Final Project

Mary Auchincloss
Department of Electrical and Computer Engineering, Temple University
Mary.Auchincloss@Temple.edu

Introduction: For our final project in this class, we were provided with labeled ECG and we were asked to build two systems for classification, one neural network and one non neural network. This ECG data had eight channels and was labeled for six different health issues. Each channel of an ECG represents the output from a different lead placed on a specific location on the patient's body, the channels all contain different information. The first task in this assignment was to read in the data from where it was stored and convert it from bytes into a usable form. Once I accomplished this, I decided to plot a sample of the data for each channel both labeled health and unhealthy. The plot of channel 0 for a sample data set is shown in Figure 1. Once I was confident, I had imported the data correctly and I was able to see the plots of each channel I researched what algorithms are most effective for these types of signals, I decided on Random Forrest and CNN.

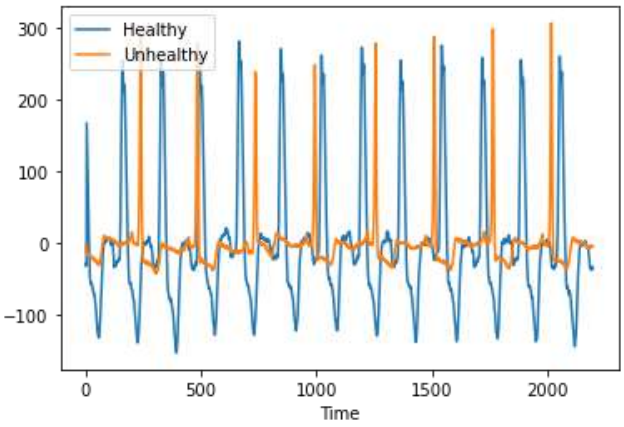

Figure 1

Algorithm No. 1 Description: Random Forrest

For my non neural network implementation I chose to use Random Forrest. To preprocess the data, I added a convolution on the data before training the model. I did this because in my observations of the plots the heartbeat peeks seemed to be different for healthy and unhealthy in all channels. I wanted the algorithm to be able to focus on the peeks during classification. After this preprocessing I used the SciKitLearn RNF package with 100 estimators to create and train my model. For my HPY files I trained on the entire training set. After my presentation I was asked to run RNF without the convolution preprocessing since the other students in the class were able to get better results with RNF without the convolution. Running the standard RNF implementation without the preprocessing yielded slightly worse results than with the convolution in my implementation but not as good as the other students results. If I were to continue to work on this project, I would keep the convolution preprocessing but I would vary the number of estimators and test the different controllable parameters of SciKitLearns RNF.

Algorithm No. 2 Description: CNN

For my neural network implementation, I chose to use CNN or a convolutional neural network. I chose this algorithm because it is known for analyzing audio signals well and the output of an ECG resembles an audio signal. For my model I used 3 convolution layers and 3 non linear layers using Relu and Maxpool. I then adjusted the loss function to give a higher penalty for guessing 0 when the truth was 1. I did this because the health issues, or classes, were not evenly distributed. Figure 2 shows the distribution of the health issues in the training set. I found that using the standard loss function yielded worse results because the model would guess 0 far to often. I also had to adjust the learning rate in the optimizer, it took several attempts before finding the correct value. I spent many hours on this project and built the aspects of my model through trial and error. In all I ran around 200 epochs training on just

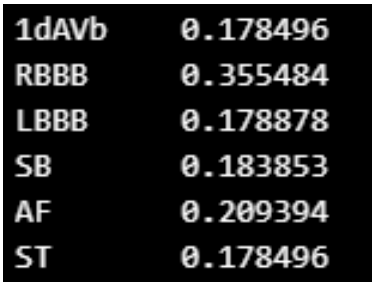

Figure 2

the unhealthy training data and around 50-100 final epochs training on the entire training set. I did see my scores go down slightly with the introduction of the healthy data but after several training rounds the difference was not too large.

Results: When I was satisfied with both of my implementations I ran them on the full training set. I had been testing on smaller sections of the data in an attempt to keep the runtime down. When both implementations finished, I was satisfied with the results from the train and dev sets, I imported the eval set and generated my HYP files. I sent these files to Dr. Picone and received my scores. These results are shown in Table 1.

	Data Set		
Algorithm	**Train**	**Dev Test**	**Eval**
RNF	0.9391 / 0.5106	0.9279 / 0.3378	0.9275 / 0.3372
CNN	0.9805 / 0.9035	0.9806 / 0.9034	0.8404 / 0.1728

Table 1. Train, Dev, and Eval macro accuracy and macro F1 for my implementation of RNF and CNN

Conclusions: This project was very challenging but also rewarding. I feel it was an excellent way to finish this class and combined everything that I learned this semester. Given more time I would have spent more time on the RNF implementation and experimented with the SciKitLearn package more, I also would have explored other ways of preprocessing the data. For CNN I would have trained more epochs on the full data set and used those results to fine tune the weights of my loss function. I enjoyed this class a great deal and feel that I learned a lot.