Electrocardiogram (ECG) Automated Interpretation

Azim Shafi - ashafi2018@my.fit.edu
Julian Robledo - jrobledo2017@my.fit.edu
Seng Jhing Goh - sgoh2018@my.fit.edu
Nicangel Sanchez - nsanchez2017@my.fit.edu

Client:

Who We Play For - Scott Hasbrouck

Advisor:

Florida Tech Professor - Dr. Philip Chan

1. **Progress of Current Milestone (Progress Matrix)**

Task	Completion (%)	Julian	SJ	Nic	Azim	To do
1. Develop, and train a CNN Model with a small dataset ~500 patients	100%	25%	25%	25%	25%	
2. Test, and measure the model performance	100%	0%	0%	50%	50%	
3. Test different data distribution to help imbalance data composition	60%	15%	15%	15%	15%	Our model is still in early stages of development so it is hard to determine the impact of an imbalanced dataset.
4. Implement the ECG extraction scripts to the GUI	100%	0%	100%	0%	0%	
5. Research a basic tutorial on creating a CNN in PyTorch and on ways to improve the accuracy and efficiency of the model as well as training and ways to evaluate the improvements over time.	100%	25%	25%	25%	25%	

2. **Discussion (at least a few sentences, ie a paragraph) of each accomplished task (and obstacles) for the current Milestone:**

Task 1: After reviewing examples and tutorials in PyTorch.org we noticed that the dataset that we had previously designed will not work for training an image classifier. The data was not returning the enumeration of the assessment. The assessment was getting returned as a string, which will not work for the tensor. This was a simple fix adding an if statement to the dataset class to return an int value ‘0’ for normal assessment and for ‘1’ abnormal. After we defined a simple Convolutional Neural Network model with 4 layers just to test that our data would get loaded into the model and that the image tensors are getting accepted by the model. We are using the torch CrossEntropyLoss() function to calculate the loss and the SGD with momentum as the optimizer. The training is defined in a simple loop that iterates through the training loader

and feeds the inputs to the model and optimizes. Next, we iterate through the test loader to evaluate the performance of the model that was trained. Early on we noticed that the model was always guessing, with something having a 100% prediction rate for normals and 0% for abnormals. This was confirmed once we implemented the data split explained in more detail in task 3.
Task 2: Implementing the evaluation portion was relatively simple as tallying the number of correct predictions for normals and abnormals allowed us to calculate the true positive and false positive rates. The only obstacle was that the confusion matrix had to be created manually as the library functions that could create a confusion matrix required arrays for the actual and predicted values, but our program stored these values as tensors.
Task 3: After our first successful training, testing and evaluation pipeline we noticed that the model was always guessing. We had a few runs with very good precision 90% of the normals and abnormals getting predicted correctly. We believed that this was the cause because our training and testing datasets were probably using the same images from the pool of ECGs. Afterwards we implemented a 70/30 split using the torch random_split() function to split the dataset into two non-overlapping new datasets of given lengths. 70% of the dataset goes to the raining and 30% for testing. After these changes to the data loaders we noticed a much better representation of the guessing that our data performed. However, there were still a few times when we would have a very high accuracy which was probably to do with the random distribution of the training dataset. We tried to fix these issues by testing different distributions of normal/abnormal ECGs for the training dataset while leaving the testing set at a random split. Because of our model simplicity we could not determine the best data distribution. We need to improve the model architecture as well as continue to test different data distributions.
Task 4: The GUI has been updated with additional features where users can extract ECG graphs using their preferred ECG Machine Model and convert them into PNG files. The ECG Diagnosis Results Display has been completed. Users are now able to view the ECG graph from the selected patient. Once we have the ECG Interpretation successfully completed, we will add this feature to the GUI and users will then be able to view the interpretation results.
Task 5: Every team member looked over various different tutorials and articles in order to better gather information rather than all looking at the same sources. The information each member found to be useful was shared with the rest of the team in order to consolidate all of the research together. One obstacle that was faced is that due to the myriad ways one could employ a CNN in Pytorch, sometimes information from one source would be incompatible with information from another source. This difficulty was overcome by choosing the implementation that was the farthest along rather than continuing in four separate directions.

3. **Discussion (at least a few sentences, ie a paragraph) of contribution of each team member to the current Milestone:**

Julian watched 3 tutorials from different developers and websites in order to help develop the architecture for the neural network. This included starting to understand how a CNN is built, what it's composed of, and the mathematics behind it. The tutorials helped determine the parameters of the layers within our CNN and to help visualize how the learning process of said

network takes place. Julian also helped with making sure that the dataset that we were using was balanced (~50% for abnormal and normal ECGs) in order to ensure that the CNN wasn't being trained on an imbalanced dataset since our initial data distribution was heavily skewed towards normal ECGs (97% of all ECGs were normal).

Azim read 2 articles on the basics of CNNs in order to get a better understanding of what they do, how they work, and how they are structured. He also read through an online course on deep learning in general alongside how to implement some of its concepts through Pytorch. Multiple articles on interpreting ECGs were also looked into in order to get an idea of which layers to use as well as how to structure them in the CNN's architecture for what the team is trying to accomplish. Azim also implemented the true positive rate and false positive rate calculations with a confusion matrix for the evaluation portion of the CNN.

SJ completed the features that were needed to be added into the GUI. In the Extraction tab, users can extract ECG graphs by browsing their file path and choosing their preferred ECG Machine Model and converting them into PNG files. In the Query tab, users can now be able to view the ECG graph from the selected patient. SJ had to implement all the features that were needed for the GUI in the first two weeks of the semester so that he could join his fellow team members and focus on the CNN portion. SJ spent a lot of time researching Convolutional Neural Network and identifying the goal/purpose of each part of building a CNN. He had to catch up with understanding the concepts of a basic Neural Network and then followed by Convolutional Neural Network.

Nicangel worked on fixing the data loader which was not returning the correct labels for the normals and abnormal ECGs. After the data loader was working properly he started to follow examples and tutorials to develop a CNN model using the torch 'nn' modules. The first implementation was using the sequential function to forward the input through the convolute layers. This had to be done because resources in the GPU were too loaded and CUDA could not run any other way. Note this was only the case for CUDA as in CPU it would work but require additional time. Also he implemented the 70/30 random data split, after this Nic focused on the learning how to visualize feature maps, as we wanted to improve the layer in the model but we could not think of what our current model looked like.

4. **Plan for the next Milestone**

Task	**Julian**	**SJ**	**Nic**	**Azim**
1. Train at least a few hundred epochs, and for every 100, calculate the confusion matrix and TPF/FPR check for improvements.	25%	25%	25%	25%

2. Vary the ratio of abnormals of the training set (look into oversampling of abnormal ECGs)	25%	25%	25%	25%
3. Generate a randomly selected dataset with a 98%/2% normal/abnormal using an 80%/20% training/testing split.	25%	25%	25%	25%
4. Research and improve the model's architecture	25%	25%	25%	25%
5. Create poster for Senior Design Showcase	25%	25%	25%	25%
6. Determine the various filter sizes for our convolutional layers	25%	25%	25%	25%

5. **Discussion (at least a few sentences, ie a paragraph) of each planned task for the next Milestone:**

Task 1: In order to see improvements on the CNN model we will need to increase the number of epochs. Following the recommendations from our advisor we will fix our dataset distribution to represent the 2-3% abnormal rate from the client dataset. Once that task is completed we will increase the number of epochs for the model to run in a loop over a few hundred epochs. Every 100 epoch we will log the TPR and FPR as well as saving an image of the confusion matrix. Doing this will allow us to visualize if the model is learning over each iteration. We will plot the number of epochs in the X-axis and the TPR and FPR in the Y-axis. We will be looking at the changes in the TPR and FPR values, if we have a good model there should be a close to 100% TPR and the FPR should be minimized over each iteration.

Task 2: We need to look at varying the abnormal ratio in our data via oversampling abnormals in order to have 2, 5, 10, 15, 20… ,50% of abnormal ECGs in the training dataset, whilst observing TPR and FPR in order to determine if the model is improving. This increase in abnormal ECGs can be done by duplicating the current 2%. To validate the improvements of the model with the different ratio we will need to run our model over a few hundred epochs as explained in task 1.

Task 3: For our dataset, we will be creating a randomly selected 500 dataset out of the 12000 dataset with the rate of 98% for normals and 2% for abnormal to avoid getting the same dataset. We will need to check and maintain the distribution of normal vs. abnormal which

should be roughly 2-3% abnormal. The data split for training and testing will be 80% and 20% because that is what our client wants it to be.
Task 4: As we have mentioned we do not have a good model that can predict normalities and abnormalities as it is only 4 convolutional layers. We still need a better understanding of how to implement different layers that can improve the model. We also need to check our layer parameters, as this is another major factor that determines the model's success.
Task 5: For the Senior Design Showcase, we are required to create a poster of our project. We might use Canva to create our poster and follow the suggested tips and tricks to create an effective poster.
Task 6: In order to properly capture the features in our ECG images, we need to determine the proper filter size. We would start by determining the size of each individual peak and trough of a heartbeat as well as the size of one whole heartbeat. These would compose the first two convolutional layers of our model, with subsequent layers being composed of multiple heartbeats. We will need to experiment with these different sizes to determine what works best for our model.

6. **Date(s) of meeting(s) with Client during the current milestone:**
    - 01/21/2022
    - 02/11/2022

7. **Client feedback on the current milestone:**
    1. Try limiting age range (14-17, 14-18)
        a. Could also try limiting other patient param.
    2. Time markers for non-cardia machines (out of scope/ possible after semester implementation)
    3. Just identifying normals as a starting point
    4. Schedule meeting w/ Chan and Klynton to discuss direction of project
        a. How many models?
        b. Workload
        c. Types of neural networks
        d. Ecg image vs numeric data
        e. Normal/abnormal vs normal/other
    5. Publication of results?
    6. Create Requirements file so client can run our code
8. **Date(s) of meeting(s) with Faculty Advisor during the current milestone:**
    - 01/31/2022
    - 02/14/2022
9. **Faculty Advisor feedback on each task for the current Milestone:**
    1. Use a simple tutorial to understand convolution and pooling layers and some of the parameters for them
        a. Next time ask what pooling layer means, how and why
    2. Set up the pipeline (training, test and evaluation)
    3. https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

4. Fix the current confusion matrix to look like this:

	Predicted: Abnormal	**Predicted:** Normal
Actual: Abnormal	True Positive	False Negative
Actual: Normal	False Positive	True Negative

5. Randomly select a 50/50 dataset from the cleaned data, and stick with the same initially randomly selected 500 rows of data.
    a. Check the distribution of normal vs. abnormal, should be roughly 2-3% abnormal.
    b. 70/30 or 80/20 data split for training/testing
6. Vary the abnormal ratio in the training set
    a. Increase : 2, 5, 10, 15, 20… ,50 of abnormal in the training dataset
    b. Duplicate the abnormal to achieve the desirable abnormal/normal ratio
7. Measure TPR and FPR, which there's usually a tradeoff between the two; as TPR increases so does FPR.
    a. The desirable TPR is 100%
    b. Minimize FPR
8. Train at least a few hundred epochs, and for every 100, calculate the confusion matrix and TPF FPR to see if there's improvement. X axis num epochs, y axis TPR FPR. This will help us see if it's improving.
9. Research what determines filter size by looking how many pixels represent the different parts of the heartbeat.
    a. how many pixels capture individually the 4 stages of a heartbeat. Maybe ¼ of the heartbeat is the filter size for the first layer
    b. for the next layer the filter size would be an entire heartbeat.
    c. Posterior convolutional layers would have filter sizes that encompass this pattern (they would then encompass multiple heartbeats).
    d. Draw a picture and take into account stride in order to determine each filter size.

Faculty Advisor Signature: _____ Date: _____

10. Evaluation by Faculty Advisor

Julian	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

SJ	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Azim	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Nicangel	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: _____ Date: _____