Diagnosis from ECG data using Neural Networks

Paul An  
Shiv Prathik

# Abstract

During our class, we have covered the Feed-Forward Neural Network. These networks have been identified as good candidate for classifying patterns as well as predicting time-series. Our homework covered classifying patterns from images. We wanted to explore the ways in which we can adapt the Neural Network to make viable predictions on time-series. A well-trained doctor or nurse can look at an ECG, and identify immediately whether there is something wrong with the patient’s heart or not. Our goal is to see if a machine can be trained to identify various problems with a patient from their ECG data. To this effect, we explore different ways in which an Electrocardiogram (ECG) can be fed into various Neural Network architectures, and compare the success rate of eat trained network.

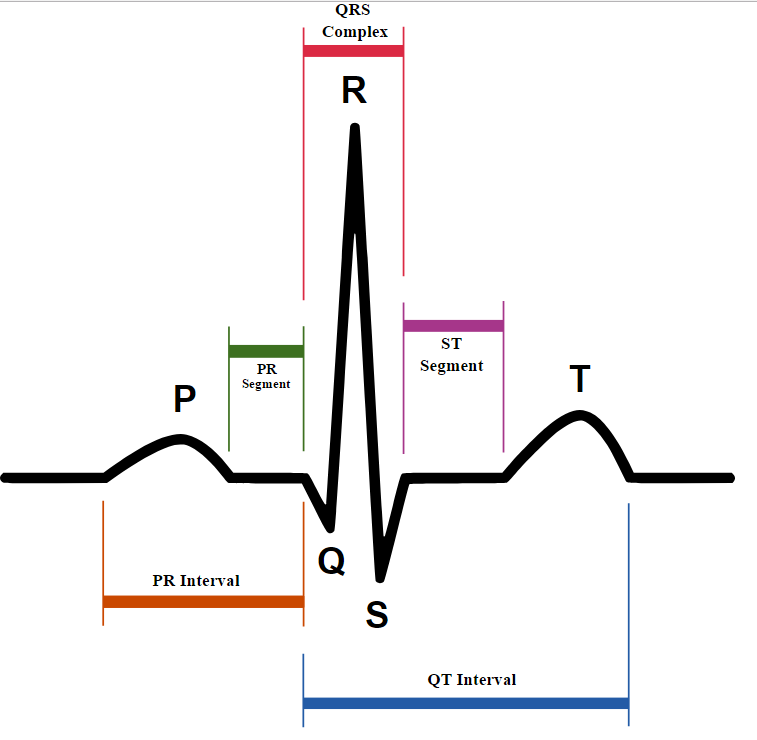
# Introduction

Electrocardiogram (ECG) is a process in which voltage reading is measured from multiple leads on a human body to analyze his heartbeat. Such mechanism is used in hospitals to monitor heart rate and activity. It can also be used to identify and diagnose some heart problems such as myocardial infarction, and dysrhythmia. Fortunately, the medical community is very active in data collection, and research, so anonymized ECG data can be easily obtained. Our data is pulled from the PhysioNet’s PhysioBank, which provides free web access to large collections of recorded physiologic signals. This database gives us access to close to 290 subjects, and 549 records.

Upon an informal conversation with medical residence acquaintances, and refined a few criteria that the input data needs to meet:

* Data from more than one single lead is necessary – each lead tells a different information about the patient.
* Several heartbeats must be examined to be sure.
* Some of the things doctors look for are inversions of P and T waves. Their inversion in certain leads (but not others) are indicative of something wrong with the heart.

The following figure taken from Wikipedia identifies the different phases of an ECG.



As we can see, the heartbeat is divided into a few different sections, and the profile of these features would dictate the diagnosis.

# ECG Data

|  |  |
| --- | --- |
| Myocardial infarction | 148 |
| Cardiomyopathy/Heart failure | 18 |
| Bundle branch block | 15 |
| Dysrhythmia | 14 |
| Myocardial hypertrophy | 7 |
| Valvular heart disease | 6 |
| Myocarditis | 4 |
| Miscellaneous | 4 |
| Healthy controls | 52 |

Each record contains a header file, which describes the patient’s conditions as well as diagnoses. Such information includes: Primary Diagnosis, Additional Diagnosis, Sex, Age, Smoker, Blood Pressure, etc. We first analyzed all headers to obtain a list of all possible diagnosis. We included secondary diagnoses to the search, but found that it yielded many irrelevant diagnoses that did not have any connection to heart-disease or ECG (such as tinnitus). Based on Physionet, the entire database contained patients with the following diagnosis (and distribution)

The signals graph for each record include 15 linear data, representing 12 leads connected to the patient, as well as 3 Frank Leads, discussed later. These leads are named: *i, ii, iii, avr, avl, avf, v1, v2, v3, v4, v5, v6,* and Frank Leads *vx, vy, vz*. The length of the dataset varies, but are around the order of magnitude of 100,000.

FIGURE HERE

# naïve Feed-Forward Network

Our first attempt is to use the feed-forward neural network (FFN) created for a class work, and adapt the input to fit that neural net. The FFN would consist of an input layer with as much nodes as there are input signals, a hidden layer (or 2) of appropriate size, and an output of each of the 15 diagnosis.

FIGURE HERE

We must consider, however, how we feed our data into the FFN. With Neural Networks, especially one simple as Feed-Forward, care must be taken to pre-process the data before applying the algorithm.

The data described in the previous section holds far too much information: each sample consists of 15 graphs, each consisting of over 200,000 data points. An FFN connects each node from layer n to every node in layer n+1. In this case, if we do not do something about the input data, we will have an input size of 1,500,000, and a very long training time (including network construction itself)

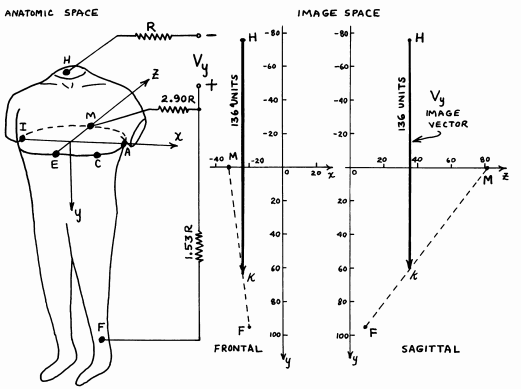
A list of possible problems is identified.

* Heartrate – each person has a different rate.
* Phase – We prefer if the features align with the other records.
* Size of data – there is simply too much data. For each record, we have well over 100 heartbeats.
* Dimensionality – 15 leads is a lot of dimensions
* Patient distribution – The data provided has a bias on diagnosis. There are too many patients with Myocardial infarction.

## Dimensionality reduction

In a normal ECG, 12 leads are connected to the body at various locations.

Ernest Frank, Ph.D. proposed a method in 1956 called vectorcardiography, which aggregates the electrical signals along each orthogonal axis (sagittal, frontal, and transverse) of the heart by via specific combination of leads. This can be achieved both via resistors in the hardware during the capture process, or via post-capture calculations



Since this information is readily available in our dataset, we propose, to use these 3 leads’ data (*vx, vy, vz*) for our learning and prediction.

## Standardizing Input Data

To address the problem of Size of data, heartrate, and phase, we devised a pre-processing algorithm that will pick out prime examples of 4 heartbeats from each record.

We first pick a starting point, and search length. For every time point in this range, we calculate the variance of a small neighborhood around that point.

FIGURE

We can see clearly that this step gives us clear distinction of where the QRS complex of the EKG is.  
Using the maximum and minimum variances, we can define some thresholds. We iterate through this variance set using a state-machine to identify the length between each peak.  
After identifying the average time length of each heartbeat (h), we can seek to the first QRS (x), and define the start and end of our region of interest as:

Once this is done, we are left with a range of data that captures the data with the number of heartbeats, as well as similar phase.

In addition to this, we needed to down sample and rescale the data to a normalized number of time-steps.  
To do this, we define an output resolution of 500 timepoints, and sample uniformly along the original range. In doing so, we also apply a convolution to reduce data loss as well as to reduce noise.

FIGURE

Our algorithm performed satisfactorily:

FIGURE

However, at times, it would capture areas of the data in which there were some input error during ECG capture process:

FIGURE

To mitigate this error, we run a small search outlined by the pseudocode below:

t <- random timepoint in entire data  
n\_data <- run algorithm with start=t  
t\_var <- total variance of n\_data  
if t\_var > threshold: run search again

Since this algorithm takes quite a long time to run, the resulting data array for each record is cached by writing out to file, and loaded on demand.

## Sample Distribution

We omitted much of the records with Myocardial Infarction diagnosis during this study. We wanted to our neural network not to have any inherent bias towards this diagnosis.

We also randomized the entire record-list before separating our training and testing data (70%/30%). This ensures that training and testing data comes from the same distribution.

Finally, it’s also worth mentioning that the training set was shuffled once again at the beginning of each epoch. The motivation for this is to avoid over-fitting, as well as to allow the neural network to overcome local-minimum during its gradient-decent search.

## Results

Finally, these standardized Frank Lead data are fed as inputs into our neural network.

Architecture FIGURE

The entire patient database is separated such that training sample make up 70% of the entire database, and the rest 30%. At the end of each epoch, we test the following:

* Total square-mean-error among all output nodes
* Ratio of Diagnosis prediction correct
* Ratio of Healthy/Unhealthy prediction correct

They are plotted below:

FIGURE

FIGURE

As you can see, the neural network began epoch 0 being only able to identify 30% of the diagnosis correctly, and 60% of healthy/unhealthy correctly.

By epoch 44, our accuracy for diagnosis is 70.1%. We can identify healthy and unhealthy patients with 99.4% accuracy.

There is obvious improvement from our learning, but learning slows down considerably around epoch 23. I believe that with our current configuration, this is the best that we can achieve.

It is also worth noting that a few other trials at each iteration of our pre-processing algorithm design, and produced abysmal results. This highlights a weakness of the FFN, in that feature extraction and pre-processing must be refined properly such that the FFN performs with positive results, and carefully such that the network does not overfit.

# Convolutional Neural Network with Tensorflow

Blah blah

CNN should be able to pick features on its own from its convolutional hidden layers…

# Conclusion

## References

Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. Circulation 101(23):e215-e220 [Circulation Electronic Pages; <http://circ.ahajournals.org/content/101/23/e215.full>]; 2000 (June 13).

ECG Data. Retrieved from <https://www.physionet.org/physiobank/database/ptbdb/>

Frank, E.(1956). "An Accurate, Clinically Practical System For Spatial Vectorcardiography". Circulation. 13:737-749. Originally published May 1, 1956

Electrocardiography, (n.d.) In *Wikipedia.* Retrieved from <https://en.wikipedia.org/wiki/Electrocardiography>