IMPROVING ECG CLASSIFICATION PERFORMANCE USING GAN GENERATED EXAMPLES FOR MINORITY CLASSES

Nicola Brazzale
Aalto University School of Science, Department of Computer Science,
{nicola.brazzale@aalto.fi}

ABSTRACT

Electrocardiogram (ECG) is performed routinely by medical personnel to identify electrical, structural and functional cardiac events. Recently, ECG has been employed in computer-aided cardiovascular events diagnosis systems. Precise and automatic detection of abnormal ECG patterns is beneficial to both cardiologists, not having to perform tedious examinations, and patients, for whom an accurate prediction could be life-saving. In this scenario, deep neural networks (DNNs) have shown significant achievements in the automatic detection of abnormal ECG patterns. However, the complex variations and imbalance of ECG beats and the lack of a large amount of labelled data make this a challenging issue. The focus of this work was to build a model able to correctly classify ECG segments of four different classes (normal beat, atrial fibrillation, abnormal beat and noisy) while paying attention to the imbalanced nature of the dataset. For this reason, a Gated Recurrent Unit (GRU) based network model is here proposed to disentangle the time dimension's features in the signals, alongside the employment of the Focal Loss that down-weights easily identifiable normal ECG examples. The advantages of the proposed network have been verified on the PhysioNet Challenge 2017 dataset containing more than 8500 ECG recordings. In this work, furthermore, the use of generative adversarial networks (GAN) is studied for the generation of synthetic single-lead ECG signals. These generated signals are then used as additional training data to improve the classifier performance, without changing the architecture of the main deep neural network. The proposed method can be deployed in telemedicine scenarios to assist cardiologists in more accurately and objectively diagnosing ECG signals.
Code available at: `https://github.com/brazzalenicola/Atrial-Fibrillation`.

Keywords ECG · Wander Removal · Convolutional Neural Network · Recurrent Neural Networks · Generative Adversarial Networks

1 Introduction

According to the World Health Organisation [1], cardiovascular diseases (CVDs) are the leading cause of death worldwide. About 17.9 million people died of CVDs in 2019, representing 32% of all global deaths. CVDs are a group of disorders of the heart and blood vessels, common manifestations are represented by myocardial infarction and strokes that are mainly caused by a blockage that prevents blood from flowing to the heart or brain. The most common reason for this is a build-up of fatty deposits on the inner walls of the blood vessels that supply the heart or brain. Strokes can be caused by bleeding from a blood vessel in the brain or from blood clots. It is important to detect cardiovascular diseases as early as possible so that management with counselling and medicines can begin.
An electrocardiogram (ECG) is a comprehensive manifestation of the electrical signal activity of the human heart. Obtaining the detailed physiological state of various parts of the heart by collecting signals is an indispensable means of clinical objective diagnosis. Atrial Fibrillation (AF) is a common type of heart disease, occurring in 1-2% of the general population [2], and it could lead to stroke, heart failure, coronary artery disease or other complications. Irregularities in the heartbeat are considered to be the most common symptom of AF and can be traced in an ECG. However, despite the importance of this disease, AF detection remains problematic, because of its episodic nature. In practice, to diagnose abnormal heart activities, cardiologists review ECG signals which is a labour-intensive and time-consuming process putting a large burden on cardiologists. For this aim, automatic detection systems of abnormal ECG are being developed

to assist physicians.
Deep learning is believed to be the spearhead of machine learning and pattern recognition. It provides a structure in which feature extraction and classification are performed together. These techniques have been widely used in many fields, such as image classification, target detection and disease prediction. In recent years, many new deep network architectures have been designed, each of those with its purpose and characteristics. Having a large pool of architecture suitable to different tasks has let to researchers to employ a specific neural network, called recurrent neural network (RNN) to process time-series data, given that RNNs are more appropriate for chronological changes in appearance of sample sequences. Although deep learning methods are showing promising results for ECG classification, they often require a large number of training samples per class to achieve good performance. In this paper, we propose a robust algorithm for classifying normal, AF, other abnormal rhythms and noisy ECG recordings. The classifier is based on a combination of Convolution Neural Networks (CNN), used to extract meaning full features from longer ECG segments, and a sequence-based network such as an RNN for capturing the time-series behaviour. The diverse ECG dataset, provided in Physionet challenge 2017 [2] is used for creating and training the models and for internal performance evaluation. Data augmentation is here employed to improve the classification performance. In this work, we overcome the sparseness of data by learning to synthetically generate ECG signals of different rhythms, which we later use to train the main deep neural network. Intuitively, we build a model that learns to recreate synthetic ECG signals using a Generative Neural Network (GAN) architecture, only for those classes that are underrepresented in the dataset, namely, Atrial Fibrillation and Noisy rhythm. The aim is to help the main network to learn significant features and prevent underfitting/overfitting. Our contributions in this work are: (1) introducing a hybrid Convolution-Recurrent neural network classifier, (2) preprocessing with noises-baseline wander removal, (3) employing GANs to create synthetic ECG to improve the classification.

2 Related Work

Many automatic classification methods have been proposed in the past years. Different types of ECG beats can be distinguished by the time-domain, wavelet transform, genetic algorithm, support vector machine (SVM), Bayesian, or other methods [3]. These algorithms' strongly depends on the engineered features, as they have to be informative, discriminating, and independent based on expert knowledge [4]. Although these methods can achieve good accuracy on experimental datasets they also present limitations regarding real-time deployment. The limitation in the applicability is due to 1) only classification of normal and AF rhythms were performed. However, there are many non-AF abnormal rhythms (like tachycardia, bradycardia, atrial flutter etc) which exhibits heartbeat patterns similar to AF. Not considering them in the dataset makes the classification task less challenging. 2) Good performance was shown on carefully-selected often clean data. However, in practical scenarios, ECG signals are often noisy. 3) Size of the test dataset are often not adequate for making a conclusion, or 4) only a small number of patients were used [2]. On top of these issues, we should consider that most of the prior methods are validated on clinically accepted 12 lead ECG signals with a relatively long duration [5].
The advent of Deep Learning has given bioinformatics a new and powerful tool to analyse bio-signals. For example, Acharya et al. [6] proposed a nine-layer CNN to automatically identify five ECG beat types. Yildirim et al. [7] designed a 1D-convolutional neural network model for detecting arrhythmia in an end-to-end fashion. Although CNNs might have proven themselves reliable with spatial data thanks to their unique weight-sharing mechanism and their image-specific inductive biases, Recurrent neural networks have obtained major successes in time-series analysis. Two very common architectures of RNNs are based on gated mechanisms like LSTM (long short-term memory) or GRU (gated recurrent unit). Both of them can effectively retain historical information and learn long-term dependencies between elements in a sequence. While these methods have been mainly applied to natural language processing (NLP) and speech recognition they recently started to be a very common choice for the detection of abnormal ECGs. Yildirim [8] proposed a new model for deep bidirectional LSTM network-based wavelet sequences to classify electrocardiogram (ECG) signals. Hou et al. [9] introduced a new algorithm based on deep learning that combines LSTM with SVM for ECG arrhythmia classification.
The imbalance of the ECG datasets has been an additional challenge that prevents methods to achieve good accuracy, since it can lead to two main problems in the training process: (1) low training efficiency because normal ECG beats occupying a large proportion of the dataset are prone to negative effects, and (2) degeneration of the model when a normal ECG beat overwhelms training [3]. Different approaches have been used to tackle this issue, some researchers used the generated oversampling method (GenOMe) [10] to solve the problem of imbalanced arrhythmias, which generated new data points with specific distributions (beta, gamma, and Gaussian) as constraints, some others employed three data-level preprocessing techniques on an extracted feature set to balance the distribution of ECG heartbeats [11].

3 Methods

3.1 Physionet Dataset

The benchmark dataset on top of which the models are built and an interval validation is performed is offered by PhysioNet in the 2017 Computing in Cardiology Challenge. The dataset consists of 8,528 single short ECG lead recordings lasting from 9 s to 61 s. In this dataset, four classes of data were considered: normal rhythm, AF rhythm, other rhythm and noisy recordings. Due to the high degree of inter-expert disagreement between a significant fraction of the expert labels, PhysioNet implemented a mid-competition bootstrap approach to expert relabeling of the data, levering the best performing Challenge entrants' algorithms to identify contentious labels [2]. The original dataset has been divided into training and validation sets with an 80-20 ratio. Examples of the signal for each of the four classes can be seen in fig.1 and fig.2.

Table 1: Data profile for the training set

Name	# recordings	% recordings
Normal	4061	59.53
AF	606	8.88
Other Abnormal	1932	28.32
Noisy	223	3.27

Table 2: Data profile for the validation set

Name	# recordings	% recordings
Normal	1015	59.50
AF	152	8.91
Other Abnormal	483	28.32
Noisy	56	3.28

3.2 Signal preprocessing

Electrocardiogram (ECG) signals contain many types of noises - baseline wander, powerline interference, electromyographic (EMG) noise, electrode motion artefact noise [12]. Baseline wander is a low-frequency noise of around 0.5 to 0.6 Hz in the ECG that arises from several factors, such as breathing, electrically charged electrodes, or subject movement. In the ECG signal, it can be seen as a low-frequency trend, which is not related to the heart activity and should be removed. Given that the preservation of the shape of the actual ECG is of primary importance since it has components also in the low-frequency band, an efficient way to estimate and remove the baseline from the signal is to use median filters (one with 0.2 seconds and one with 0.6 seconds window size).
Normalization is another important step. Depending on the subject, on its skin resistivity, on the quality of the electrical connection of the electrodes and many other factors, the amplitude of the ECG may be highly affected. In this case, since we are more interested in the shape of the signal than in its amplitude, we perform an element-wise normalization, which means that the normalization is applied independently on each trace. For this purpose, we simply estimate the signal amplitude using the 99th and the 5th percentile, which allow us to exclude possible outliers, making this approach more robust compared to using max/min when using real-world data.
Another important remark is that the signals in the dataset have different lengths. Most of the signals have 9000 samples (30 seconds, considering the 300 Hz sampling frequency), so we use that value as the fixed input shape. If the signal is too short (< 9000 samples), we pad the vector with zeros to match the required size, in the other case, so if the signal is too long (> 9000 samples), we select a portion of the signal (crop) to match the required size (all the experiments conducted are based on the central crop of the lengthy signals). Examples of some signals at each preprocessing step can be seen in fig.3.

3.3 Convolutional Recurrent Neural Network - CRNN

3.3.1 Convolutional Neural Network

Convolutional neural networks (CNN) are powerful tools that have shown advantages on both accuracy and efficiency in image recognition, audio classification and semantic identification. The connectivity between neurons in a CNN is similar to the organization of the visual cortex in animals, which gives CNNs image-specific inductive biases that makes this architecture superior to other methods in the recognition of the pattern and structure of items. Recent studies have also shown the great potentials of CNN in dealing with biomedical applications [13], such as skin cancer diagnosis, animal behavior classification, histopathological diagnosis, protein structure prediction and electromyography (EMG) signal classification [14]. More in general, CNNs usually consist of the input layers, convolutional layer, pooling layer, batch-normalisation layer, fully-connected layer and output layer. Some of these layers help the network to extract and map features from input data to speed up learning and reduce over-fitting. Different arrangements of these layers, or the lack of one or more of them, allow the creation of networks with different architectures and purposes. In this work, we made use primarily of convolutional and batch normalisation layers for the convolutional part of the network.

1. **Convolutional layer**:
   This kind of layer performs a convolution operation between the input and a kernel with specified dimensions. In this work, a 1D convolution kernel is employed and it convolutes independently of the feature map of the previous layer. The output of a convolutional layer is obtained adding a bias to the convolution operation and passing it to a nonlinear activation function.

$$h_i^{l,k} = f(b_i^{l,k} + \sum_{n=1}^{N} W_{n,i}^{l,k} * x_{i+n-1}^{l-1,k}) \tag{1}$$

   Where $h_i^{l,k}$ is the ouput of the ith layer l, $f()$ is the activation function and $b_i^{l,k}$ is the bias of the lth layer. $W_{n,i}^{l,k}$ and $x_{i+n-1}^{l-1,k}$ are the k^{th} convolutional kernel and the ouput of neuron in layer $l-1$, respectively.
2. **Batch-Normalisation layer**:
   It usually helps to have intermediate signals with zero mean and unit variance. To this end, batch-normalisation layers are here used after each 1D convolutional layer. In these layers the mean μ and the standard deviation σ are computed from the current batch of input signals $\{x^{(1)}, ..., x^{(N)}\}$ and intermediate signals are normalised to zero mean and unit variance.

$$\tilde{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{2}$$

   Finally, the layer can control the mean and the variance of the outputs y with two trainable parameter γ and β: $y = \gamma \odot \tilde{x} + \beta$.
   Batch normalisation makes the optimization landscape smoother. This smoothness induces a more predictive and stable behaviour of the gradients, allowing for faster training, faster convergence and better generalization.

3.3.2 Recurrent Neural Network

A Recurrent Neural Network (RNN) is a type of artificial neural network architecture with recurrent connections to process sequential input data. RNN is categorized as a deep learning technique due to an automatic process of feature calculation without predetermining some appropriate features. Every node at a time step consists of input from the previous node and it proceeds using a feedback loop. Each node produces a current hidden state (h_t) and outputs $\hat{y}_t$ by using current input and the previous hidden state as follows:

$$\begin{aligned} h_t &= f(W_h h_t - 1 + V_h x_t + b_h) \\ o_t &= f(W_o h_t + b_o) \end{aligned} \tag{3}$$

Where h_t denoted the hidden state for each time step t, W and V are the weights for the hidden layers in recurrent connection. while b is the bias term for hidden and output states and f represents an activation function applied to each node of the network. Original RNN, also known as vanilla RNN, has similar forward pass and backward pass processes as other artificial neural networks. The difference is only in the backpropagation process where the term being backpropagation is defined through time (BPTT). The original formulation has some drawbacks such as the inability to derive context from steps of previous states much far behind, in other words, is unable to capture long term dependencies. Furthermore, Vanilla RNN suffers from exploding and vanishing of gradients.

3.3.3 Gated Recurrent Unit - GRU

One of the most famous and most utilised solutions to the vanilla RNN problem is represented by a gated mechanism called Gated Recurrent Unit (GRU). GRU models deploy memory cells with several gates in the hidden layer.
Forget and input gates o_t control both which part of the input should be added to long term state c_t and which part of c_t should be omitted. When z_t is 1, the forget gate is opened and the input gate is closed, whereas z_t is 0, the forget gate is closed and the input gate is opened.
With this architecture, described with equations 4, the input of a time step t is deleted every time the previous $(t-1)$ memory is stored. Intuitively, the reset gate determines how to combine the new input with the previous memory, and the update gate decides how much of the previous memory information is retained to calculate the new state.

$$\begin{aligned} r_t &= \sigma(W_{xr}^T x_t + W_{or}^T o_{t-1} + b_r) \\ z_t &= \sigma(W_{xz}^T x_t + W_{oz}^T o_{t-1} + b_z) \\ z_t &= tanh(W_{x\tilde{o}}^T x_t + W_{o\tilde{o}}^T (r_t \otimes o_{t-1}) + b_{\tilde{o}} \\ o_t &= z_t \otimes o_{t-1} + (1 - z_t) \otimes \tilde{o}_t. \end{aligned} \tag{4}$$

Where W_{xr},W_{xz},W_{xo} denote the weight matrices for the corresponding connected input vector, W_{or}, W_{oz}, W_o represent the weight matrices of the previous time step, and b_r ,b_z, b are bias.

3.3.4 Proposed Architecture

Given the input training set, $X = (x^{(1)}, y^{(1)}), ..., (x^{(i)}, y^{(i)}), ..., (x^{(n)}, y^{(n)})$, where n is the number of ECG beats containing the class labels, $x^{(i)}$ is an ECG beat and $y^{(i)} \in \{0, 1, 2, 3\}$ is the corresponding label of the $x^{(i)}$. To the Atrial Fibrillation class label 0 has been assigned, Normal rhythms class has label 1, while abnormal rhythms and noisy samples have labels 2 and 3, respectively. To achieve the detection of atrial fibrillation, we started using the methods proposed in Gao et al. [3] however several modifications are made in the network structure. In this paper, we propose a network architecture combining convolutional and recurrent blocks. The convolutional block consist in 3 one-dimensional convolutional layers in order to process a one-dimensional time series. This convolutional block was used to extract features and to lower the dimensions of the signals, while the recurrent block was used to capture long time dependencies. After each convolutional layer, the intermediate signals go trough a batch-normalisation layer and rectified linear unit (ReLU) activation. The output of the the convolutional part is a signal with 138 samples and 128 feature maps, reducing by a significant factor the length of the original input. These feature vectors are then fed into the recurrent block, composed of a gated recurrent unit (GRU), in contrast with Gao et al. [3] architecture, where they employed a LSTM layer. The choice of choosing GRU over LSTM layer, came from the fact that GRU is believed to achieve comparable result to LSTM while having has fewer tensor operations and therefore, they are faster to train. The recurrent layer produces one feature vector for the whole feature vectors and finally, this output, is then fed into fully connected layers whose aim is to perform the final classification, thanks also to the last softmax layer which calculates the probability of each ECG beat category. The full list of the proposed architecture's layers is shown in tab.3.

Table 3: Network's summary

Layer	# parameters	kernel	stride
Conv1D	160	9	4
BatchNorm1D	32	-	-
ReLu	-	-	-
Conv1D	4640	9	4
BatchNorm1D	64	-	-
ReLu	-	-	-
Conv1D	18496	9	4
BatchNorm1D	128	-	-
ReLu	-	-	-
GRU	39168	-	-
Dense	2080	-	-
Dense	132	-	-
Softmax	-	-	-

3.4 Focal Loss

The focal loss [3] is believed to be a more effective way to deal with imbalanced datasets. Its formulation is obtained by transforming the cross-entropy loss (CE).

$$CE(y, p) = \begin{cases} -log(p) & \text{if } y = 1 \\ -log(1-p) & \text{otherwise} \end{cases} \tag{5}$$

In the above $y \in \{\pm 1\}$ specifies the ground-truth class and $p \in [0, 1]$ is the model's estimated probability for the class with label $y = 1$. For convenience we can define p_t as:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases} \tag{6}$$

Therefore we can rewrite the CE loss as: $CE(p_t) = -log(p_t)$.
We can see the focal loss as a dynamically scaled version of CE, where the scaling factor decays to zero as the confidence of the classification increases. The focal loss can be expressed by

$$\text{FL}(p_t) = -(1-p_t)^{\gamma}\, log(p_t),\ \gamma \geq 0 \tag{7}$$

where the factor $(1-p_t)^{\gamma}$ operates as a modulator and γ is a focusing parameter. The purpose of the modulation factor is to reduce the weights of easily recognisable ECG beats so that the model is more focused on ECG beats that are difficult to classify during training. When an ECG beat is misclassified and p_t is small, the value of the modulation factor is close to 1 and the loss is barely affected. Easily classified negatives comprise the majority of the loss and dominate the gradient. As p_t goes to 1, the factor goes to 0 and the loss for well-classified examples is down-weighted.

3.5 Generative Adversarial Network - GAN

Our approach is to create a generative model for each type of under-represented heartbeat type, for a total of two models. Each such model includes a generator network and a discriminator network. The architecture of the generator and discriminator networks is the same across the two models, but each has unique values for the parameters. A generator network takes as input a random latent vector and outputs a synthetic ECG signal. A discriminator network takes as input an ECG signal and outputs a binary decision representing whether the input signal is real or was generated by a generator network. More in detail, the generator starts from noise to try to create an imitation of the data and the discriminator looks at both the real data and fake data provided by the generator and tries to predict what's real and what's fake while the generator keeps trying to improve its imitation of the data. We denote the generator network as G_{wg} and the discriminator network as D_{wd}, where w_g and w_d represent the parameters of the Generator and Discriminator networks, respectfully. We consider a data point x_r originated from a real-world distribution P_r and z a random vector from a random noise distribution $p(z)$. The aim of the generator is to make a distribution P_G of generated values $x_g = G_{wg}(z)$ that is very similar to the real-world distribution P_r. The generator's weights w_g are optimised using a discriminator that is trained with weights w_D to distinguish between the fake x_g and the real x. The discriminator's output $D_{wd}(x)$ gives out the probability of the input being fake or not given real-world example, x. We train D to maximize the probability of assigning the correct label to both training examples and samples from G and we simultaneously train G to minimize $log(1 - D(G(z)))$. We can say that D and G play a two-player min-max game with value function $V(G, D)$ and the goal is to optimise the following equation:

$$min_G max_D V(D, G) = E_{x \sim p_r(x)}[logD(x)] + E_{z \sim p_z(z)}[log(1 - D(G(z)))] \tag{8}$$

The generator's loss function is a binary cross-entropy loss (BCE) between the discriminator's prediction and the real data label. Regarding the discriminator's loss, it is a combination of binary cross-entropy loss computed on the real samples and binary cross-entropy loss computed on the fake samples. The training consists of repeatedly updating the discriminator before and the generator after. To update the discriminator, we sample N example x_i from the training set and we generate N samples $g(z_i)$ using the generator and finally, we compute the binary cross-entropy loss as

$$L_d = -\frac{1}{N}\sum_{i=1}^{N} log\ d(x_i) - \frac{1}{N}\sum_{i=1}^{N} log(1 - d(g(z_i))) \tag{9}$$

The second step consists in updating the generator by generating N samples $g(z_i)$ and computing the loss function as

$$L_g = -\frac{1}{N}\sum_{i=1}^{N} log\ d(g(z_i)) \tag{10}$$

Our proposed architectures for the generator and the discriminator are described in tab.9 and in tab.10. Thanks to the training of these two networks we were able to generate new synthetic samples using the trained generators. We chose to generate as many samples as the cardinality of the corresponding classes, in so doing we doubled the size of those under-represented classes, improving the class distribution inequality, as shown in tab.4 and tab.5.
A comparison of generated signals and original signals can be seen in fig.4.

Table 4: Augmented data profile for the training set

Name	# recordings	% recordings
Normal	4061	53.07
AF	1213	15.85
Other Abnormal	1932	25.25
Noisy	446	5.83

Table 5: Augmented data profile for the validation set

Name	# recordings	% recordings
Normal	1015	53.06
AF	303	15.84
Other Abnormal	483	25.25
Noisy	112	5.85

4 Results

4.1 Results without Data Augmentation

The hybrid network was trained using the original dataset, described in tab.1, for 80 epochs, with a learning rate of 0.001, the Focal Loss and Nesterov-accelerated Adaptive Moment Estimation (NAdam) optimiser, which is an effective gradient descent optimization algorithm that combines the Adam and the Nesterov's Accelerated Gradient (NAG) algorithms to calculate adaptive learning rates for different parameters. NAdam has been chosen since it has shown

Table 6: Evaluation without data augmentation

Name	Accuracy	Recall (TPR)	Specificity (TNR)	Precision	F1
Atrial Fib	0.89	0.18	0.96	0.31	0.23
Normal	0.66	0.86	0.37	0.66	0.75
Other Abnormal	0.68	0.24	0.85	0.38	0.29
Noisy	0.96	0.00	0.99	0.00	0.00

better performance than other gradient descent optimization methods in practical applications. The training profile of the accuracy and the loss on the original dataset can be seen in fig.5.

Although the model seems to be achieving satisfactory results in terms of accuracy for each class, the imbalanced nature of the classification makes that measure not reliable. In light of this, the Recall (True positive rate), the Specificity (True Negative Rate), the precision and the F1 score have been calculated to assess the quality of the classifier. As it can be seen in 6 the recall is very low especially for minority classes. Precision and, in addition, F1 scores do not show very high values, meaning that the overall accuracy of the classifier can be subject to improvements.

4.2 Results with Data Augmentation

The two GAN models were trained for 1000 epochs using Adam optimiser with eq.9 and eq.10 as loss functions for the discriminator and the generator, respectively. With the augmented dataset tab.4, the hybrid network was trained again for 80 epochs, with a learning rate of 0.001, the Focal Loss and NAdam optimiser. The training profile of the accuracy and the loss on the augmented dataset can be seen in fig.5.

Table 7: Evaluation with data augmentation

Name	Accuracy	Recall (TPR)	Specificity (TNR)	Precision	F1
Atrial Fib	0.89	0.63	0.93	0.63	0.64
Normal	0.73	0.86	0.60	0.71	0.78
Other Abnormal	0.73	0.35	0.87	0.46	0.39
Noisy	0.94	0.04	0.99	0.25	0.06

The results in tab.7 show that thanks to data augmentation, the classification has benefited. Accuracy, recall, specificity, precision and F1 score all show better values meaning that the classification is more accurate.

4.3 Scoring result

The scoring for the PhysioNet challenge was based on the F1 measure, which is an average F1 value from the classifier for each class. This score was calculated on a hidden test set, however, in the absence of such a test set, in this work, a validation test consisting of 20% of all the original dataset, was used to calculate the final score. The F1 score is defined as:

$$F_{1i} = \frac{2TP_i}{2TP_i + FP_i + FN_i} \text{ for each } i \in \{N, A, O, P\} \tag{11}$$

with $\{N, A, O, P\}$ being the Normal rhythm class, Atrial fibrillation, abnormal rhythm and noisy recordings, respectively. The final score of the challenge is generated as:

$$F_1 = \frac{F_{1N} + F_{1A} + F_{1O}}{3} \tag{12}$$

Despite the proposed method lack far behind the top-4 entries of the challenge tab.8, it can be seen that the presence of more data has brought a significant improvement to the final method's evaluation, helping to achieve an overall $F1$ score of 0.633.

Table 8: Final scores of the Challenge

Method	F1-score
Teijeiro et al. [15]	0.831
Zabihi et al. [16]	0.826
Baydoun et al. [2]	0.822
Zihlmann et al. [17]	0.821
Proposed method with augmentation	0.633
Proposed method	0.423

5 Conclusion

In this study, we proposed a hybrid CNN-GRU based network to capture both meaning full features and the inherent temporal structure in ECG signals, and the Focal Loss used to improve the training effect by down-weighting the impact of normal ECG beats that are easier to classify. The results show that the hybrid network with FL achieved an average accuracy of 0.785 but very low recall and precision. We also propose a Generative Adversarial Network (GAN) architecture that can be trained to generate ECG signals which are meant to resemble real ECG signals observed in the data. Experimental results of the PhysioNet dataset demonstrate the effectiveness and robustness of the proposed network and its performance, in terms of precision, recall and F1 score, significantly improve when the training set has been augmented using synthetic data. We were able to achieve an average accuracy of 0.823 but with higher recall and precision with respect to the baseline model. The study was conducted only on four different ECG beat types. To make the scenario more realistic, various types and numerous beats should be incorporated in future studies. Another important improvement would be to use different architectures or more data to train the GAN models to achieve higher quality and visually better generated examples closer to reality. These results alongside further research improve ECG classification have the potential to enable automated ECG analysis technology. This type of technology can support applications such as quality control on expert diagnoses and home-ECG kits.

References

[1] World Health Organization. Cardiovascular diseases (CVDs), 2019.

[2] Gari D. Clifford, Chengyu Liu, Benjamin Moody, Liwei H. Lehman, Ikaro Silva, Qiao Li, A. E. Johnson, and Roger G. Mark. AF classification from a short single lead ECG recording: The PhysioNet/computing in cardiology challenge 2017. In *Computing in Cardiology*, volume 44, pages 1–4. IEEE Computer Society, 2017.

[3] Junli Gao, Hongpo Zhang, Peng Lu, and Zongmin Wang. An Effective LSTM Recurrent Network to Detect Arrhythmia on Imbalanced ECG Dataset. *Journal of Healthcare Engineering*, 2019, 2019.

[4] Tomer Golany, Gal Lavee, Shai Tejman Yarden, and Kira Radinsky. Improving ECG Classification Using Generative Adversarial Networks. Technical report.

[5] Shreyasi Datta, Chetanya Puri, Ayan Mukherjee, Rohan Banerjee, Anirban Dutta Choudhury, Rituraj Singh, Arijit Ukil, Soma Bandyopadhyay, Arpan Pal, and Sundeep Khandelwal. Identifying normal, AF and other abnormal ECG rhythms using a cascaded binary classifier. In *Computing in Cardiology*, volume 44, pages 1–4. IEEE Computer Society, 2017.

[6] U. Rajendra Acharya, Shu Lih Oh, Yuki Hagiwara, Jen Hong Tan, Muhammad Adam, Arkadiusz Gertych, and Ru San Tan. A deep convolutional neural network model to classify heartbeats. *Computers in biology and medicine*, 89:389–396, 10 2017.

[7] Özal Yıldırım, Paweł Pławiak, Ru San Tan, and U. Rajendra Acharya. Arrhythmia detection using deep convolutional neural network with long duration ECG signals. *Computers in Biology and Medicine*, 102:411–420, 11 2018.

[8] Özal Yildirim. A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification. *Computers in biology and medicine*, 96:189–202, 5 2018.

[9] Borui Hou, Jianyong Yang, Pu Wang, and Ruqiang Yan. LSTM-Based Auto-Encoder Model for ECG Arrhythmias Classification. *IEEE Transactions on Instrumentation and Measurement*, 69(4):1232–1240, 4 2020.

[10] H. R. Sanabila, Ilham Kusuma, and Wisnu Jatmiko. Generative oversampling method (GenOMe) for imbalanced data on apnea detection using ECG data. In *2016 International Conference on Advanced Computer Science and Information Systems, ICACSIS 2016*, pages 572–579. Institute of Electrical and Electronics Engineers Inc., 3 2017.

[11] Kandala N.V.P.S. Rajesh and Ravindra Dhuli. Classification of imbalanced ECG beats using re-sampling techniques and AdaBoost ensemble classifier. *Biomedical Signal Processing and Control*, 41:242–254, 3 2018.

[12] Rahul Kher. Signal Processing Techniques for Removing Noise from ECG Signals. Technical report, 2019.

[13] Jia Li, Yujuan Si, Tao Xu, and Saibiao Jiang. Deep Convolutional Neural Network Based ECG Classification System Using Information Fusion and One-Hot Encoding Techniques. *Mathematical Problems in Engineering*, 2018, 2018.

[14] Mengze Wu, Yongdi Lu, Wenli Yang, and Shen Yuong Wong. A Study on Arrhythmia via ECG Signal Classification Using the Convolutional Neural Network. *Frontiers in Computational Neuroscience*, 14, 1 2021.

[15] Tomás Teijeiro, Constantino A. García, Daniel Castro, and Paulo Félix. Arrhythmia Classification from the Abductive Interpretation of Short Single-Lead ECG Records. 11 2017.

[16] Morteza Zabihi, Ali Bahrami Rad, Aggelos K. Katsaggelos, Serkan Kiranyaz, Susanna Narkilahti, and Moncef Gabbouj. Detection of atrial fibrillation in ECG hand-held devices using a random forest classifier. In *Computing in Cardiology*, volume 44, pages 1–4. IEEE Computer Society, 2017.

[17] Martin Zihlmann, Dmytro Perekrestenko, and Michael Tschannen. Convolutional recurrent neural networks for electrocardiogram classification. In *Computing in Cardiology*, volume 44, pages 1–4. IEEE Computer Society, 2017.

Appendix

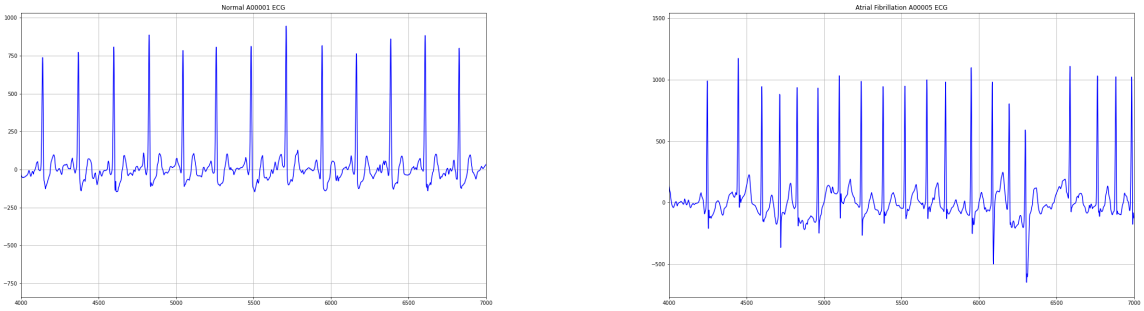

Figure 1: Normal rhythm (left) and Atrial Fibrillation rhythm (right)

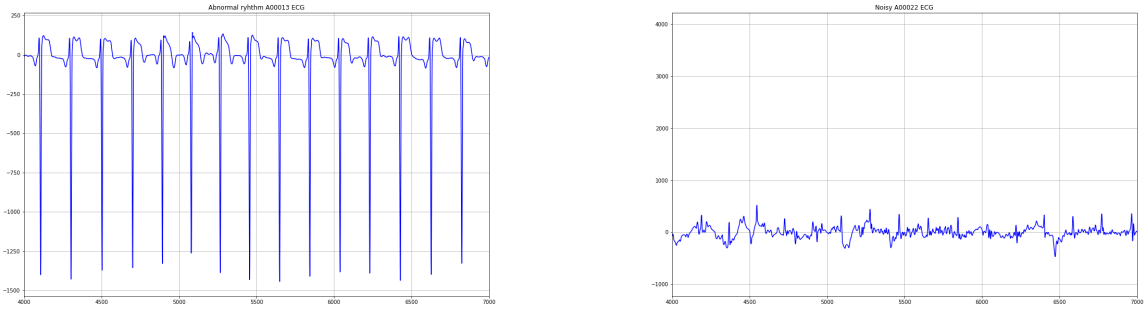

Figure 2: Abnormal rhythm (left) and Noisy recording (right)

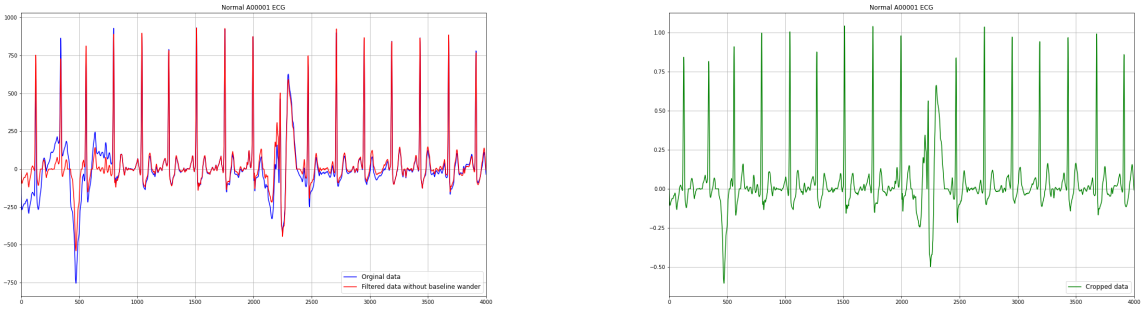

Figure 3: Preprocessed signal after baseline wander removal (left) and normalisation (right)

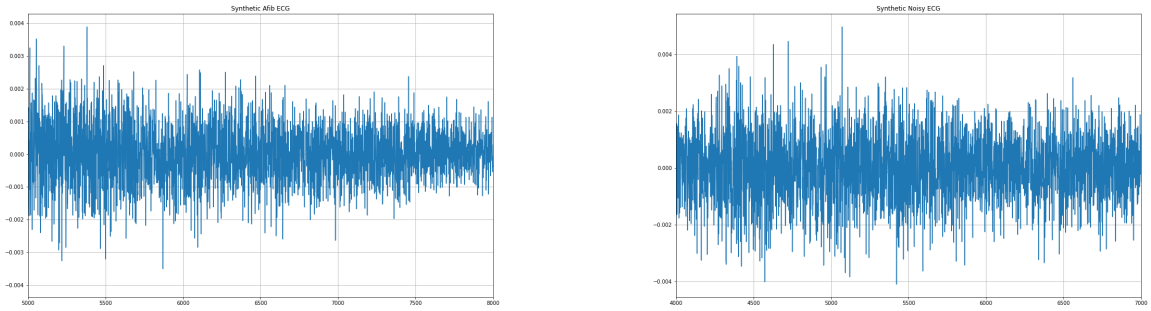

Figure 4: Synthetics signal for Atrial fibrillation (left) and Noisy recordings (right)

Table 9: Generator's summary

Layer	# parameters	kernel	stride
ConvTranspose1d	1024	256	2
ConvTranspose1d	2272	71	2
ConvTranspose1d	8192	64	2
ConvTranspose1d	18432	36	2
ConvTranspose1d	1088	34	2
LSTM	84,992	-	-
Tanh	-	-	-

Table 10: Discriminator's summary

Layer	# parameters	kernel	stride
Conv1D	160	256	4
LeakyReLU	-	-	-
Conv1D	4640	128	3
LeakyReLU	-	-	-
Conv1D	18496	64	3
LeakyReLU	128	-	-
LSTM	39168	-	-
Conv1D	2080	8	2
Dense	132	-	-
Dense	132	-	-
Sigmoid	-	-	-

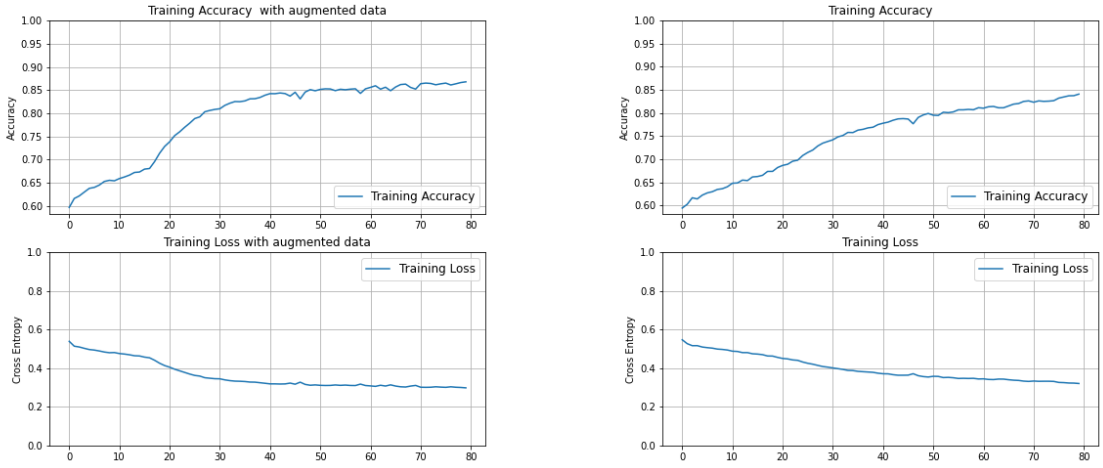

Figure 5: Training profile with augmentation (left) and without augmentation (right).