.edu/~jcorso/t/CSE555/ZhDaICPR2004.pdf`). Specific comments on the implementation:

(a) Be sure to implement weighted kernel-density estimation (based on the current probability of foreground and probability of background).

(b) Use the same exact color-based feature space that they do.

(c) You do not need to implement the method of normalized KL-divergence for selecting the kernel scale for initialization. Just set it to some reasonable value manually.

(d) You do not need to use the method based on sample variance to set the kernel bandwidth. Just set it to something reasonable (0.1).

(e) You have the choice of either implementing a sampling-based version as Zhao and Davis have done (i.e., take 6% of the pixels each round), or you can simply process all of the pixels.

(f) Rather than implementing the Gaussian kernel as they have, use the Epanechnikov kernel:

$$K(u) = \frac{3}{4}(1 - u^2)\delta(|u| \leq 1) \tag{1}$$

(g) Have your system iterate for a fixed number of iterations (say 25).

2. Set the bandwidth to 0.1 and apply your implementation to the `flower.ppm` file and to the `butterfly.ppm` file. Output the initial probability maps (P_{fg} and P_{bg}), the initial classification, and the final maps and classification.

3. Repeat the previous step but vary the bandwidth from among (0.05, 0.1, 0.15, and 0.2). Explain the results.

4. Now, do a Google^{TM}Image Search for *your favorite wild animal* (e.g., tiger, leopard, hawk) and take the first photographic result. Execute your automatic figure-ground segmenter on this image. Show the resulting separation and explain it. (Note, you probably want to make the image some reasonable size like 320x240 before processing it.)

How and what to submit: explains, figures and tables should be part of the "written" assignment. The code (whether it is Matlab, C or Java) should be tar'd and submitted via the CSE submit script:

1. Login to pollux (the code MUST work on the CSE machines).

2. Use "tar -cvf homework3.tar list-of-files-or-directories".

3. Then, type "submit_cse555 homework3.tar".

Include a README file to explain what is there, but only submit the code. If it is C/C++ code, you need to include a makefile that will compile on the CSE machines (it is thus encouraged to link to any libraries available there rather than submitting them). If it is Java code, you need to include an ant file that will compile on the CSE machines. In all cases, you need to include a script named "go" that will (once compiled) do everything requested in the assignment (in the case of C or Java, it is sufficient to generate image files of the plots and save them to reasonably named files). The script can be bash/tcsh for C/Java and should be a simple Matlab script for Matlab. Note, the script need not automatically do the Google search; just have it operate on the image you've already downloaded and saved.