NEURAL NETWORKS FINAL PROJECT

UNIVERSITY OF GRONINGEN

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

Detection and Classification of Pathological Heartbeats from ECG Recordings

Authors:

Anna Gumenyuk
s3893464

Charlotte Hessels
s4236696

Zeynep Metin
s4236599

Mara Nedelcu
s4245350

Date: September 5, 2023

Abstract

ECG recording is a life-saving diagnosis tool that is used in detection of arrhythmia -abnormalities in heart's rhythm. Due to its challenging nature, ECG-reading has been done by experienced doctors. However, recent developments in the area of Neural Networks prove that machine learning models trained on ECG-recordings are now able to detect and classify present arrhythmias more accurately than trained cardiologists. Similarly, our aim in this paper is to create and train a 1-Dimensional Convolutional Neural Network (CNN) that can classify pathological heartbeats, based on data given in the MIT-BIH dataset. We used Accuracy, Loss, Recall, Precision and the F1 Score in order to assess the performance of our model. The results suggested that our model performed significantly better on classification of Normal heartbeats than other types of arrhythmias.

Keywords: Classification; CNN; Arrhythmia

Contents

1 Introduction

Electrocardiogram (ECG) is a diagnostic tool used in the analysis and determination of heart-related problems, and works by measuring the electrical activity in the heart through the electrodes placed on the patient's body. The classification of measured heart rhythms plays a crucial role in the diagnosis of the patient, and the healthcare procedure that follows after. However, due to its frequent use and its dependency on human interpretation, understanding these heartbeats, and detecting any arrhythmia -an abnormality in the heartbeat rhythm- can be a costly and time-consuming practice.

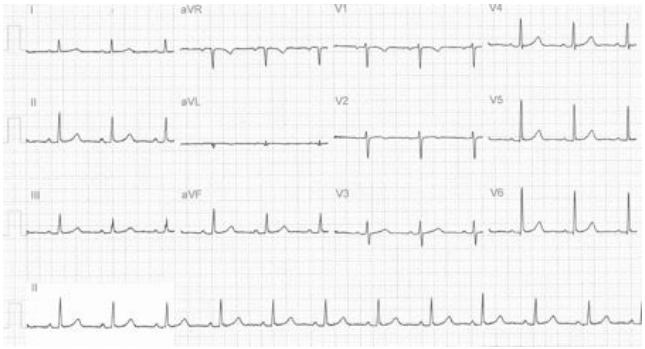

Figure 1: A typical ECG recording (Dekie, 2017)

Therefore, in order to speed up this procedure and make it more accessible, there has been much research done in this area to automatize the classification of ECG readings. An example of the research done in this area is described in the paper written by Rajpurkar et al., who trained a 34-layer Convolutional Neural Network (CNN) that can detect and classify 12 different types of arrhythmias, in addition to normal heartbeats, in an ECG recording. The results of this study show that the trained CNN in fact performed better than a certified cardiologist in the classification of the ECG input. CNN's are a type of neural network that are generally preferred in the detection and classification of images due to their high accuracy. However, they can also be used in 1D signal classification as they are able to learn features from input data in a one-dimensional format, such as time series data. CNNs are able to learn these features by convolving the input data with a set of filters which extract certain features from the data.

Another example of automatized arrhythmia classification is described in the paper written by Hadaeghi, who modelled an Echo State Network (ESN) -an instance of Reservoir computing- to detect and classify different types of abnormalities in heart rhythms. This ESN was trained separately for each patient, making it more accustomed to detecting arrhythmias specific to each patient. In contrast to a model that is trained on data from all patients, a model customized to the patient requires fewer data and makes it possible to adapt the model to the unique features of ECG recordings of each patient.

Inspired by these pieces of research, we have decided to create and train a CNN with multiple layers: as applying transfer learning proved to be a challenge considering our lack of practical experience in this subject, and the time constraints, all layers will be trained on data gathered from all patients in the data set. We will be using MIT-BIH dat aset as it is one of the most commonly used ones in this subject. The data set consists of 30-minute ECG recordings from 47 patients, with every heartbeat annotated with a letter representing the type of rhythm. Detailed information about the data set is given in the next section. Finally, we will compare our results with the model described in paper written by Hadaeghi, and analyze the performance of our model.

2 Data

We decided to use the data that was proposed in the Semester Project Suggestion for classifying pathological heartbeats in our model. This data set, the MIT-BIH arrhythmia database, was accessed from (Moody & Mark, 2001) and maintained by Goldberger et al., which we studied to gain a deeper understanding of what the data consists of and used in our model.

The data we used contains 48 Electrocardiogram (ECG) recordings of 47 patients. Each recording is 30 minutes in total, meaning 1800 seconds of ECG data per recording. Since the recordings were sampled at 360 samples per second, there are 648000 samples per ECG recording available. As mentioned before, the ECG data from 47 patients was used. Around half were randomly selected from a set of ECG recordings taken from a mix of in- and outpatients, and the other half was selected from the same set in order to assure that different types of arrhythmias were all represented (Goldberger et al., 2000).

During an ECG, electrodes are placed on the skin of a patient, which records the electrical signal from the heart. This electrical signal is expressed in millivolts (mV). There are different ways of graphically describing the electrical activity in someone's heart and it is often done by analyzing different groups of electrodes, which are called 'leads' (*The ECG leads: electrodes, limb leads, chest (precordial) leads, 12-Lead ECG (EKG) – ECG & ECHO*, 2017). There are different leads to describe the electrical activity of a patient's heart, but the data we used describes the MLII lead and the V5 lead. Each sample in the data, therefore, consists of two values, one describing the voltage measured by the MLII lead and the other describing the V5 lead. We plotted the heartbeats together with the given annotations to gain a better visual intuition of the data. This is shown in Figure 2.

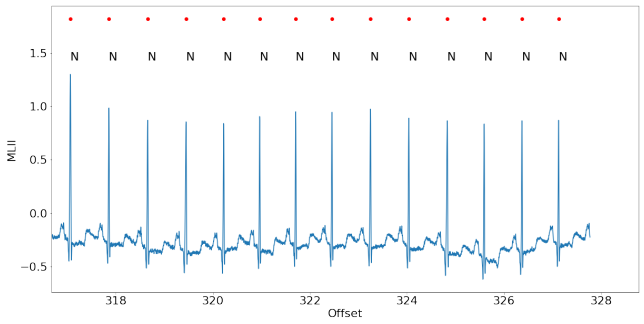

Figure 2: The heartbeat patterns with their corresponding classification in the MLII channel.

As can be seen in Figure 2, the heartbeats of a person become visible. In this project, we are aiming to classify these heartbeats. Therefore, it is important to understand exactly what the samples consisting of this heartbeat represent and what elements make up a heartbeat.

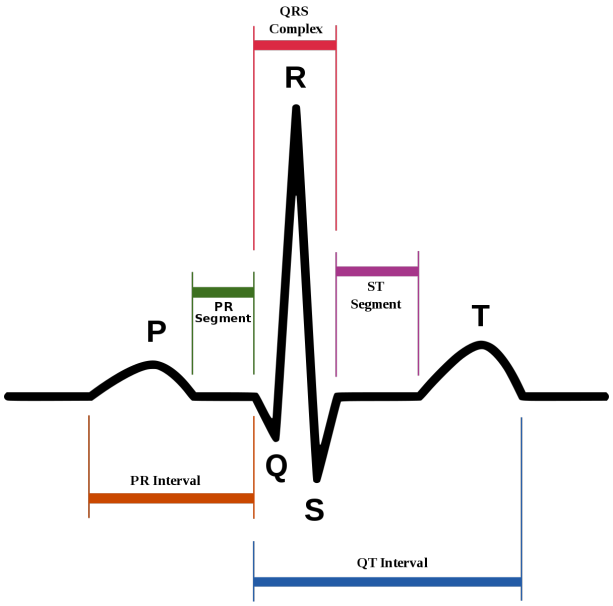

Figure 3: Normal QRS complex (Atkielski, 2007)

An important aspect of interpreting the data from an ECG is the QRS complex, which is one of the most prevalent features seen on an ECG as part of the heartbeat of a person. A visualization of a QRS complex is seen in Figure 3.

The peak of this QRS complex, the R-peak, is used to define the sample windows over which a heartbeat is classified. Each of these CSV files is accompanied by a text file which consists of the different annotations that cardiologists made and reviewed. These annotations include beat labels that were given to each heartbeat to indicate if it was normal or of which abnormal type the beat was.

Having this understanding of the data helped us to determine the next steps in designing the model. However, before the data could be used, some pre-processing steps were necessary, which will be described below.

3 Methods and Experiments

3.1 Preprocessing

Inspired by the preprocessing steps outlined by Kachuee, Fazeli, and Sarrafzadeh (2018) and applying the methods of Aerts (2019), the data was preprocessed as follows:

1. Instead of using the annotations to split MIT-BIH records into individual heartbeats, we looked for R-peaks which are the maximum amplitudes of the R wave in the QRS complex. We used this method because it could also be applied to the ECG data that was not annotated. hence, the annotations were used only to classify each beat as one of the beat types mentioned in the research by Kachuee et al. (2018). The table 1 summarizes the mappings between beat annotations and AAMI EC57 categories.

2. In order to cover a complete QRS complex, each heartbeat also included the first 40 readings of the upcoming heartbeat.

3. Every heartbeat was resampled to $125Hz$ as suggested by (Kachuee et al., 2018). While the reason for this choice of frequency was not specified, we assume that it is due to the conventional ECG frequency limitations that range from $0.05Hz$ to $150Hz$ (Tragardh & Schlegel, 2006).

4. As suggested, the values of the readings were normalized in the $0-1$ range.

5. As described by Aerts (2019) each heartbeat was then presented as an array of 187 values (or, a row vector $\mathbf{x} \in \mathbb{R}^{187}$). The heartbeat records longer than 187 entries were discarded. The heartbeat records that were shorter than 187 were padded with zeroes so that they contained 187 values.

6. Then, the classification of the heartbeat record was appended at the end of the array which implies that the $\{x_{188}\}$ entry of the vector was the classification of the heartbeat. Which value corresponds to which classification category can be found in Table 1. This resulted in a heartbeat information being represented as vector $\mathbf{x} \in \mathbb{R}^{188}$ with entries $[x_1, \ldots, x_{187}]$ denoting the actual heartbeat recordings and $\{x_{188}\}$ being the classification of that heartbeat.

7. Heartbeats that could not be classified based on the existing categories were discarded.

8. The heartbeats were then stored in a comma-separated values (CSV) file for each patient. They were stored in a matrix $M \times 188$ where M is the number of beat recordings that could be obtained from the records of the participant. In total, we have obtained 96 CSV files representing 48 recordings per each channel, V5 and MII.

Category	Annotation	Value
N	Normal Left/Right bundle branch block Atrial escape Nodal Escape	0
S	Atrial premature Aberrant atrial premature Nodal premature Supra-ventricular premature	1
V	Premature ventricular contraction Ventricular escape	2
F	Fusion of ventricular and normal	3
Q	Paced Fusion of paced and normal Unclassifiable	4

Table 1: The table displaying the heartbeat classifications and the corresponding value label for the test and training data. An explanation: for example, if the beat is annotated as nodal premature, it will belong to the category labelled as S and its classification value is 1.

After these preprocessing steps, we visualized the data again. The Figure 4 displays the examples of normalized classified heartbeat recordings.

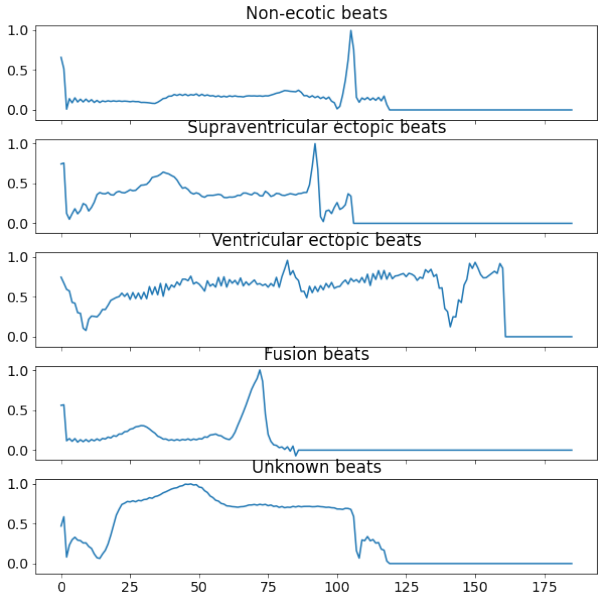

Figure 4: Normalized beat types.

3.2 Learning architecture

To solve our learning task, namely the accurate classification of the pathological heartbeats, we decided to use a one-dimensional Convolutional Neural Network, as suggested in the ECG project proposal. We used a CNN since it is refined to classify data and extract important features, which is what we aim to do in this project. We built our model using Keras in Google Colab, and started with setting up a sequential model. This means that every layer that we add to the model has one input and one output. Next, we will describe all layers we added to the model. Some explanations are based on the developer guides from Keras (Keras, n.d.).

The first layer we add is a one-dimensional convolutional layer that 'slides' a kernel of size 5 over the input sequence. The rectified linear unit activation function is used in this layer, which was discussed in the course. One advantage of this activation function is that it is less computationally expensive than, for example, the sigmoid function. This activation function is used for later layers in the model as well. The filter size of the 1D convolutional layer is set to 64, which means that 64 output convolutions will be created.

After this, a pooling layer is added to reduce the number of parameters that need to be learned by reducing the dimensions of the feature maps that were created (GeeksforGeeks, 2021). The exact pooling layer we used is the 'Average Pooling Layer 1D' from Keras. This pooling layer computes the average elements of the feature map with a pooling window size that we set to 2.

We then added a flattening layer in order to convert the output of the convolutional layer to one feature vector (Jeong, 2021).

The fourth layer is a dense layer. This layer is fully connected to the layer before it, the flattening layer, and has a size of 64 and again uses the rectified linear unit activation function. After this dense layer, a dropout layer is added. In this dropout layer, some input units are randomly set to 0 with a rate of 0.5 in order to prevent overfitting.

After this, we add two final dense layers. The first again uses the rectified linear unit activation function, but this time with a size of 32. The second dense layer functions as the output layer by using the softmax function. This converts all the obtained values to a probability distribution of size 5. This means that when the model is given an input, 5 probabilities (or a probability vector) will be outputted and each probability (or each entry of the vector) will correspond to one of the classifications that can be given to the heartbeat.

3.3 Loss Function

As our loss function, we used the categorical cross entropy function. This function is used when there are two or more labels, which is the case in this project. The loss is calculated by the sum

$$L = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i, \tag{1}$$

where N is the total output size, y_i is the i'th target output and $\hat{y}_i$ the i'th model output (*Categorical crossentropy loss function — Peltarion Platform*, n.d.).

3.4 Optimization

For understanding and explaining the Adam optimizer, we mainly used the sources (Ruder, 2020a) and (GeeksforGeeks, 2020).

The optimization algorithm that we used in our model to optimize the gradient descent is the Adam optimizer. The default learning rate of 0.001 was used, as well as the default values for β_1 and β_2 being 0.9 and 0.999 respectively. The Adam optimizer calculates the 'decaying averages of the past and past squared gradients' (Ruder, 2020b). Below are the estimates of the means and uncentered variance of the gradients, m_t and v_t,

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3}$$

where β_1 and β_2 are constants (0.9 and 0.999 respectively), t is the current time-step and g is the gradient at time-step t.

These estimates are used to calculate the bias-corrected versions of both, as seen in equations (4) and (5).

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{4}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{5}$$

Now, combining these equations discussed above, the final Adam update equation is given below.

$$\theta_{t+1} = \theta_t - \frac{l}{\sqrt{\hat{v}_t} + \varepsilon} \tag{6}$$

Here, θ are the model's parameters at time-step t, l is the learning rate (set to 0.001) and ε is a constant that is set by default to 10^{-8} in order to prevent divisions by zero.
The Adam optimizer is generally faster and still effective even for bigger data sets compared to other optimization algorithms and was therefore used in our model (Brownlee, 2021).

3.5 Training Procedure

Before training the model, the data was split into three sets: training, validation, and testing. First, the data was split in the following way: 80% of the data was selected for the training of our model, and the remaining 20% for testing purposes. The validation set was generated from the training set by setting aside 20% of the training set for the validation process.

Before generating the validation data set, when examining the training data, it was found that some of the classification categories were undersampled: most of the heartbeats were classified as normal (Figure 5). As such, we augmented each category to 20000 samples using resampling with replacement, so each classification category counted an equal number of sample beats. As a result, in total the training set contained 100000 (a hundred thousand) samples, with five classification categories counting 20000 samples of each. Then, since 20% of the training set was turned into a validation set, the training set was reduced to 80000 samples.

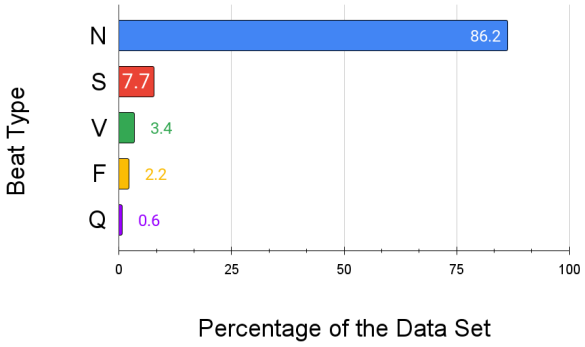

Figure 5: The distribution of the beats in the training set. N corresponds to Normal heartbeats, S - to Premature beats, V - to Ventricular abnormalities, F - to Fusion beats, and Q - to paced/unclassifiable beats.

Formally, it gives a training set

$$S = (\mathbf{u}_i, \mathbf{y}_i)_{i=1\ldots N}$$

where $\mathbf{u}_i \in \mathbf{R}^{187}$ is a heartbeat record, $\mathbf{y}_i \in \mathbf{R}$ is a classification of the heartbeat, and N is the number of training samples, which was 80000 in our case.
Then, the validation set is

$$V = (\mathbf{u}_i^{'}, \mathbf{y}_i^{'})_{i=1\ldots N}$$

where $N = 20000$.
Finally, the testing set can be represented as

$$T = (\mathbf{u}_i^{test}, \mathbf{y}_i^{test})_{i=1\ldots N}$$

where $N = 29872$.

When the data was ready and the network architecture was determined as specified earlier, the network was trained through k-fold cross validation with 5 folds ($k = 5$) as it is suggested to be one of the common choices of the number of folds (Renbo Olsen, 2021). In every fold, we were looking at such performing measures as loss and accuracy. When the training of the network was over, the results of the training were saved after each fold for later evaluation. The hyperparameters (such as learning rate, optimizer, etc.) of the network remained constant during the whole process of training. The parameters that were tuned are the connection weights.

We had to make sure that the network would not run out of data during the training, so it was trained for 25 epochs with a step size of 2500 with a batch size of 32.

To set up the condition to terminate the training we used an **EarlyStopping** function that would monitor the loss

value with a patience level of 8, meaning that the training would stop if the loss value was no longer decreasing for 8 consecutive epochs. The model that performed the best was then saved and its performance was analyzed based on the criteria that will be discussed in the following section.

4 Results

In evaluating the goodness of our model we have employed the accuracy, f1 score, precision, recall and loss of the model. The evaluation has been done both through the validation set prepared before the training, and the k-fold cross validation, as well as the test set that was created with the aim of ultimately testing the performance of our ECG classifier. The generated Confusion Matrix (Figure 6), summarises the overall fitness of the classifier.

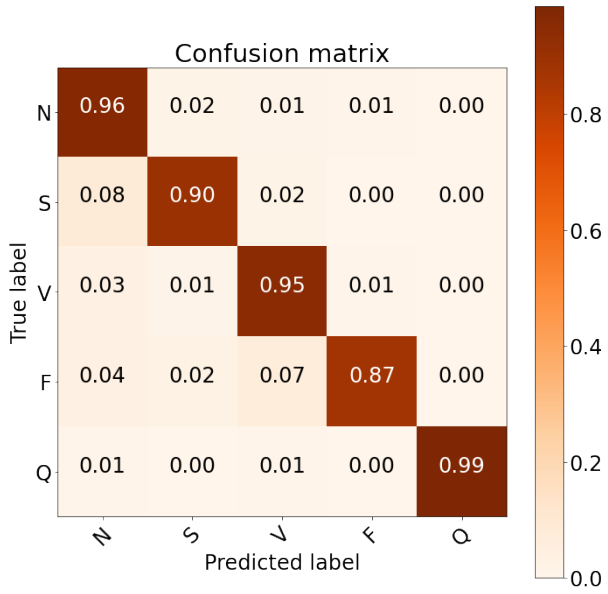

Figure 6: The Confusion Matrix of the classifier describing its overall performance

4.1 Accuracy

Accuracy, alongside the loss, are the de facto standard metrics when it comes to assessing the fitness of a classification model. Thus, it was only fit to evaluate our model based on accuracy. As such, our model reached an accuracy of 0.99 when tested on the validation data set prepared before the training and the k-fold cross validation, and an overall accuracy of 0.96 for the actual test data set.

4.2 Loss

The loss function that we have used to evaluate the fitness of the model computed the cost of mismatch between the expected value and the actual output as 0.03 for the validation data set prepared before the training and the k-fold cross validation, and an overall loss of 0.16 for the actual test data set.

4.3 Recall

Recall depicts the proportion of the actual positives of the data which was accurately identified by the classifier. Hence, recall is calculated as the number of true positives divided by the sum of true positives and false negatives. Upon assessing the performance of our model, we have found that the recall for Normal heartbeats was of 0.96, 0.90 for the Premature beats, 0.95 for the Ventricular abnormalities, 0.87 for the Fusion of Ventricular and Normal beats, and eventually 0.99 for the paced/unclassifiable beats. These values can be assisted on the main diagonal of generated Confusion Matrix. Lastly, the macro recall average is computed to be 0.93, while the weighted recall average is of 0.96.

4.4 Precision

The precision of a classification model shows what proportion of positive identifications was, in fact, true to its form. As such the precision of the model is calculated as the total number of true positives, divided by the sum of true and false positives. Looking at the precision of our model, as seen in Figure 7, we have achieved a precision of 1.00 for the classification of Normal heartbeats, 0.57 for the Premature beats, 0.69 for the Ventricular abnormalities, 0.50 for the Fusion of Ventricular and Normal beats, and eventually 0.97 for the paced/unclassifiable beats. Overall, our model achieved a macro precision average of 0.75, and a weighted precision average of 0.97. Therefore, it can be said that the model correctly classifies the heartbeats 0.97 percent of the time.

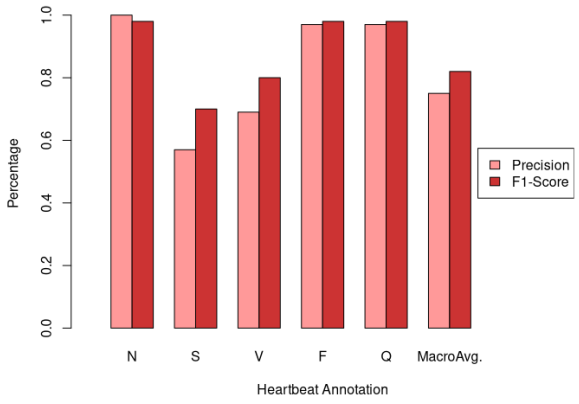

Figure 7: Barplot visualizing the Precision and F1 Score for each predicted heartbeat label.

4.5 F1 Score

The F1 score combines the two metrics of Precision and Recall into one, and it works especially well on imbalanced data sets, which is the case for our testing data set. By conveying the balance between precision and recall, the F1 score is calculated by computing the harmonic mean between Precision and Recall. As such, when looking at Figure 7, it can be seen that the F1 score for the Normal beats is 0.98, for the Premature beats it is 0.70, 0.80 for the Ventricular abnormalities, 0.63 for the Fusion of Ventricular and Normal beats, and eventually 0.98 for the paced/unclassifiable beats. Eventually, the macro f1 score average is 0.82, while the weighted F1 score average is of 0.96.

5 Discussion

5.1 Results Analysis

The goal of our project was to compute a model that classifies, as accurately as possible, pathological heart beats. In training the model we have used the MIT-BIH data set, and we have attempted to prevent overfitting using 5-fold cross validation, to eventually test the fitness of our model on a testing data set, that has been extracted from the main data set before starting the model training.

In testing the goodness of our model we have evaluated the overall accuracy of the model, the precision, recall and f1 scores of the model's performance on each of the 5 data categories, as well as the overall loss.

The overall high accuracy of our model of 96%, depicts its general good performance. Consequently, the computed accuracy of our model proves to be higher than the one captured by Kachuee et al. (2018), who reached a 93.4% classifier accuracy. Thus, it can be concluded that, while our classifier seems to reach high performance, its accuracy may be somewhat too high, pointing towards a slight tendency towards overfitting. It seems that, against our best efforts in employing 5-fold cross validation, our classifying model still reached some low level of overfitting.

When looking at the computed values of our chosen loss function, we can see quite a stringent difference between the emergent values from the validation data set and the testing set. This difference between the two is to be expected from a classifier, nonetheless, the reason for this occurrence is still somewhat unclear to us. We assume that the main culprit is the unbalance found within the testing data set, seeing as the undersampled categories in this set have not been previously balanced or augmented, with the majority of the data consisting of Normal heart beats. This, of course, leaves room for improvement on the matter. Hence, with a loss of 0.16, the model scored worse when classifying the data in the testing set, which points toward a more natural behaviour of the model.

Focusing on the precision, recall and f1 score of the five distinctive categories found in the testing data set, we can underline some differences between them. It seems that the greatest discrepancies can be observed in the precision, with the Normal heartbeats being classified with the highest precision of 100%, while the worst precision could be observed for the Fusion of Ventricular and Normal beats, of only 50%. The recall scores are, however, more balanced, with the highest difference between two classes of only 12%. All in all, the f1 scores describe the fitness of the model best, as we are dealing with an imbalanced data set. Summarising the exactness and completeness of our model, the f1 scores of each of the classes highlight the adequate performance of our model in the classification of Normal and Paced/Unclassifiable beats. In contrast, the Fusion of Ventricular and Normal beats seems to be the class in which our model performs worst, with a f1 score of 0.63.

5.2 Reflection

Implementing a classifier for ECG data proved to be a true challenge both in regards to the amount of background research needed, as well as the actual algorithm-development process.

It seems that the initial milestone that needed to be overcome was properly grasping the meaning behind the data that we set out to work with. As we have learned, ECG analysis is a challenge in itself, even for the trained eye of a medical professional. Thus, training a neural network to classify this type of data meant for us to initially comprehend what each nook and crannie of an ECG depicts about a patient's heart beat.

Upon understanding the actual meaning behind the data set the next step taken in the creation of our model was the preprocessing of the data, which proved to be no easy task in itself. Considering our lack of experience in preprocessing medical data, combining our coding knowledge in python with the issue at hand showed its mettle as challenging. Aside from the issues we encountered when dealing with the various versions of python code, that we seemed to have an ongoing battle with, while dealing with the raw data, the preprocessing was otherwise somewhat smooth sailing. Even if we were provided with an already preprocessed batch of data, we decided that doing the preprocessing ourselves will bring us one step closer to properly grasping the full extent of what building and training a neural network is, in fact, about.

Thus, even if it proved itself a challenging endeavour, preprocessing, was in fact a fruitious experience that helped us understand the data set that we were working with better, Even if various python versions of code were, indeed, some of the setbacks that we encountered along our coding journey, our main problems arose with our decision to use Google Collab Notebooks as our coding platform. Seeing as we were to work in a group, we took the collective decision to work on our code using Google Collab Notebook, as the code was supposed to update in real time, as the others were developing it, and we could work using multiple devices at the same time. Nonetheless, Collab did not bring to the table what it promised. Its execution was consistently lagging

when more than one person was working on the notebook, and we had to run the notebook repeatedly on each of our laptops, as the data would often be inexplicably lost from one person to the next, as well as one coding attempt to the next. Apart from being time consuming, this feature rapidly became bothersome, so we decided to simply just work solely on one computer.

Even though in the beginning we assumed that building and training the model would prove itself to be the most challenging part of our project, we were pleasantly surprised to see that this part was, in fact, the most enjoyable and rewarding of them all. Observing the model training itself on the provided data, to eventually give overall satisfying results upon testing, was a truly fulfilling experience. Wistfully, there is one meager setback that we encountered when testing the model. To be more precise, the results provided by the testing data set were not as consistent as we would have hoped with the ones emerging from the validation data. This may have been caused by the manner in which we have dealt with the preprocessing of the data.

Nevertheless, looking beyond the few setbacks encountered along the way, the experience of building, training and testing a neural network, was truly fruitful and rewarding, both in the gained knowledge, as well as the final results that we have eventually obtained.

5.3 Future Research and Potential Improvements

One of the papers discussed previously, specifically the research conducted by Kachuee et al. (2018), mentions the use of transfer learning as a method of improving the detection and classification performance of a model. Accordingly, they touch upon customizing the classification algorithm, in order to tune the model on the individual features of the ECG recordings gathered from the patients. Particularly, they aimed to initially train the model on one data set, to concurrently transfer the knowledge to a different one. This process has successfully proved that the knowledge learned from the training on the former set can be successfully transferred to a different but related learning task. Due to time constraints, we were not able to work on integrating extra layers that would be trained on data from certain patients and implement transfer learning. However, we believe that further making use of this practice would improve the applicability of our model.

Furthermore, we trained the model with a fixed architecture and constant hyperparameters. This implies that we did not tune the number of layers or neurons in the layer, the learning rate, optimizer, loss function, or other explicitly defined parameters. Thus, a potential improvement would be to try out various combinations of different parameters and determine which one outperforms the others. Hence, there is a high chance of different parameters being better suited for the task at hand. Therefore, a further tuning of hyperparameters may prove to be a wonderful starting point for additional research on the ECG classifier development scene.

Acknowledgements

We mean to note the aid that we have received in the completion of this project from the additional sources that have helped in the implementation of our architecture, as mentioned in the code itself. Moreover, we thank prof. dr. Herbert Jaeger for the offered guidance in understanding the needed material for the smooth sailing of implementing the semester project.

References

Aerts, K. (2019). *Ecg machine learning.* Retrieved from https://github.com/koen-aerts/ECG_ML

Atkielski, A. (2007, 01). *QRS Complex.* Retrieved from https://en.wikipedia.org/wiki/QRS_complex

Brownlee, J. (2021, 01). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning.* Retrieved from https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Categorical crossentropy loss function — Peltarion Platform. (n.d.). Retrieved from https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy

Dekie, L. (2017, 07). *High Quality ECG Recording Assurance.* Retrieved from https://www.appliedclinicaltrialsonline.com/view/high-quality-ecg-recording-assurance

The ECG leads: electrodes, limb leads, chest (precordial) leads, 12-lead ECG (EKG) – ECG & ECHO. (2017, May). https://ecgwaves.com/topic/ekg-ecg-leads-electrodes-systems-limb-chest-precordial/. ECG & Echo Waves. (Accessed: 2022-6-28)

GeeksforGeeks. (2020, 10). *Intuition of Adam Optimizer.* Retrieved from https://www.geeksforgeeks.org/intuition-of-adam-optimizer/

GeeksforGeeks. (2021, 07). *CNN — Introduction to Pooling Layer.* Retrieved from https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/#:%7E:text=Why%20to%20use%20Pooling%20Layers,generated%20by%20a%20convolution%20layer.

Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., ... Stanley, H. E. (2000, June). PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation*, *101*(23), E215–20.

Hadaeghi, F. (2019). Reservoir computing models for patient-adaptable ecg monitoring in wearable devices. *arXiv preprint arXiv:1907.09504*.

Jeong, J. (2021, 12). *The Most Intuitive and Easiest Guide for Convolutional Neural Network.* Retrieved from https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480#:%7E:text=Rectangular%20or%20cubic%20shapes%20can,a%20single%20long%20feature%20vector.

Kachuee, M., Fazeli, S., & Sarrafzadeh, M. (2018). Ecg heartbeat classification: A deep transferable representation. In *2018 ieee international conference on healthcare informatics (ichi)* (pp. 443–444).

Keras. (n.d.). *Keras documentation: Developer guides.* Retrieved from https://keras.io/guides/

Moody, G. B., & Mark, R. G. (2001, May). The impact of the MIT-BIH arrhythmia database. *IEEE Eng. Med. Biol. Mag.*, *20*(3), 45–50.

Renbo Olsen, L. (2021). *Multiple-k: Picking the number of folds for cross-validation.* Retrieved from https://cran.r-project.org/web/packages/cvms/vignettes/picking_the_number_of_folds_for_cross-validation.html#:~:text=When%20performing%20cross%2Dvalidation%2C%20it,common%20to%20use%2010%20folds

Ruder, S. (2020a, 03). *An overview of gradient descent optimization algorithms.* Retrieved from https://ruder.io/optimizing-gradient-descent/index.html#gradientdescentvariants

Ruder, S. (2020b, 03). *An overview of gradient descent optimization algorithms.* Retrieved from https://ruder.io/optimizing-gradient-descent/index.html#adam

Tragardh, E., & Schlegel, T. T. (2006). High-frequency ecg.