CISG5810 Course Project

Pedestrian Detection by SVM

1. Objective

This project is an exercise for detecting pedestrians in common video surveillance scenes. See examples in the figure below. For each image, the SVM method slides a window over the image, each window extracts a patch to be classified. The positive window is then displayed in different colors.

In this project, you will use the training examples to learn the SVM classifier for pedestrian. Then on a set of real testing images, you will side windows over the images and classify these windows by the SVMs that you trained.

2. Training Data

INRIA Person Dataset, http://pascal.inrialpes.fr/data/human/


3. What to hand in

a) Visualize the top 5 positive and negative support vectors respectively. To visualize the support vectors, you show the HoG vectors as images.

b) For each of the testing images, you show the detection windows in red.

4. Tools and codes
For the project, you will use the HoG (Histogram of Oriented Gradients) feature to extract the information \( X = (x_1, \ldots, x_n) \) from each image patch, and then train the SVM for classification.

HOG:

http://pascal.inrialpes.fr/software/olt/

http://www.mathworks.com/matlabcentral/fileexchange/28689-hog-descriptor-for-matlab

SVM:

http://svmlight.joachims.org/

http://www.csie.ntu.edu.tw/~cjlin/libsvm/

In LibSVM, binaries for both Matlab and Octave are included for 32bit Windows, just run svmtrain and svmpredict from within the directory you expand the files. If you are using a different architecture/platform, then you need to rebuild the binaries. This is done in matlab by first running mex -setup followed by make. In Octave, simply run make_octave.

This package trains a model from a matrix of feature vectors, and a vector of class labels. A simple example is provided:

matlab> load heart_scale.mat
matlab> model = svmtrain(heart_scale_label, heart_scale_inst, '-c 1 -g 0.07'); % train the model
matlab> [predict_label, accuracy, dec_values] = svmpredict(heart_scale_label, heart_scale_inst, model); % test the training data

The third parameter of svmtrain contains a string with the following options:

options:
-s svm_type : set type of SVM (default 0)
  0 -- C-SVC
  1 -- nu-SVC
  2 -- one-class SVM
  3 -- epsilon-SVR
  4 -- nu-SVR
-t kernel_type : set type of kernel function (default 2)
  0 -- linear: u'v
  1 -- polynomial: (gamma*u'v + coef0)^degree
  2 -- radial basis function: exp(-gamma|u-v|^2)
  3 -- sigmoid: tanh(gamma*u'v + coef0)
-d degree : set degree in kernel function (default 3)
-g gamma : set gamma in kernel function (default 1/num_features)
-r coef0 : set coef0 in kernel function (default 0)
-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
-n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
-p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)
-m cachesize : set cache memory size in MB (default 100)
-e epsilon : set tolerance of termination criterion (default 0.001)
-h shrinking: whether to use the shrinking heuristics, 0 or 1 (default 1)
-b probability_estimates: whether to train a SVC or SVR model for probability estimates, 0 or 1 (default 0)
-wi weight: set the parameter C of class i to weight*C, for C-SVC (default 1)

The k in the -g option means the number of attributes in the input data.

6. Reference for HoG feature