Received 15 June 2024, accepted 29 June 2024, date of publication 2 July 2024, date of current version 10 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3422329

RESEARCH ARTICLE

Atrial Fibrillation Detection From Electrocardiogram Signal on Low Power Microcontroller

MUHAMMAD YAZID[1], (Senior Member, IEEE), MAHRUS ABDUR RAHMAN[2], NURYANI NURYANI[3], AND ARIPRIHARTA[4], (Member, IEEE)

[1]Department of Biomedical Engineering, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia
[2]Faculty of Medicine, Universitas Airlangga, Surabaya 60115, Indonesia
[3]Department of Physics, Faculty of Science, Universitas Sebelas Maret, Surakarta 57126, Indonesia
[4]Department of Electrical Engineering and Informatics, Faculty of Engineering, Universitas Negeri Malang, Malang 65145, Indonesia

Corresponding author: Muhammad Yazid (yazid@bme.its.ac.id)

This work was supported by the Indonesian Collaborative Research Program (Riset Kolaborasi Indonesia, RKI) 2023 under Grant 1671/PKS/ITS/2023.

ABSTRACT In this paper, we proposed the implementation of a simple and lightweight Atrial Fibrillation detection based on an improved Variable Step Dynamic Threshold Local Binary Pattern algorithm. Using feature selection based on correlation and statistical significance, we can reduce the input feature size to just 44 features without significantly degrading classification accuracies. Tested on 15-second signal segments from the MIT-BIH Atrial Fibrillation Database and combined with a support vector machine classifier, the proposed method can achieve 99.14% sensitivity, 99.12% specificity, and 99.13% accuracy. When the input signal length is 60 seconds, the sensitivity, specificity, and accuracy are 99.49%, 99.46%, and 99.47%, respectively. The reduced input feature size results in a machine learning model size as small as 132.86kB when the input signal length is 60 seconds. When implemented on an Arm Cortex M4-based STM32F413ZHT3 microprocessor with 100MHz clock frequency, the proposed method can achieve similar performance as a PC-based system with an average current consumption of just 27mA. The embedded C program can fit in as small as 114.46kB of flash memory and complete one SVM inference as fast as 11.28ms. The results presented in this paper show that it is possible to do highly accurate machine learning classification that can detect Atrial Fibrillation from ECG signals on a low-power, constrained resource microcontroller. Our results will make developing a high quality, low cost, and low power wearable smart medical electronic device for detecting atrial fibrillation from ECG signal much easier.

INDEX TERMS Health, cardiology, preventable disease, electrocardiogram, arrhythmia, atrial fibrillation, machine learning.

I. INTRODUCTION

A. BACKGROUND

Atrial Fibrillation (AF) is one of the most common types of arrhythmia in the world [1]. It has been associated with increasing risk of other diseases, including potentially deadly cardiovascular diseases such as stroke [2] or non-cardiovascular diseases like dementia or Alzheimer's disease [3], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Fabian Khateb.

Researchers have identified AF burden as an important parameter in predicting AF-related risk of stroke. AF burden is defined as the overall time spent in AF during a specified period and is expected to change dynamically within a minimum and a maximum range. For this reason, continuous monitoring of the occurrence of AF is important in the diagnosis and characterization of AF in a patient [5].

Conventionally, long-term monitoring of AF is done using a wearable Holter device, capable of recording electrocardiogram (ECG) signals for a long period of time. A trained medical expert will then check the long ECG recording for abnormal beats, including AF beats. This process is tedious and time-consuming, which will be a huge burden to the medical staff. Furthermore, in some countries, there are not enough medical experts available who are trained to diagnose cardiovascular diseases from ECG signals.

B. PREVIOUS WORKS ON DETECTION OF ATRIAL FIBRILLATION BY MACHINE LEARNING

In order to solve the problems associated with conventional methods, an automatic method of detecting atrial fibrillation beats from ECG signals is necessary. Based on machine learning, this algorithm can learn from expert annotated signals and use the learned patterns to detect similar ECG beats in new signals. Previous research has shown the necessity and potential of using machine learning-based analysis of ECG signals [4], [6], [7], [8], [9]

Due to its ability to automatically detect significant features, several previous works have proposed deep learning-based methods for detecting abnormalities, including atrial fibrillation and other arrhythmia, from ECG signals. These methods enable researchers to leverage advances in electronic engineering technologies, such as the latest central processing unit (CPU) or graphical processing unit (GPU) having large computing powers.

In work by Mahmud et al., [10], a two-dimensional deep convolutional neural network (2D CNN) is modified for use with one-dimensional (1 D) signals. The authors implemented a method called depthwise separable 2D convolution, where the spatial and inter-channel convolution operations are performed separately, reducing model complexities and computation time. The method proposed in [10] achieved a high accuracy of 99.28% when tested on a 5-class classification of ECG data in the MIT-BIH Arrhythmia Dataset. However, the work by [10] assumes a beat-by-beat segmented input ECG signal, making its accuracy dependent on the accuracy of the beat detection and segmentation algorithm [11].

Petmezas et al. [12] combined convolutional neural network (CNN) and long short-time memory (LTSM) to achieve 97.87% sensitivity and 99.29% sensitivity when used on the MIT-BIH Atrial Fibrillation Database (AFDB) to classify ECG beats into four classes (Normal, Atrial Fibrillation, Atrial Flutter, and AV Junctional Rhythms). While this method can work with a very short input signal length (one beat), similar to the work by [10], its accuracy depends significantly on the accuracy of the beat detection method used.

C. PREVIOUS WORKS ON RESOURCE-CONSTRAINED DEVICE-BASED AUTOMATIC ATRIAL FIBRILLATION DETECTION

Recent advances in semiconductor and microelectronic technologies have allowed the development of small-sized, battery-powered electronic biomedical devices that can be worn by a person and used continuously with minimal disturbance in their daily activities. If monitoring and detection of patients' conditions can be implemented in these devices, the quality of life of the patients and their families will be improved significantly, and the work burden of medical staff will be reduced.

Deep learning-based methods can achieve relatively high classification accuracies at the cost of a high computation load and large memory requirement. These trade-offs mean implementing a deep learning-based detection algorithm on a low-power and low-cost wearable microcontroller device is not trivial. Previous works have been able to walk around this problem using cloud-based detection [13] or edge hardware with a relatively large computing power such as a Raspberry Pi board [14] or NVIDIA Jetson Nano [15].

Tseng et al. [13] propose a sliding large kernel-based deep learning algorithm for processing and classifying ECG signals acquired by mobile devices. The method proposed by [13] takes advantage of the currently ubiquitous wireless high-speed internet connection to transmit ECG data from a mobile device to a network server. The ECG signals are converted into images in the server and processed using a deep learning method.

Farag et al. [14] proposed using a short-time Fourier transform (STFT) based one-dimensional convolutional layer to generate a spectrogram from the input ECG signal. These one-dimensional features are then converted into two-dimensional heat maps and classified using two-dimensional convolutional neural networks. Using the MIT-BIH Arrhythmia Database and running on an edge device based on Raspberry-Pi 3 model B+, [14] achieved 99.1% classification accuracy while using up to 12MB of random access memory.

Seitanidis et al. [15] used two-dimensional convolutional neural networks (2D-CNN) to classify two-dimensional image representations of ECG signals. The input ECG signals are segmented by beats, and each beat is converted into a 128 $\times$ 128 pixel grayscale image. Using NVIDIA Jetson Nano as the edge device, [15] achieved 95.3% accuracy when classifying ECG signals from the MIT-BIH Arrhythmia Database.

The requirement of using cloud-based or high computational power hardware imposes severe limitations on a wearable medical device. The necessity to transmit all recorded signals to the cloud for classification means large data transmission costs will be incurred. The energy required for the wireless data transmission will also be significant, reducing the device's battery life or requiring larger batteries to be used, degrading comfortability and usability [16], [17]. Using a system with large computing power means there will be a higher heat dissipation on top of the larger energy consumption, both of which are not ideal for a wearable medical device. In addition, transmitting and storing a patient's raw biomedical signal to the cloud will pose significant security and privacy-related risks [18]. Platforms with higher computing power and larger memory capacity are also typically more expensive, increasing production costs and the financial burden for patients.

For the above reasons, developing an ECG classification method that can be run on an edge device with limited computing resources is essential.

Reducing the computation load of the detection system can be achieved by using manually selected features-based machine learning. Instead of having the machine learning model automatically find significant features of the input signals, they are previously selected by a human expert. The selected input features are then used as the input vector for a machine learning classifier such as a support vector machine (SVM). Using extracted features as classifier input reduces the input dimension and the search space for the machine learning model training. This approach will reduce computing power and memory size requirements for the hardware.

Chen et al. [19] selected five features consisting of statistical parameters of the RR intervals for use with a simple artificial neural network (ANN) classifier with only a single hidden layer with three nodes. While the training phase of the machine learning model was done on a personal computer (PC), the trained model was embedded in a low-power microcontroller for the on-the-edge inference. This method achieved 94.5% classification accuracy and a 2.22ms processing time when used on 15 seconds length ECG signals from the China Physiological Signal Challenge 2018 (CPSC2018) [20].

Ganapathy et al. [21] proposed a method based on dynamically assigned symbolic representation of RR intervals of an ECG signal as the input feature to conventional classifiers. In [21], each RR intervals are divided into several classes based on its lengths, using threshold values dynamically determined from the average value of the RR intervals of the whole signal. The generated time series of codes are then converted into a cooccurrence matrix, which will be the input features for machine learning. Tested on the Atrial Fibrillation Prediction Challenge Database (AFPDB) and the Atrial Fibrillation Termination Challenge Database (AFTDB), the method proposed by [21] achieved 94.0% and 99.8% classification accuracies, respectively.

Previously, in the work by Yazid and Mahrus [22], we proposed a new feature extraction algorithm for the detection of atrial fibrillation from ECG signals called Variable Step Dynamic Threshold Local Binary Pattern (VSDTLBP). Consisting of only simple arithmetic and logic operation, our proposed feature extraction method can result in a 99.11% sensitivity and 99.29% specificity when combined with an SVM classifier and applied to 60 seconds length ECG signals from the MIT-BIH Atrial Fibrillation Database.

The relatively simple computation involved in the VSDTLBP algorithm and its significantly higher classification accuracy among the reported works on Atrial Fibrillation detection algorithms make it a suitable candidate for implementing a wearable AF detection device on a low-power microcontroller.

D. CONTRIBUTION OF THIS WORK

In this paper, we present an improvement of the algorithm proposed in [22] and discuss its efficient implementation on low-power microcontrollers. The method proposed in this work will be shown to achieve high classification accuracy even when implemented as an embedded application in a low-power and low-cost microcontroller. The memory footprint and the computation time required by the proposed method are relatively small and suitable for an at-the-edge classification of Atrial Fibrillation.

The contributions of this work are:

1) Improvements to the VSDTLBP algorithm to make it more suitable for use in low-power microcontrollers.
2) Efficient implementation of the VSDTLBP algorithm in constrained resource edge device/microcontroller.
3) Classification accuracy of the proposed method is shown to be comparable with previous works that are based on high computing power systems such as PC or GPU.
4) When compared with previous works on edge device-based systems, the proposed work can achieve a higher classification accuracy with less power consumption and production cost.

The organization of the paper is as follows. First, in section II, we will explain the dataset, data segmentation, hardware, and analysis methods used in this paper. In section III, we will elaborate on our proposed method. Section IV will show and discuss the experiment results, which will then be analyzed and compared with results of previous works in section V. Finally, section VI will be the conclusion.

II. MATERIALS AND METHODS

A. ECG SIGNAL DATASET

In this work, ECG signals from the MIT-BIH Atrial Fibrillation Database (AFDB) [23], [24] and MIT-BIH Arrhythmia Database (MITDB) [24], [25] are used to verify and compare the quality of the developed method with previous works. These publicly available ECG signal databases are commonly used in AF detection research in previous works [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], which will facilitate objective comparison of our proposed method with those works.

1) MIT-BIH ATRIAL FIBRILLATION DATABASE

The MIT-BIH Atrial Fibrillation Database consists of 25 long-time records of ECG signals from human subjects with AF. Each recording is 10 hours long and consists of two ECG signals sampled with 250 samples/second sampling frequency, and provided with manually annotated information of each beats [23], [24].

Only the first of the two available ECG channels in the MIT-BIH Atrial Fibrillation Database is used [22].

TABLE 1. **The number of input signal segments generated from MIT-BIH atrial fibrillation database.**

Input Length [seconds]	All Segments		Balanced	
	Non AF	AF	Non AF	AF
10	50566	33482	33482	33482
15	33660	22277	22277	22277
20	25200	16675	16675	16675
30	16755	11064	11064	11064
40	12521	8270	8270	8270
50	9983	6598	6598	6598
60	8293	5475	5475	5475

TABLE 2. **The number of input signal segments generated from MIT-BIH arrhythmia database 200 series.**

Input Length [seconds]	All Segments		Balanced	
	Non AF	AF	Non AF	AF
10	3666	752	752	752
15	2434	490	490	490
20	1818	357	357	357
30	1204	226	226	226
40	898	165	165	165
50	716	125	125	125
60	596	98	98	98

Following [22], [38], the ECG signals are split into non-overlapping segments of 10, 15, 20, 30, 40, 50, and 60 seconds lengths. The ECG segments are labeled according to the beat annotations included in the database. Segments containing both AF and non-AF beats are discarded, and only segments that are fully AF or fully non-AF are used [22]. Since input data classes are unbalanced, random undersamplings are performed on the larger class to balance it with the other class [22], [36].

Table 1 shows the number of segments generated for each input length from the MIT-BIH Atrial Fibrillation Database. The numbers listed under "Balanced" are the number of signal segments used in the 10-fold stratified cross-validation test after random undersampling to balance the classes.

2) MIT-BIH ARRHYTHMIA DATABASE

The MIT-BIH Arrhythmia Database consists of 48 ECG records from 47 human subjects. Each recording is approximately 0.5 hours long and sampled with 360 samples per second sampling frequency. This database also includes manually annotated information of each beat [24], [25].

The MIT-BIH Arrhythmia Database signals can be divided into 100 series and 200 series signals. Among these, only the 200 series signals contain Atrial Fibrillation episodes. For this reason, only the 200 series signals are used in this work. The signals are resampled into 250 samples/second sampling frequency. Other preprocessing, segmentation, and labeling processes are the same as the MIT-BIH Atrial Fibrillation Database signals. Table 2 shows the number of segments generated for each input length from the MIT-BIH Arrhythmia Database 200 series.

B. HARDWARE IMPLEMENTATION

The proposed method is implemented as an embedded application in STM32F413ZH microcontroller [39], contained in a Nucleo-144 development board produced by STMicroelectronics [40]. The microcontroller is based on the Arm Cortex M4 core [41], with 1.5MB of flash memory and 320kB of random access memory (RAM). The STM32F413ZH microcontroller is also equipped with a hardware floating point unit (FPU) and can run with a clock frequency up to 100MHz [39]. The microcontroller is programmed on the STM32CubeIDE, an integrated development environment (IDE) provided by the manufacturer [42]. To compare its performances on different types of microcontrollers, the proposed method will also be tested on several other microcontrollers produced by STMicroelectronics, including STM32F207ZG, STM32L4R5ZIP, and STM32H7A3ZIQ. Specifications of the microcontrollers used in this work are shown in Table 3.

Unless explicitly specified, if an MCU is equipped with a hardware floating point unit (FPU), it is enabled in all experiments reported in this paper.

C. ANALYSIS METHOD

The performances of the proposed algorithm are measured and compared to previous works using the commonly used sensitivity, specificity, and accuracy metrics. In this work, two experiment settings are used:

1) Experiment Setting 1: In order to compare the performance of the proposed method with previous works, 10-fold stratified cross-validation against the whole balanced segments of the MIT-BIH Atrial Fibrillation Dataset and the MIT-BIH Arrhythmia Dataset is done. The average results of five experiments are presented in this paper.
2) Experiment Setting 2: This experiment setting is mainly used for experiments related to embedding trained machine-learning models in a microcontroller. The available data (after balancing) are first split into train and test segments (80% of the balanced data for training and 20% for testing). In this experiment, 10-fold stratified cross-validation is done on the training segments, and the model with the best accuracy is saved. This model is then used to do inference on the test segments, both as a Python program running on a personal computer (PC) and as an embedded C program on the microcontroller, so that performances can be compared. For the inference on the microcontroller, the test data is loaded onto an SD card for the MCU to access. This experiment is repeated five times and the average results are shown in this paper. Experiment Setting 2 is done only with the MIT-BIH Atrial Fibrillation Database signals.

TABLE 3. Specifications of microcontrollers used in this work.

MCU	Board	Core	Max Clock Freq. [MHz]	RAM Memory [kB]	Flash Memory [MB]	FPU
STM32F207ZG	Nucleo-F207ZG	Arm Cortex M3	120	128	1	No
STM32F413ZH	Nucleo-F413ZH	Arm Cortex M4	100	320	1.5	Yes
STM32L4R5ZI	Nucleo-L4R5ZI-P	Arm Cortex M4	120	640	2	Yes
STM32H7A3ZI	Nucleo-H7A3ZI-Q	Arm Cortex M7	280	1376	2	Yes

In order to show the suitability of the proposed method for use in an embedded system, the size of the generated machine learning model and the extracted support vector array size are also shown and compared with previous work.

This paper also reports embedded memory usage, computation time, and average current consumption of the embedded implementation of our proposed method. Memory usage is provided by the integrated development environment (IDE) used in development (STM32CubeIDE), while computation time is measured using the internal microcontroller timer. MCU average current consumption is measured at the current measurement location specified in the Nucleo-144 board datasheets provided by the manufacturer.

III. IMPROVING VARIABLE STEP DYNAMIC THRESHOLD LOCAL BINARY PATTERN FOR MICROCONTROLLER-BASED MACHINE LEARNING

A. PREVIOUSLY PROPOSED VARIABLE STEP DYNAMIC THRESHOLD LOCAL BINARY PATTERN

Variable Step Dynamic Threshold Local Binary Pattern (VSDTLBP) was proposed in Yazid and Mahrus [22] for use as a feature extraction method in AF detection from ECG signal. In [22], histograms of LBP codes were used as the input to the SVM classifier to classify ECG segments into AF or non-AF classes.

An 8-bit VSDTLBP code corresponding to a particular target data point is calculated from the values of its eight neighbor data points (4 data points before and 4 data points after). The VSDTLBP differs from the conventional LBP in that it specifies an interval (called step) between these data points, which can be adjusted according to signal characteristics and the specific application target. Furthermore, the VSDTLBP algorithm used a dynamically computed threshold value rather than using the value of the center data point as the threshold as in the conventional LBP algorithm. These made the VSDTLBP algorithm easily adaptable to different types of time series signals and classification problems.

An illustration of the operation of the VSDTLBP as described in [22] is shown in Figure 1.

$$LBP(x[i]) = \sum_{r=0}^{\frac{P}{2}-1} \times \left\{ S\Big(x\big[i+(r-\frac{P}{2})\times step\big]-f(x,i,P,step)\Big)2^{r} \right.$$

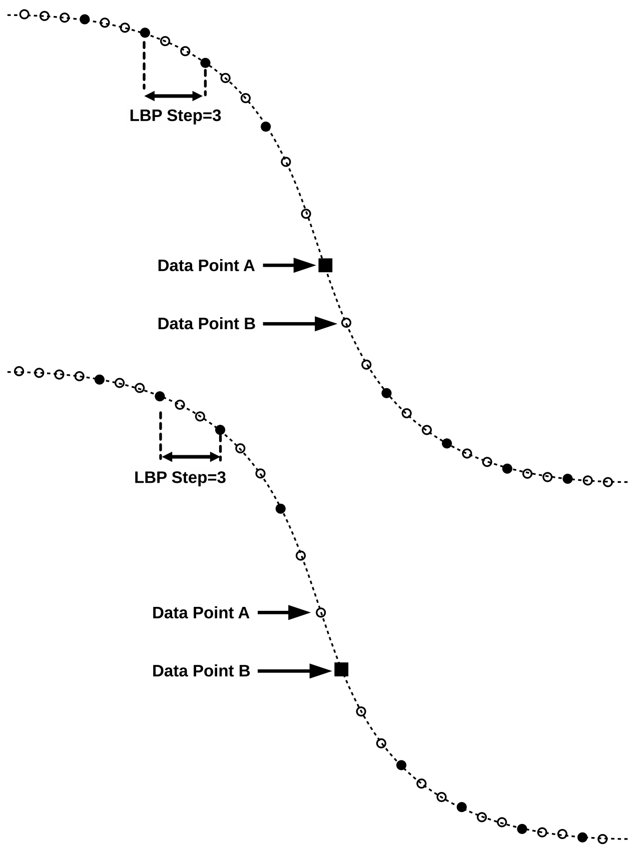

FIGURE 1. Illustration of the variable step LBP method as described in [22]. The black points are data points used for calculating an LBP code. The top picture shows the points associated with the generation of LBP code corresponding to reference data point A, and the lower picture shows the points associated with the generation of LBP code for reference data point B. The LBP Step chosen in this example is 3, which is the distance between adjacent black points [22].

$$\left. + S\Big(x\big[i+(r+1)\times step\big]-f(x,i,P,step)\Big)2^{r+\frac{P}{2}} \right\}$$

$$\text{where} \quad S(a) = \begin{cases} 1, & a \geq 0 \\ 0, & a < 0 \end{cases} \tag{1}$$

$$f(x,i,P,step) = \frac{1}{P+1} \sum_{r=i-(\frac{P}{2})\times step}^{i+(\frac{P}{2})\times step} x[r] \tag{2}$$

Equation 1 and 2 shows the VSDTLBP calculation method mathematically [22]. In these equations, $f(x, i, P, step)$ is the dynamic threshold function, P is the binary bit length of the LBP code, and *step* is the selected step value between data points being considered for generating an LBP code.

In [22], from the 256 possible 8-bit codes generated by Equation 1 and 2, only 58 types of local binary pattern (LBP) codes which are defined as the "uniform" LBP codes by Ojala et al. [43] are used. Uniform LBP codes are LBP codes that have at most two transitions between "0" to "1" or "1" to "0" in its binary form representation. More detailed information about VSDTLBP and LBP can be found in previous works [22] and [43], respectively.

B. ADDITION OF LOW PASS FILTER TO REMOVE ALIASING EFFECT

Since VSDTLBP with step value n will ignore $n - 1$ data points between the data points used in the calculation of each LBP value, any signal component with wavelength less than n times the sampling period will create an aliasing effect, a commonly known negative side effect of downsampling. Using a low pass filter to preprocess the input signal will reduce the aliasing effect and improve the ability of the LBP codes to represent signal features much more accurately, increasing classification accuracy.

In the proposed work, a Butterworth low pass filter (LPF) was used with a cut-off frequency calculated based on Equation 3, where f_c is the LPF cut-off frequency, f_s is the signal sampling frequency, and *step* is the LBP step value. The signal sampling frequency used in this work is 250 samples/second, and the LBP step value is 6, which was found to be optimal for 250 S/s ECG signals in previous work [22].

$$f_c = \frac{f_s}{2 \times step} \tag{3}$$

C. FEATURE SIZE REDUCTION BASED ON CORRELATION AND P-VALUES

While the size of uniform LBP feature vectors is relatively small compared to those of other methods, further reducing the feature vector size can result in a smaller machine-learning model size, minimizing memory usage and computation time. The size of the machine learning model is particularly important for a microcontroller implementation since, typically, there are only a few hundred kilobytes up to a few megabytes of internal flash memory available, which must also be shared with other parts of the programs, such as the graphical user interface.

One way to minimize the feature vector size is by removing redundant features. If more than one features are highly correlated to each other, it is logical that only one of them is necessary since the inclusion of the other correlated features will only add a little information for the machine learning model.

Pearson correlation coefficient has been proposed in previous works as an indicator of feature redundancy [44], [45], [46], [47], [48], [49]. In this work, Pearson correlation coefficient between input features is used as a criterion of feature selection. When two or more features are highly correlated, only one is selected, and the rest are removed. Only 10000 (5000 AF and 5000 non-AF) randomly selected 10-second ECG segments from the MIT-BIH Atrial Fibrillation database ECG signals are used in the feature selection process to prevent overfitting.

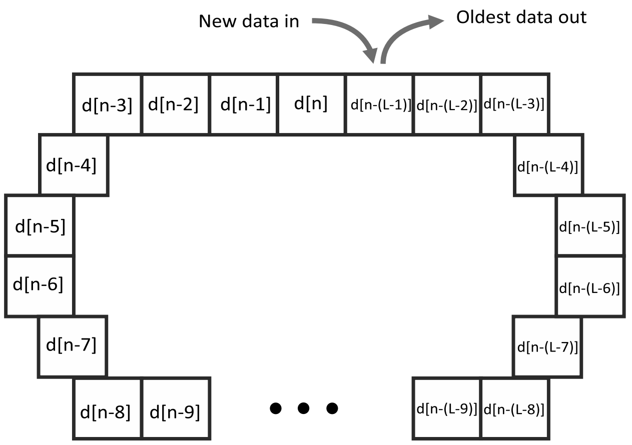

FIGURE 2. Illustration of a circular buffer data architecture with length L. $d[n]$ is the newest data in the circular buffer, while $d[n-(L-1)]$ is the oldest data. When new data arrives, the oldest data is removed, and the newest data is put in its place.

P-value has also been widely used for feature selection, particularly to remove irrelevant features [50], [51]. We perform statistical tests on each feature to determine its importance in distinguishing the two classes (AF and non-AF). Features with large p-values are determined as insignificant and removed.

D. EFFICIENT IMPLEMENTATION USING CIRCULAR BUFFER

Circular buffer is a commonly used data structure in resource-constrained systems such as low-power microcontrollers [52], [53], [54], [55]. Figure 2 illustrates a basic circular buffer architecture that has the length L. In the circular buffer illustrated in Figure 2, $d[n]$ is the latest data inserted in the circular buffer, $d[n-1]$ the data inserted immediately before the latest, and so on. If the length of the circular buffer is L, then $d[n-(L-1)]$ contains the oldest data in it. When new data arrives, the oldest data must be taken out, and this new data is put in its place and becomes the current latest data in the buffer ($d[n]$).

The VSDTLBP and its derivation proposed in this paper can be implemented very efficiently using circular buffer architecture.

In our implementation presented in this work, the AF detection system continuously accepts new ECG data points, generates the corresponding LBP codes, updates the LBP histogram, and uses a trained machine learning model to classify the ECG signal using the LBP histogram as input features.

To save data storage and transmission costs, only the detected AF ECG segment and its surrounding data points need to be stored or transmitted to the cloud. This can be easily implemented using a circular buffer with a length corresponding to the amount of data points to be stored or transmitted whenever AF is detected.

In the proposed implementation, whenever a new ECG datapoint $d[n]$ arrives, a new LBP code corresponding to datapoint $d[n-(4 \times step)]$ can be calculated. The delay of $4 \times step$ is due to the fact that for each datapoint, the proposed algorithm needs the previous four datapoints and the following four datapoints to calculate its corresponding LBP code, assuming an eight-bit length LBP code. The calculated LBP codes are stored in a circular buffer with a length similar to the input segment length. After the initial ECG segment has been received, the oldest LBP code in the buffer will always be replaced by the newest, and the LBP circular buffer will always contain the LBP codes corresponding to the latest segment of the ECG input signal.

Further, after calculating each new LBP code, the LBP histogram will be updated. This process involves adding 1 to the value stored in the histogram bins corresponding to the new LBP code and reducing 1 from the value stored in the histogram bin corresponding to the oldest LBP codes removed in the previous process. At this point, the LBP histogram is ready for use as input features for a machine-learning model. To improve classification accuracy, input scaling or normalization, a common part of a machine learning inference process flow, can also be added after this step. A flow chart illustrating the process outlined in the above paragraphs can be seen in Figure 3.

A significant benefit of the proposed method, particularly when used in real-time on a stream of ECG data points, is that for each arriving new data point, only one new LBP code needs to be calculated, followed by a simple process of replacing the oldest value in a circular buffer and updating the value of two bins in a histogram. At this point a complete input vector is ready for the inference process, assuming at least one initial segment length is already processed. Since, generally, this process is faster than the ECG signal sampling period, a machine learning inference using the latest VSDTLBP histogram as its input can be done at every sampling period. The limitation on the inference frequency will be the computation time of the inference itself.

E. IMPLEMENTATION OF MACHINE LEARNING INFERENCE IN MICROCONTROLLER

In this work, we propose doing the machine learning inference process at the edge using a low-resource microcontroller. The target microcontrollers are Arm Cortex M-based microcontrollers produced by STMicroelectronics.

The support vector machine (SVM) model is first trained in a personal computer having large memory and computing power. Afterward, support vectors are extracted from the generated model and included as an array in the C program embedded in the target microcontroller.

SVM model training is done in Python using the sci-kit-learn library and the radial basis function (RBF) kernel [56]. 10-fold stratified cross validation is used, and the trained

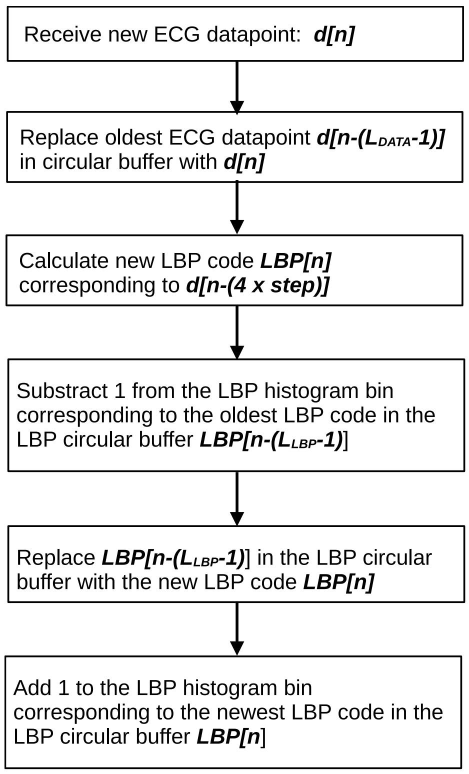
Receive new ECG datapoint: d[n]
Replace oldest ECG datapoint d[n-(L_DATA-1)] in circular buffer with d[n]
Calculate new LBP code LBP[n] corresponding to d[n-(4 x step)]
Substract 1 from the LBP histogram bin corresponding to the oldest LBP code in the LBP circular buffer LBP[n-(L_LBP-1)]
Replace LBP[n-(L_LBP-1)] in the LBP circular buffer with the new LBP code LBP[n]
Add 1 to the LBP histogram bin corresponding to the newest LBP code in the LBP circular buffer LBP[n]

FIGURE 3. Flowchart illustrating the process flow every time a new ECG datapoint is received. *n* is the current time, L_{DATA} is the length of the circular buffer storing ECG data, and L_{LBP} is the length of the circular buffer storing generated LBP codes.

model having the best accuracy is stored for use in the microcontroller. The microcontroller embedding is based on the Arm CMSIS-DSP library provided by Arm [57].

IV. RESULTS AND DISCUSSION

A. EFFECT OF ADDITION OF LOW PASS FILTER

Figure 4 shows the effect of low-pass filtering before the LBP calculation for different input data segment lengths. It can be seen that classification accuracy improved significantly compared to the original algorithm proposed in [22].

B. EFFECT OF FEATURE SELECTION

The feature selection process resulted in 44 features left from the original 58 features (VSDTLBP codes) proposed by Yazid and Mahrus in [22].

Figure 5 shows the effect of feature selection on classification accuracy. It can be seen that the proposed feature reduction does not affect classification results significantly. On the other hand, the smaller model size achieved by reducing 14 features from the original 58 (about 24% reduction in the number of features) is essential for allowing implementation of the machine learning algorithm on a low-resource microcontroller.

Table 4 shows the number of support vectors generated after SVM training for several input segment lengths when using the original 58 LBP codes of VSDTLBP [22] and the 44 LBP codes selected by the feature selection process.

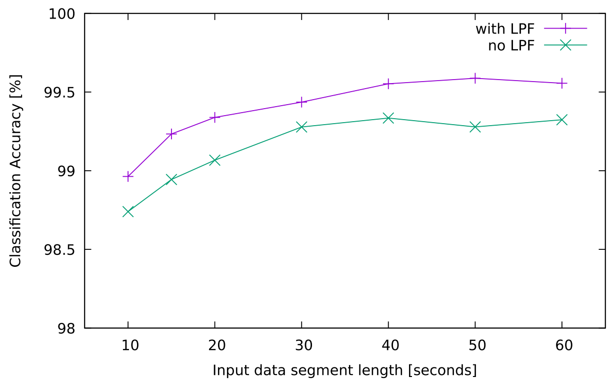

FIGURE 4. Plot showing the effect of adding a low pass filter before the LBP calculation on classification accuracy, tested against the MIT-BIH Atrial Fibrillation Database with Experiment Setting 1. The no LPF results are achieved using the original VSDTLBP algorithm proposed by [22].

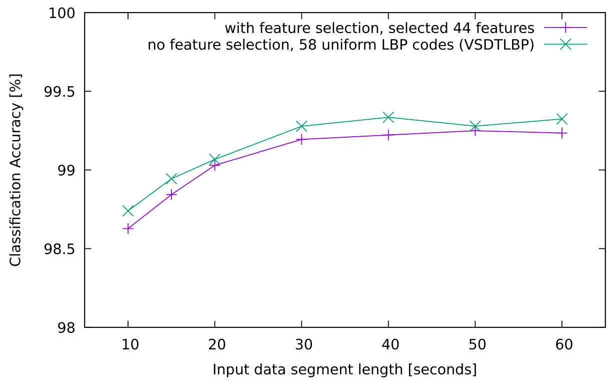

FIGURE 5. Plot showing the effect of feature selection on classification accuracy, tested against the MIT-BIH Atrial Fibrillation Database with Experiment Setting 1. The result without feature selection is achieved using the original VSDTLBP algorithm as proposed by [22].

TABLE 4. Effect of Feature Selection on the number of generated support vectors when trained on ECG segments from the MIT-BIH atrial fibrillation database, using experiment setting 1.

Input Length	Number of Support Vectors		Reduction
[seconds]	58 Features	44 Features	[%]
10	167144	131384	21
15	101338	77273	24
20	73996	55370	25
30	49694	34848	30
40	38350	26884	30
50	31935	21991	31
60	27341	18621	32

The feature selection resulted in a significant decrease in the number of support vectors generated after training. Since the size of the machine learning model is directly proportional to the number of support vectors, this also means a similar decrease in machine learning model size.

It should be noted that while Figure 5 shows a slightly decreased classification accuracy as a result of the feature reduction, it will be canceled out by the larger accuracy improvement due to the addition of the low pass filter.

TABLE 5. Comparison of the classification results between the proposed method and the previous work [22] when used on the MIT-BIH atrial fibrillation database with experiment setting 1.

Input Length	VSDTLBP [22] [%]			Proposed [%]		
[seconds]	Sens.	Spec.	Acc.	Sens.	Spec.	Acc.
10	98.52	98.96	98.74	98.94	98.88	98.91
15	98.73	99.15	98.94	99.14	99.12	99.13
20	98.88	99.25	99.07	99.27	99.26	99.26
30	99.16	99.40	99.28	99.38	99.29	99.33
40	99.20	99.47	99.33	99.51	99.35	99.43
50	99.16	99.40	99.28	99.56	99.42	99.49
60	99.19	99.46	99.32	99.49	99.46	99.47

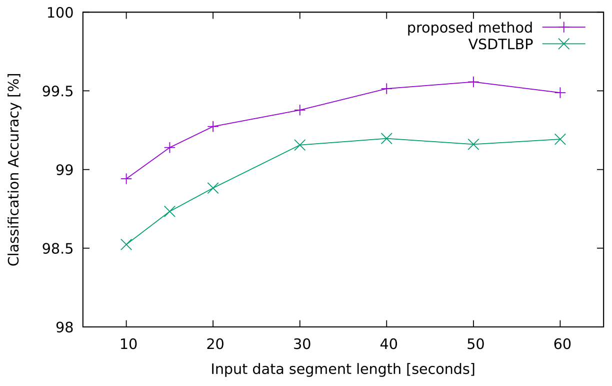

FIGURE 6. Plots showing a comparison of the classification sensitivity of the proposed method compared to the original method [22] for different lengths of input ECG segments from the MIT-BIH Atrial Fibrillation Database using Experiment Setting 1.

C. EXPERIMENT SETTING 1

1) CLASSIFICATION ACCURACY

Figure 6, Figure 7, and Figure 8 shows plots of the classification sensitivity, specificity, and accuracy against input signal segment length, when the improvements specified in the previous sections are implemented and tested against the whole of the MIT-BIH Atrial Fibrillation Database using Experiment Setting 1, compared to the results of the original VSDTLBP [22] algorithm. Table 5 shows these results in numbers.

The plots show that the improvements proposed in this paper particularly improve the classification sensitivity compared to the previous results reported in [22]. Classification sensitivity is the ability of the algorithm to identify Atrial Fibrillation segments in ECG signals. A high sensitivity value is especially important for abnormality detection since it reduces the risk of potentially dangerous abnormal conditions being undetected.

Table 6 shows the comparison results when using the MIT-BIH Arrhythmia Database, showing similar trends as those achieved when using the MIT-BIH Atrial Fibrillation database.

2) MACHINE LEARNING MODEL SIZE

Table 7 compares the size of the generated machine learning model between the original VSDTLBP [22] and the proposed method. The model used for this table is the best model

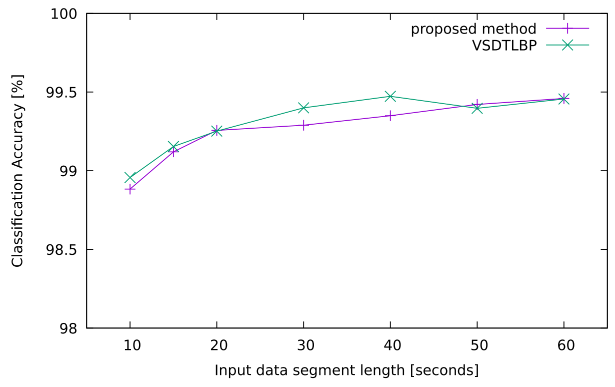

FIGURE 7. Plots showing a comparison of the classification specificity of the proposed method compared to the original method [22] for different lengths of input ECG segments from the MIT-BIH Atrial Fibrillation Database using Experiment Setting 1.

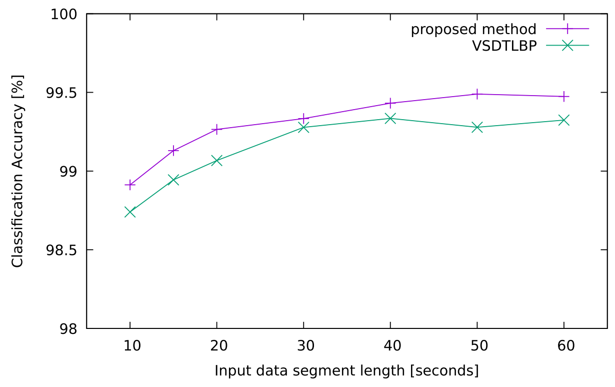

FIGURE 8. Plots showing a comparison of the classification accuracy of the proposed method compared to the original method [22] for different lengths of input ECG segments from the MIT-BIH Atrial Fibrillation Database using Experiment Setting 1.

TABLE 6. Comparison of the classification results between the proposed method and the previous work [22] when used on the MIT-BIH arrhythmia database with experiment setting 1.

Input Length	VSDTLBP [22] [%]			Proposed [%]		
[seconds]	Sens.	Spec.	Acc.	Sens.	Spec.	Acc.
10	98.72	99.12	98.92	98.91	99.12	99.02
15	99.02	99.06	99.04	98.90	98.98	98.94
20	98.99	99.22	99.10	99.22	99.27	99.24
30	99.20	99.20	99.20	99.20	99.29	99.25
40	98.06	99.39	98.73	99.39	99.52	99.45
50	97.12	99.84	98.48	98.72	99.84	99.28
60	97.96	98.98	98.47	98.16	99.39	98.78

generated in the 10-fold stratified cross-validation using the balanced MIT-BIH Atrial Fibrillation Database signals with Experiment Setting 1. It can be seen that the proposed method has a significantly reduced machine-learning model size. Table 8 also shows the extracted support vector array size for each saved model, which generally correlates with the model file size.

A typical low-cost microcontroller from STMicroelectronics used in small electronic devices includes as small as

TABLE 7. Comparison of generated trained model size between the proposed method and VSDTLBP [22], using the MIT-BIH atrial fibrillation database with experiment setting 1.

Input Length	Trained Model Size [kB]		Reduction
[seconds]	VSDTLBP [22]	proposed	[%]
10	1411.4	945.8	33.0
15	841.4	555.9	33.9
20	631.3	413.0	34.6
30	430.8	262.3	39.1
40	331.1	200.6	39.4
50	261.0	168.9	35.3
60	229.0	145.1	36.6

TABLE 8. Comparison of extracted support vector array size between the proposed method and VSDTLBP [22], using the MIT-BIH atrial fibrillation database with experiment setting 1.

Input Length	Support vector array size		Reduction
[seconds]	VSDTLBP [22]	proposed	[%]
10	167144.4	110255.2	34.0
15	101337.6	64618.4	36.2
20	73996.4	48523.2	34.4
30	49694.4	31090.4	37.4
40	38349.6	23760.0	38.0
50	31934.8	20495.2	35.8
60	27341.2	16966.4	37.9

a few hundred kB of flash memory to store its programs. Our proposed method's machine learning model sizes are sufficiently small to embed in this type of microcontroller. This point is particularly significant in applications that include a graphical user interface since the graphics data necessary for the GUI usually take a large amount of memory, limiting available memory space for other uses.

D. EXPERIMENT SETTING 2

For implementation on the microcontroller and its performance comparison with a PC-based Python program, Experiment Setting 2 as specified in section II-C is used. In this experiment, the balanced segments from the MIT-BIH Atrial Fibrillation Database are randomly split into training and testing segments using an 80:20 proportion. 10-fold stratified cross-validation is done on the training segments, and the model with the best accuracy is saved for use against the test segments. The experiment is repeated five times and the average results are reported in this paper.

1) EMBEDDED CLASSIFICATION ACCURACY

Table 9 shows the classification accuracy of the trained machine learning model (using the 80% training segments) when tested against the 20% test segments, showing both the results of inference as a python program on PC and as an embedded program on the MCU.

Our experiments showed that embedding a trained support vector machine (SVM) machine learning model as a

TABLE 9. Comparison of classification results of the proposed method on PC and STM32F413ZH MCU, using the MIT-BIH atrial fibrillation database with experiment setting 2.

Input Length	Results on PC [%]			Results on MCU [%]		
[seconds]	Sens.	Spec.	Acc.	Sens.	Spec.	Acc.
10	98.57	98.94	98.75	98.70	98.94	98.82
15	99.37	99.01	99.19	99.35	99.03	99.19
20	99.28	99.13	99.21	99.22	99.19	99.21
30	99.19	99.10	99.14	99.14	99.10	99.12
40	99.46	99.40	99.43	99.46	99.33	99.40
50	99.09	99.70	99.39	99.09	99.70	99.39
60	99.45	99.18	99.32	99.45	99.18	99.32

TABLE 10. Memory usage and computing time of the proposed method on STM32F413ZH MCU, using the MIT-BIH atrial fibrillation database with experiment setting 2.

Input Length [seconds]	Model Size [kB]	Flash Usage [kB]	Computing Time	
			1 LBP Code [μs]	SVM [ms]
10	791.67	424.49	8.18	67.53
15	473.61	274.09	8.27	40.42
20	340.81	211.45	8.18	29.03
30	227.34	157.83	8.18	19.37
40	189.77	140.08	8.18	16.15
50	148.11	120.39	8.18	12.59
60	132.86	114.46	8.27	11.28

C program on the STM32 MCU using the Arm CMSIS library does not result in significant classification accuracy degradation. This result is a significant achievement since, typically, MCU implementation of machine learning algorithms results in reduced classification accuracy.

2) MEMORY USAGE AND COMPUTING TIME

Table 10 shows flash memory usage and computation time required to calculate a single LBP code and an SVM classification on the STM32F413ZH microcontroller. The best machine learning models generated in the training processes using Experiment Setting 2 are used in these experiments.

The STM32F413ZH microcontroller unit (MCU) is produced by STMicroelectronics. It is a 32-bit MCU with a hardware floating point unit (FPU), 1.5MB of flash memory, 320kB RAM, and a maximum clock frequency of 100MHz [39].

The flash memory usage in Table 10 is taken from the values reported by the integrated development environment (IDE) STM32CubeIDE provided by the manufacturer [42]. The computing time is calculated using the internal timer of the MCU.

Table 10 shows that even the longest calculation time specified in the table is still well below the average length of a single heartbeat of a human, indicating that our proposed method can achieve a near real-time detection of Atrial Fibrillation even when implemented on a resource-constrained microcontroller.

TABLE 11. Effect of hardware FPU use on the performance of the proposed method, tested on STM32F413ZH using 60-second input signals from the MIT-BIH atrial fibrillation database with experiment setting 2.

FPU Use	Sens. [%]	Spec. [%]	Acc. [%]	Computing Time	
				1 LBP Code [μs]	SVM [ms]
Yes	99.45	99.18	99.32	8.27	11.28
No	99.45	99.18	99.32	27.55	55.92

3) EFFECT OF HARDWARE FLOATING POINT UNIT (FPU)

The STM32F413ZH microcontroller contains a hardware floating point unit (FPU), which can handle resource-intensive calculations involving floating numbers, greatly reducing processor computing burden and the algorithm calculation time. Table 11 compares algorithm performance when embedded in the STM32F413ZH microcontroller with and without enabling the hardware FPU. As expected, the calculation time is significantly increased when the hardware FPU is disabled because the compiler is forced to emulate floating point calculations in software. On the other hand, consistent classification results are achieved irrespective of the use of hardware FPU unit on this MCU.

4) PERFORMANCE COMPARISON ON MULTIPLE MCU TYPES

Table 12 compares the proposed method's performance when embedded in several STM32 microcontrollers from different series based on several different Arm Cortex-M microprocessor cores. The models and test data for 60 seconds input data generated in PC based experiment using Experiment Setting 2 are used in this test. When hardware FPU is not available (as in STM32F207ZG) or is not enabled, such as in the first result for STM32F413ZHT3 in Table 12, the computing times are significantly larger. However, all microcontroller tested in this work shows similar high classification accuracy as the PC based results. This result shows that the proposed method does not require sacrificing classification accuracy, even when implemented on the relatively cheap Arm Cortex-M3 or M4-based microcontrollers. The simple process used by our proposed feature extraction method does not depend on advanced features that are only available in expensive high-end microcontrollers.

V. COMPARISON WITH PREVIOUS WORKS

A. COMPARISON WITH PREVIOUS WORKS ON ATRIAL FIBRILLATION CLASSIFICATION

Table 13 compares our proposed method with previous works on atrial fibrillation classification using the MIT-BIH Atrial Fibrillation Database, and Table 14 compares with those using the MIT-BIH Arrhythmia Database, respectively. Our proposed work achieved better results than most previous works in both tables, including the work presented in the original paper [22].

Our proposed method achieved better accuracy while reducing machine learning model size, which is very beneficial for embedded implementation in low-resource

TABLE 12. Comparison of performance of the proposed method on several types of STM32 MCU, using 60-second input signals from the MIT-BIH atrial fibrillation database with experiment setting 2.

MCU Type	Core Type	Clock Freq. [MHz]	FPU Use	Sens. [%]	Spec. [%]	Acc. [%]	Computing Time		Avg. Current [mA]
							1 LBP Code [μs]	SVM [ms]	
STM32F207ZG	Cortex-M3	120	No	99.45	99.18	99.32	27.09	54.67	37
STM32F413ZHT3	Cortex-M4	100	No	99.45	99.18	99.32	27.55	55.92	26
STM32F413ZHT3	Cortex-M4	100	Yes	99.45	99.18	99.32	8.27	11.28	27
STM32L4R5ZIP	Cortex-M4	120	Yes	99.45	99.18	99.32	7.73	12.73	32
STM32H7A3ZIQ	Cortex-M7	280	Yes	99.45	99.18	99.32	7.31	8.39	23

TABLE 13. Comparison with previous works in atrial fibrillation classification using MIT-BIH atrial fibrillation database.

Authors	Year	Input Length [seconds]	Sens. [%]	Spec. [%]	Acc. [%]
Tateno and Glass [26]	2001	100 beats	94.40	97.20	
Logan & Healey [27]	2005	10	96.00	89.00	
Couceiro et al. [28]	2008	12	93.80	96.09	
Dash et al. [29]	2009	128 beats	94.40	95.10	
Lake & Moorman [30]	2010	12 beats	91.00	94.00	
Huang et al. [31]	2010	50 beats	96.10	98.10	
Lee et al. [58]	2013	128 beats	98.20	97.70	
Ladavich et al. [32]	2015	7 beats	98.09	91.66	
Asgari et al. [33]	2015	30	97.00	97.10	97.1
Garcia et al. [34]	2016	7 beats	91.21	94.53	93.32
Xia et al. [35]	2018	5	98.79	97.87	98.63
He et al. [36]	2018	5 beats	99.41	98.91	99.23
Mousavi et al [59]	2020	5	98.88	98.78	98.83
Yazid & Mahrus [22]	2020	15	98.63	99.00	98.81
Yazid & Mahrus [22]	2020	60	99.11	99.29	99.2
Wang et. al. [60]	2021	2	98.60	99.30	99.1
Zhang et. al. [61]	2021	8	99.15	99.43	99.32
Radhakrishnan et al. [62]	2021	4	99.17	99.18	
Duan et. al. [63]	2022	29 beats	98.48	98.40	98.43
Kumar et al. [64]	2022	30 beats	98.27	98.84	98.62
Phukan et al. [65]	2023	5	99.26	95.30	97.68
Li et al. [66]	2023	5	99.86	99.72	99.79
Zhang et al. [67]	2024	30	98.35	96.41	97.58
This work	2024	15	99.20	99.10	99.15
This work	2024	60	99.42	99.48	99.45

TABLE 14. Comparison with previous works in atrial fibrillation classification using MIT-BIH arrhythmia database.

Authors	Year	Input Length [seconds]	Sens. [%]	Spec. [%]	Acc. [%]
Tateno and Glass [26]	2001	100 beats	88.20	87.60	
Dash et al. [29]	2009	128 beats	90.20	91.20	
Lee et al. [58]	2013	128 beats	91.10	89.70	
Yazid & Mahrus [22]	2020	15	99.14	98.81	98.98
Yazid & Mahrus [22]	2020	60	99.38	98.97	99.18
Wang et. al. [60]	2021	2	97.60	98.70	98.40
Kumar et al. [64]	2022	30 beats	93.06	91.67	91.82
Zhang et al. [67]	2024	30	94.63	99.00	95.04
This work	2024	15	99.18	98.41	98.80
This work	2024	60	99.18	99.18	99.18

microcontrollers. The classification sensitivity, specificity, and accuracy are also among the best in the list. Some previous results that are slightly better are achieved at the cost of using more resource-demanding methods, such as deep learning or computationally complex feature extractions.

B. COMPARISON WITH PREVIOUS WORKS ON EDGE DEVICE-BASED ECG SIGNAL CLASSIFICATION

Table 15 shows a comparison of our work with previous works on edge device/constrained device-based ECG signal classification. Since previous works on Atrial Fibrillation classification that include implementation on edge device/constrained devices are still rare, we also include works on Arrhythmia classification in the comparison table.

Compared to the systems based on the Raspberry-Pi devices such as those proposed by [14], [65], and [68] and the NVIDIA Jetson Nano device used by [15] and [69], our proposed method and system are better due to the much lower power consumption of the STM32 microcontrollers. Our measurement shows that the average current consumption of the STM32F413ZHT3 MCU when running the proposed method is just around 27mA. Since the supply voltage of the MCU on the Nucleo-144 board is 3.3V, the average power consumption can be calculated to be just around 89.1mW. On the other hand, the idle power consumption of Raspberry-Pi4 and NVIDIA Jetson Nano is reported to be around 2.1W and 0.9W, respectively [70]. Our proposed method's much lower power consumption increases battery life and reduces heat dissipation, making a wearable device more convenient to use for a long period of time.

The STM32 microcontroller-based system proposed in this work also has significant benefits in terms of size and production cost compared to Raspberry Pi or NVIDIA Jetson Nano-based systems since the latter two typically have a relatively large form factor and are more expensive compared to an STM32 microcontroller-based system.

TABLE 15. Comparison with previous works on ECG classification based on constrained resource device.

Author (year)	Detection Target	MCU /MCU Board	Method	Database	Acuracy [%]	Inference Time [ms]	Model Size [kB]	Price[*6] [USD]
Faraone et al., [75] (2020)	Atrial Fibrillation	nRF52832	ANN	CinC2017[*1]	86.1	94.8	195.6	2.58
Farag et al., [14] (2022)	Arrhythmia	Raspberry Pi 3B+ (BCM2837B0)	STFT+1D CNN +2D CNN	MITDB [*2]	99.1	9	89.88	35
Seitanidis et al., [15] (2022)	Arrhythmia	NVIDIA Jetson Nano	2D CNN	MITDB	95.3	1.48		499
Chen et al., [19] (2022)	Atrial Fibrillation	Arduino Mega (ATmega2560)	HRV+Freq. +MLP	CPSC2018[*3]	94.5	2.12		14.49
Mohebbanaaz et al. [69] (2022)	Arrhythmia	NVIDIA Jetson Nano	DNN	MITDB	99.56			499
Falaschetti et al. [73] (2022)	Arrhythmia	STM32 (STM32L476RG)	LSTM	MITDB	90.62	659	305.6	5.97
Phukan et al. [65] (2021)	Atrial Fibrillation	Raspberry Pi 4 (BCM2711)	1D CNN	AFDB [*4]	97.68	2.54	3110	35
Karri et al., [68] (2023)	Arrhythmia	Raspberry Pi 4 (BCM2711)	DWT,DSM +LTSM	MITDB	99.64	58		35
Chen et al., [71] (2023)	Atrial Fibrillation	Arduino Mega (ATmega2560)	HRV+MLP	CinC2017	86			14.49
				AFDB	87.97			
Zylinsky et al [74] (2023)	Atrial Fibrillation	STM32 (STM32WB55RG)	HRV+SVM	CinC2020[*5]	96.9	0.72	2.5	4.55
Zishan et al. [72] (2024)	Arrhythmia	Arduino Nano (ATmega328)	DNN	MITDB	96.8		1.267	2.8
This work (2024)	Atrial Fibrillation	STM32 (STM32F413ZH)	VSDTLBP +SVM	AFDB	99.45	11.28	132.86	13.82

[*1] CinC2017: Physionet Computing in Cardiology Challenge 2017 Dataset [24], [76].
[*2] MITDB: MIT-BIH Arrhythmia Database [24], [25].
[*3] CPSC2018: China Physiological Signal Challenge 2018 [20]
[*4] AFDB: MIT-BIH Atrial Fibrillation Database [23], [24].
[*5] CinC2020: Physionet Computing in Cardiology Challenge 2020 Dataset [24], [77].
[*6] Device price informations are current as of June 16th 2024, taken from the manufacturer's website if provided and from mouser.com otherwise.

In Table 15, there are also previous works using 8-bit microcontrollers such as ATmega2560 and ATmega328 [19], [71], [72]. These microcontrollers are known to have low power consumption and are relatively cheap. However, their downside is the relatively small memory size. Previous works implementing machine learning algorithms on these devices can only use a small number of input features in order to minimize model size. They also need to optimize the machine learning model further to make it fit in the memory, and the optimization process will typically reduce classification accuracy.

Chen et al. [19] proposed a system based on the 8-bit ATmega2560 microcontroller using only 22 heart rate variability features to achieve 94.5% accuracy when tested against the China Physiological Signal Challenge (CPSC) 2018 dataset. The method proposed in [19] needs only 2.12ms to execute inference on the microcontroller, not including the time required for feature extraction. The features used by [19] include computationally complex functions such as standard deviation, root mean square, skewness, and coefficient of variation, potentially adding significant computing time. Our proposed method does not use such expensive calculations in the feature extraction process. Furthermore, [19] also depends on the accurate detection of R peaks from the ECG signals, which will add more processing time and significantly affect the system's overall accuracy.

Falaschetti et al. [73] proposed an STM32 based system using a long short time memory (LSTM) classifier to classify arrhythmia. Reference [73] does not use feature extraction in their method, directly feeding 145 samples of ECG signal into the machine learning model. While this makes the process relatively simple, it results in a significantly higher machine learning model size and longer inference time than our proposed method.

Zylinski et al. in [74] also proposed a system based on a low-power STM32 microcontroller, which can achieve a fast inference time of just 0.72 ms for each input ECG signal with a length from 6 to 60 seconds. However, the RR-interval-based feature extraction used by [74] requires accurate detection of R peaks from ECG signals, adding more computation complexity and time. Our proposed method is better because it has a relatively simpler feature extraction process and does not depend on R peak detection.

VI. CONCLUSION

In this work we proposed an improved method to detect Atrial Fibrillation from ECG signal. The proposed method successfully increased classification accuracy and reduced the trained model size. Implemented on an Arm Cortex M4-based microcontroller, the proposed method can run an SVM inference within 11.28ms using as low as 27mA average current, enabling near real-time atrial fibrillation detection on a low-power and low-cost device. In contrast to many previously proposed edge device-based inference systems, which usually need to implement methods that reduce classification accuracy (such as model pruning) in order to be able to run as an embedded system, our proposed method can achieve as high accuracy classification as a PC based solution when implemented in a low power microcontroller. While further works such as testing with real ambulatory ECG data recorded from a wearable device are necessary, the work presented in this paper can contribute to reducing the cost and improving the quality of smart wearable ECG monitor devices, helping reduce the burden on medical professionals, patients, and their families.

REFERENCES

[1] C. R. Wyndham, "Atrial fibrillation: The most common arrhythmia," *Texas Heart Inst. J.*, vol. 27, no. 3, p. 257, 2000.

[2] G. L. Botto, G. Tortora, M. C. Casale, F. L. Canevese, and F. A. M. Brasca, "Impact of the pattern of atrial fibrillation on stroke risk and mortality," *Arrhythmia Electrophysiology Rev.*, vol. 10, no. 2, pp. 68–76, Jul. 2021.

[3] T. J. Bunch, "Atrial fibrillation and dementia," *Circulation*, vol. 142, no. 7, pp. 618–620, Aug. 2020.

[4] F. K. Wegner, L. Plagwitz, F. Doldi, C. Ellermann, K. Willy, J. Wolfes, S. Sandmann, J. Varghese, and L. Eckardt, "Machine learning in the detection and management of atrial fibrillation," *Clin. Res. Cardiol.*, vol. 9, no. 111, pp. 1010–1017, 2022.

[5] G. Boriani, M. Vitolo, I. Diemberger, M. Proietti, A. C. Valenti, V. L. Malavasi, and G. Y. H. Lip, "Optimizing indices of atrial fibrillation susceptibility and burden to evaluate atrial fibrillation severity, risk and outcomes," *Cardiovascular Res.*, vol. 117, no. 7, pp. 1–21, Jun. 2021.

[6] P. Vardas, M. Cowie, N. Dagres, D. Asvestas, S. Tzeis, E. P. Vardas, G. Hindricks, and J. Camm, "The electrocardiogram endeavour: From the Holter single-lead recordings to multilead wearable devices supported by computational machine learning algorithms," *EP Europace*, vol. 22, no. 1, pp. 19–23, Jan. 2020.

[7] A. Rizwan, A. Zoha, I. B. Mabrouk, H. M. Sabbour, A. S. Al-Sumaiti, A. Alomainy, M. A. Imran, and Q. H. Abbasi, "A review on the state of the art in atrial fibrillation detection enabled by machine learning," *IEEE Rev. Biomed. Eng.*, vol. 14, pp. 219–239, 2021.

[8] J. Xue and L. Yu, "Applications of machine learning in ambulatory ECG," *Hearts*, vol. 2, no. 4, pp. 472–494, Oct. 2021.

[9] J. R. Giudicessi, M. Schram, J. M. Bos, C. D. Galloway, J. B. Shreibati, P. W. Johnson, R. E. Carter, L. W. Disrud, R. Kleiman, Z. I. Attia, P. A. Noseworthy, P. A. Friedman, D. E. Albert, and M. J. Ackerman, "Artificial intelligence–enabled assessment of the heart rate corrected qt interval using a mobile electrocardiogram device," *Circulation*, vol. 143, no. 13, pp. 1274–1286, 2021.

[10] T. Mahmud, S. A. Fattah, and M. Saquib, "DeepArrNet: An efficient deep CNN architecture for automatic arrhythmia detection and classification from denoised ECG beats," *IEEE Access*, vol. 8, pp. 104788–104800, 2020.

[11] P. Bonizzi, J. Karel, S. Zeemering, and R. Peeters, "Sleep apnea detection directly from unprocessed ECG through singular spectrum decomposition," in *Proc. Comput. Cardiol. Conf. (CinC)*, Sep. 2015, pp. 309–312.

[12] G. Petmezas, K. Haris, L. Stefanopoulos, V. Kilintzis, A. Tzavelis, J. A. Rogers, A. K. Katsaggelos, and N. Maglaveras, "Automated atrial fibrillation detection using a hybrid CNN-LSTM network on imbalanced ECG datasets," *Biomed. Signal Process. Control*, vol. 63, Jan. 2021, Art. no. 102194.

[13] K.-K. Tseng, C. Wang, T. Xiao, C.-M. Chen, M. M. Hassan, and V. H. C. de Albuquerque, "Sliding large kernel of deep learning algorithm for mobile electrocardiogram diagnosis," *Comput. Electr. Eng.*, vol. 96, Dec. 2021, Art. no. 107521.

[14] M. M. Farag, "A self-contained STFT CNN for ECG classification and arrhythmia detection at the edge," *IEEE Access*, vol. 10, pp. 94469–94486, 2022.

[15] P. Seitanidis, J. Gialelis, G. Papaconstantinou, and A. Moschovas, "Identification of heart arrhythmias by utilizing a deep learning approach of the ECG signals on edge devices," *Computers*, vol. 11, no. 12, p. 176, Dec. 2022.

[16] M. Yazid, F. Fahmi, E. Sutanto, R. Setiawan, and M. Aziz, "Simple authentication method for vehicle monitoring IoT device with verifiable data integrity," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 7027–7037, Apr. 2023.

[17] G. Sivapalan, K. K. Nundy, A. James, B. Cardiff, and D. John, "Interpretable rule mining for real-time ECG anomaly detection in IoT edge sensors," *IEEE Internet Things J.*, vol. 10, no. 15, pp. 1–12, Aug. 2023.

[18] M. M. Farag, "A tiny matched filter-based CNN for inter-patient ECG classification and arrhythmia detection at the edge," *Sensors*, vol. 23, no. 3, p. 1365, Jan. 2023.

[19] J. Chen, Y. Zheng, Y. Liang, Z. Zhan, M. Jiang, X. Zhang, D. S. da Silva, W. Wu, and V. H. C. d. Albuquerque, "Edge2Analysis: A novel AIoT platform for atrial fibrillation recognition and detection," *IEEE J. Biomed. Health Informat.*, vol. 26, no. 12, pp. 5772–5782, Dec. 2022.

[20] F. Liu, C. Liu, L. Zhao, X. Zhang, X. Wu, X. Xu, Y. Liu, C. Ma, S. Wei, Z. He, J. Li, and E. N. Yin Kwee, "An open access database for evaluating the algorithms of electrocardiogram rhythm and morphology abnormality detection," *J. Med. Imag. Health Informat.*, vol. 8, no. 7, pp. 1368–1373, Sep. 2018.

[21] N. Ganapathy, D. Baumgärtel, and T. Deserno, "Automatic detection of atrial fibrillation in ECG using co-occurrence patterns of dynamic symbol assignment and machine learning," *Sensors*, vol. 21, no. 10, p. 3542, May 2021.

[22] M. Yazid and M. Abdur Rahman, "Variable step dynamic threshold local binary pattern for classification of atrial fibrillation," *Artif. Intell. Med.*, vol. 108, Aug. 2020, Art. no. 101932.

[23] G. B. Moody and R. G. Mark, "A new method for detecting atrial fibrillation using RR intervals," *Comput. Cardiol.*, vol. 10, pp. 227–230, Jan. 1983.

[24] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, Jun. 2000.

[25] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, Sep. 2001.

[26] K. Tateno and L. Glass, "Automatic detection of atrial fibrillation using the coefficient of variation and density histograms of RR and δRR intervals," *Med. Biol. Eng. Comput.*, vol. 39, no. 6, pp. 664–671, Nov. 2001.

[27] B. Logan and J. Healey, "Robust detection of atrial fibrillation for a long term telemonitoring system," in *Computers Cardiology*, 2005, pp. 619–622.

[28] R. Couceiro, P. Carvalho, J. Henriques, M. Antunes, M. Harris, and J. Habetha, "Detection of atrial fibrillation using model-based ECG analysis," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–5.

[29] S. Dash, K. H. Chon, S. Lu, and E. A. Raeder, "Automatic real time detection of atrial fibrillation," *Ann. Biomed. Eng.*, vol. 37, no. 9, pp. 1701–1709, Sep. 2009.

[30] D. E. Lake and J. R. Moorman, "Accurate estimation of entropy in very short physiological time series: The problem of atrial fibrillation detection in implanted ventricular devices," *Amer. J. Physiol.-Heart Circulatory Physiol.*, vol. 300, no. 1, pp. H319–H325, Jan. 2011.

[31] C. Huang, S. Ye, H. Chen, D. Li, F. He, and Y. Tu, "A novel method for detection of the transition between atrial fibrillation and sinus rhythm," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 4, pp. 1113–1119, Apr. 2011.

[32] S. Ladavich and B. Ghoraani, "Rate-independent detection of atrial fibrillation by statistical modeling of atrial activity," *Biomed. Signal Process. Control*, vol. 18, pp. 274–281, Apr. 2015.

[33] S. Asgari, A. Mehrnia, and M. Moussavi, "Automatic detection of atrial fibrillation using stationary wavelet transform and support vector machine," *Comput. Biol. Med.*, vol. 60, pp. 132–142, May 2015.

[34] M. García, J. Ródenas, R. Alcaraz, and J. J. Rieta, "Application of the relative wavelet energy to heart rate independent detection of atrial fibrillation," *Comput. Methods Programs Biomed.*, vol. 131, pp. 157–168, Jul. 2016.

[35] Y. Xia, N. Wulan, K. Wang, and H. Zhang, "Detecting atrial fibrillation by deep convolutional neural networks," *Comput. Biol. Med.*, vol. 93, pp. 84–92, Feb. 2018.

[36] R. He, K. Wang, N. Zhao, Y. Liu, Y. Yuan, Q. Li, and H. Zhang, "Automatic detection of atrial fibrillation based on continuous wavelet transform and 2D convolutional neural networks," *Frontiers Physiol.*, vol. 9, p. 1206, Aug. 2018.

[37] J. Wang, "Automated detection of atrial fibrillation and atrial flutter in ECG signals based on convolutional and improved Elman neural network," *Knowl.-Based Syst.*, vol. 193, Apr. 2020, Art. no. 105446.

[38] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, "A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors," *Sensors*, vol. 19, no. 22, p. 5026, Nov. 2019.

[39] *High-Performance Access Line, Arm Cortex-M4 Core With DSP and FPU, 1,5 MByte of Flash Memory, 100 MHz CPU, ART Accelerator, DFSDM*, document STM32F413ZH, 2024.

[40] *STM32 Nucleo-144 Development Board With STM32F413ZH MCU, Supports Arduino, ST Zio and Morpho Connectivity*, document NUCLEO-F413ZH, 2024.

[41] *Cortex-m4*, Arm Holdings Plc, Cambridge, U.K., 2024.

[42] *Integrated Development Environment for STM32*, document STM32CubeIDE, 2024.

[43] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[44] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," *J. Bioinf. Comput. Biol.*, vol. 3, no. 2, pp. 185–205, Apr. 2005.

[45] M. Osl, S. Dreiseitl, F. Cerqueira, M. Netzer, B. Pfeifer, and C. Baumgartner, "Demoting redundant features to improve the discriminatory ability in cancer data," *J. Biomed. Informat.*, vol. 42, no. 4, pp. 721–725, Aug. 2009.

[46] J. Xu, B. Tang, H. He, and H. Man, "Semisupervised feature selection based on relevance and redundancy criteria," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 1974–1984, Sep. 2017.

[47] I. Jo, S. Lee, and S. Oh, "Improved measures of redundancy and relevance for mRMR feature selection," *Computers*, vol. 8, no. 2, p. 42, May 2019.

[48] I. M. Nasir, M. A. Khan, M. Yasmin, J. H. Shah, M. Gabryel, R. Scherer, and R. Damasevicius, "Pearson correlation-based feature selection for document classification using balanced training," *Sensors*, vol. 20, no. 23, p. 6793, Nov. 2020.

[49] Y. Liu, Y. Mu, K. Chen, Y. Li, and J. Guo, "Daily activity feature selection in smart homes based on Pearson correlation coefficient," *Neural Process. Lett.*, vol. 51, no. 2, pp. 1771–1787, Apr. 2020.

[50] F. Bagherzadeh-Khiabani, A. Ramezankhani, F. Azizi, F. Hadaegh, E. W. Steyerberg, and D. Khalili, "A tutorial on variable selection for clinical prediction models: Feature selection methods in data mining could improve the results," *J. Clin. Epidemiology*, vol. 71, pp. 76–85, Mar. 2016.

[51] S. E. Awan, M. Bennamoun, F. Sohel, F. M. Sanfilippo, B. J. Chow, and G. Dwivedi, "Feature selection and transformation by machine learning reduce variable numbers and improve prediction for heart failure readmission or death," *PLoS One*, vol. 14, no. 6, Jun. 2019, Art. no. e0218760.

[52] D. Lucani, G. Cataldo, J. Cruz, G. Villegas, and S. Wong, "A portable ECG monitoring device with Bluetooth and Holter capabilities for telemedicine applications," in *Proc. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Sep. 2006, pp. 5244–5247.

[53] T. Bijlsma, M. Bekooij, P. Jansen, and G. Smit, "Communication between nested loop programs via circular buffers in an embedded multiprocessor system," in *Proc. 11th Int. Workshop Softw. Compil. Embedded Syst.*, Mar. 2008, pp. 33–42.

[54] G.-y. Ding, W. Sun, Z.-j. Li, and J. Xing, "The design and implementation of CNC system based on the circular buffer," in *Proc. Int. Conf. Syst. Sci. Eng. (ICSSE)*, Jun. 2012, pp. 534–538.

[55] C. Moreno and S. Fischmeister, "Fast and energy-efficient digital filters for signal conditioning in low-power microcontrollers," in *Proc. 54th ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, Jun. 2017, pp. 1–6.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-Learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[57] T. Lorenser, "The DSP capabilities of arm cortex-m4 and cortex-m7 processors," *ARM White Paper*, vol. 29, pp. 1–19, Nov. 2016.

[58] J. Lee, Y. Nam, D. D. McManus, and K. H. Chon, "Time-varying coherence function for atrial fibrillation detection," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 10, pp. 2783–2793, Oct. 2013.

[59] S. Mousavi, F. Afghah, and U. R. Acharya, "HAN-ECG: An interpretable atrial fibrillation detection model using hierarchical attention networks," *Comput. Biol. Med.*, vol. 127, Dec. 2020, Art. no. 104057.

[60] J. Wang, "An intelligent computer-aided approach for atrial fibrillation and atrial flutter signals classification using modified bidirectional LSTM network," *Inf. Sci.*, vol. 574, pp. 320–332, Oct. 2021.

[61] H. Zhang, Z. Dong, M. Sun, H. Gu, and Z. Wang, "TP-CNN: A detection method for atrial fibrillation based on transposed projection signals with compressed sensed ECG," *Comput. Methods Programs Biomed.*, vol. 210, Oct. 2021, Art. no. 106358.

[62] T. Radhakrishnan, J. Karhade, S. K. Ghosh, P. R. Muduli, R. K. Tripathy, and U. R. Acharya, "AFCNNet: Automated detection of AF using chirplet transform and deep convolutional bidirectional long short term memory network with ECG signals," *Comput. Biol. Med.*, vol. 137, Oct. 2021, Art. no. 104783.

[63] J. Duan, Q. Wang, B. Zhang, C. Liu, C. Li, and L. Wang, "Accurate detection of atrial fibrillation events with R-R intervals from ECG signals," *PLoS ONE*, vol. 17, no. 8, Aug. 2022, Art. no. e0271596.

[64] D. Kumar, A. Peimankar, K. Sharma, H. Domínguez, S. Puthusserypady, and J. E. Bardram, "Deepaware: A hybrid deep learning and context-aware heuristics-based model for atrial fibrillation detection," *Comput. Methods Programs Biomed.*, vol. 221, Jun. 2022, Art. no. 106899.

[65] N. Phukan, M. Sabarimalai Manikandan, and R. Bilas Pachori, "AFibri-Net: A lightweight convolution neural network based atrial fibrillation detector," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 12, pp. 4962–4974, Dec. 2023.

[66] Y. Li, L. Zhang, L. Zhu, L. Liu, B. Han, Y. Zhang, and S. Wei, "Diagnosis of atrial fibrillation using self-complementary attentional convolutional neural network," *Comput. Methods Programs Biomed.*, vol. 238, Aug. 2023, Art. no. 107565.

[67] H. Zhang, H. Gu, G. Chen, M. Liu, Z. Wang, and F. Cao, "An atrial fibrillation classification method based on an outlier data filtering strategy and modified residual block of the feature pyramid network," *Biomed. Signal Process. Control*, vol. 92, Jun. 2024, Art. no. 106107.

[68] M. Karri and C. S. R. Annavarapu, "A real-time embedded system to detect QRS-complex and arrhythmia classification using LSTM through hybridized features," *Expert Syst. Appl.*, vol. 214, Mar. 2023, Art. no. 119221.

[69] Mohebbanaaz, Y. Padma Sai, and L. V. Rajani Kumari, "Cognitive assistant DeepNet model for detection of cardiac arrhythmia," *Biomed. Signal Process. Control*, vol. 71, Jan. 2022, Art. no. 103221.

[70] S. P. Baller, A. Jindal, M. Chadha, and M. Gerndt, "DeepEdgeBench: Benchmarking deep neural networks on edge devices," in *Proc. IEEE Int. Conf. Cloud Eng. (IC2E)*, Oct. 2021, pp. 20–30.

[71] J. Chen, M. Jiang, X. Zhang, D. S. da Silva, V. H. C. de Albuquerque, and W. Wu, "Implementing ultra-lightweight co-inference model in ubiquitous edge device for atrial fibrillation detection," *Expert Syst. Appl.*, vol. 216, Apr. 2023, Art. no. 119407.

[72] M. A. O. Zishan, H. M. Shihab, S. S. Islam, M. A. Riya, G. M. Rahman, and J. Noor, "Dense neural network based arrhythmia classification on low-cost and low-compute micro-controller," *Expert Syst. Appl.*, vol. 239, Apr. 2024, Art. no. 122560.

[73] L. Falaschetti, M. Alessandrini, G. Biagetti, P. Crippa, and C. Turchetti, "ECG-based arrhythmia classification using recurrent neural networks in embedded systems," *Proc. Comput. Sci.*, vol. 207, pp. 3479–3487, Oct. 2022.

[74] M. Zylinski, A. Nassibi, and D. P. Mandic, "Design and implementation of an atrial fibrillation detection algorithm on the ARM cortex-M4 microcontroller," *Sensors*, vol. 23, no. 17, p. 7521, Aug. 2023.

[75] A. Faraone and R. Delgado-Gonzalo, "Convolutional-recurrent neural networks on low-power wearable platforms for cardiac arrhythmia detection," in *Proc. 2nd IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Aug. 2020, pp. 153–157.
[76] G. D. Clifford, C. Liu, B. Moody, H. L. Li-wei, I. Silva, Q. Li, A. Johnson, and R. G. Mark, "Af classification from a short single lead ecg recording: The physionet/computing in cardiology challenge 2017," in *Computing in Cardiology (CinC)*. Piscataway, NJ, USA: IEEE Press, 2017, pp. 1–4.
[77] E. A. P. Alday, A. Gu, A. J. Shah, C. Robichaux, A.-K. I. Wong, C. Liu, F. Liu, A. Bahrami Rad, A. Elola, S. Seyedi, Q. Li, A. Sharma, G. D. Clifford, and M. A. Reyna, "Classification of 12-lead ECGs: The PhysioNet/Computing in cardiology challenge 2020," *Physiological Meas.*, vol. 41, no. 12, Dec. 2020, Art. no. 124003.

MUHAMMAD YAZID (Senior Member, IEEE) received the B.Eng. degree in electronic engineering from Tohoku University, in 2004, and the M.Eng. degree in electronic engineering from The University of Tokyo, in 2006.

After several years of working experience with semiconductor industry, he is currently with the Biomedical Engineering Department, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, as a Lecturer. His main research interests include analog IC design, biomedical instrumentation, biomedical signal analysis, and machine learning.

MAHRUS ABDUR RAHMAN received the M.D. degree from Universitas Airlangga, in 1987, and the Ph.D. degree in medicine from Universitas Airlangga, in 2016.

He specializes in pediatric cardiology and a Medical Staff with the Pediatric Cardiology Division, Faculty of Medicine, Universitas Airlangga, and the Dr. Soetomo General Hospital, Surabaya, Indonesia.

NURYANI NURYANI received the Ph.D. degree from the University of Technology Sydney, specializing in health technologies. He is a Professor of medical instrumentation with the Department of Physics, Universitas Sebelas Maret, Indonesia. With a focus on medical instrumentation, biomedical engineering, artificial intelligence, and soft computing, he has spearheaded numerous research initiatives. His expertise encompasses coordinating research projects dedicated to advanced detection systems for cardiovascular issues, including atrial fibrillation, heart arrhythmia, and hypertension.

ARIPRIHARTA (Member, IEEE) received the S.T. and M.T. degrees in electrical engineering from Brawijaya University, Indonesia, in 2004 and 2012, respectively, and the Ph.D. degree in electronic engineering from NKUAS (currently NKUST), Taiwan, in 2017.

Since 2005, he has been with the Faculty of Engineering, State University of Malang, Indonesia, as a Lecturer, and a Research Leader with Power Electronics and Drives Laboratories, the IoT, and algorithms. In 2019, he joined the Centre of Advanced Materials for Renewable Energy (CAMRY), Indonesia, as the General Secretary. He received many grants from Indonesia Government, especially under DRPM and PNBP schemes.