1. The dataset in 'data_for_cnn.mat' consists of 1000 ECG signals and each row corresponds to one ECG signal. The class label for each ECG signal is given in 'class_label. mat' file. Implement the 1D convolutional neural network with BPCNN as the learning algorithm for the evaluation of optimal weight matrices in FC layers and optimal kernels or filters in convolution layer. The network consists of one convolutional layer, one pooling layer and two fully connected (FC) layers. The network flow is given by

   Input-Convolution Layer-Pooling layer- Convolution Layer-Pooling layer -FC1-FC2-FC3-Output

Consider the square loss function as cost function in the output layer. You can select number of hidden neurons by your own choice. In the pooling layer, you can use average pooling with down-sampling factor as 2. (For implementation of the BPCNN algorithm, please refer to the class notes or slides).

(For MATLAB, you can use the inbuilt functions from deep learning toolbox)

(You can use Python with Keras, and tensorflow at the backend.)

2. Implement the 1D convolutional autoencoder for the dataset given in data_for_cnn.mat file. The architecture is given as

   (input-convolution layer-pooling layer-FC-upsampling layer-transpose convolution layer)

   (For MATLAB, you can use the inbuilt functions from deep learning toolbox)
   (You can use Python with Keras, and tensorflow at the backend.)

3. Implement the neuro-fuzzy inference system (NFIS) classifier for the classification task. You can use data4.xlsx file. The training and test instances can be selected using hold out cross-validation.