1D Convolutional Neural Networks for Detecting Atrial Fibrillation

Khalil Ben Kalboussi, Lazhar Khriji, Farah Barika Ktata and Abdulnasir Hossen

March 3, 2020

1D CONVOLUTIONAL NEURAL NETWORKS FOR DETECTING ATRIAL FIBRILLATION

Khalil BEN KALBOUSSI[*], Lazhar KHRIJI[**], Farah BARIKA KTATA[*], and Abdulnasir HOSSEN[**]

[*]High. Inst. of Applied Sciences and Technology Sousse ISSATSO, University of Sousse, Tunisia.
[**]College of Engineering, Sultan Qaboos University, Oman.
kalboussikhalil@gmail.com, lazhar@squ.edu.om, farah.ktata@gmail.com, abhossen@squ.edu.om

Abstract—**This work is a part of the mobile health monitoring system project in Sultan Qaboos' university, Muscat Oman. We explain in this work an effective and precise method of detecting Atrial Fibrillation from a single channel short electrocardiogram (ECG). The used ECG signals are downloaded from the Physionet/Computing in Cardiology Challenge 2017. Signals lengths varies between thirty and ninety seconds. The outputs are 3 different classes, Atrial Fibrillation (AF) Normal (N) and Noisy ($\sim$). The proposed model is based on a deep learning one dimensional Convolutional Network, eliminating the need to manually extract features. R-peaks are detected using python's BioSPPy library then R to R intervals are calculated, stacked into a dataframe, amputated and parsed with a manually chosen value then injected into the neural network. The RR records are classified next into one of the three classes. The proposed model has reached 98% training accuracy, 96% validation accuracy and 94.07% testing accuracy.**

Index Terms—**ECG, Machine Learning, Signal processing, E-Health, Biomedicine**

I. INTRODUCTION

The Ministry of Health of Oman showed in its annual report of year 2017 that about 25 per cent of all hospital deaths were due to heart diseases and circulatory system ailments. Atrial fibrillation (AF) is the most common cardiac arrhythmia, and is the major risk factor for death, stroke, hospitalization, heart failure and coronary artery disease [1]. AF can be detected by visually analyzing Electrocardiograph (ECG) signals from the patients, and detecting arrhythmic heart beats. Detecting AF automatically have been the subject of research in the last years. Many techniques have been applied to extract the most relevant features in order to detect AF, based on time-frequency analysis of ECG [2]–[4], RR intervals analysis [3] and heart rate variability analysis (HRV) [4]. In this work we try to automatically detect AF, by classifying the ECG signals into one of the three classes (Normal, Atrial Fibrillation, Noisy). The paper will be divided into two parts, in the first part *Methodology* we will explain the methodology and the proposed Model, second part will demonstrate our *results*.

II. RELATED WORK

A lot of research works have been conducted in the field of preprocessing and classifying ECG signals. Many methods have been proposed, in this section we conduct a profound study in order to familiarize some of the important and effective techniques for preprocessing, and models for classification. Joachim A. in [5] proposed a pre-processing method to extract features after detecting R-peaks. The RR interval time-series was first calculated then the signal quality was estimated on a second-by-second basis and the continuous sub-segment with the highest quality was selected for further analysis. A number of features were estimated: heart rate variability (time domain based, fragmentation, coefficient of sample entropy etc.), ECG morphology (QRS length, QT interval etc.) and the presence of ectopic beats. The features were used to train support an SVM. The included test metric is only F1, which reached 80%. In [6] Lucia B. and al. have used fifty features based on the ECG signal, derived from the RR series and obtained combining QRS morphology and rhythm. After applying a stepwise linear discriminant analysis, only thirty features are used in LSVM classifier. The F1 score reached 81%. Guangyu B. proposed a model in [7] including a method for extracting thirty features based on AF Features, ECG Morphology, RR intervals, Similarity of QRS, Similarity of R amplitude, Ratio of high similarity, and Signal Qualify. A decision tree based classifier model was utilized in the work to obtain an F1 = 86%. Another work conducted by Pietro B. et al. [8] includes a two stages RUSBoost model to classify the signals. The first stage aims to distinguish between noisy and non-noisy recordings. In the second stage, the recordings not classified as noisy are delineated and the AF features are extracted, and provided as inputs to an ensemble learning classifier. Before that a modified Pan and Tompkins algorithm is implemented to detect QRS peaks then a range of AF is extracted from the recording describing the spectral properties of the hearth rate variability (HRV), the ECG signal morphology, the complexity of both the ventricular and the atrial activity, and a variety of other atrial activity indices for irregularity and variability analysis. An F1 score = 75% is reached. In other works like [9] Chandra B. and al. did not extract features. First the signal is normalized to be in [-1,1] then it is passed thru a filter to detect the base line, then that base line is subtracted from the original signal to smooth it. R peaks are detected to form templates then injected into a CNN. An overall F1 score of 71% is reached. Another work [10] by Ivaylo C. and al. used two techniques to eliminate noise and low amplitude in normal ECG signals 1) The signal is stepwise amplified until the detector starts to register QRS. 2) The noise immunity is improved by zeroing the signal during 300ms around the detected noise and then the QRS detector is restarted. Then HRV features are calculated from RR-Tachogram like mean value, median value, standard deviation, mean deviation, ratio of mean-to-median value, etc. and from dRR-Tachogram like proportion of RR intervals differing by $>$ 50ms from the preceding RR interval, square root of the mean squared differences of successive RR intervals, etc. along with some other features like Average beat, P-waves and amplitudes. A Linear Discriminant Analysis (LDA) classifier has been implemented in this work to give an accuracy of 80%. Erin E. C. and al. [11] proposed a classic features extraction technique based on the ECG characteristics: Ventricular response features, Atrial activity features and Other ECG features like Average spectral power, Variance of spectral power, Root mean square fluctuation of time series, Average total power of time series, etc. a decision tree is implemented with a SMOTE (Synthetic Minority Oversampling Technique) for increasing the sample size of the minority classes. F1 score of this model have reached 78,55%. Matthieu D. S. and al. in [12] also proposed a classic method beginning with filtering ach ECG signal with cutoff frequencies of 1 and 50Hz. A Pan and Tompkins s QRS detector was then used RR intervals were extracted then features were extracted depending on the heart rate variability. Template-based features were also used along with signal quality features. After that an adaboost classifier was implemented to give an F1 score = 76% .

III. Electrocardiogram sampling

Electrocardiogram (ECG), representing a patients heart beats in the time interval, was first used for calculating the intervals between heart beats (RRI) an therefore calculate the heart rate variability (HRV) to use as a marker of fetal distress in 1965, and it was recognized subsequently as a noninvasive marker of autonomic activity [13]. HRV is used for predicting mortality in ceveral cenarios, and the early detection of certain heart diseases. It have proved it's effeciency in prediction of sudden heart attacks [14] and diagnosis of overdose and poisoning [15] . The ECG sampling frequency required to ensure sufficient precision and texture of the signal have been the axis of multiple studies. Although physicists recommends a minimum sampling frequency of $\geq$ 500 Hz, saying that a smaller sampling frequency may result in stronger high-frequency noises in the signal, some studies have proved that lower sampling frequencies, such as 100 Hz or even 50 Hz, might be sufficient with interpolation [16], [17]. While acquiring an ECG signal, a vector of points is created to form a graph. The length of this vector varies with the sampling frequency and the recording duration. In this work we have used a dataset of real patients ECG signals, taken with a frequency of 300Hz and varying in duration from 10 to 60 seconds.

IV. Dataset

The ECG signals we used are one lead, short signals collected from real patients, downloaded from physionet's website . These signals has been put to the public in the frame of a medical computing challenge ”PhysioNet CinC Challenge 2017” [2]. The signals files are in .MAT extension files which is developed by Mathworks for use with the MATLAB software. In this work, these files were read and treated using Python. The data inside the files are in the form of a matrix describing the sample number and the signal amplitude. The length of signals vary from one to another (from thirty to ninety seconds). The signals has been sampled with a 300 Hz frequency. 8528 Records are available on the site. After removing the ”Other” class, leaving only AF, N and ~, a data set of 5971 records were used in the training. A separate 300 records were downloaded, 230 are used for testing and verification.

V. Methdology

In this section we will present the method we used in pre-processing then we will explain the architecture of the proposed CNN model. The idea is to make use of the internal filters of the CNN so that we don't have to apply any features extraction techniques like adaptive threshold and principal component analysis, wavelet transform or any signal processing manipulation. To work on clean signals and reduce errors caused by noise, we have applied a bandpass Butterworth. The next section explains the caracteristics of the filter and demonstrates the results of the learning process while changing the order of this filter.

A. Butterworth filter

The Butterworth filter is a type of signal processing filter designed to have a frequency response as flat as possible in the passband. It is also referred to as a maximally flat magnitude filter. The general transfer equation of this filter is as follows

$$H_{(jw)} = \frac{1}{\sqrt{1+\varepsilon^2(\frac{\omega}{\omega_p})^{2n}}}$$

Where: n represents the filter order, Omega ω is equal to $2\Pi f$ and Epsilon ε is the maximum pass band gain, (Amax). If Amax is defined at a frequency equal to the cut-off -3dB corner point (fc), ε will then be equal to one and therefore ε^2 will also be one. In this work we applied a butterworth passband filter with a cutoff frequencies of 3 and 40Hz, which is the range of typic noise in ECG signals. We used a 7th order filter based on a comparative experimental study applied on filters to observe their response time and effect of the final precision of the model explained later in thin paper. Demonstrated in table I the best precision is for 7th order filter.

Filter order	Execution time	Obtained final precision
2nd order	5.015 minutes	93.87%
3rd order	4.384 minutes	94.01%
4th order	4.358 minutes	93.28%
5th order	4.331 minutes	94.07%
6th order	4.303 minutes	93.74%
7th order	4.320 minutes	94.07%
8th order	4.343 minutes	93.15%

TABLE I: Effect of filter order on the final precision

Yet the application of this filter is perfect for our application, it may not be suitable for other techniques used on ECG. As our technique, explained later, uses only the R peaks to generate another graph, other techniques uses many other caracteristics from ECG signal that maybe lost using this time of filter. Fig 1 illustrates the behaviour of a butterworth filter on a noisy ECG signal.

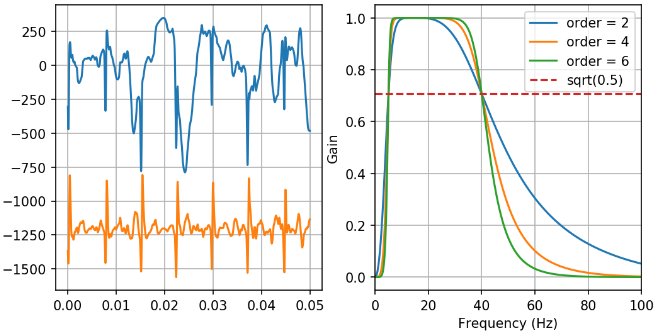


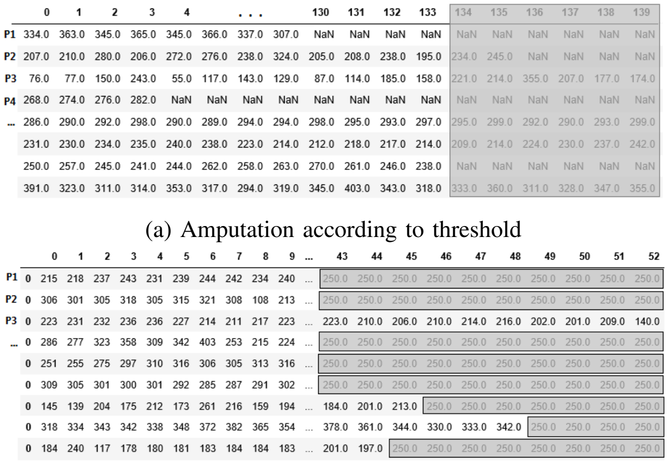
Fig. 1: Butterworth filter behaviour

B. Pre-Processing technique

As R peaks are the most important component in ECG signals and are the most important agents that are used to diagnose AF, detecting R peaks is a critical task, which led us to implement a library “BioSPPy” which offers many bio-signals processing tools like filtering, reading, plotting, detecting R peaks and many other functions. We used this library to detect R peaks, and then with a simple procedure, calculate each interval between two peaks. Each signal has its duration and each one contains a number of heart beats and R-R intervals different from the other which leads to a non-uniform data set as shown in fig 2.

Supposing that S is the ECG signal with $x = x_0, x_1, x_2, \ldots, x_n$, with n+1 the number of samples inside the signal, n can be calculated by $n =$ signal duration*sampling rate. After detecting R peaks another vector is obtained $p = p_0, p_1, p_2, \ldots, p_m$where $m+1$ is the number of elements in the vector, m cannot be calculated because it depends on the patient's heart rate. R-R intervals are also stored in a vector defined as follows $i = i_0, i_1, i_2, \ldots, i_k$ where $k = m - 1$ and i is defined as a sequence $i_n = p_{(n+1)} - p_n$ The difference in length will lead to empty (NAN) cells inside the data-frame fig 2 because the width of the data frame will be determined by the longest signal. To treat this issue we have dropped the columns that exceeds a certain threshold Fig 2a and replace the missing values, in the columns that didn't reach the threshold, with a chosen value Fig 2b. The resulting dataframe will be similar to a matrix, with each row representing the data of one patient. By changing the value of the threshold the number of the columns in the dataframe change accordingly.

the psedudo code and the flochart in Fig 3. explains the preprocessing procedure:



	0	1	2	3	4		. . .		130	131	132	133	134	135	136	137	138	139
P1	334.0	363.0	345.0	365.0	345.0	366.0	337.0	307.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
P2	207.0	210.0	280.0	206.0	272.0	276.0	238.0	324.0	205.0	208.0	238.0	195.0	234.0	245.0	NaN	NaN	NaN	NaN
P3	76.0	77.0	150.0	243.0	55.0	117.0	143.0	129.0	87.0	114.0	185.0	158.0	221.0	214.0	355.0	207.0	177.0	174.0
P4	268.0	274.0	276.0	282.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	286.0	290.0	292.0	298.0	290.0	289.0	294.0	294.0	298.0	295.0	293.0	297.0	295.0	299.0	292.0	290.0	293.0	299.0
	231.0	230.0	234.0	235.0	240.0	238.0	223.0	214.0	212.0	218.0	217.0	214.0	209.0	214.0	224.0	230.0	237.0	242.0
	250.0	257.0	245.0	241.0	244.0	262.0	258.0	263.0	270.0	261.0	246.0	238.0	NaN	NaN	NaN	NaN	NaN	NaN
	391.0	323.0	311.0	314.0	353.0	317.0	294.0	319.0	345.0	403.0	343.0	318.0	333.0	360.0	311.0	328.0	347.0	355.0

(a) Amputation according to threshold

		0	1	2	3	4	5	6	7	8	9	...	43	44	45	46	47	48	49	50	51	52
P1	0	215	218	237	243	231	239	244	242	234	240	...	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
P2	0	306	301	305	318	305	315	321	308	108	213	...	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
P3	0	223	231	232	236	236	227	214	211	217	223	...	223.0	210.0	206.0	210.0	214.0	216.0	202.0	201.0	209.0	140.0
...	0	286	277	323	358	309	342	403	253	215	224	...	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
	0	251	255	275	297	310	316	306	305	313	316	...	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
	0	309	305	301	300	301	292	285	287	291	302	...	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
	0	145	139	204	175	212	173	261	216	159	194	...	184.0	201.0	213.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0
	0	318	334	343	342	338	348	372	382	365	354	...	378.0	361.0	344.0	330.0	333.0	342.0	250.0	250.0	250.0	250.0
	0	184	240	117	178	180	181	183	184	184	183	...	201.0	197.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0	250.0

(b) replacing missing values

Fig. 2: Amputation and replacing missing data

```
Begin
    Read .CSV Labels file
        For each Label
            If Label = N then Label <= 0
            If Label = A then Label <= 1
            If Label = ~ then Label <= 2
        End for
        Drop others
    Read .MAT Signals files
        For each file
            Treat ECG signal using Biosppy
            library(filtering and R peaks
            detection)
            Calculate R-R interval
            append to dataframe
        End for
    Drop columns with number of NaN > Thresh
    Replace remaining NaN values with Val
    Split Data (25% Testing, 75% training)
    Inject to Neural Network,
    begin Learning
    End Learning and save model
END
```

C. Convolutional Neural Networks

Convolution is a function derived from two other functions by integration which describes how one of them modifies the other. The mathematical expression of convolution is defined as follows:

$$(f * g)(t) \equiv \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d(\tau)$$

A Convolutional Neural Network (CNN) applies filters (also called kernels or feature detectors) to the input to detect patterns and extract a feature map. Different types of CNNs exist, depending on the application we can choose between one, two or three dimensional CNN. The key difference is the dimensionality of the input data and how the feature detector (or filter) slides across the data. In the one dimensional CNN

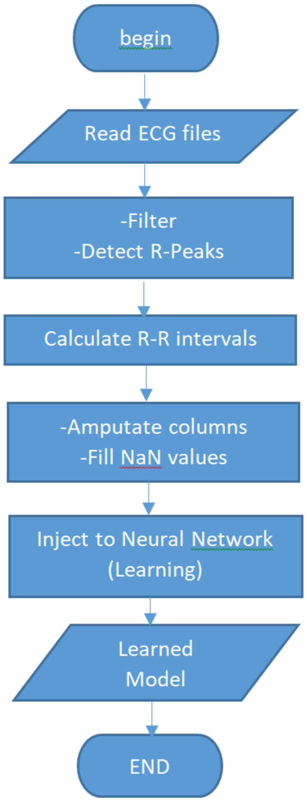


Fig. 3: Method's Flowchart

the filters run thru the data row by row (assuming that the data is a table) where the selected dimension of the data is one by X where X is the number of features, but in two dimensional CNN the filters go matrix by matrix with two dimensions, X and Y. this is called Kernel convolution, it is mainly used Computer Vision algorithms. The process of CNN filtering or applying kernels is where we take a small matrix of numbers (so called kernel or filter), we run it thru the data or image and transform it based on the values from this filter. A calculated feature map values according to the following formula is obtained, where the input data is denoted by f and our filter by h. The indexes of rows and columns of the result matrix are marked with m and n respectively.

$$G[m,n] = (f * h)[m,n] = \sum_{j}\sum_{k} h[j,k]f[m-j,n-k]$$

In this work we have chosen to implement a one dimensional CNN applying the filters on each row at a time in the dataframe, representing one patient. this choice was made considering the natural structure of the data, as one vector representing one patient is not connected by any means to any of the other vectors. In this case using a two dimensional CNN and applying filters to more than one row together is useless and can affect the learning negatively.

D. Proposed machine learning model

The model we are proposing is based on a one dimensional Convolutional Neural Network [fig 4]. After the pre-processing the resulting R-R records will be injected into the input layer. After it three hidden layer are applied with respectively 100, 150 and 100 nodes. The input and the first hidden layer are activated using the Rectified Linear Unit function (ReLU), the second and third hidden layers are activated using nonlinear activation (sigmoid). Then a flattening function is applied to convert the two dimensional output of the previous layer to a single dimension array to inject into a fully connected output layer.

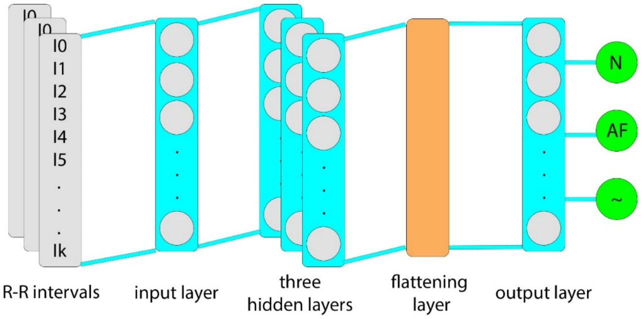


Fig. 4: Proposed 1D Convolutional Neural Network Model

VI. Tests and Resluts

Precision is the number of True Positives divided by the number of True Positives and False Positives.

$$Precision = \frac{TP}{TP + FP}$$

Put another way, it is the number of positive predictions divided by the total number of positive class values predicted. It is also called the Positive Predictive Value (PPV). Recall is the number of True Positives divided by the number of True Positives and the number of False Negatives.

$$Recall = \frac{TP}{TP + FN}$$

Put another way it is the number of positive predictions divided by the number of positive class values in the test data. It is also called Sensitivity or the True Positive Rate. F1 is also called the F Score or the F Measure and defined by the equation

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Put another way, the F1 score conveys the balance between the precision and the recall

The tests have been conducted on two machines 1) an intel I3 CPU, 8Gb Ram machine running python in anaconda environment and jupyter notebook IDE 2) an intel I7 CPU, 8Gb Ram machine also running python in anaconda environment and jupyter notebook IDE. The results and comparative study are illustrated in table II

	Filtering technique	Features extraction	classifier	F1 score
Joachim A. et al.	Filter	R-peaks detection Feature calculation	SVM	80%
Lucia B. et al.	Filter	QRS detection Feature calculation	LSVM	81%
Guangyu B. et al.	Filter	Feature calculation	Decision tree	86%
Pietro B. et al.	Filter	QRS detection Feature calculation	Decision tree	75%
Chandra B. et al.	Normalizing Filter	R-peaks detection Feature calculation	CNN	71%
Ivaylo C. et al.	Adaptive amplification Noise detection	QRS detection Feature calculation	LDA	80%
Erin E. C. et al	Filter	Feature calculation	Decision tree + SMOTE	78,5%
Matthieu D. S. et al.	Filter	QRS detection Feature calculation	Decision tree	76%
Proposed Architecture	Filter ND&SD technique	R-peaks detection	1DCNN	**94%**

TABLE II: Proposed architecture results and comparative study

VII. Conclusion and Perspectives

In this work, we showed the developing process of a very cost-effective mobile health monitoring system. The platform has successfully passed the real case usage tests. A machine-learning model has been developed and successfully tested. An accuracy of 94% on the testing data set has been reached yet many improvements can be done. Although the fourth class *"Other"* is not mandatory in our context and can be discarded, it is not practical in other uses to classify ECG signals without recognising other abnormalities than Atrial Fibrillation. For this issue we are working on adding the Other class without dropping the accuracy of the classifier. A hybrid classifier can be used for this purpose, conserving the actual three class CNN classifier and adding a two classes seperate classifier to distinguish between AF and Other Classes. The classifier can be a two dimensional CNN comparing QRS templates, or a decision tree treating the ECG as a single dimensional vector.

Acknowledgment

This work have been conducted in collaboration with Sultan Qaboos university, Muscat, Oman and the Higher institut of applied sciences and technology, Sousse, Tunisia within OmanTel grant "EG/SQU-OT/18/01"

References

[1] L. GYH et al. "Atrial fibrillation, Nature Reviews". Disease Primers 2 2016;16016.

[2] F. Andreotti et al. “Comparing Feature-Based Classifiers and Convolutional Neural Networks to Detect Arrhythmia from Short Segments of ECG” Computing in Cardiology 2017; VOL 44

[3] L. Billeci et al. “Detection of AF and Other Rhythms Using RR Variability and ECG Spectral Measures” Computing in Cardiology 2017; VOL 44

[4] Z. MEI et al. “Automatic Atrial Fibrillation Detection Based on Heart Rate Variability and Spectral Features” IEEE Access 2018 ; Vol 6

[5] A. Joachim et al., "Rhythm and Quality Classification from Short ECGs Recorded Using a Mobile Device," IEEE Computing in Cardiology, 2017.

[6] B. Lucia et al., "Detection of AF and Other Rhythms Using RR Variability and ECG Spectral Measures," IEEE Computing in Cardiology, 2017.

[7] G. B. et al., "Detection of Atrial Fibrillation Using Decision Tree Ensemble," Computing in Cardiology , 2017.

[8] B. Pietro et al., "Detection of Atrial Fibrillation Episodes from Short Single Lead Recordings by Means of Ensemble Learning," Computing in Cardiology, 2017.

[9] B. Chandra et al., "Atrial Fibrillation Detection Using Convolutional Neural Networks," IEEE Computing in Cardiology , 2017.

[10] C. Ivaylo et al., "Multi-parametric Analysis for Atrial Fibrillation Classification in ECG," Computing in Cardiology , 2017.

[11] E. C. Erin et al., "Atrial Fibrillation Classification from a Short Single Lead ECG Recording Using Hierarchical Classifier," Computing in Cardiology, vol. 44, 2017.

[12] D. S. Mathieu et al., "Combining Template-based and Feature-based Classification to Detect Atrial Fibrillation from a Short Single Lead ECG Recording," Computing in Cardiology, vol. 44, 2017.

[13] Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology. "Heart rate variability: standards of measurement, physiological interpretation and clinical use." Circulation. 1996;93(5):1043–1065.

[14] Liu T, Lin Z, Ong ME, Koh ZX, Pek PP, Yeo YK, et al. "Manifold ranking based scoring system with its application to cardiac arrest prediction: a retrospective study in emergency department patients." Comput Biol Med. 2015;67:74–82.

[15] Jeong J, Kim Y. "Heart rate variability analysis in acute poisoning by cholinesterase inhibitors." Signa Vitae. 2017;13(2):33–40

[16] Ziemssen T, Gasch J, Ruediger H. "Influence of ECG sampling frequency on spectral analysis of RR intervals and baroreflex sensitivity using the EUROBAVAR data set." J Clin Monit Comput. 2008;22(2):159–168.

[17] Mahdiani S, Jeyhani V, Peltokangas M, Vehkaoja A. "Is 50 Hz high enough ECG sampling frequency for accurate HRV analysis?" Conf Proc IEEE Eng Med Biol Soc. 2015;2015:5948–5951.

[18] Ohhwan Kwon, MD,1 Jinwoo Jeong et al. "Electrocardiogram Sampling Frequency Range Acceptable for Heart Rate Variability Analysis." Healthcare Informatics Research. 2018;24(3):198