IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 41, No. 1, February, 2024, pp. 179-188
Journal homepage: http://iieta.org/journals/ts

FPGA Implementation of a CNN Application for ECG Class Detection

Marwa Fradi[1*], Khriji Lazhar[2], El-Hadi Zahzah[3], Mohsen Machhout[1]

[1] Physic Department, Faculty of Sciences of Monastir, Laboratory of Electronics and Micro Electronics, University of Monastir, Monastir 5000, Tunisia
[2] Department of Electrical and Computer Engineering, College of Engineering, Sultan Qaboos University, P.O. No. 34, PC 123, Muscat, Sultanate of Oman
[3] Laboratory of Informatics, Image and Interaction (L3i, France), La Rochelle University, La Rochelle 17042, France

Corresponding Author Email: marwa.fradi@fsm.rnu.tn

https://doi.org/10.18280/ts.410114

Received: 29 June 2022
Revised: 1 January 2023
Accepted: 11 January 2023
Available online: 29 February 2024

Keywords:
ECG, Pynq-Z2, CNN, Co-Design (Hard/Soft), time, accuracy

ABSTRACT

Convolutional Neural Networks (CNN) show huge necessity in medical area diagnosis, CNN can be used for ECG features extraction, heartbeats classification and abnormal beats detection, helping clinicians to get the true diagnosis of cardiovascular diseases at early stage. In this context, an optimized CNN is proposed to be implemented on Pynq-Z2 board for Electrocardiography (ECG) signal class detection. As first step, a CNN has been implemented on the processor ARM Cortex A9 of Pynq Z2. Implementation results show the efficiency of our purpose, achieving accuracies results of 98.86%, 98.61% and 98.39% for the training, validation and test process respectively. Then to improve the inference results a Hardware/Software Codesign has been proposed due to the parallel architecture of FPGA, time process acceleration has reached 10 times compared to the implementation on the Processor. Moreover, a gain of the surface has been achieved by using low number of resources. Thus, a real time application has been reached with a very excellent class accuracy detection going to 99.45% for the training, 99.12% for the validation and 99.03% for the testing processes, when tested on MIT-BIH ECG signals in a short time process with 0.0018s/signal through the test process and 0.005s/ signal during the training.

1. INTRODUCTION

Nowadays, medical image and signal processing applications are preferred due to the importance of human health alongside technological developments in deep learning or (DL), which currently represents the key solution for related medical applications to an automatic diagnosis. In this context, a major interest is given to convolutional neural networks (CNN) for the automatic detection of different heart diseases from real-time ECG signals. Where, the electrocardiogram (ECG) has become a useful tool for the diagnosis of cardiovascular diseases as it is fast and noninvasive. The most important criteria for the applications used in these fields is to guarantee that the system operates at high speed and in real time. Thus, FPGAs are commonly used in such applications. In this work, a CNN algorithm for class detection of ECG signals is implemented in real time on the Pynq Z2 based FPGA board. The detected signals are implemented on Pynq FPGA It has been widely used for research and machine learning prototyping for medical applications. In this context, a medical application is developed based CNN architecture for the detection of ECG signals in real time on Pynq Z2 FPGA.

The implementation of CNN application on Pynq Z2 is done to more accelerate time processing, due to the parallel FPGA architecture, which speed up the mathematical operations and enhance precision results, thanks to the creation of a new additionner multiplier IP.

1.1 Contributions

The contribution in this paper is illustrated as follows: First, a design of a new IP has been made, which plays a huge role to accelerate time execution of the proposed CNN algorithm. Then a deep CNN architecture has been proposed consisting of convolutional layers, where more we are going deeper through layers, more we get excellent precision and detection results. Finally, our proposed codesign architecture present a medical real time application that can be needed in hospitals; helping clinicians to get the true diagnostic in a short time process. Thus, the importance of an embedded medical system.

2. STATE OF THE ART

Deep Learning neural networks models such as Convolutional neural networks CNN and Reccurent Neural Network RNN have shown a huge development [1, 2]. The CNN has shown an interesting role [3] in various area such medical and internet of Things axes [4, 5]. Moreover, it has shown excellent classification precision, in cardiovascular diseases detection such atrial fibrillation [6] and myocardial infarction [7]. Moreover, it shows excellent role in heartbeats

classification [8] given its ability to extract features from ECG signals to avoid manual extraction complexity. Furthermore, ECG signals shows an important need to be we analyzed with high accuracy in a short time process. Kotareddy [9] have implemented CNN model on different platforms for ECG class detection. Achieved accuracies are reaching to accuracy of the model over the validation set is 92.83% and test set is 95.00%. Yu et al. [10] in 2021 and thanks to specially built hardware, proposed a variety of FPGA-based accelerator systems that incorporated software and hardware optimization techniques to attain great performance and energy efficiency. CNN are strongly recommended for embedded applications by Bettoni et al. [11]. Binary CNN was implemented on a SOC , surpassing the state of art CNN based ECG identification method. Wei et al. [12] proved that trends of AI based biomedical processing algorithmss and Hardware method. Rana and Kim [13] have proposed a CNN model, to be implemented on Pynq-Z2, achieving an accuracy of 95% using the MIT-BIH database. The SNN has acieved 90% of accuracy for ECG signal classification.. the proposed DNN, by Ney et al. [14] proved their ability to solve the problem in medical domains related to embedded systems to demonstrate their performances in terms of energy consumption. Pynq Z2 has been used by Mándi et al. [15] for AI algorithms implementation for image processing acceleration. Pynq Z2 has been developed by TUL corporation [16]. A CNN shows to be an automatic extractor of deep features from ECG-data without complexity requirement by Zairi et al. [17]. The performance of the ECG classifier models is evaluated based on accuracy and power. Among the three AI algorithms, the SNN requires the lowest power consummation of 0.226 W on-chip, followed by MLP (1.677 W), and CNN (2.266 W). However, the highest accuracy is achieved by the CNN (95%), followed by MLP (76%) and SNN (90%) [10]. Li et al. [18] in 2018 have implemented CNN model on different devices for atrial fibrillation detection. Pynq proves to be the best devices in terms of accuracy, time processing and power. Chourasia et al. [19] in 2020 have proposed a 1D CNN architecture for ECG classification into five classes; the achieved accuracy has reached 97.36% that is a promising result, thus the importance of using a CNN architecture. Yıldırım et al. [20] in 2018 have proposed a deep CNN architecture, for fast ECG signals classification, with less complexity achieving 92% of accuracy. Wu et al. [21] have proposed a one dimensional CNN architecture, after MIT-BIH ECG signals processing. The proposed CNN combined with the preprocessing step has given excellent accuracies for ECG classification going to 99.35%. Thus, the huge role of pre-processing in increasing accuracies results [22, 23].

3. PROPOSED ARCHITECTURE METHOD

Our Convolutional Neural Network (CNN) algorithm used for the detection of ECG signals has been implemented on the Pynq Z2 board. In order to have a real-time medical application on an embedded system, two implementations methods are successfully completed with an acceleration of the execution speed. In a first step, we use a pure software implementation of our CNN on the ARM processor of Pynq Z2. While in a second step, in order to obtain better acceleration results, we resort to an implementation of a hard / soft co-design on the Pynq Z2 FPGA.

3.1 Software implementation on ARM processor

The synoptic flow of the soft implementation is presented by Figure 1, where we develop a software application based on the convolutional neural network for the detection of different cardiac pathologies. Subsequently, the image of Pynq Z2 has been downloaded and saved i in the SD card, this image carries the Linux OS of Pynq Z2. Finally, the connection is made between the computer and the Pynq card using the Ethernet cable and the USB cable, and our CNN code will be implemented on the ARM processor of Pynq Z2.

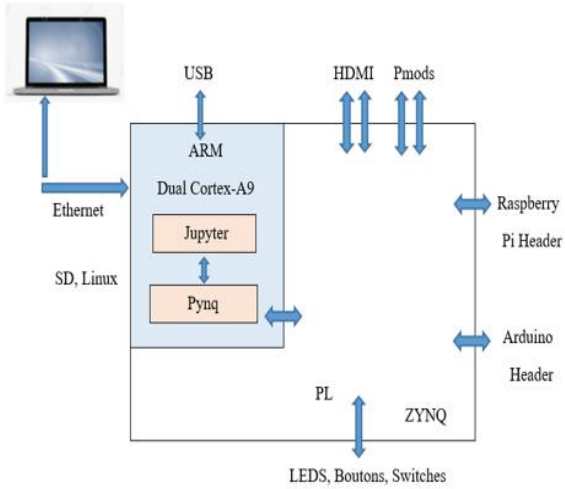

Figure 1. Synoptic flow of software implementation on ARM processor

3.1.1 Dataset preparation

Our ECG dataset presents three categories: the Mit-Bih, the PTB and the processed PTB ECG signals. These signals are accessible free from Physionet. Each category is devided into three processes: Training, validation and test with 70%, 20% and 10% respectively as described in Table 1.

Table 1. Dataset preparation

Dataset	Train	Validation	Test
Mit-Bih	78798=70%	21892=20%	8756=10%
PTB	10548=70%	3031=20%	1172=10%
Processed PTB	10548=70%	3031=20%	1172=10%

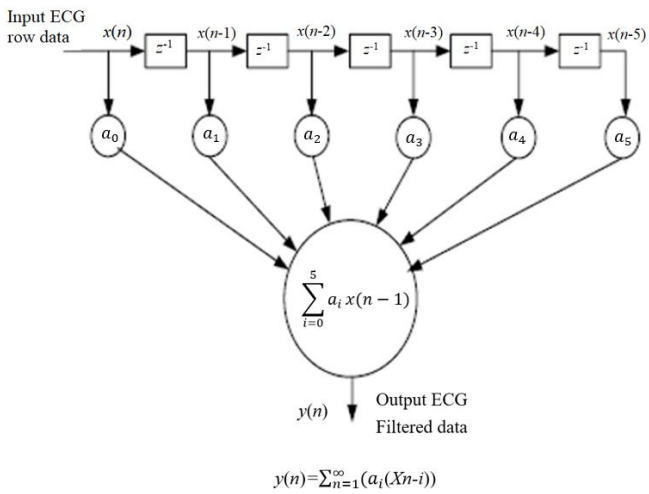

Figure 2. FIR-I of 6 order

3.1.2 Dataset processing step

A FIR-6 as presented in Figure 2 has been applied on 1D-ECG signals, with the aim to remove noise, getting filtered signal as depicted in Figure 3, where the used filter removes high frequencies from ECG signals. As a result, as described in Figure 3 (b), noisy oscillations have been removed.

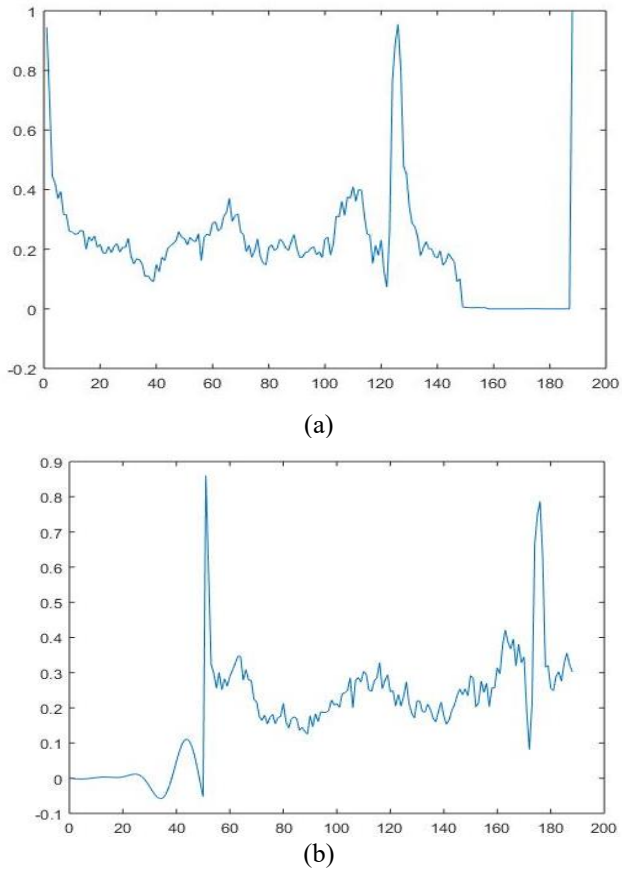

Figure 3. Filtered ECG signal: (a) PTB signal, (b) Filtered PTB signal

3.1.3 CNN architecture

The proposed CNN architecture as illustrated by Figure 4 is applied on a one-dimensional input ECG signal of size (187×1). It includes 8 convolutional layers, using 16 convolutional filters, playing a huge role in feature maps extraction. Then, a Maxpooling layer is applied to extract the most important feature information from ECG signals, a dropout layer to ovoid the overfitting during the training process, two fully connected layers, to compute and determine the class score. Finally, a Softmax classier layer is applied for ECG class decision.

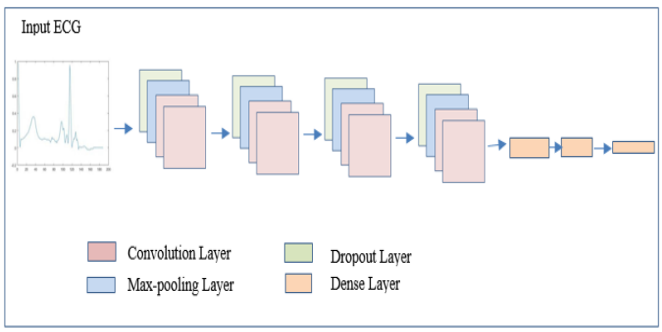

Figure 4. Proposed CNN architecture

3.1.4 Optimizer networks

Adam-optimizer is based on the moment's estimations, using them to optimize functions. Indeed, Adam determines an exponential weighted calculation, moving average of the gradient, Then the calculated gradient is squared. The use of Adam is referred to its crucial role that plays in decreasing the number of fluctuations, minimizing the error rate. Then its implementation is easy and depends on little memory space. Moreover, its use is sufficient enough for both small and large dataset. Finally, it works well with noisy problems as depicted in depicted results with the noisy PTB, accuracy results is reaching 99.28%.

3.2 Co-Design implementation on FPGA based Pynq-Z2 board

Our design consists of two main parts: the PL (bitstream) design and the Tcl file of the project block diagram. The whole proposed co-design is described in Figure 5.

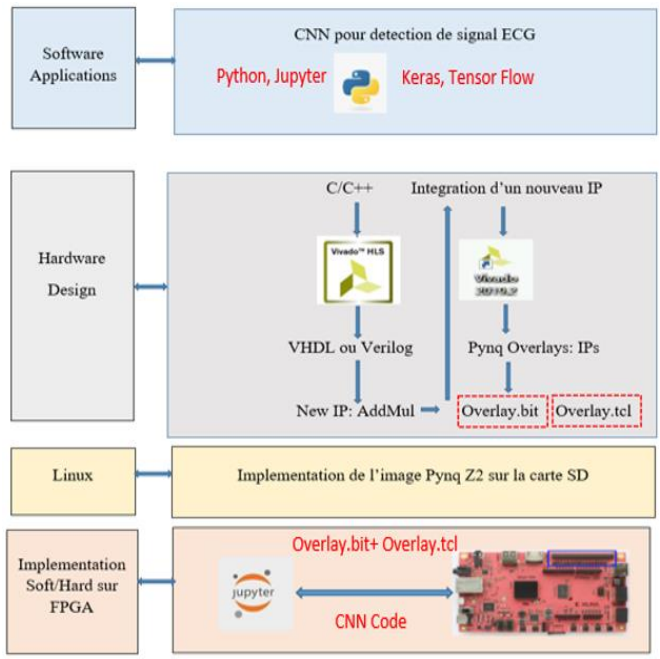

Figure 5. Proposed co-design soft/hard

3.2.1 Creation of a new IP: Multiplier adder

For our hardware design, we resort to creating a new IP to speed up the process of running our CNN code for automatic detection of ECG signals. For this, we used Vivado HLS with the C / C ++ tool to create a multiplier adder. The choice to add a multiplier adder is based on the large role of the addition operation as well as the multiplication operation which presents the most critical operation in a signal processing application. It is an operation which is very often requested and which deserves to be optimized in surface as well as in speed. This new IP, as presented in Figure 6, that we have programmed from Vivado HLS, constitutes a multiplier adder which is easily, instantly will be imported and integrated and used in Vivado Block Design.

3.2.2 Material conception of our "Block design"

Our block design consists of a new IP that we created from Vivado, layered with other IPs in Vivado SDK. This description, close to the electronic architecture of the final system, can be easily integrated by the main tool Vivado as a

component in the block design of the complete architecture, which must include some additional elements like the IP block of the system. Zynq processing as shown in the following figure. This IP speeds up the CNN execution process with its addition and multiplication operations. Also, Vivado's auto connect function makes it easier to connect the PS modules and the new created IP, creating two other auxiliary blocks: the reset module and the interconnection devices as shown by the block design in the following figure.

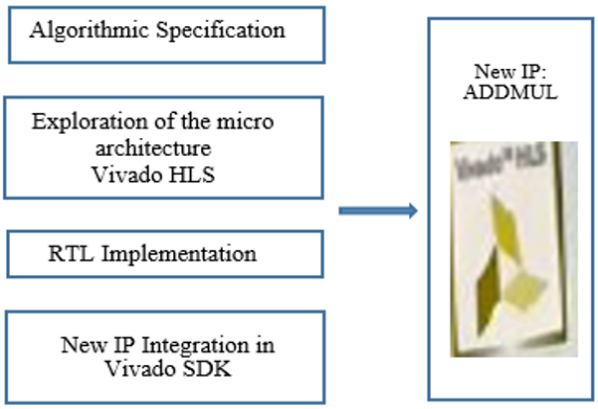

Figure 6. Creation of a new IP: Addmul

Then the entire system can be synthesized in a process that transforms the IP-based design into a low-level description based on logic gates. The implementation process then maps this description at the gate level to the resources of the selected FPGA, optimizing the location of the electronics in the available CLBs and their interconnection (routing). Our block design is described in Figure 5. The following figure shows the synoptic flow of our soft / hard codesign for the automatic detection of the ECG signal class. integrated as shown in the following figure. These circuits commonly called FPGA (Field Programmable Gate Array) contain in their structure, CLB (Configurable Logic Block) blocks linked together by a matrix of connections. Each CLB block can contain multiple cells which form the basic building block of the set. This programmable cell can be configured to generate a particular function. The interconnection of a set of programmable cells constitutes the overall function of the implanted system.

Today, several tens of thousands of CLBs are available on a single integrated circuit. To this are added RAM blocks and recently multiplier blocks wired on the same circuit. This architecture has naturally appealed to circuit designers for real-time signal processing applications, which generally require very significant resources. FPGAs also contain hardened components for commonly used functions. Additionally, FPGAs support partial reconfiguration dynamics, which can have positive impacts for large DL models, where individual layers could be reconfigured on FPGAs without disrupting computation in other layers. To speed up hardware designs, the FPGA board could be a perspective over GPU synthesis, implementation and export of the new IP. Finally, after the overall synthesis and implementation processes, the bit stream file which contains the configuration information can be generated to be up-loaded to the FPGA, thus building the custom hardware design. Once our block design is designed and the final synthesis guidelines are applied, the system is ready to be exported to an FPGA, using Vivado-SDK, a bitstream or binary file (.bit file) will be generated which can be used. To program the Zynq PL.

The Vivado IP Integrated Block Design Tcl file for PL Design is used by PYNQ to automatically identify the Zynq system configuration, including versions, interrupts, resets, and other control signals. Based on this information, parts of the system configuration can be changed automatically from PYNQ, drivers can be automatically assigned, features can be enabled or disabled, and signals can be connected to the corresponding Python methods. The Tcl file must be generated and supplied with the bitstream file as part of an overlay. The Tcl file can be generated in Vivado by exporting the IP Integrator block diagram at the end of the overlay design process. The PYNQ PL class will automatically analyze the Tcl. This automatically generated Tcl should ensure that it could be properly analyzed by the PYNQ. These two files are uploaded under our CNN code.

The hardware conception design is as described by Figure 7.

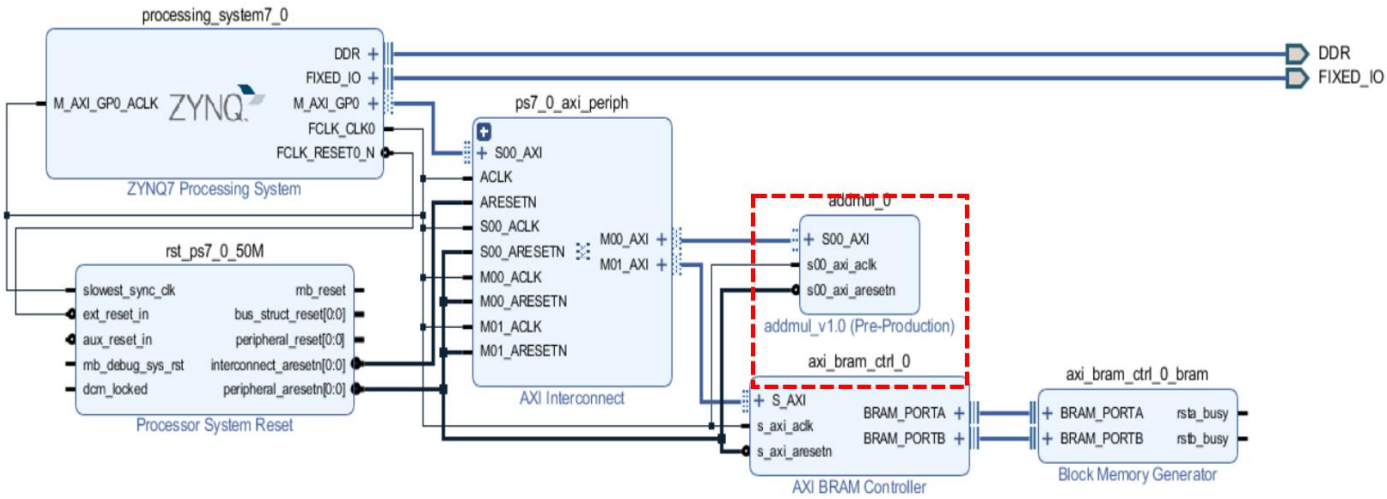

Figure 7. Conception of the hardware design using Vivado

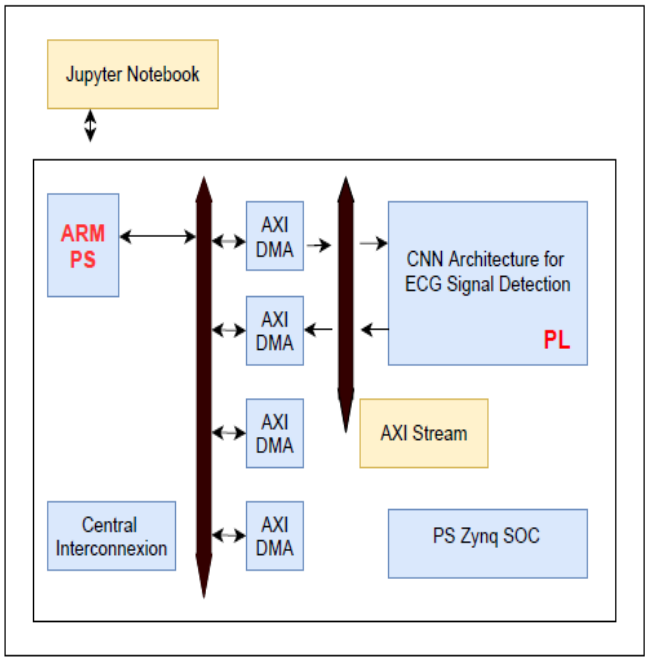

Figure 8. Connection between Jupyter notebook and Pynq Z2

3.2.3 Connection to jupyter notebooks

To establish the connection between the card and our computer, the pynq Z2 card must be connected to the Ethernet port of the router, as well as our computer must be connected to Ethernet, then we access the following address http: // pynq: 9090. To connect Jupyter Notebooks to Pynq Z2, we open the web browser following http://192.168.2.99:9090. The connection is as depicted in Figure 8.

3.2.4 Implementation process

During this phase, and after making sure that our FPGA card is connected to our PC and to the jupyter, we import the design.

Bit and the design.tcl files under the code to be implemented. These files carry the design hardware description that we designed from Vivado SDK and Vivado HLS as shown in the Figure 9. The execution time of our CNN for the detection of ECG signals on the Pynq Z2 ARM processor is 58.68s to classify 10,605 signals, where each signal has 3600 samples. Which means 5ms to classify and detect each signal.

4. RESULTS

4.1 Software implementation results for ECG signals detection on the ARM processor

4.1.1 Accuracy results

Table 2 represents the achieved results through training, validation and test processes which is explained by the huge role played by used optimizer networks such as Adam and SGD. Optimizers networks have decreased the error rate and optimized accuracies results.

Table 2. CNN based Adam optimizer results on ARM - Processor

Dataset	Train	Validation	Test
Mit-Bih	98.86%	98.61%	98.39%
PTB	98.51%	98.11%	98.21%
Processed PTB	**99.12%**	99.08%	**98.79%**

Figure 9. Pynq Z2 connection to PC and Jupyter notebook

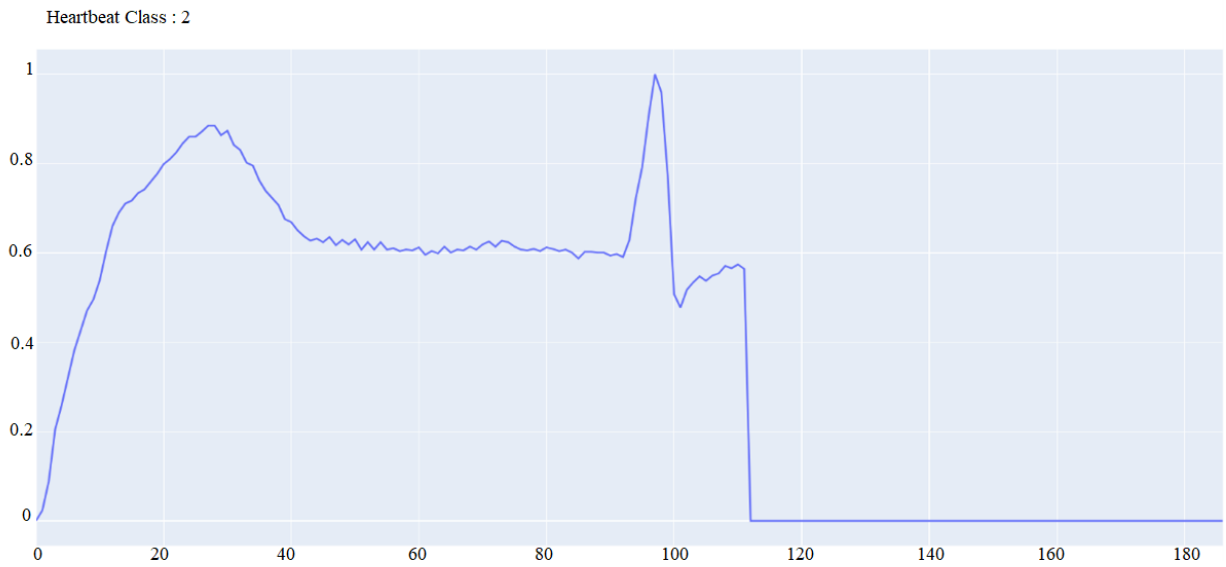

Figure 10. Mit-Bih Class 2 recognition

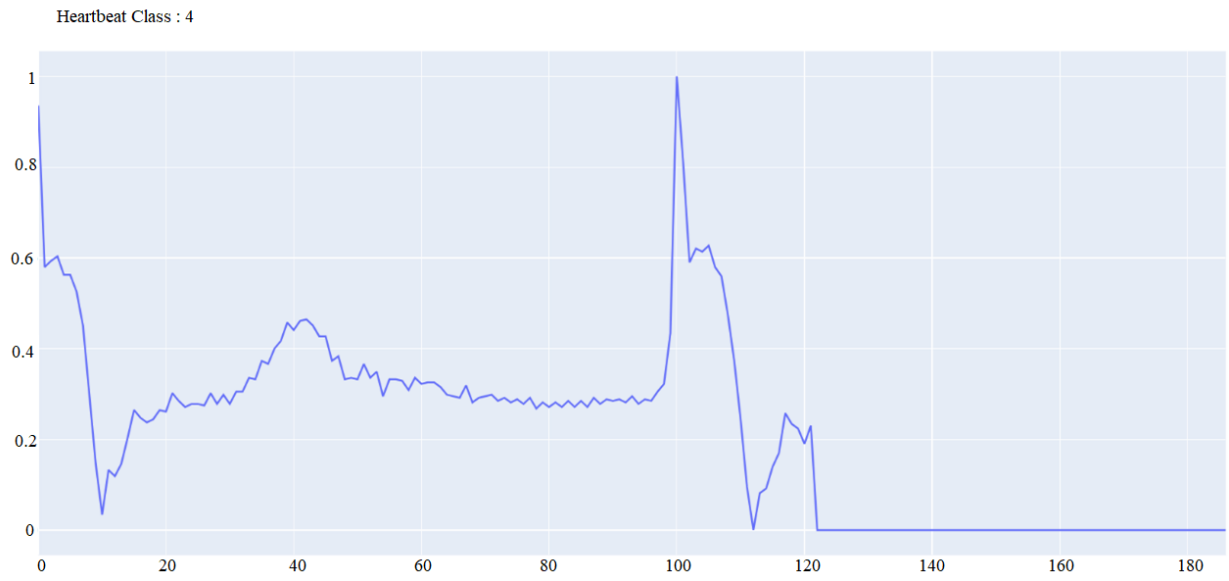

Figure 11. Mit-Bih Class 4 recognition

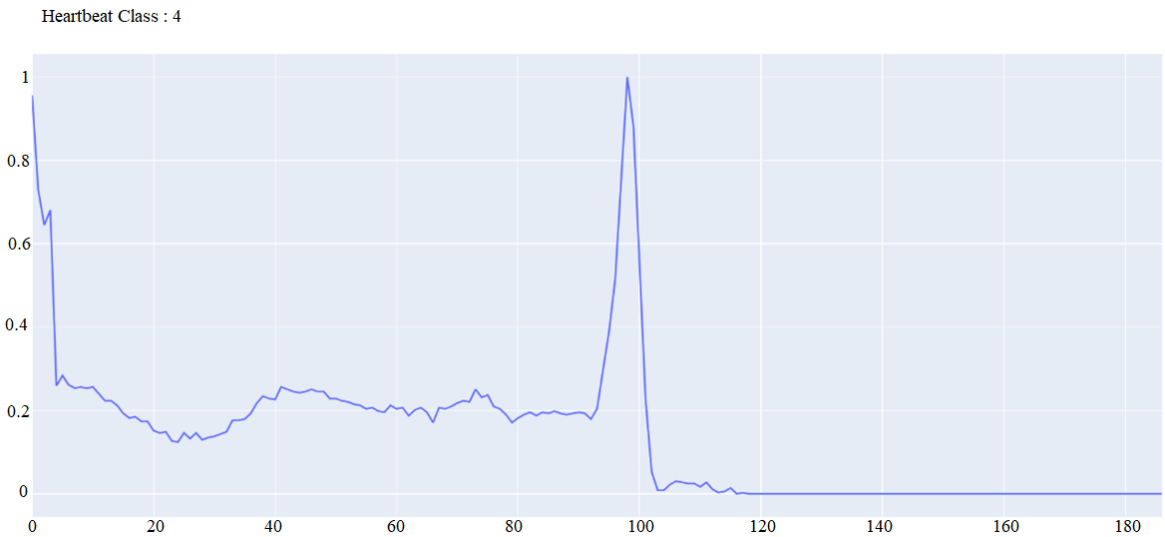

Figure 12. Mit-Bih Class 4 recognition

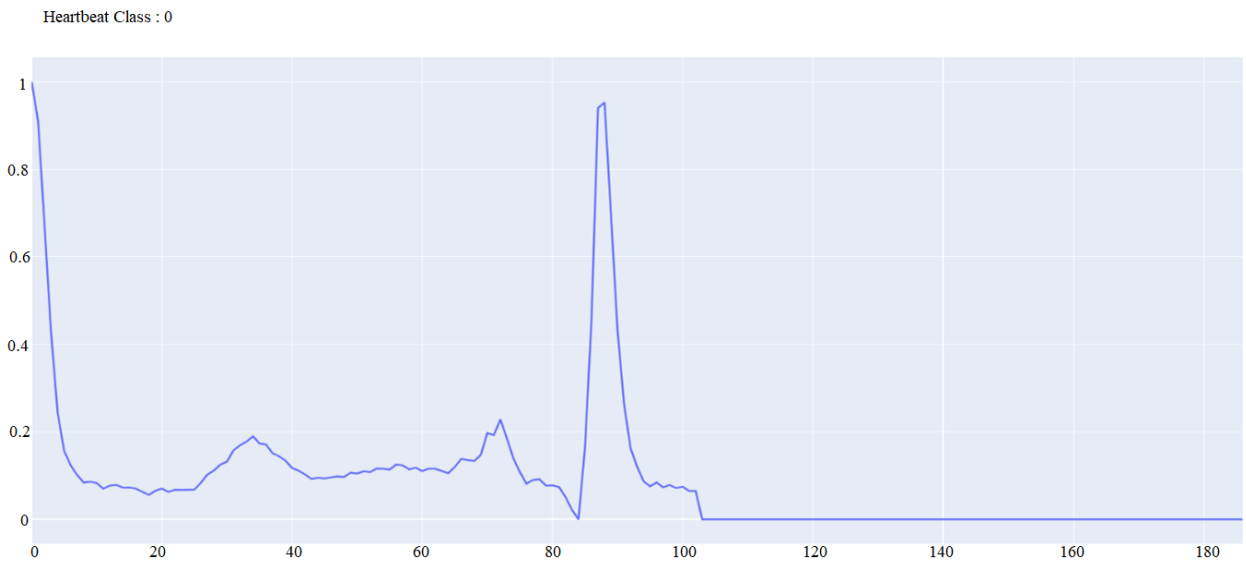

Figure 13. PTB class 0 recognition

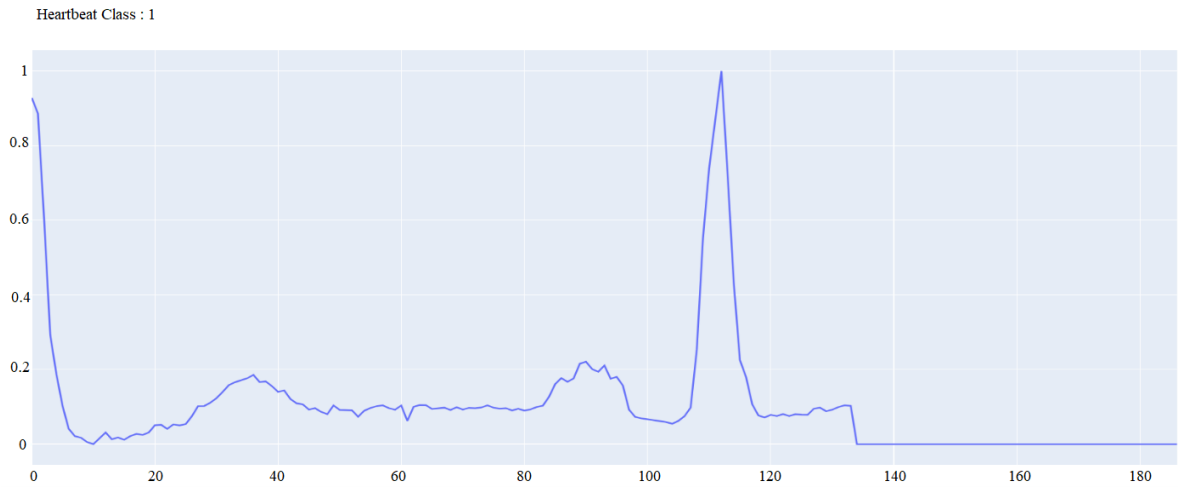

Figure 14. PTB class 1 recognition

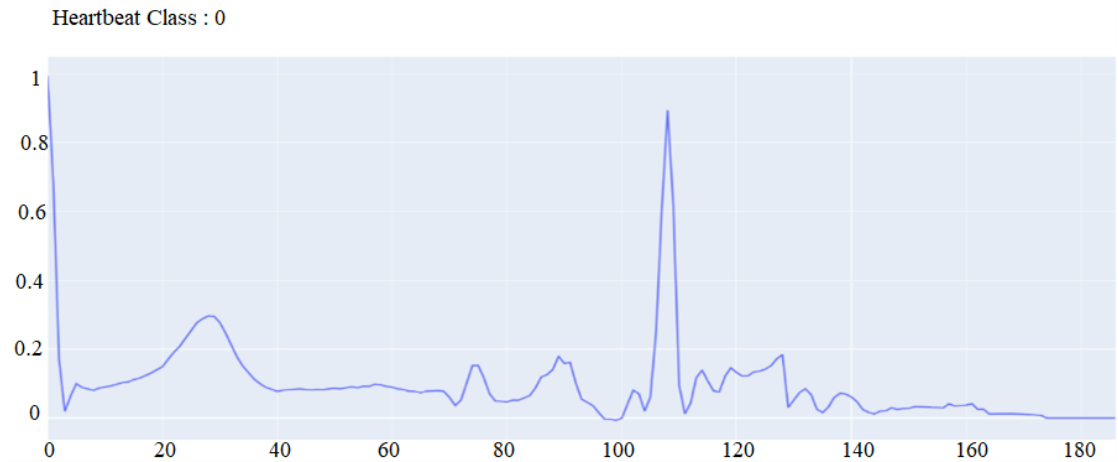

Figure 15. Processed PTB class 0 recognition

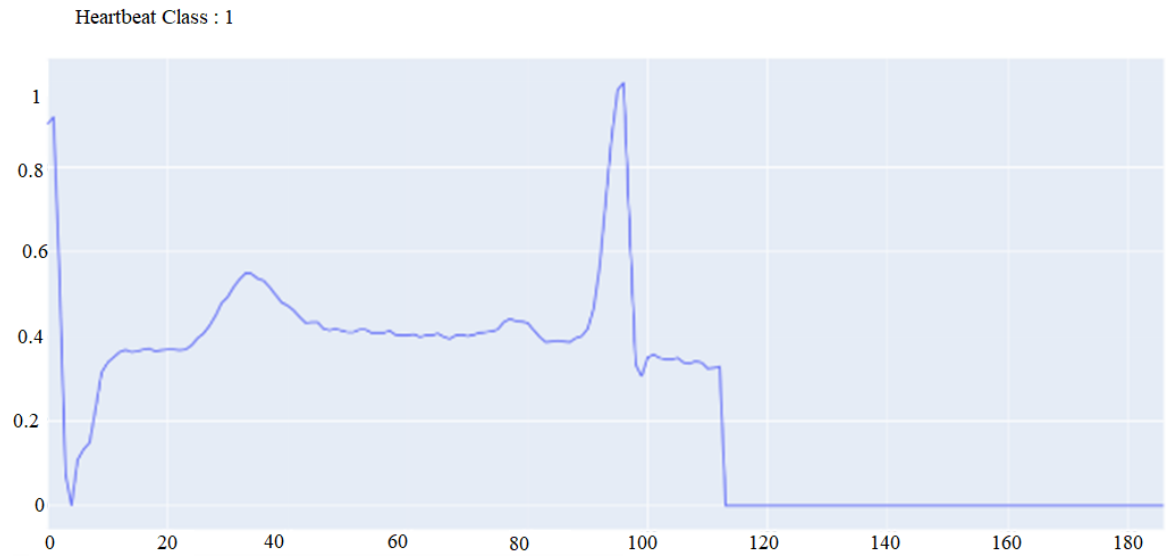

Figure 16. Processed PTB class 1 recognition

4.1.2 ECG class detection results on ARM processor

As presented by the following figures, MIT-BIH classes were well detected, each class represents a pathologic abnormality disease. Figures 10-12 represent different classes such as premature ventricular contraction (Class 2), and inclassable beats for (Class 4), whereas, Figures 13 and 14 show two detected PTB classes such as normal and abnormal ECG signal, respectively class 0 and class1. Finally, Figures 15 and 16 show both normal and abnormal ECG signal for the processed PTB signals.

4.2 Co-design implementation results

4.2.1 Used resources

The used resources are shown in Table 3, where we have used a limited number of resources, which saves space.

Table 3. Used resources

Ressources	Utilisation	Total	Used
LUT	1477	53200	2.78%
LUT RAM	74	17400	0.43%
FF	2191	106400	2.06%
DSP	3	220	1.36%
PS1	1	1	100%

4.2.2 Accuracies results using FPGA based Pynq Z2

Proposed CNN architecture based on Convolutional layers, Maxpooling layer, dropout, two dense layers and Softmax classifier layer has achieved promising accuracies for both Mit-Bih and PTB signals as depicted in Table 4 and Table 5. Adam and SGD optimizers were introduced through the propounded CNN, to ovoid overfitting. Adam has proved to be better than SGD in terms of accuracies results. This phenomenon is explained by of Adam behavior, that enables the convergence to a global minimum, however, SGD 's fluctuations enable to jump to better local minimum. The preprocessing step, that aims to remove noise from PTB signal has a big effect on enhancing accuracies as depicted in Tables 4 and 5, achieved results are reaching to 99.47%, 99.07% using the SGD optimizer. Moreover, results are coming to 99.58%and 99.22% for both train and test processing using Adam optimizer. Compared to accuracies results using ARM processor, Table 4 and 5 guarantee better results due to the implementation on Pynq FPGA, thus the importance of FPGA to speed up the calculation operations and thus getting results that are more accurate.

Table 4. Accuracies results using SGD optimizer

Dataset	Train	Validation	Test
Mit-Bih	99.22%	98.89%	98.76%
PTB	99.13%	98.92%	98.57%
Processed PTB	**99.47%**	99.18%	99.07%

Table 5. Accuracies results using Adam optimizer

Dataset	Train	Validation	Test
Mit-Bih	99.45%	99.12%	99.03%
PTB	99.21%	99.11%	98.89%
Processed PTB	**99.58%**	99.34%	**99.22%**

4.2.3 ECG-class detection on Pynq-FPGA

Figure 17 presents the implementation results of our codesign in terms of execution speed where we achieve a time of 0.0022s/signal for the implementation process on FPGA and 0.0055s for the implementation process on ARM -pynqZ2. Hence, the importance of our codesign in accelerating execution processes.

4.2.4 Time–processing results for both software implementation and co-design implementation on Pynq-Z2 board

The execution time of our CNN for the detection of ECG signals on the Pynq Z2 ARM processor is 58.68s to classify 10,605 signals, where each signal has 3600 samples. Which means 5ms to classify and detect each signal. These results are very encouraging in view of the large database. The result obtained can be considered as a real-time processing speed, but our goal is to accelerate this execution time more so as to have medical information in a very short time. To achieve this goal, we are going to develop a hard / soft codesign to implement our CNN on the Pynq FPGA board, given the very important role that FPGAs play in speeding up computational operations. Moreover, GPUs help in accelerating deep learning frameworks with large datasets. Achieved results are as depicted in Table 6.

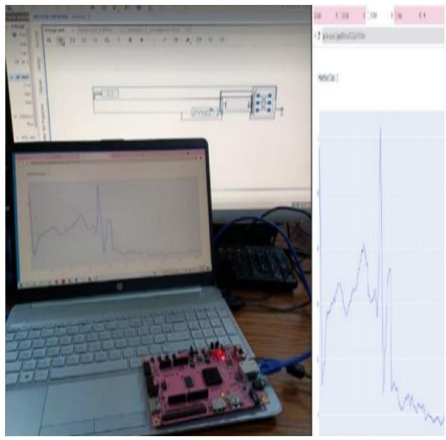

Figure 17. Processed PTB Class 1 detection

Table 6. Time processing results

Dataset	Train	Validation	Test
Mit-Bih	0.0055s/signal	0.0022s/signal	0.007s/signal
Processed PTB	0.0042s /signal	0.0018s/signal	0.006s/signal

5. DISCUSSIONS

5.1 Implementation comparison on processor and Pynq FPGA

The implementation of CNN application on Pynq Z2 is done to more accelerate time processing, due to the FPGA architecture. The created new IP which is an additionner-multiplier aims to accelerate time execution application to get more accurate class in a short time. To conclude, the Field-programmable gate arrays (FPGAs) have been used to construct hardware accelerators for CNNs. Today the Python language is becoming the most promising language for FPGA [24] and GPUs implementation [25, 26], showing their huge role in CNN algorithms acceleration using a huge number of ECG dataset and GPUs implementation, showing their huge role in CNN algorithms acceleration using a huge number of ECG dataset.

5.2 Comparison with the state of the art

The comparative study is summarized in Table 7, where we surpass all state of the art methods. Compared with Burger in 2020, we gain more than 2% in terms of accuracy and in terms of execution speed with 0.008s. Furthermore, the implementation of a CNN code on FPGA for the analysis of ECG signals takes 0.003 s for each signal. In addition, we have

outperformed the state of the art in terms of precision where we gained 9% compared to Ran and Kim [13] in 2020. This is because of the architecture of our hardware description for programming pynq Z2 FPGAs.

Moreover, we have outperformed [19], by a value of 5% where authors have used ANN architecture. This gain is explained by the whole role that plays our proposed CNN architecture, which is deep, and robust for real time signal classification. To conclude, more we are going deeper through the proposed neural network, more we get accurate results. Compared with the proposed state of the art model in the studies [7, 8], we outperform them. This is due to the pre-processing stage by the FIR-6 application and by the reasonable design of the proposed CNN. Thus, the importance of the pre-processing stage. The comparative study has demonstrated, that CNN architecture has a big effect on optimizing accuracies results, in fact, more we are going deeper, more we get accurate precision results, surpassing state of the art methods. Then, process to implement a CNN system on an embedded device, show a high speed of mathematical operations, that leads to get excellent accuracies through epochs during train, validation and test. The need to make a medical application on embedded system such as Pynq based FPGA is explained by the necessity of our proposed hardware co-design, more explained in Table 3, which need 3 processors, to make a good acceleration process. Thus, the necessity of a CNN implementation on FPGA for medical application, which can be used as a good candidate in hospitals and clinics.

Table 7. Comparative study with the state of the art

Dataset	Neural Network	Pynq Z2 FPGA	ARM Pynq Z2	GPU
ECG [22]	CNN	Time = 0.008s/signal Accuracy = 97%		
ECG [13]	SNN	Accuracy= 90%		
	CNN	Accuracy=95%		
	MLP	Accuracy=76%		
MIT-BIH [20]	CNN	Accuracy= 92%		
MIT-BIH [21]	CNN	Accuracy= 99.35%		
ECG-Physionet [9]	CNN	Time = 1.93s /		2.58s
ECG [23]	DCNN- BiLSTM	Train = 89.3% F1 = 0.89		
ECG-Mit-Bih [8]	CNN-LSTM	Accuracy =99.02%		
	CNN-GRU	Accuracy=99.61%		
ECG [7]	CNN	Accuracy= 87%		
Mit-Bih	**Proposed Work**	0.0042s Accuracy=99.45%	0.0018s	Time= 0.007s

6. CONCLUSION

Through this work, we have developed a medical real time application based codesign application for ECG class detection. Our proposed CNN architecture has proved to be robust to get the true class for three types of ECG dataset such as MITBIH, PTB and the processed PTB. Achieved results have shown to be excellent and promising, reaching up to 99.65% for the processed PTB dataset in a short time process due to its implementation on PynqZ2 based FPGA.

ACKNOWLEDGMENTS

This work is supported by the Ministry of High education and scientific research in Tunisia, La Rochelle University, L3I Laboratory, France and a financial support by Sultan Quaboos University, Oman.

REFERENCES

[1] Marwa, F., Zahzah, E.H., Bouallegue, K., Machhout, M. (2022). Deep learning based neural network application for automatic ultrasonic computed tomographic bone image segmentation. Multimedia Tools and Applications, 81(10): 13537-13562. https://doi.org/10.1007/s11042-022-12322-3

[2] Efendi, S., Nurahmadi, F., Nainggolan, P.I. (2023). The role of recurrent convolutional neural network in IoT for building a security artificial intelligence and home assistance system. International Journal of Safety and Security Engineering, 13(3): 403-408. https://doi.org/10.18280/ijsse.130303

[3] Vasudeva, S.T., Rao, S.S., Panambur, N.K., Shettigar, A.K., Mahabala, C., Kamath, P., Linul, E. (2022). Development of a convolutional neural network model to predict coronary artery disease based on single-lead and twelve-lead ECG signals. Applied Sciences, 12(15): 7711. https://doi.org/10.3390/app12157711

[4] Olayiwola, J.O., Badejo, J.A., Okokpujie, K., Awomoyi, M.E. (2023). Lung-related diseases classification using deep convolutional neural network. Mathematical Modelling of Engineering Problems, 10(4): 1097-1104. https://doi.org/10.18280/mmep.100401

[5] Singh, P., Rathee, N., Sharda, S., Kumar, S. (2023). Comparative study of Rough Set-based FCM and K-Means clustering for tumor segmentation from brain MRI images. Revue d'Intelligence Artificielle, 37(4): 921-927. https://doi.org/10.18280/ria.370412

[6] Ping, Y., Chen, C., Wu, L., Wang, Y., Shu, M. (2020). Automatic detection of atrial fibrillation based on CNN-LSTM and shortcut connection. In Healthcare, 8(2): 139. https://doi.org/10.3390/healthcare8020139

[7] Makimoto, H., Höckmann, M., Lin, T., Glöckner, D., Gerguri, S., Clasen, L., Kelm, M. (2020). Performance of a convolutional neural network derived from an ECG database in recognizing myocardial infarction. Scientific Reports, 10(1): 1-9. https://doi.org/10.1038/s41598-020-65105-x

[8] Yao, G., Mao, X., Li, N., Xu, H., Xu, X., Jiao, Y., Ni, J. (2021). Interpretation of electrocardiogram heartbeat by CNN and GRU. Computational and Mathematical Methods in Medicine, 2021. https://doi.org/10.1155/2021/6534942
[9] Kotareddy, S.K.R.K. (2021). Edge Devices for Detecting Atrial Fibrillation Using Deep Learning. The University of Texas at Dallas.
[10] Yu, X., Majoros, T., Oniga, S. (2021). Hardware Implementation of CNN Based on FPGA for EEG Signal Patterns Recognition. In 2021 International Conference on e-Health and Bioengineering (EHB), Iasi, Romania, pp. 1-4. https://doi.org/10.1109/EHB52898.2021.9657679
[11] Bettoni, M., Urgese, G., Kobayashi, Y., Macii, E., Acquaviva, A. (2017). A convolutional neural network fully implemented on FPGA for embedded platforms. In 2017 New Generation of CAS (NGCAS), Genova, Italy, pp. 49-52. https://doi.org/10.1109/NGCAS.2017.16
[12] Wei, Y., Zhou, J., Wang, Y., Liu, Y., Liu, Q., Luo, J., Huang, L. (2020). A review of algorithm & hardware design for AI-based biomedical applications. IEEE Transactions on Biomedical Circuits and Systems, 14(2): 145-163. https://doi.org/10.1109/TBCAS.2020.2974154
[13] Rana, A., Kim, K.K. (2020). Comparison of artificial neural networks for low-power ECG-classification system. Journal of Sensor Science and Technology, 29(1): 19-26. https://doi.org/10.5369/JSST.2019.29.1.19
[14] Ney, J., Loroch, D., Rybalkin, V., Weber, N., Krüger, J., Wehn, N. (2021). HALF: Holistic auto machine learning for FPGAs. In 2021 31st International Conference on Field-Programmable Logic and Applications (FPL), Dresden, Germany, pp. 363-368. https://doi.org/10.1109/FPL53798.2021.00069
[15] Mándi, Á., Máté, J., Rózsa, D., Oniga, S. (2021). Hardware accelerated image processing on FPGA based PYNQ-Z2 board. Carpathian Journal of Electronic & Computer Engineering, 14(1): 20-23.
[16] TUL corpo ration. (2021). TUL PYNQ-Z2 Product Specification, https://www.tul.com.tw/productspynq-z2.html.
[17] Zairi, H., Kedir Talha, M., Meddah, K., Ould Slimane, S. (2020). FPGA-based system for artificial neural network arrhythmia classification. Neural Computing and Applications, 32(8): 4105-4120. https://doi.org/10.1007/s00521-019-04081-4
[18] Li, J., Si, Y., Xu, T., Jiang, S. (2018). Deep convolutional neural network based ECG classification system using information fusion and one-hot encoding techniques. Mathematical Problems in Engineering. https://doi.org/10.1155/2018/7354081
[19] Chourasia, M., Thakur, A., Gupta, S., Singh, A. (2020). ECG Heartbeat Classification Using CNN. In 2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Prayagraj, India, pp. 1-6. https://doi.org/10.1109/UPCON50219.2020.9376451
[20] Yıldırım, Ö., Pławiak, P., Tan, R.S., Acharya, U.R. (2018). Arrhythmia detection using deep convolutional neural network with long duration ECG signals. Computers in Biology and Medicine, 102: 411-420. https://doi.org/10.1016/j.compbiomed.2018.09.009
[21] Wu, M., Lu, Y., Yang, W., Wong, S.Y. (2021). A study on arrhythmia via ECG signal classification using the convolutional neural network. Frontiers in Computational Neuroscience, 14: 564015. https://doi.org/10.3389/fncom.2020.564015
[22] Burger, A., Qian, C., Schiele, G., Helms, D. (2020). An embedded CNN implementation for on-device ECG analysis. In 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Austin, TX, USA, pp. 1-6. https://doi.org/10.1109/PerComWorkshops48775.2020.9156260
[23] Cheng, J., Zou, Q., Zhao, Y. (2021). ECG signal classification based on deep CNN and BiLSTM. BMC Medical Informatics and Decision Making, 21(1): 1-12. https://doi.org/10.1186/s12911-021-01736-y
[24] Bouguezzi, S., Fredj, H.B., Belabed, T., Valderrama, C., Faiedh, H., Souani, C. (2021). An efficient FPGA-based convolutional neural network for classification: Ad-MobileNet. Electronics, 10(18): 2272. https://doi.org/10.3390/electronics10182272
[25] Kotareddy, S.K.R.K. (2021). Edge Devices for Detecting Atrial Fibrillation Using Deep Learning. The University of Texas at Dallas.
[26] Karyonov, M. (2023). Accelerating code assembly: Exploiting heterogeneous computing architectures for optimization. Ingénierie des Systèmes d'Information, 28(4): 951-958. https://doi.org/10.18280/isi.280415

NOMENCLATURE

FPGA	Field-programmable gate arrays
ECG	Electrocardiogram
Adam	Adadelta Momentum
GPU	Graphic Processor Unit
CPU	Central Processor Unit