

# End-to-End Multi-Modal Tiny-CNN for Cardiovascular Monitoring on Sensor Patches

Mustafa Fuad Rifet Ibrahim<sup>\*,1,2,3</sup>, Tunc Alkanat<sup>1</sup>, Maurice Meijer<sup>1</sup>, Alexander Schlaefer<sup>2</sup>, Peer Stelldinger<sup>3</sup>

<sup>1</sup>NXP Semiconductors, Eindhoven, The Netherlands

<sup>2</sup>Institute of Medical Technology and Intelligent Systems, Hamburg University of Technology, Hamburg, Germany

<sup>3</sup>Department of Computer Science, Hamburg University of Applied Sciences, Hamburg, Germany

{mustafa.ibrahim, tunc.alkanat, maurice.meijer}@nxp.com

schlaefer@tuhh.de

{mustafa.ibrahim, Peer.Stelldinger}@haw-hamburg.de

**Abstract**—The vast majority of cardiovascular diseases are avoidable or treatable by preventive measures and early detection. To efficiently detect early signs and risk factors, cardiovascular parameters can be monitored continuously with small sensor patches, which improve the comfort of patients. However, processing the sensor data is a challenging task with the demanding needs of robustness, reliability, performance and efficiency. The field of deep learning has tremendous potential to provide a way to analyze cardiovascular sensor data to detect anomalies which alleviates the workload of doctors for more effective data interpretation. In this work, we show the feasibility of applying deep learning for the classification of synchronized electrocardiogram and phonocardiogram recordings under very tight resource constraints. Our model employs an early fusion of data and uses convolutional layers to solve the problem of binary classification of anomalies. Our experiments show that our model matches the accuracy of the current state-of-the-art model on the “training-a” dataset of the Physionet Challenge 2016 database while being more than two orders of magnitude more efficient in memory footprint and compute cost. Further, we demonstrate the applicability of our model on edge devices, such as sensor patches, by estimating processor performance, power consumption, and silicon area.

**Index Terms**—patient monitoring, edge computing, tiny-ml, smart sensing, sensor patches

## I. INTRODUCTION

Cardiovascular diseases are the leading cause of death, globally accounting for around a third of all deaths in 2021 [1]. Earlier detection of cardiovascular diseases or their associated risk factors yields more effective preventive measures and better prognosis [2], [3].

One way to achieve efficient detection of early signs of cardiovascular diseases is patient monitoring. Using different sensor modalities can aid the diagnosis if these modalities complement each other. For this reason, this work uses two data types, namely electrocardiogram (ECG) and phonocardiogram (PCG). Realizing a multi-modal long-term monitoring while minimizing the impact on daily activities of a patient requires a ubiquitous, lightweight and small measurement device. However, manual interpretation of the enormous amount

of data from multiple sensors is not feasible at a large scale when many patients are monitored simultaneously. Thus, this data has to be processed automatically.

The area of Deep learning (DL) provides methods to analyze this data. Many DL methods have been proposed and were shown to yield improved performance in the analysis of ECG and PCG data [4], [5]. However, this increase in performance comes at the expense of increased computational cost. DL methodologies are usually demanding in computational resources, which hinders their adoption in portable medical applications.

Today, the required computational power is available both in the cloud as well as in hub devices like smartphones or tablets. However, transmission and remote processing of sensitive health data gives rise to privacy concerns, additional energy costs, and risk of service disruptions due to, e.g., loss of connection to hub device or cloud. Another option, which this work focuses on, is to run the DL model on the data collection device. This allows it to be used as a standalone solution that requires transmission only in case of a detected cardiovascular anomaly, thus reducing data transmission, providing better privacy and fulfilling the requirement of continuous monitoring while allowing for independence and freedom of the patient. However, this option puts very tight constraints on the memory footprint (chip size), computational cost and energy consumption of a DL model. In this work we show the feasibility of using a DL model in this challenging context. First, we use an efficient convolutional neural network (CNN) building block to build an efficient end-to-end CNN for our cardiovascular monitoring use-case. Then, we show its competitive performance by applying it to a public dataset containing synchronized ECG and PCG recordings, achieving a reduction in parameter count as well as a reduction in Floating Point Operations (FLOPs) of more than two orders of magnitude compared to the state-of-the-art while matching its accuracy. Finally, we estimate the required processor performance, power consumption and silicon area of our model to show the feasibility of hardware application with strict boundary conditions of a typical sensor patch.

\* Corresponding author

The remainder of this paper is organized as follows. In Section II, we briefly review the related work. In Section III, we describe our method in detail. In Section IV, we describe the experimental setup and present results on the quantitative performance of our method. In Section V, we present a detailed analysis on hardware requirements to efficiently run our model on a sensor patch. In Section VI, we discuss the performance and hardware requirements results. Finally, in Section VII, we give concluding remarks.

## II. RELATED WORK

In the literature, various ML-based approaches to the problem of classification of ECG and PCG signals have been proposed. Susič *et al.* [6] first filter the PCG signal to reduce the influence of noise and differences in experimental setup between recordings. Then, after normalization, the PCG is segmented into individual cardiac cycles in order to determine the temporal location of the first and second heart sound, systole and diastole. They manually extract features in time and frequency domains, as well as statistical features and features resulting from wavelet decomposition. The authors use a support vector machine (SVM) to classify the extracted features. Similarly, Low *et al.* [7] segment the heart sound, as well as the synchronized ECG signal for the same time period. Then, the authors detect the location of the QRS complex in the ECG signal and extract the first and second heart sounds from the PCG signal. Hettiarachchi *et al.* [8] make use of continuous wavelet transforms to construct scalograms from the ECG and PCG signals. Then, a CNN [9] is used. This model has two separate branches for either signal type which are then fused in the deeper dense layers. Gjoreski *et al.* [10] use a pre-processing that is similar to [6], however, Gjoreski *et al.* utilize both classic ML and DL models. In this approach, first, the PCG signal is divided into multiple segments. Then, a set of features are manually extracted, and together with the raw signals, the features are given as input to both a random forest model and a CNN model. Finally, the resulting segment-level predictions, manually extracted features and CNN features are all analyzed by another classic ML model, which outputs the final classification scores for the entire recording. This way, Gjoreski *et al.* take into account that different segments might contain different amounts of relevant information for the classification task. Li Pengpai *et al.* [11] also make use of both classic ML and DL. However, in this approach, the DL models, which are composed of both convolution and long-short-term-memory (LSTM) [12] layers, are used to extract features for a support vector machine (SVM) [13] classifier. Here, Li *et al.* make use of the raw ECG signal while decomposing the PCG signal into multiple frequency bands. Li Jinghui *et al.* [14] also utilize both an LSTM and a CNN model to fit better to the sequential nature of the ECG and PCG signals. The authors use raw inputs for the LSTM and use a wavelet transform pre-processing for the CNN. Thomae *et al.* [15] use a very light-weight model without feature extraction. The authors use the raw PCG signal

as the only input for their deep learning model and achieve an average of specificity and sensitivity of 0.55.

As summarized, there exist previous works aiming at solving the problem of anomaly detection with ECG and PCG signals. Similarly, in this study, we are using both sensor modalities with a DL-based model. However, unlike the aforementioned related works, the main focus of our approach is the efficiency on extremely resource-constrained edge devices, such as sensor patches. Our work differs from the previous work by one or more of the following aspects. Our methodology is (1) end-to-end, i.e., does not require the usually costly step of feature extraction, (2) provides state-of-the-art classification performance, (3) is more than two orders of magnitude less demanding in terms of memory footprint and computational cost compared to the current state-of-the-art model.

## III. METHODOLOGY

This section describes our method in detail. Figure 1 illustrates the general system which includes three components: sensing of the ECG and PCG signals; processing of the data to get a classification result; and, finally, communication in case of a detected cardiovascular anomaly. All three main operations outlined here are assumed to be run on a resource-constrained, and possibly battery-operated healthcare sensor patch. This means that, in addition to the demands imposed by the application of the DL model, the other functionalities, such as sensing and communication, also incur computational cost, memory requirements (silicon area) and energy consumption. However, for the scope of this section, we limit ourselves to the analysis of the processing portion of the pipeline, since it usually constitutes the majority of the required resources.

### A. Training dataset

To train and test our model, we use the synchronized ECG and PCG data from the Physionet/Computing in Cardiology Challenge 2016 dataset [16]. This is a heart sound dataset that consists of nine separate databases collected by seven different research teams. These databases differ in many aspects such as the recording devices, recording locations, data quality, and patient types. Among these, we use the Massachusetts Institute of Technology heart sounds database (designated as "training-a" in the data files) which is the only database that contains synchronized ECG and PCG recordings. In this database, there are 409 recordings from 121 subjects, of which 405 contain both ECG and PCG data. There are 117 samples from healthy patients, and 288 samples from patients with various conditions, such as mitral valve prolapse, aortic disease, benign murmurs, or miscellaneous pathological conditions. The diagnosis for each patient was verified through echocardiographic examination at the Massachusetts General Hospital. Each recording in this dataset has a duration of 9 – 37s and was recorded with a Welch Allyn Meditron electronic stethoscope during in-home visits or in the hospital in an uncontrolled environment. The samples in this database are corrupted by various sources of noise and the sampling rate

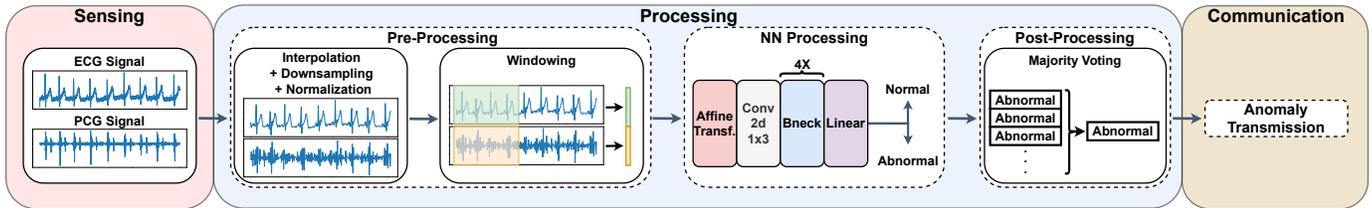


Fig. 1. An overview of the system pipeline. After the ECG and PCG signals are measured the data is interpolated, downsampled and normalized. After that windows are extracted. The windows of ECG and PCG signals are concatenated and fed into our model which classifies them into normal or abnormal. A majority voting over multiple windows gives the final classification. In case of a cardiovascular anomaly the data is transmitted to a doctor.

of the data was reduced from 44,100 Hz to 2000 Hz when the challenge database was assembled by Liu *et al.* [16].

### B. Pre-Processing

Before we input the raw ECG and PCG data to our network, we apply a simple pre-processing (see Figure 1). First, we interpolate any missing values from the ECG and PCG signals and then downsample the signals from 2000 Hz to  $\sim 42.67$  Hz. We then normalize the signals separately to a mean of 0 and a standard deviation of 1. Then, 3 s windows with an overlap duration of 2 s are extracted. Given the sampling rate of  $\sim 42.67$  Hz, this corresponds to having a 1D tensor with a length of 128 samples for each window and sensor. Finally, for both of the sensors, tensors for the same time interval are concatenated in the width dimension, resulting in an input of shape  $1 \times 1 \times 256$  (Channels  $\times$  Height  $\times$  Width), which is then fed to the neural network. This implies that our network does not need any heart beat segmentation of the signals or feature extraction which saves computational resources and energy consumption. We choose a sampling frequency of  $\sim 42.67$  Hz since our model achieves the best results with it. Results with other sampling frequencies are omitted due to space constraints

### C. Network Design

The neural network used in this paper is a CNN and consists of four parts. Table I shows the operations and parameters used in the network architecture. The first part of the network is an affine transform layer that allows for better data fusion by enabling the network to assign a different weight on the ECG and PCG portions of the input. This is required because the two signal types might contain a different amount of relevant information for a sample. Separate, trainable shift and scale parameters are used for ECG and PCG. Before training, we determine the initial values for these parameters with a simple grid search. The second part of our network is a convolutional layer that uses a stride of 2 in order to reduce the signal length further. This serves to reduce the computational cost in the following convolution layers. The third part consists of four bottleneck blocks introduced in the MobileNetV2 paper [17]. The main idea behind a bottleneck block is to maintain low dimensionality in input and output while allowing for higher expressiveness when performing the spatial filtering with the depthwise convolution. We choose this building block because it was shown to lead to a significant reduction in computational

TABLE I  
NETWORK OPERATIONS AND PARAMETERS

Input	Operator	Exp. size	Non-Linear	Stride
1x1x256	Affine Transform	-	-	-
1x1x256	Conv2d, 1x3	-	HardSwish	2
16x1x128	Bottleneck, 1x3	16	ReLU	1
16x1x128	Bottleneck, 1x3	72	ReLU	1
24x1x128	Bottleneck, 1x3	88	ReLU	1
24x1x128	Bottleneck, 1x5	96	HardSwish	1
40x1x128	Avg. Pool	-	-	-
40x1x1	Linear	-	-	-

cost [17]. We choose channel numbers and non-linearities of convolution layers according to the architecture of the MobileNetV3-Small [18]. Compared to the MobileNetV3-Small we use 1D convolutions instead of 2D convolutions and reduce the stride length in the bottleneck blocks to 1 to avoid further downsampling. Following the feature extraction by a series of bottleneck blocks, the fourth and final part of our network classifies the input. For this, a fully-connected layer with the softmax activation is used, and the class scores for the two classes, namely normal and abnormal, are computed.

### D. Post-Processing

In order to combine the class scores for individual windows into a classification result for the entire recording, we use majority voting over all windows belonging to the same recording. In a practical implementation of our system, we envision the majority voting to be done following the classification of every window over  $N$  amount of windows. In other words, every second we classify the current 3 s window and redo the majority voting based on the current window and the last  $N - 1$  windows. Note, that the parameter  $N$  has to be chosen carefully since it has an effect on how sensitive the system is to short-term fluctuations. In case of the detection of a cardiovascular anomaly the data containing the anomaly has to be transmitted to a doctor in order for the doctor to assess the classification of the network. The precise implementation of this has implications for memory footprint and is discussed in Section VI.

## IV. MODEL PERFORMANCE

In this section, we first describe the evaluation metrics and experimental setup and then present the results.

### A. Metrics

The following metrics are used to measure the performance of the model: accuracy, sensitivity, specificity, precision and F1 score [19]. In addition to those metrics, the area under the receiver operating characteristic curve (AUC) is also calculated. In order to measure the memory cost, we measure the parameter count of the model and we estimate the computational cost by calculating the number of FLOPs.

### B. Training

We train the network for 300 epochs on an NVIDIA RTX A6000 using the PyTorch framework [20]. We use the AdamW optimizer [21] to train our network to minimize the cross-entropy loss. We use an initial learning rate of  $10^{-3}$  and a batch size of 32. During training, we use weighted random sampling to counter the class imbalance problem of the training dataset.

### C. Evaluation

To evaluate our model, we split the recordings in the full dataset randomly into a group of 325 recordings for training and a group of 80 for testing. The random splitting is done in a way that ensures a balanced test set of 40 normal and abnormal recordings. The resulting splits contain 7395 abnormal, 2281 normal samples for training, and 1234 abnormal, 1217 normal samples for testing. Due to the fact that each subject contributed several recordings it is likely that recordings from the same patient are present in both the train and test set. We train and test our model five times and report back on the performance averages and standard deviations.

### D. Results

We compare our model with the three previous works with the highest accuracy in literature that used a DL approach trained and evaluated on the same database, as well as made use of both the ECG and PCG data [8], [11], [14]. Results are summarized in Table II. The performance results of the other models are taken from the respective works and the parameter count and FLOPs were calculated by rebuilding their models. The model of Li Jinghui *et al.* [14] includes a slightly modified GoogLeNet architecture and LSTM. However, the parameter count is dominated by the GoogLeNet which is why for ease of calculation we only included the parameter count and FLOPs of this modified GoogLeNet as lower bounds in Table II indicated by the ">". The table shows our approach matches the performance of the state-of-the-art model of Li Jinghui *et al.* [14] while being more than two orders of magnitude more efficient in both parameter count as well as FLOPs.

In order to examine the effect of using both ECG and PCG signals we also compare the performance of our model when using ECG only, PCG only and both signals together (Table III). The table shows our model benefits significantly from using both types of signals. The model using both signal types clearly outperforms the other two models across all performance metrics. The difference is especially large for specificity and precision. We also examine the impact of the

affine transform layer at the beginning of our model. As it can be seen from Table III, the layer has a large impact on all performance metrics. The combination of ECG and PCG without the affine transform layer performs worse than with the affine transform layer. Nevertheless, despite the drop in performance when using only the ECG signal our model performs better than both the approach of Li Pengpai *et al.* [11] and the approach of Hettiarachchi *et al.* [8] that utilize both ECG and PCG signals.

For the analysis of hardware requirements in Section V we quantize our model to 8-bit. We use quantization aware training [22]. Table IV shows the performance of the quantized version is on average 2.75 percentage points lower than that of the unquantized version but still achieves high accuracy, performing better than two of the three baseline methods.

## V. HARDWARE REQUIREMENTS ANALYSIS

In this section we estimate the required processor performance, power consumption and silicon area of our model to show its applicability on hardware typically found on sensor patches. In order to do that we establish upper bounds for those variables to compare our model against. Song *et al.* [23] present a fully-integrated low-power SoC for measuring multiple vital signs in sensor patches, including a prior-art comparison. From these works we extract the following upper bounds for SoC in terms of performance, power consumption and silicon area::

- Processing clock frequency of up to 100 *MHz*;
- Power consumption during processing of up to 3.7 *mW*;
- Total silicon area of up to 18.5  $mm^2$  in 55 *nm* CMOS.

With these bounds, we assess the feasibility of applying our model on sensor patches. As mentioned in Section IV, we train and subsequently quantize our model to 8-bit operation. We make a first-order hardware cost assessment using a neural processing unit (NPU) as the targeted hardware accelerator next to a microprocessor core. We select the ARM Ethos-U55 NPU due to its small size and cost-effective implementation [24]. The following hardware configuration applies:

- 40 *nm* CMOS process node;
- Compute core configuration with 32  $\frac{MACs}{cycle}$ , which is the smallest option for Ethos-U55;
- Memory configuration supporting a total of 64 *KB* SRAM with 64-bit R/W access, which we implement as two instances of 32 *KB*.

### A. Assessment of Processor Performance

We assess the NPU hardware performance while running our model by utilizing ARM's Vela compiler [25]. The obtained results for the ARM Ethos-U55 NPU are as follows:

- Inference total cycle count of 461726 and  $1854792 \frac{MACs}{inference}$
- Compute cycle count of 461726
- Memory access of 54030 cycles

As our model requires 1  $\frac{inf}{sec}$ , the low cycle count of 461726 allows for operating at a NPU clock frequency of 500 *kHz*,

TABLE II  
PHYSIONET 2016 CLASSIFICATION RESULTS COMPARISON WITH PREVIOUS WORKS

Author	Accuracy	Sensitivity	Specificity	Precision	F_1 Score	AUC	Params	FLOPs
Hettiarachchi, Ramith, et al. (2021)	0.9041	0.9474	0.75	-	-	0.9106	~711K	~277M
Li, Pengpai, Yongmei Hu, and Zhi-Ping Liu. (2021)	0.873 ±0.010	0.903 ±0.006	0.845 ±0.018	-	0.874 ±0.010	0.936 ±0.011	~445K	~32.8M
Li, Jinghui, et al. (2022)	0.9613	<b>0.9848</b>	0.908	0.9604	<b>0.9724</b>	-	>5.8M	>1.5G
<b>Ours</b>	<b>0.97</b> ± <b>0.0143</b>	0.975 ±0.0177	<b>0.965</b> ± <b>0.0224</b>	<b>0.966</b> ± <b>0.0208</b>	0.970 ±0.0139	<b>0.986</b> ± <b>0.0053</b>	<b>15.9K</b>	<b>1.86M</b>

TABLE III  
CLASSIFICATION RESULTS OF OUR MODEL WITH DIFFERENT SENSOR MODALITIES

Model	Accuracy	Sensitivity	Specificity	Precision	F_1 Score	AUC
ECG Only	0.92 ±0.0112	0.97 ±0.0209	0.87 ±0.0326	0.883 ±0.0244	0.924 ±0.0096	0.938 ±0.0075
PCG Only	0.6625 ±0.0234	0.835 ±0.0379	0.49 ±0.0335	0.621 ±0.0175	0.712 ±0.0218	0.651 ±0.0300
ECG + PCG (No affine)	0.935 ±0.0056	0.955 ±0.0209	0.915 ±0.0224	0.919 ±0.0189	0.936 ±0.0054	0.951 ±0.0138
<b>ECG + PCG (Affine)</b>	<b>0.97</b> ± <b>0.0143</b>	<b>0.975</b> ± <b>0.0177</b>	<b>0.965</b> ± <b>0.0224</b>	<b>0.966</b> ± <b>0.0208</b>	<b>0.970</b> ± <b>0.0139</b>	<b>0.986</b> ± <b>0.0053</b>

TABLE IV  
CLASSIFICATION RESULTS COMPARISON BETWEEN OUR 8-BIT QUANTIZED AND NON-QUANTIZED 128 SIGNAL LENGTH MODEL

Model	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUC
Not quantized	<b>0.97</b> ± <b>0.0143</b>	0.975 ±0.0177	<b>0.965</b> ± <b>0.0224</b>	<b>0.966</b> ± <b>0.0208</b>	<b>0.970</b> ± <b>0.0139</b>	<b>0.986</b> ± <b>0.0053</b>
<b>Quantized</b>	0.9425 ±0.0068	<b>0.985</b> ± <b>0.0224</b>	0.9 ±0.0306	0.909 ±0.0250	0.945 ±0.0058	0.962 ±0.0142

leading to an inference time of about 0.92 s. Utilizing a higher NPU clock frequency reduces the inference time and allows for duty-cycled processing in combination with a low-power standby mode. For example, a 100 MHz clock can reduce the inference time to about 5 ms and a standby time of about 995 ms every second. The final NPU clock frequency choice depends on the system requirements as well as process technology characteristics. However, this estimation shows that our model is suitable for sensor patch application from an NPU clock frequency viewpoint.

### B. Assessment of Power Consumption

We estimate the NPU power consumption while running our model based on the data published in [26]. The energy consumed by a single multiply-accumulate (MAC) unit is estimated to be about  $0.55 \frac{pJ}{MAC}$  at 0.9 V (ref. 8-bit MULT, 16-bit ACC, 3 registers). For the same operating condition, a 32 KB SRAM with 64-bit access consumes about 20 pJ per access, which is about 36x more energy consuming.

We calculate a first-order estimation of the NPU power consumption as follows:

$$P_{NPU} = (P_{comp} + P_{mem}), \quad (1)$$

where  $P_{NPU}$  is the total NPU active power when running a given DL model,  $P_{comp}$  is the power consumed by the compute portion and  $P_{mem}$  is the power consumed by the SRAM portion. We calculate  $P_{comp}$  as follows:

$$P_{comp} = E_{MAC} \cdot \frac{\#MACs}{cycle} \cdot util \cdot \frac{\#cycles}{inf} \cdot \frac{inf}{sec} + P_{comp\_leak}, \quad (2)$$

where  $E_{MAC}$  is the energy consumed by a single MAC operation,  $\frac{\#MACs}{cycle}$  refers to the size of the MAC array supported by the hardware,  $util$  is the average MAC array utilization,  $\frac{\#cycles}{inf}$  is the total compute cycles per inference,  $\frac{inf}{sec}$  refers to the inferences per second and  $P_{comp\_leak}$  is the leakage power of the compute portion. We calculate  $P_{mem}$  as follows:

$$P_{mem} = E_{SRAM\_RW} \cdot \frac{\#cycles}{inf} \cdot \frac{inf}{sec} + P_{SRAM\_leak}, \quad (3)$$

where  $E_{SRAM\_RW}$  is the energy consumed by the SRAM R/W access,  $\frac{\#cycles}{inf}$  is the total memory access cycles per inference,  $\frac{inf}{sec}$  refers to the inferences per second and  $P_{SRAM\_leak}$  is the leakage power of the 64 KB memory portion. Generally, DC-DC conversion is applied from sensor

TABLE V  
ESTIMATED NPU TOTAL POWER CONSUMPTION AND BATTERY CURRENT

<b>General</b>		
Battery voltage ( $V_{BAT}$ )	<i>Li-ion</i> : 4.2 <i>LiMgO<sub>2</sub></i> : 3.0 <i>Zinc-Air</i> : 1.2	V
Supply voltage ( $V_{DD}$ )	0.9	V
Ambient temperature ( $\eta$ )	25	$^{\circ}C$
DCDC efficiency ( $\eta$ )	0.8	
<b>Estimated NPU power consumption</b>		
Total power	11.9	$\mu W$
$P_{comp}$ ( $P_{comp\_leak}$ )	2.8 (1.8)	$\mu W$
$P_{mem}$ ( $P_{SRAM\_leak}$ )	9.1 (0.5)	$\mu W$
<b>Estimated NPU current from battery</b>		
<i>Li-ion</i> @4.2V $V_{BAT}$	3.6	$\mu A$
<i>LiMgO<sub>2</sub></i> @3.0V $V_{BAT}$	5.0	$\mu A$
<i>Zinc-Air</i> @1.2V $V_{BAT}$	12.4	$\mu A$

patch input battery supply ( $V_{BAT}$ ) down to the NPU supply voltage ( $V_{DD}$ ); we assume a DC-DC efficiency of 80%. For a given battery type, the first-order current from the battery can be estimated as follows:

$$I_{BAT} = \frac{P_{NPU}}{\eta \cdot V_{BAT}}, \quad (4)$$

where  $\eta$  is the DC-DC efficiency and  $V_{BAT}$  is the battery voltage. Table V shows the estimated NPU total power consumption and battery current for typical battery types. It can be observed that NPU power consumption is very low when running our model, and so is the current consumed from the battery regardless of battery type. With such low power consumption, our model acceleration does not face any concerns to be deployed on a sensor patch.

### C. Assessment of Silicon Area

We make a first-order area estimation of the NPU and related memories based on an example design in 40 nm CMOS. The ARM Ethos U55 micro NPU is as small as about 0.6 mm<sup>2</sup> including a row utilization of 60% for place-and-route. The NPU SRAM of 64 KB is about 0.3 mm<sup>2</sup>, assuming two 32 KB SRAM with an instance placement utilization of 90%. The pre-processing block requires an SRAM of 128 Bytes per channel to store the incoming sensor data for a 3 s window which is  $\ll$  0.01 mm<sup>2</sup>. Adding some further digital logic for control and data path, we estimate the NPU total area to be about 0.9 mm<sup>2</sup>.

From this assessment, we have demonstrated that the overhead of adding such an NPU to the SoC is small compared to the upper bound of total silicon area. Thus, we conclude that utilizing such NPU for sensor patches can be considered as a realistic scenario from an area viewpoint.

## VI. DISCUSSION

The comparison with previous DL approaches in table II shows that our model can achieve both high classification performance as well as high efficiency. As mentioned in Section III the main part of our network design resembles a miniaturized MobilenetV3-Small that is modified to fit our specific input data of 1D biosignals as opposed to RGB images. We thus repurpose what has been developed in the field of computer vision to achieve both high performance as well as efficiency. In our analysis we have so far not considered network designs that are traditionally applied to sequential data such as LSTM networks or transformers [27]. While being inherently suitable to sequential data LSTMs suffer from the fact that they are not parallelizable. CNNs are parallelizable which is beneficial for inference time and can thus allow less power consumption for a given processor clock frequency due to increased standby time. Transformers on the other hand are parallelizable and as the main building block of large language models achieve remarkable results in areas such as natural language processing (NLP) [28] but they usually require more parameters which implies higher memory cost and a substantial amount of data in order to surpass the performance of other network architectures. Furthermore, CNNs have already been successfully deployed for time-series data processing [29], [30]. Considering the fact that in our case we are dealing with a small dataset of 405 recordings and our goal is to achieve high efficiency for the application on low power sensor patches we consider efficient CNN architectures already established in the computer vision field.

The experiments with different sensor modalities (Table III) show that our model benefits from both ECG and PCG signals. The PCG recordings in the dataset that we used are corrupted by various different noise sources such as kids playing or dogs barking. PCG recordings are based on sound signals and are thus more impacted by noise factors like these than ECG recordings which are based on electrical signals. Using PCG signals along the ECG signals as input to our DL model does not reduce performance, in fact it leads to an increase in performance. Thus, we show that even a tiny DL model such as ours can utilize different sensor modalities in order to achieve better performance than in the single sensor modality case.

As mentioned in Section III-D in case of a detected cardiovascular anomaly we envision our system to send the current N windows over which the majority voting has led to a positive classification to a doctor for final assessment. Aside from a legal perspective, reliability and trustworthiness also demand the data suspected to contain the cardiovascular anomaly to be sent to a doctor for final assessment. This of course requires further memory and thus silicon area for the past N-1 windows. However, since we downsample the data we end up with only 128 bytes per signal type and window which, as mentioned in Section V-C, is insignificant in terms of required silicon area.

Our analysis of the sensor patch applicability is based on simulations and reasonable assumptions informed by existing

hardware. However, this only allows rough estimations to be made. Since in our case our model is well below the considered upper bounds we believe that our conclusion holds even when assuming a sizeable margin of error due to first-order approximations.

## VII. CONCLUSION

In this work, we show the feasibility of applying a deep learning model on resource-constrained medical edge devices, such as sensor patches. We evaluate our model on the "training-a" dataset of the Physionet/Computing in Cardiology Challenge 2016 database. Our model matches the current state-of-the-art performance while being more than two orders of magnitude more efficient in both parameter count and FLOPs. To analyze the applicability of our model on low power edge hardware we approximate the required processor frequency, energy consumption as well as silicon area based on ARM's Ethos-U55 NPU using their Vela compiler. We show that the requirements imposed by our model are well within reasonable bounds based on existing SoCs for healthcare sensor patches.

## VIII. ACKNOWLEDGMENT

We thank the Bundesministerium für Bildung und Forschung (BMBF) for their support under the project 16KISK221 ("Holistische Entwicklung leistungsfähiger 6G-Vernetzung für verteilte medizintechnische Systeme (6G-Health)").

## REFERENCES

- [1] British Heart Foundation. Global heart & circulatory diseases factsheet. Accessed Jan. 23, 2024. [Online]. Available: <https://www.bhf.org.uk/-/media/files/for-professionals/research/heart-statistics/bhf-cvd-statistics-global-factsheet>
- [2] M. J. Stampfer, F. B. Hu, J. E. Manson, E. B. Rimm, and W. C. Willett, "Primary prevention of coronary heart disease in women through diet and lifestyle," *New England Journal of Medicine*, vol. 343, no. 1, pp. 16–22, 2000.
- [3] S. E. Chiuve, M. L. McCullough, F. M. Sacks, and E. B. Rimm, "Healthy lifestyle factors in the primary prevention of coronary heart disease among men: benefits among users and nonusers of lipid-lowering and antihypertensive medications," *Circulation*, vol. 114, no. 2, pp. 160–167, 2006.
- [4] S. Hong, Y. Zhou, J. Shang, C. Xiao, and J. Sun, "Opportunities and challenges of deep learning methods for electrocardiogram data: A systematic review," *Computers in biology and medicine*, vol. 122, p. 103801, 2020.
- [5] W. Chen, Q. Sun, X. Chen, G. Xie, H. Wu, and C. Xu, "Deep learning methods for heart sounds classification: A systematic review," *Entropy*, vol. 23, no. 6, p. 667, 2021.
- [6] D. Susič, G. Poglajen, and A. Gradišek, "Identification of decompensation episodes in chronic heart failure patients based solely on heart sounds," *Frontiers in Cardiovascular Medicine*, vol. 9, p. 1009821, 2022.
- [7] J. X. Low and K. W. Choo, "Automatic classification of periodic heart sounds using convolutional neural network," *World Academy Science Engineering and Technology, International Journal of Electrical and Computer Engineering*, vol. 5, pp. 100–105, 2018.
- [8] R. Hettiarachchi, U. Haputhanthri, K. Herath, H. Kariyawasam, S. Munasinghe, K. Wickramasinghe, D. Samarasinghe, A. De Silva, and C. U. Edussooriya, "A novel transfer learning-based approach for screening pre-existing heart diseases using synchronized ecg signals and heart sounds," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.
- [9] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in neural information processing systems*, vol. 2, 1989.
- [10] M. Gjoreski, A. Gradišek, B. Budna, M. Gams, and G. Poglajen, "Machine learning and end-to-end deep learning for the detection of chronic heart failure from heart sounds," *Ieee Access*, vol. 8, pp. 20 313–20 324, 2020.
- [11] P. Li, Y. Hu, and Z.-P. Liu, "Prediction of cardiovascular diseases by integrating multi-modal features with machine learning methods," *Biomedical Signal Processing and Control*, vol. 66, p. 102474, 2021.
- [12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [13] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.
- [14] J. Li, L. Ke, Q. Du, X. Ding, and X. Chen, "Research on the classification of ecg and ppg signals based on bilstm-googlenet-ds," *Applied Sciences*, vol. 12, no. 22, p. 11762, 2022.
- [15] C. Thomae and A. Dominik, "Using deep gated rnn with a convolutional front end for end-to-end classification of heart sound," in *2016 Computing in Cardiology Conference (CinC)*. IEEE, 2016, pp. 625–628.
- [16] C. Liu, D. Springer, Q. Li, B. Moody, R. A. Juan, F. J. Chorro, F. Castells, J. M. Roig, I. Silva, A. E. Johnson *et al.*, "An open access database for the evaluation of heart sound algorithms," *Physiological measurement*, vol. 37, no. 12, p. 2181, 2016.
- [17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
- [18] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.
- [19] J. K. Blitzstein and J. Hwang, *Introduction to probability*. Crc Press, 2019.
- [20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.
- [21] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.
- [22] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.
- [23] S. Song, M. Konijnenburg, R. Van Wegberg, J. Xu, H. Ha, W. Sijbers, S. Stanzone, D. Biswas, A. Breeschoten, P. Vis *et al.*, "A 769  $\mu$ w battery-powered single-chip soc with ble for multi-modal vital sign monitoring health patches," *IEEE transactions on biomedical circuits and systems*, vol. 13, no. 6, pp. 1506–1517, 2019.
- [24] ARM. Ethos-u55. Accessed Sep. 20, 2023. [Online]. Available: <https://www.arm.com/products/silicon-ip-cpu/ethos/ethos-u55>
- [25] ———. Arm ethos-u npu application development overview version 5.0. Accessed Sep. 20, 2023. [Online]. Available: <https://developer.arm.com/documentation/101888/0500/NPU-software-overview/NPU-software-tooling/The-Vela-compiler>
- [26] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*. IEEE, 2014, pp. 10–14.
- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [28] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.
- [29] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 1578–1585.
- [30] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.