A Deep Learning Approach to ECG Heartbeat Categorization

Amaury Chauvin | Matthieu Thienpont | Oscar Levy | Ruan Van Schalkwyk | Valentin Foucault

Background

This project explores how different neural network models perform heartbeat classification using an ECG Heartbeat Categorization Dataset on Kaggle.com (Fazeli, S. 2018). It is comprised of two well-known collections of signals, the BIT-MIH Arrhythmia Dataset (MIT.edu., 2020) and the PTB Diagnostic ECG Database (Bousseljot, R., D. Kreiseler, and A. Schnabel, 1995), each split into five and two categories, respectively. We will refer to these as MITBIH and PTBDB from this point onwards.

An ECG (ElectroCardioGram) is a recording of electrical activity in the heart which provides information on cardiac problems such as arrhythmia. Developing an effective heartbeat classification model can be considered essential because it allows for rapid diagnosis of heart-related complications, assisting medical professionals in making potentially life-saving decisions.

Motivation

The main motivation for this project revolves around the creation of a diagnostic toolset. Applying Machine Learning and creating domain specific intelligence to a critical medical problem would therefore constitute a valuable approach. The technology would enable medical professionals to detect and classify the different arrhythmia characteristics in patients. Such an application would bring speed and efficiency to the process and in the case of good model performance, better diagnostics. Outsourcing part of the medical process in times of emergency such as the recent Covid-19 Pandemic. This would further allow for medical staff to be more reactive and augment general infrastructure capacity.

Datasets

Collection	Samples per category	Total Samples
MIT-BIH Arrhythmia	Normal - 72471	109446
	Supraventricular - 2223	
	Ventricular - 5788	
	Fusion - 641	
	Q (Unclassifiable) - 6431	
PTB Diagnostic ECG	Normal - 4046	14552
	Abnormal - 10506	

Table 1: List of the datasets

The collections contain time series data, where each sample is a segment corresponding to a single heartbeat as represented by an electrocardiogram. The MITBIH dataset is composed of 5 classes, where N is normal heartbeats and S, V, F, Q are subcategories of different arrhythmias and myocardial infarctions. The PTBDB dataset just contains two general classes for Normal and Abnormal beats.

Considering the difference in class number, models were initially built around the two collections separately, partially to preserve the level of granularity in the MITBIH dataset. The types of models used to tackle the classification problem at this stage include a standard ANN, CNNs, and an LSTM, and are described further in the Method section. Transfer learning was also used as a second approach, which required collapsing the four abnormal categories of the MITBIH dataset into one but allowed a single model to cover both datasets with minimal additional training.

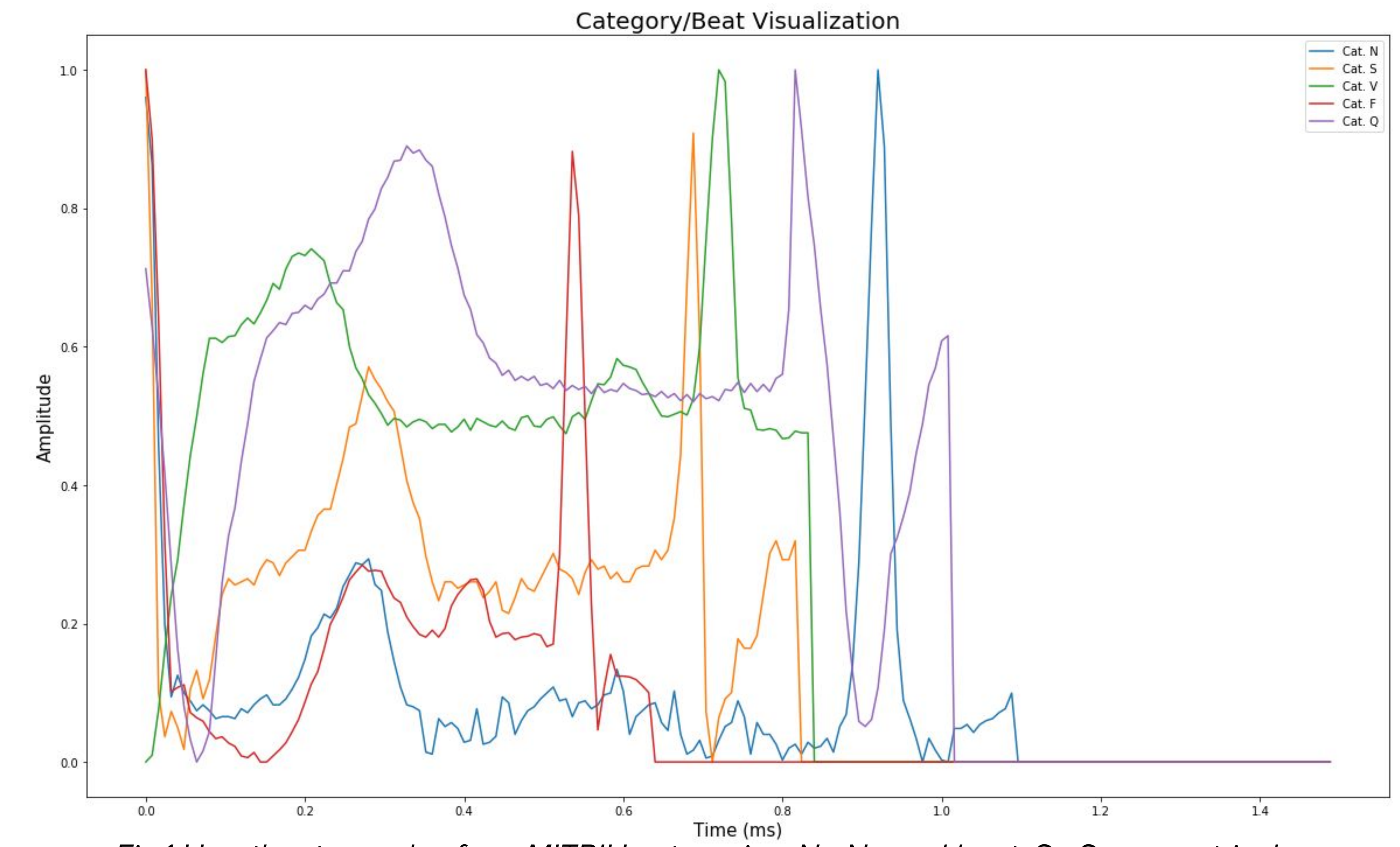

Fig.1 Heartbeat samples from MITBIH categories; N - Normal beat; S - Supraventricular premature beat; V - Premature ventricular contraction - ; F - Fusion (of ventricular and normal beat); and Q - Unclassifiable beat.

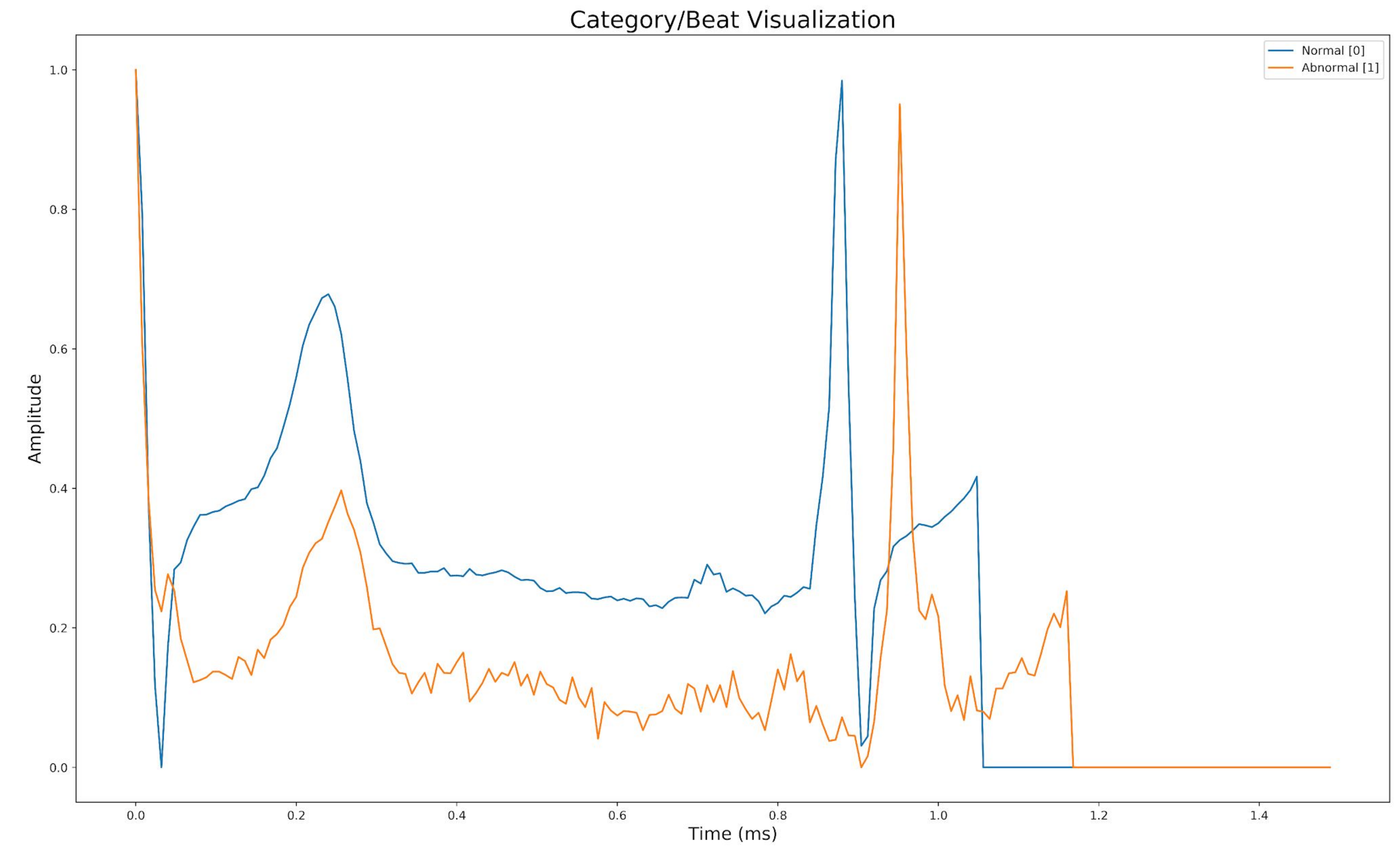

Fig.2 Heartbeat samples from PTBDB categories; N - Normal beat and A - Abnormal beat.

Data Exploration

As part of the exploration made on the data, non-linear (Spearman) and linear (Pearson) correlations were ran over the data in order to extrapolate connections between temporal pointers in the samples. The first temporal moments (upper right) had the highest covariance ratios which indicated that there was a need to focus on initial timesteps in the research.

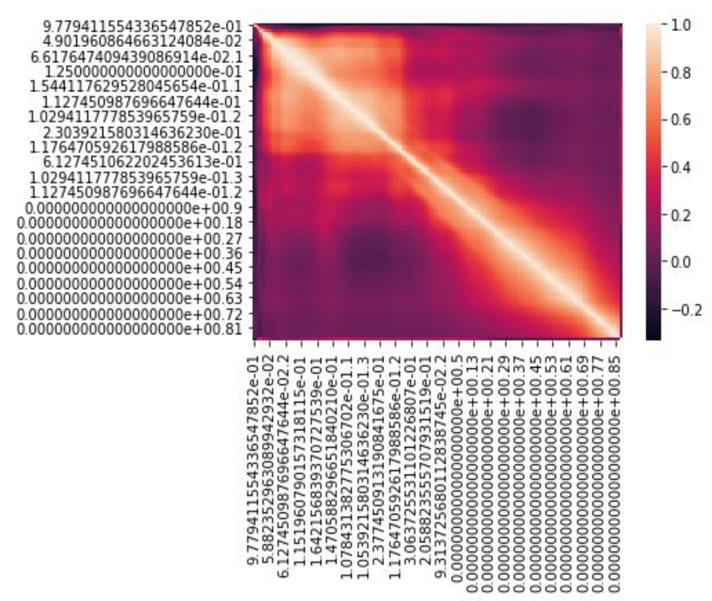

Fig.3 Spearman Correlation MITBIH

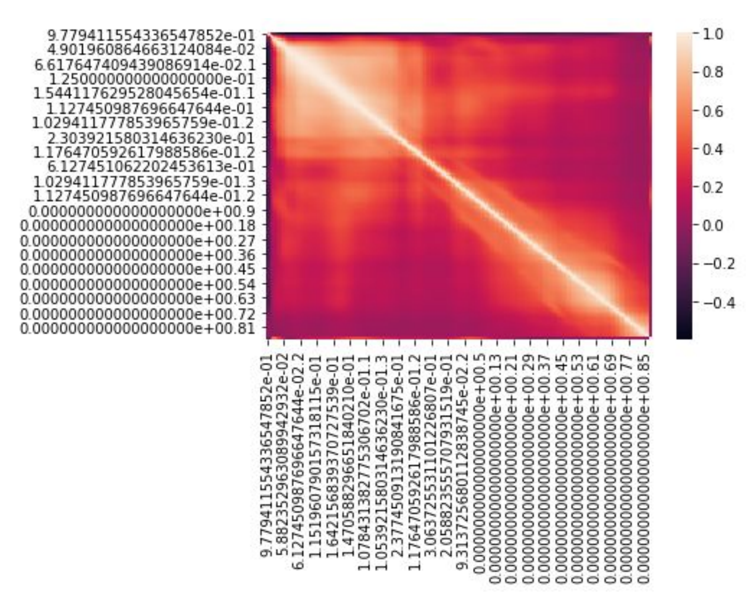

Fig.4 Pearson Correlation MITBIH

Preprocessing

The collections were provided preprocessed to a certain degree—cropped, downsampled and padded where needed—but the decision was made to upsample under-represented classes. Fig. 5 displays before and after ratios for the PTBDB dataset; the abnormal class initially contained almost three times more samples before rebalancing, which would have likely caused our models to suffer from the accuracy paradox.

Note that we considered downsampling undesirable, as too little data would be left (see table). A shuffle method was also applied to ensure models would be correctly tested and trained.

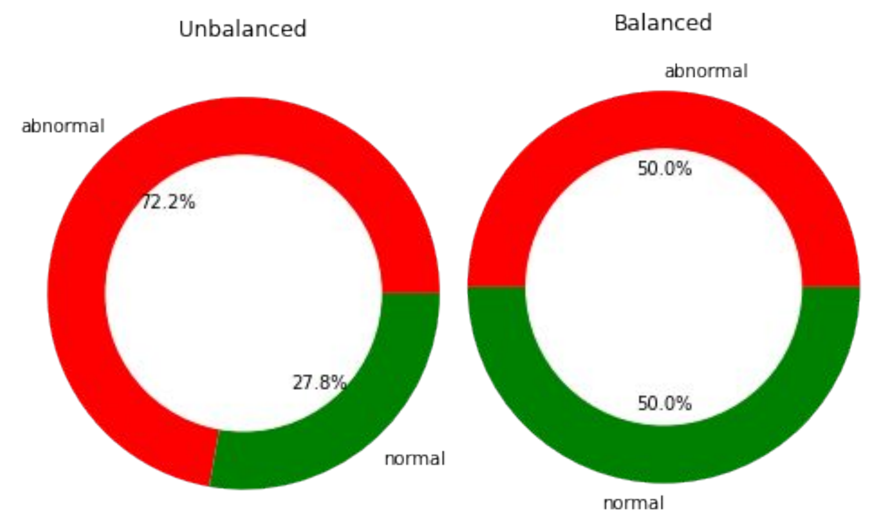

Fig.5 Rebalancing classes for PTBDB

	Norm. sample #	Abnorm. sample #
Original	10506	4046
Undersampled	4000	4000
Upsampled	10000	10000

Table 2: 1:1 ratio rebalancing options on PTBDB

Augmenting the classes served to overcome the possibility of imbalanced learning on the available data, but also curbed the possibility of overfitting. It is well known that more data translates to more stable learning.

Method

Classic methods

The following models were tested on both MITBIH and PTBDB datasets:

- **Convolutional Neural Network (CNN) - Low complexity**
- **Convolutional Neural Network (CNN) - High complexity**
- **Long Short-Term Memory (LSTM)**
- **Artificial Neural Network (ANN)**

Transfer learning (TL)

Transfer learning is another machine learning technique that focuses on storing the knowledge gained while solving one problem and applying it to a different but related problem . PTBDB trained classifiers were therefore briefly re-trained for a few additional epochs on the MITBIH dataset.

Training Considerations

1. **Overfitting**
   Throughout the course of the project we were constantly aware of the dangers overfitting a model could pose, since we are working with models that analyze crucial information related to the health of patients overfitting could be fatal. Therefore we utilised the method of early stopping for each of the models and stopping the training when there was no improvement in the loss of our models.
1. **Size of the models**
   Since we were limited on data diversity and had to rely on techniques to augment our data, we were hyper aware of the dangers that too large or small a model can pose to learning by exacerbating overfitting and underfitting respectively
1. **Design of model internals**
   While designing our models we also paid close attention to the design of the layers - this is evidenced by the inclusion of an appropriate amount of dropout and batch normalisation layers within our Convolution and LSTM models. We also make sure to design our standard feedforward neural networks with an appropriate amount of neurons in each layer.

A Deep Learning Approach to ECG Heartbeat Categorization

Amaury Chauvin | Matthieu Thienpont | Oscar Levy | Ruan Van Schalkwyk | Valentin Foucault

Model Architectures

LSTM Model

Layer (Type)	Output Shape	Number of Parameters
Dense 1 (Dense)	None, 187, 32	64
Bidirectional LSTM (LSTM)	None, 187, 128	49664
Dropout 1 (Dropout)	None, 187, 128	0
Batch Norm 1 (Batch Normalization)	None, 187, 128	512
Dense 2 (Dense)	None, 187, 64	8256
Dropout 2 (Dropout)	None, 187, 64	0
Batch Norm 2 (Batch Normalization)	None, 187, 64	256
Dense 3 (Dense)	None, 187, 5	325
Flatten (Flatten)	None, 935	0
Dense 4 (Dense)	None, 5	4680

Table 3: Architecture of the LSTM model

During the course of our project we wanted to examine the behaviour and performance of different models on our data. One of the models we were keenly interested in implementing is an LSTM.

The above diagram lays out the architecture for one of the LSTM models we chose to implement for this project. The one specified above is a less complex version consisting of only one bidirectional layer followed by batch normalization layers and dropout layers.

The initial motivation for choosing to implement and examine the performance of LSTM models stemmed from their good reputation in being able to model sequential data. We wanted to examine their performance on tabulated data and see if they were able to deliver good quality performance as well.

There was some work involved in formatting the data to make it applicable to the LSTM, we had to reshape our data into a three-dimensional format.

One thing that we noticed was that the efficiency of run time to accuracy was subpar when compared to the other models. With our more complex model we noticed that the run times increased exponentially with little improvement in the accuracy.

CNN Model

Layer (Type)	Output Shape	Number of Parameters
Convolution 1 (Convolution1D)	None, 182, 64	448
Batch Normalization 1 (Batch Normalization)	None, 182, 64	256
Max Pooling 1 (MaxPooling1D)	None, 91, 64	0
Convolution 2 (Convolution1D)	None, 89, 64	12352
Batch Normalization 2 (Batch Normalization)	None, 45, 64	256
Max Pooling 2 (MaxPooling1D)	None, 45, 64	0
Convolution 3 (Convolution1D)	None, 43, 64	12352
Batch Normalization 3 (Batch Normalization)	None, 43, 64	256
Max Pooling 3 (MaxPooling1D)	None, 22, 64	0
Flatten (Flatten)	None, 1408	0
Dense 1 (Dense)	None, 64	90176
Dense 2 (Dense)	None, 32	2080
Dense 3 (Dense)	None, 5	165

Table 4: Architecture of the CNN model

The choice was made to implement a Convolutional Neural Network as one of the approaches to this problem. Furthermore, two different architectures were designed, with the less complex detailed in Table 4. The decision to build two models of differing complexities was supported by the need for varied experimental results. The data had to be reshaped to a three-dimensional format in order to be fitted by the models.

The ability of CNN models to capture grouped dependencies in data was very attractive in the context of a dependant based time series. Furthermore, the data flow consisting of MaxPooling and Convolutional Layers allowed for the models to group data points in the time series which were helpful in explaining the patient condition at each step. Slight bumps or unsual waves are therefore detected and used as features for classification.

Model Results

Classic Learning

Model	Dataset	Training Samples	Testing Samples	Accuracy	Epochs/Batch Size
CNN (low complexity)	**MITBIH**	**70000**	**30000**	**99.38%**	**40/32**
CNN (high complexity)	MITBIH	70000	30000	98.11%	40/32
LSTM	MITBIH	70000	30000	96.14%	10/32
ANN	MITBIH	70000	30000	98.01%	40/32
CNN (low complexity)	PTBDB	5600	2400	97.37%	40/32
CNN (high complexity)	**PTBDB**	**5600**	**2400**	**98.33%**	**40/32**
CNN (HC, upsampled)	**PTBDB**	**14000**	**6000**	**99.63%**	**40/32**
LSTM	PTBDB	5600	2400	95.50%	40/32
LSTM (High complexity upsampled)	PTBDB	14000	6000	97.60%	40/32
ANN	PTBDB	5600	2400	82.46%	40/32

Table 5: Model results using classic learning

Transfer Learning

Model	Dataset	Training Samples	Testing Samples	Accuracy	Epochs/Batch Size
CNN (low complexity downsampled)	PTBDB → MITBIH	42000	18000	97.62%	10/32
CNN (low complexity upsampled)	**PTBDB → MITBIH**	**42000**	**18000**	**97.80%**	**10/32**
CNN (high complexity)	PTBDB → MITBIH	42000	18000	97.32%	10/32
CNN (high complexity upsampled)	PTBDB → MITBIH	42000	18000	97.27%	10/32
LSTM	PTBDB → MITBIH	42000	18000	96.23%	10/32
ANN (downsampled)	PBTDB → MITBIH	42000	18000	96.31%	10/32
ANN (upsampled)	PTBDB → MITBIH	42000	18000	96.42%	10/32

Table 6: Model results using transfer learning

A Deep Learning Approach to ECG Heartbeat Categorization

Amaury Chauvin | Matthieu Thienpont | Oscar Levy | Ruan Van Schalkwyk | Valentin Foucault

Results Visualization

Graphs of the Model Accuracy and Model Loss for each of the two best models for each dataset.

CNN(High complexity) PTBDB

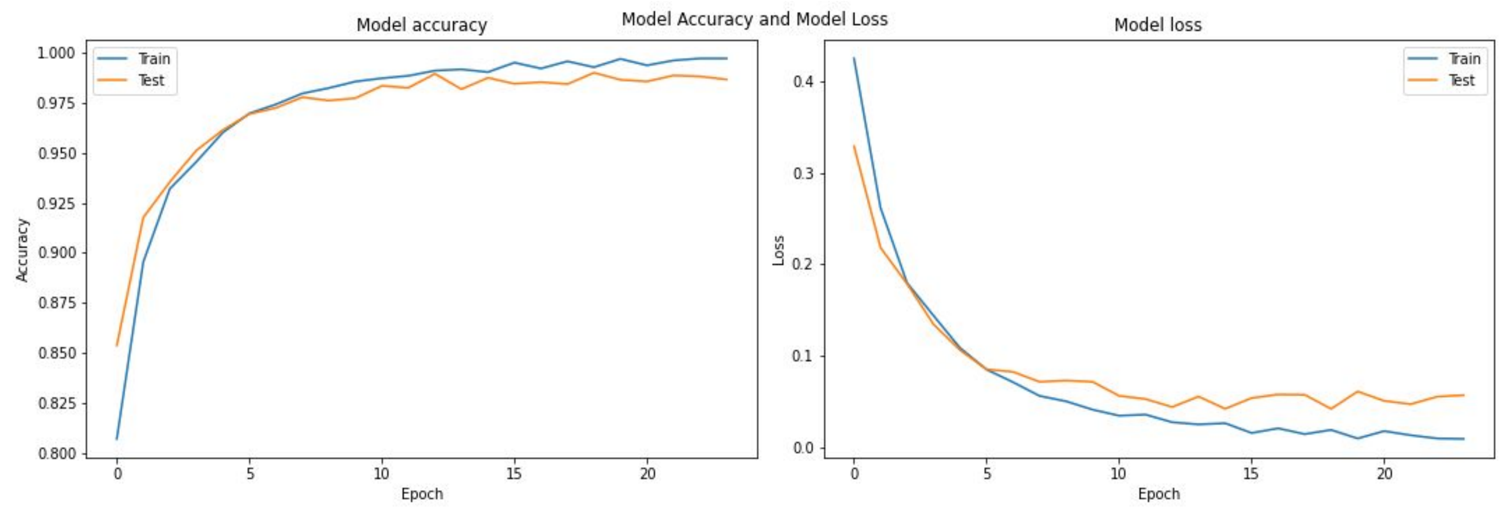

Fig.6 Accuracy and Loss Graph for the High Complexity CNN with upsampled data

LeNet-5 MITBIH

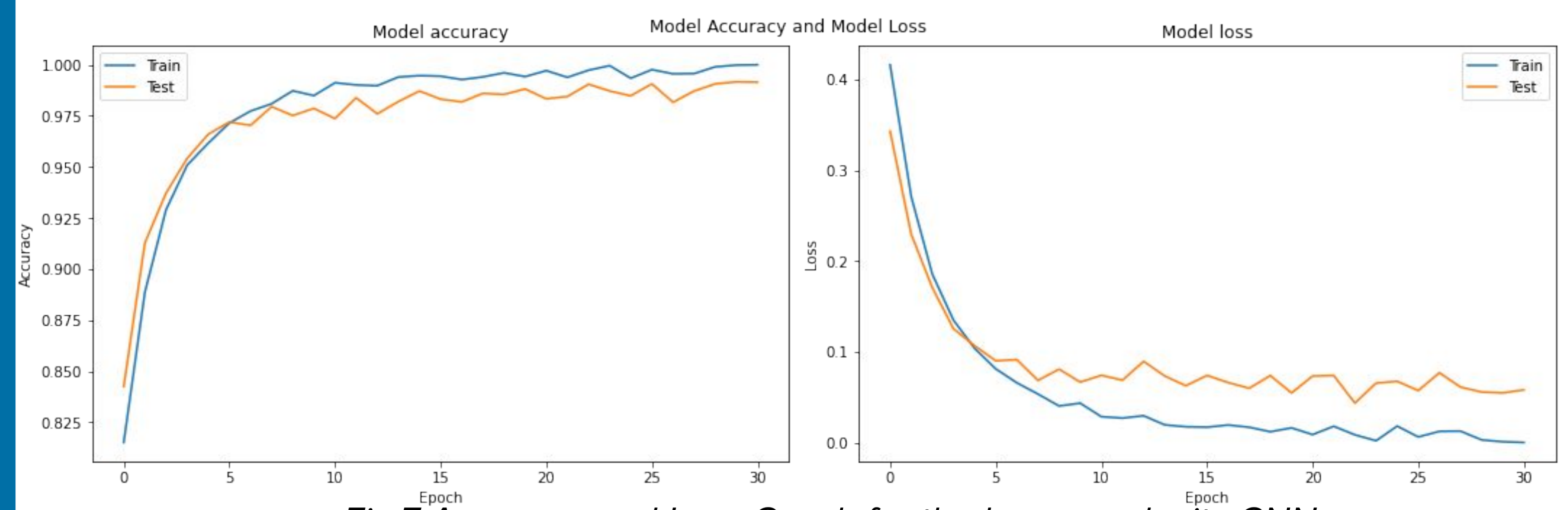

Fig.7 Accuracy and Loss Graph for the low complexity CNN

Confusion Matrices and ROC Curves

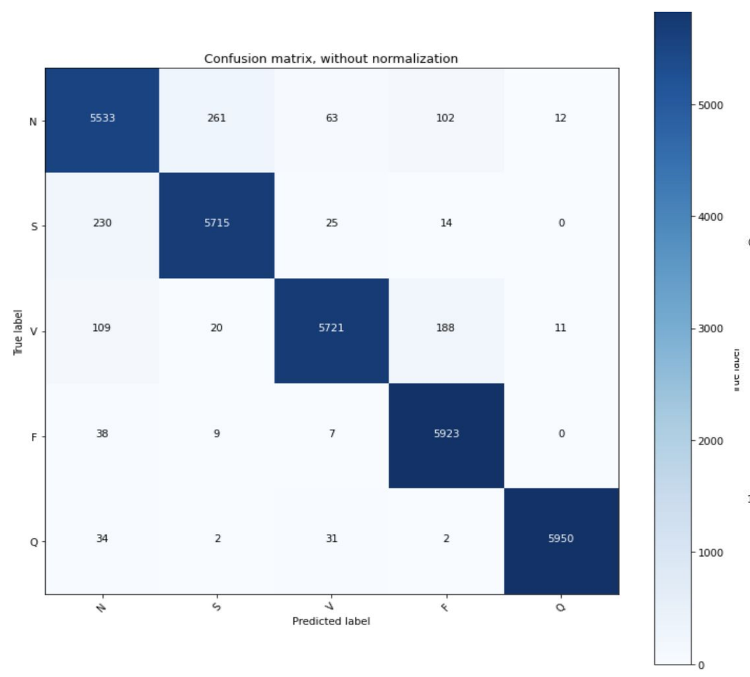

Fig.8 Confusion matrix for MITBIH

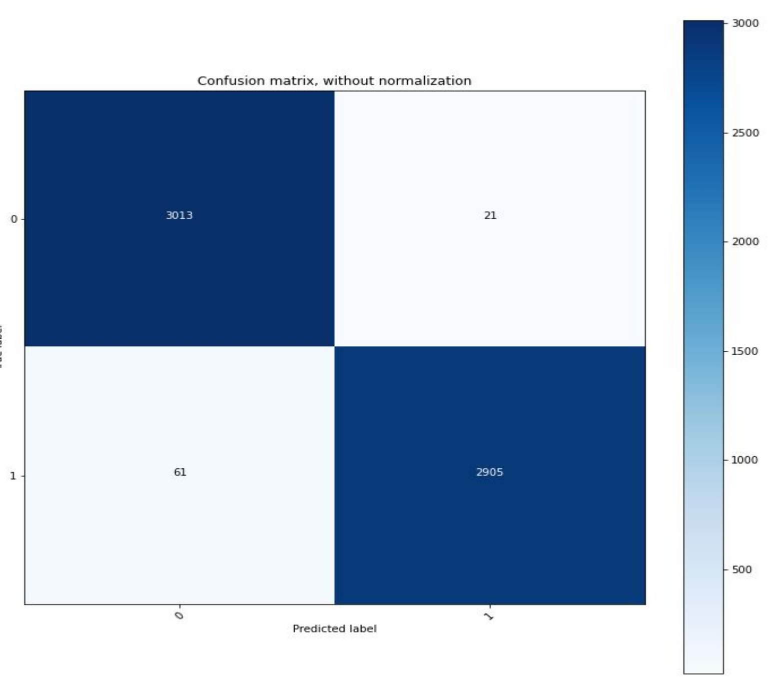

Fig.9 Confusion matrix for PTBDB

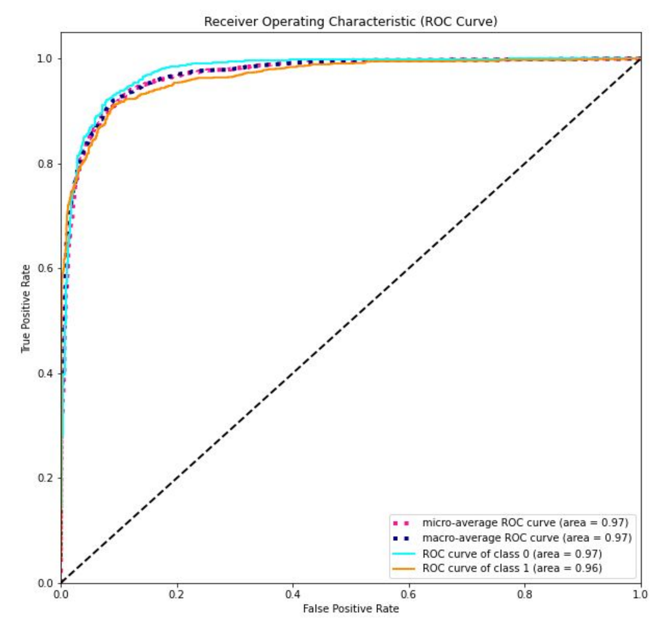

Fig.10 ROC for PTBDB

Figures 6, 7, and 8 along with the accuracy and loss curves located above allow us to visualise the extent to which the models were able to learn.

These curves helped us in identifying whether our models were overfitting on the data which we were training them on. Additionally, they aided us in visualising whether upsampling helped our models better generalise the patterns present in the data.

We feel that their inclusion in this poster is important as they provide a more accurate way in determining the validity of models rather than just stating an accuracy value.

Discussion

- The Low Complexity and High Complexity Convolutional Neural Networks were respectively the highest performers on MITBIH and PTBDB data
- There is a clear advantage in using Convolutional Neural Networks over Long Short Term Memory Models in this context of medical time-series. This is obvious not only in terms of performance but also efficiency in training.
- Transfer Learning proved very interesting and effective over the re-encoded set of MIT BIH data points. After conversion from a multi label to binary label paradigm, additional training of 10 epochs was sufficient to yield high accuracies on a completely new dataset.
- This Transfer Learning result is important in terms of efficiency as it proves that instead of training two models over 40 epochs on two datasets, it is possible to train one model over 40 epochs and 10 additional epochs for the two datasets. Relative training time is therefore reduced by 75%.

Applications

Fig.11 Apple Watch product (source Apple.com website)

Fig.12 ECG Monitor (source Nmmedical.fr website)

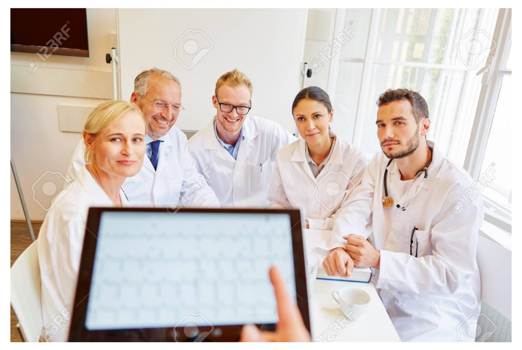

Fig.13 Doctors in training (source 123rf.com website)

- **Smartwatches**: By using laser receptors that are capable of monitoring the pulse of a patient in real time, detect potential cardiac anomalies in order to warn the patient in case of an upcoming heart attack.

  We could also think about having a system in place that would automatically warn the emergency services in case of a heart failure.
- **"Doctor helper"**: Help the doctors to interpret ECGs (or confirm their existing interpretation) by having an AI that is able to guess what the pathology is.
- **Doctor training**: Help future doctors to learn about the different types of cardiac pathologies by allowing them to guess the pathology of a specific ECG and providing them with an interface that would give them the real answer.

Future work

The following work could be done to improve the models later on:

- Improve the transfer learning part
- Transformers
- Deep belief Nets (Hinton, 2017)
- Temporal Convolutional Network (Roy, 2019)
- More experimental preprocessing e.g. different rebalancing ratios, using synthetic samples...

References

Bousseljot, R., D. Kreiseler, and A. Schnabel. 1995 Nutzung der EKG-Signaldatenbank CARDIODAT der PTB über das Internet. Biomedizinische Technik / Biomedical Engineering 40.s1 pp.317-318.

Dictionary.cambridge.org. 2020. ELECTROCARDIOGRAM | Meaning In The Cambridge English Dictionary. [online] Available at: <https://dictionary.cambridge.org/dictionary/english/electrocardiogram> [Accessed 24 May 2020].

Fazeli, S. (2018). ECG Heartbeat Categorization Dataset. Retrieved May 23, 2020, from https://www.kaggle.com/shayanfazeli/heartbeat

Hannun, A., Rajpurkar, P., Haghpanahi, M., Tison, G., Bourn, C., Turakhia, M. and Ng, A., 2019. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nature Medicine, 25(1), pp.65-69.

Hinton, G., 2017. 2007 NIPS Tutorial On: Deep Belief Nets.

Kachuee, M., Fazeli, S. and Sarrafzadeh, M., 2018. ECG Heartbeat Classification: A Deep Transferable Representation. 2018 IEEE International Conference on Healthcare Informatics (ICHI),.

MIT.edu. 2020. MIT-BIH Arrhythmia Database V1.0.0. [online] Available at: <http://doi.org/10.13026/C2F305> [Accessed 21 April 2020].

Porumb, M., Stranges, S., Pescapè, A. and Pecchia, L., 2020. Precision Medicine and Artificial Intelligence: A Pilot Study on Deep Learning for Hypoglycemic Events Detection based on ECG. Scientific Reports, 10(1).

G. B. Moody and R. G. Mark (2001). The impact of the MIT-BIH Arrhythmia Database, in IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3, pp. 45-50

Rajput, K., 2018. On Arrhythmia Detection by Deep Learning and Multidimensional Representation. Arxiv, [online] Available at: <https://arxiv.org/ftp/arxiv/papers/1904/1904.00138.pdf> [Accessed 21 April 2020].

Roy, R., 2019. Temporal Convolutional Networks. [Blog] Available at: <https://medium.com/@raushan2807/temporal-convolutional-networks-bfea16e6d7d2> [Accessed 13 May 2020].