Exercise 4: Feature Extraction and SVM Classification

Due date: 11:59pm, Nov 27th, 2015

1 Introduction

In this exercise, you will use support vector machine (SVM) to classify on-body placement of accelerometer sensors. Data has been collected by strapping a smart phone to the left and right ankles as well the left and right biceps of a subject. The goal is to train a multi-class SVM classifier to determine one of the four placements of the device.

Your implementation should be based on Matlab/Octave. For instructions on downloading and installing Octave, check out Download GNU Octave. Further documentation for Octave functions can be found at the Octave documentation pages. MATLAB documentation can be found at the MATLAB documentation pages.

The basic SVM classifiers work with two classes. To extend the SVM to multiple classes, one approach is to train multiple one-vs-all SVM classifiers. To ease your implementation, you may use LibSVM for this task. LibSVM is a library for support vector machines including multi-class classifiers. It is written in both Java and C++, and can interface with many other languages such as Python and Matlab. A copy of LibSVM has been included in the dataset for this project. To interface with MATLAB/Octave, please follow the instructions in `dataset/libsvm-3.20/matlab`. Upon successful compilation, you will see mex files for your platform under the same directory. To use LibSVM, you will need to include the path to the mex files from your current workspace. Of most relevance to the project are two functions:

- `svmtrain` – train the SVM classifier. The resulting model is stored in a struct `model`

- `svmpredict` – predict the labels of testing samples and evaluate the accuracy of the classifier.

Run `svmtrain` and `svmpredict` in the command window of Matlab or Octave to see the usage of both functions. As inputs to these function, the labels can be stored in an $n \times 1$ vector and the feature data is stored in an $n \times p$ matrix, where n and p are the number of data points and the number of features used.

2 Dataset

In the dataset `acc.mat` provided, there are four matrices, each containing the 3-axis accelerometer readings from left/right bicep and left/right ankle. The first column of the matrix corresponds timestamps of data collection in ms. The 2nd – 4th column correspond to the accelerometer reading in the x-, y-, and z-axis respectively. The last column corresponds to the magnitude computed as $\sqrt{acc_x^2 + acc_y^2 + acc_z^2}$. The accelerometer is roughly sampled at 50Hz.

To compensate possible missing data and uneven sampling intervals in smart phone sensor data, you can interpolate the data to a fixed sampling frequency.

3 Feature Extraction

Many time domain and frequency domain features can be utilized for this task. It is generally observed that the magnitude of acceleration at the ankles are much higher than those from biceps. There are slight differences in the swing range and frequency in the movements of left and right sides of the body during walk.

Some commonly used time-domain features for time-series data X are,

- *Mean absolute value (MAV):* The estimate of mean absolute value in window of L samples given by,

$$\overline{X}_i = \frac{1}{L} \sum_{k=1}^{L} x_k.$$

- *Zero crossings (ZC):* This feature is a simple frequency measure representing the number of times a waveform changes polarity. However,

due to the possibility for noise, a threshold T must be defined. Therefore, a pair of consecutive samples constitutes a zero crossing only if their absolute difference exceeds both a noise threshold and their absolute sum (i.e. they have different polarity). Given two consecutive samples x_k and x_{k+1}, this condition is described by:

$$|x_{k+1} - x_k| > max\left(|x_{k+1} + x_k|, T\right).$$

- *Slope sign changes (SSC):* The number of times that a waveform changes slope polarity may provide additional frequency information about the signal. Given three consecutive samples x_{k-1}, x_k and x_{k+1}, the conditions for a slope sign change at sample k can be expressed as,

$$\begin{gathered} x_k > \max(x_{k-1}, x_{k+1}) \text{ or } x_k < \min(x_{k-1}, x_{k+1}) \\ and \\ \max(|x_{k+1} - x_k|, |x_k - x_{k-1}|) > T_2, \end{gathered} \tag{1}$$

  where T_2 is a threshold for the tolerance to noise.

- *Waveform length (WL):* The total length of the signal over the data window represents a combined measure of amplitude, frequency, and duration. This feature is calculated as the sum of absolute voltage differences between each pair of adjacent samples within the classification window, or

$$WL = \frac{1}{L-1} \sum_{k=2}^{L} |x_k - x_{k-1}|.$$

All above features are computed over a sample window of length L. L, T, T_2 are tunable parameters. One design option is to allow a percentage of overlapping between neighboring windows.

In this exercise, you can extract the above features from each axis of the accelerometer readings and use them directly in classification. Alternatively, you can apply principle component analysis (PCA) to project the features to a much smaller feature space. PCA is useful to reduce the computation complexity and extract more robust features. In feature selection, you can choose the top m features whose sum principal component variance is no less than 99% of the total variance.

4 Supervised Classification using SVM

For multi-class classification, LibSVM provides several options including `svm_type` 0 and 1, `kernel_type` (0 – 3).

To evaluate the accuracy of your classifier, you can divide the data randomly into 70% and 30% for training and testing. Note that the division should keep the data as balanced as possible, namely, roughly the same number of (labelled) data exist for each class in the training and testing data sets. Accuracy and confusion matrices should be computed by averaging over multiple instances of divisions between the training and testing sets.

5 Submission

In the report, give a brief description of the procedure you have implemented, list the features and parameters used, and summarize the performance results (e.g., accuracy and confusion matrix) and key observations. Compare the results with and without PCA, and different type of SVM classifiers. Submit your report (in pdf) and codes (in .m) and output files (in .mat) if applicable through SVN.

Acknowledgement

Many thanks to Ala Shaabana for data collection.