End-to-end Scalable and Low Power Multi-modal CNN for Respiratory-related Symptoms Detection

Haoran Ren, Arnab Neelim Mazumder, Hasib-Al Rashid,
Vandana Chandrareddy, Aidin Shiri, Nitheesh Kumar Manjunath, Tinoosh Mohsenin
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, USA
Email: {rhaoran1, arnabm1, hrashid1, vandanc1, aidins1, n67, tinoosh}@umbc.edu

***Abstract*—With the onset of the highly contagious COVID-19 pandemic, early-stage and clinic-independent machine assistance is essential for initial disease diagnosis based on its symptoms such as fever, dry cough, fatigue, and dyspnea. This paper proposes a scalable and low power architecture based on end-to-end Convolutional Neural Networks (CNN) for respiratory-related symptoms (cough and dyspnea) detection. The CNN-based model will be part of the final product running on general computing processors that can assess patients similar to what doctors do at triage and telemedicine using passively recorded audio and other information. The proposed model consists of 1D-convolutions to extract audio features and combinations of 2D-convolutions and fully-connected neurons for classification. The architecture achieves a detection accuracy of 87.5% for cough and 87.3% for dyspnea respectively. The proposed work involves extensive optimization of parameters in order to develop a model architecture that can be implemented on highly constrained power budget devices while maintaining high classification accuracy. This optimization allows us to achieve the model size of 960 KB for cough detection which is 193x smaller than the related works employing the end-to-end CNN architecture. The hardware architecture is designed to provide more versatility in terms of the number of input channels, filters, data width and processing engine (P.E.) in a parameterized manner with the target of proposing a reconfigurable hardware. The proposed architecture is fully synthesized and placed-and-routed on Xilinx Artix–7 FPGA. At 47.6 MHz operating frequency, our cough detection hardware architecture consumes 211 mW of power. On the other hand, dyspnea detection hardware architecture consumes 207 mW power at an operating frequency of 50 MHz. In addition, the proposed hardware architecture meets the latency deadline of 1s needed for the efficient operation of hardware while still being energy-effective compared to related work.**

***Index Terms*—Audio and speech processing, time-series, deep learning, low power CNN hardware, FPGA implementation**

I. Introduction

The recent COVID-19 pandemic and other lower-respiratory infectious diseases have created enormous pressure over the health sector due to the sheer number of patients that are contracting the virus every day. Traditionally, when people feel symptoms, they either call a doctor or have themselves checked by medical experts at walk-in clinics, where extensive use of vital signs, visual and auditory information are applied to make diagnostic decisions. Such practice during a pandemic is unsuitable and impractical as a result of limited capacity on existing facilities and human resources, and, ironically, can expedite spreading the infection. Early research [1] shows that machine learning tools on a limited unpublished dataset can distinguish solely between coughs from COVID-19 patients and those who are healthy or with upper-respiratory coughs at a high accuracy of 96.8%. Our ultimate goal is to allow machine learning models running on low power mobile devices to assess patients similar to what doctors do at triage and telemedicine, using passively recorded audio, and later by video and self-declared information.

Two of the most common symptoms that medical professionals have attributed to the lower respiratory infection are coughing and dyspnea. Coughing is one of the most frequently reported symptoms among patients [2], and it is usually the first symptom of almost all respiratory illnesses. Dyspnea or shortness of breath is a cardiopulmonary system disorder caused by a lung and heart disease or any physical load. Both cough and dyspnea detection fall in the domain of audio recognition task for deep learning. Although, Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) Networks have shown impressive performance in image and time-series tasks over the years [3]–[6], it has gained much popularity in audio recognition tasks [7]–[9] as well recently. Researchers in [10], [11] proposed end-to-end models for audio recognition task, which can perform feature extraction in conjunction with classification from raw audio signals. These large models optimize the configuration of the feature extractor automatically as relation weights of the neurons so that they can extract a new discriminatory feature which humans cannot design. However, these models are not well suited for embedded low power implementations due to their large model sizes and computations. As a part towards our goal, this paper introduces an energy-efficient, scalable hardware implementation of a novel end-to-end CNN-based cough and dyspnea detection system. The major contributions of this paper are as follows:

- Proposing a flexible end-to-end CNN-based framework that can take audio recording from individuals and be configured for detecting cough and dyspnea.
- Performing substantial parameter optimization and

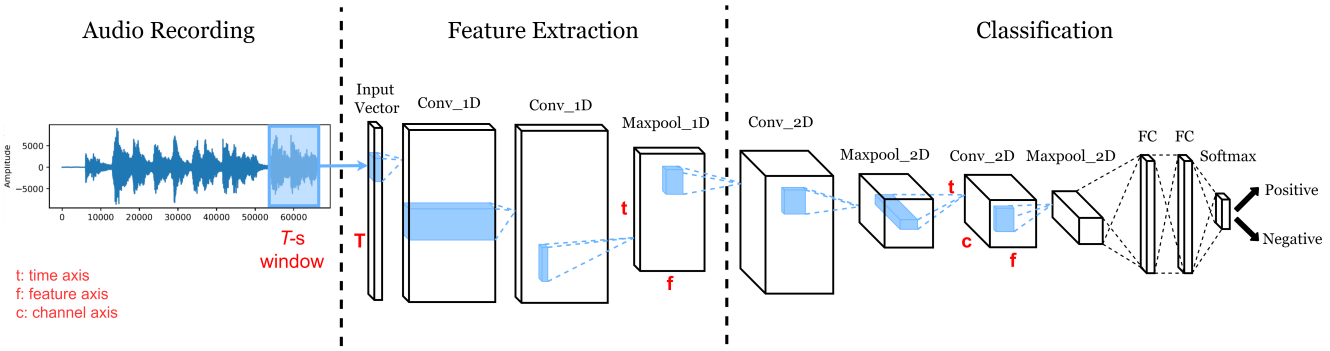


Fig. 1: The proposed flexible EnvNet-like [11] end-to-end CNN architecture that can be used for both cough detection and shortness of breath. Depending on the selected audio window size, the input size and the computation will be varied. The numbers of 1D-convolution layers and fully-connected layers can be altered, as well as their parameters are tuneable.

input audio window size tuning, with the goal of reducing computation complexity and memory requirements for low power hardware implementation while meeting the accuracy requirements.

- Implementing a parameterized and scalable hardware for different numbers of processing engines (P.E.) that replicate the end-to-end CNN architecture for low-power deployment.
- A comprehensive implementation and benchmarking of the proposed work with different numbers of parallel P.E.s and comparisons with state-of-the-art FPGA implementation results.

II. Related Work

There has been a number of software-based cough detection systems suggested in the literature. Previous studies have shown that cough has distinct latent features from different respiratory syndromes [12] which can be detected using machine learning algorithms. The Leicester Cough Monitor (LCM) [13], and VitaloJAK [14] are examples of both software-based devices for recording patient audio, and algorithms for detecting cough from recorded data. The LCM algorithm is however just semi-automated, model parameters need manual tuning for each person recording [15].

There has been a considerable interest recently in applying deep learning to automated detection of cough. The authors in [16], [17] used CNN and in [18], [19] used dense layers to detect the cough sounds for wearable systems. Authors in [20] showed tremendous work on different deep neural networks for low power hardware implementation for speech processing. Authors in [21], [22] used deep learning models to detect shortness of breath from the speech recordings. To our knowledge there is no existing work for low power hardware-based solution for detecting cough and dyspnea.

III. Proposed Architecture

Figure 1 depicts the proposed scalable convolution neural network architecture for both cough and dyspnea detection. The proposed system consists of two main parts: the feature extraction and the classification sections. The input to the model is a vector of samples read from a T seconds window of the audio recordings. The feature extraction section takes the one dimensional vector and extracts the useful features with performing several consecutive one-dimensional convolutions. Afterwards, a non-overlapping max-pooling operation downsamples the feature map. The classification section includes multiple two-dimensional convolutions with a non-overlapping max-pooling downsamples each of the outputs, followed by two or more fully-connected layers where the last one is the output.

TABLE I: One Example of the proposed convolutional neural network architecture in details

Cough detection model trained on ESC-50 dataset [23], with 1 second window as input at 44.1 kHz sample rate. Extracted features are downsampled to 10ms. Model for experiment Set 3 in Table III.

Layer	Description	Output
Input Layer	Audio Window Vector	44100
Conv_1D	Kernels = 40 x [8x1] - BN - ReLU	44100x40
Conv_1D	Kernels = 40 x [8x1] - BN - ReLU	44100x40
Maxpool_1D	Pool size = [441x1] - 10ms	100x40
Conv_2D	Kernels = 32 x [16x6] - BN - ReLU	85x35x32
Maxpool_2D	Pool size = [5,5]	17x7x32
Conv_2D	Kernels = 16 x [4x1] - BN - ReLU	14x7x16
Maxpool_2D	Pool size = [2,1]	7x7x16
Flatten	7x7x16	784
Dense	Neurons = [256] - ReLU - 50% Dropout	256
Dense	Neurons = [128] - ReLU - 50% Dropout	128
Dense	Neurons = [50] - Softmax	50

Specifically, in order to avoid noise, padding is only applied to 1D-convolutions. All the pooling and 2D-convolution operations should be performed without padding the feature map. The pooling filter sizes of the max-pooling operations in classification section should be precisely decided to evenly divide the output shape of the preceding 2D-convolution on both of the two dimensions, so that all the features are accounted during the operation.

In addition, batch normalization and ReLU (Rectified Linear Unit) activation function are applied to the output of each convolution layer. ReLU activation function and 50% dropout are applied to the fully-connected layers, except for the output layer which is activated by a softmax function.

IV. Experimental Setup

A. Case Studies for Evaluation of the Proposed Architecture

1) Cough Detection: To evaluate the performance of the proposed system on cough detection, we use the ESC-50 dataset [23]. The ESC-50 dataset contains 50 classes of environmental sound, where cough is one of the classes. This dataset has a total of 2,000 audio recordings with 40 recordings in each class. Each recording is a 5-second single-channel audio waveform file at a sampling rate of 44.1 kHz. The dataset comes with an equally separated 5-fold cross-validation scheme. We use this default separation so that every model is trained with 1200 recordings, validated by 400 recordings, and tested on the other 400 recordings.

For each audio recording, we load the raw data with the default 44.1 kHz sampling rate and regularize to the range within -1 to 1. Then, we extract all the *T* seconds windows with a stride *S*, and filter out silent windows if the maximum amplitude within a window is smaller than a certain threshold. Meanwhile, we label each extracted window with the same label as the audio recording it belongs to. At last, we perform regularization again on each window individually to the range of -1 to 1 before feeding it into the model.

During training and validation, we input the model with each window and its label. However, during testing, we classify each audio recording based on probability-voting [8] over all the windows generated from it. In other words, we sum up the softmax outputs for all the windows extracted from one test audio recording and predict a label for it with the summed-up output. To evaluate the proposed architecture, We consider both the overall accuracy of the 50-class classification, and the recall score for all the cough-class test data as our metric to classify cough.

2) Dyspnea Detection: To evaluate the performance of the proposed system on dyspnea detection, we used our own collected dataset. The dataset contains 13 normal recordings and 10 short-of-breath recordings. Each recording contains human voice of reading article paragraphs with or without winding. The recordings are captured by different devices and are re-sampled to 44.1 kHz sampling rate. The recording lengths lie in the range between 30 to 60 seconds. We extract windows from the recordings without removing silent windows. The final dataset contains around 3000 windows for our experiments, depending on window size and window stride setting. Also, when splitting into train, validation, and test subsets, we ensure that no window in the test set is overlapped with any window in the train set. Since we are working on a relatively small dataset, we use window level prediction at testing.

B. Training Method

The models are trained for 100 epochs under categorical cross-entropy criterion using stochastic gradient descent (SGD) with a momentum of 0.6. The learning rate is initially assigned as 0.01 for the first 40 epochs, and afterward, it is multiplied by 0.1 for every 15 epochs. The amplitude threshold for silent window removal is 0.2. The models and related methods are implemented with TensorFlow [24]. Audio processing are done with librosa [25].

V. Model Optimization for Low Power Hardware

The goal of this work is to construct an optimized model that is energy efficient and implementable on FPGA in terms of device memory constraints. This leads us to reform our model architecture to decrease model size and number of computations. In this section, we investigate the effects of changing the following parameters *1) Window Size,* and *2) Parameter* tuning in the 1D convolutions and dense layers.

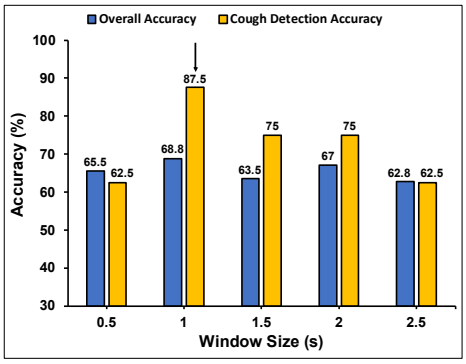
A. Window Size

Selecting a sufficient and balanced window size *T* is vital to extract sound features, especially when distinguishing a single sound and a continuous sound. Since the ESC-50 dataset has a variety of sound classes, we start our experiment for cough detection with a very small window size of 0.5s, and increase it by a step of 0.5s, until 2.5s. For dyspnea detection, the dataset contains only continuous sound, so that we use window sizes between 2s and 8s, with a step of 1s. A window stride $S = 0.25$s used for all of our experiments.

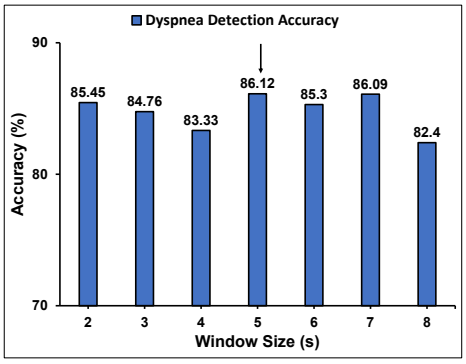
TABLE II: Model Layer Parameters for the Two Case Studies with Respect to Window Size T seconds

Layer	Cough Detection	Dyspnea Detection
Input Layer	$44100 * T$	$44100 * T$
Conv_1D	40 x [8x1]	40 x [8x1]
Conv_1D	40 x [8x1]	40 x [8x1]
Maxpool_1D	[441x1]	[441x1]
Conv_2D	32 x [16x6]	32 x [32x8]
Maxpool_2D	[5,5]	[10,5]
Conv_2D	16 x [4x1]	16 x [8x4]
Maxpool_2D	[$2 * T$,1]	[T,1]
Dense	[256]	[128]
Output	[50]	[2]
Model Size (KB)	1456	842

During evaluating our proposed architecture with different input window sizes in the variable-controlling approach, a mechanism of how to generate the same model for different input shapes is essential. In order to achieve this, as shown in Table II, aside from setting up all the other layers with the same parameters across the experiments, two of the maxpooling layers are specified with respect to window size *T*. First, we define the purpose of the max-pooling operation in the feature extraction section to be making each element on the time axis of the downsampled feature map to represent a certain duration of the input audio window. That is, we define the pooling filter size of this layer to be $sampling\_rate * duration\_size \,/\, 1\ second$. We use 10ms duration for this work. Second, for the last pooling operation before fully-connected layers, we define the pooling filter size on the time axis to be $c * T$, which is directly proportional to *T* by a constant *c*. To be specific, we use $2*T$ for the cough detection, and $1*T$ for the dyspnea detection. In addition, for window size experiments only, we also apply



(a)



(b)

Fig. 2: Detection Accuracy with different window sizes for (a) cough detection and (b) dyspnea detection architecture. For window size experiments only, padding is applied to 2D-convolutions.

padding for the 2D-convolutions to avoid mismatch in output shape.

B. Parameter tuning in the 1D Convolution and dense layers

With the consideration of an upper bound on model size at 1MB, we continue our experiment with different model architectures in terms of the number of filters and layers, to further reduce the model size and investigate the effect on accuracy. We only take into account variations in 1D-convolution and fully-connected layers and the filters associated with them. We keep the configuration for the 2D-convolution layers the same as with the architectures depicted in Table II. The primary reason behind this lies in the fact that 2D-convolution and maxpooling layers included in the model are carefully designed in relation to their receptive fields to extract the features from the input samples conveniently, and any tweaking in these layers will reduce accuracy significantly.

VI. Experimental Results

Figure 2 shows the accuracy results for both case studies with respect to window size. Figure 2 (a) illustrates the overall accuracy and cough detection accuracy from the ESC-50 dataset. The overall accuracy stands for accuracy across the 50 classes available in the ESC-50 dataset and the cough detection accuracy represents the recall score for the cough class. As evident in Figure 2 (a), both the window size of 1s and 2s show good and balanced performances of extracting distinctive features between single sounds and continuous sounds. Consider the fact of even with same model size as defined in Table II, a higher window size will increase the number of computations, thus, a window size of 1s is chosen for our implementation scenario. Similarly, from Figure 2 (b) it is apparent that the window size of 5s and 7s work best for the dyspnea detection model, thus we decide to use 5s window as our input for this case study.

With a specific window selected, Table III and Table IV show the six different sets that we consider in our design. The first four rows in Table III and Table IV represent the number of filters in the respective layers. We compare our work with EnvNet [11], another end-to-end CNN network built for ESC-50 , which proposed a model with 48.6 million parameters at 185.30 MB model size. They reported an average accuracy of 64% with a best accuracy of 66.4%. The results show that our work achieves similar average accuracy and a higher best accuracy, whereas our architecture for cough detection is 193× smaller in size.

TABLE III: Analysis of Different Layer Configurations for Cough Detection Model with 1-second Window Size

Configuration	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
CONV_1D_1	8	**8**	8	16	16	16
CONV_1D_2	8	**8**	8	N/A	N/A	N/A
FC_1	128	**256**	256	128	256	256
FC_2	N/A	**N/A**	128	N/A	N/A	128
Overall Acc (%)	65	**67.3**	62	64.5	63.8	56.5
Cough Acc (%)	75	**87.5**	62.5	75	87.5	62.5
Parameters (K)	125.5	**232.4**	258.9	112.9	219.8	246.3
Model Size (KB)	543	**960**	1066	486	903	1010
Computations (M)	74.9	**75.2**	75.2	74.9	75.2	75.2

TABLE IV: Analysis of Different Layer Configurations for Dyspnea Detection Model with 5-second Window Size

Configuration	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
CONV_1D_1	8	8	8	16	16	**16**
CONV_1D_2	8	8	8	N/A	N/A	**N/A**
FC_1	64	128	128	64	128	**128**
FC_2	N/A	N/A	64	N/A	N/A	**64**
Accuracy (%)	84.6	86.1	83.6	82.3	84.5	**87.3**
Parameters (K)	120.5	202.6	210.7	107.8	189.9	**198**
Model Size (KB)	521	842	877	465	785	**821**
Computations (M)	405.3	405.5	405.5	405.3	405.5	**405.5**

It is obvious from the Figure 3 that set 2 and set 5 provide us the optimum results in terms of model size (Memory), computations, and cough detection accuracy (Cough Acc). However, the overall accuracy (Overall Acc) for set 5 is not satisfactory and this prompts us to use the architecture of set 2 for our hardware implementation. In a similar context from Table IV and Figure 4 it is evident that set 6 is the most suitable architecture for hardware design in relation to the dyspnea model with the highest detection accuracy (Accuracy) for the recordings along with appropriate size (Model Size) and computation setup (Computations).

VII. Hardware Architecture Design

Figure 5 shows the block diagram for the proposed flexible hardware model. The design is configured for

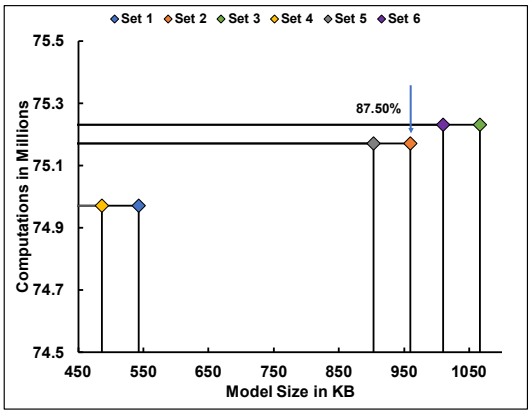


Fig. 3: Trend for computations and memory with different sets for cough architecture.

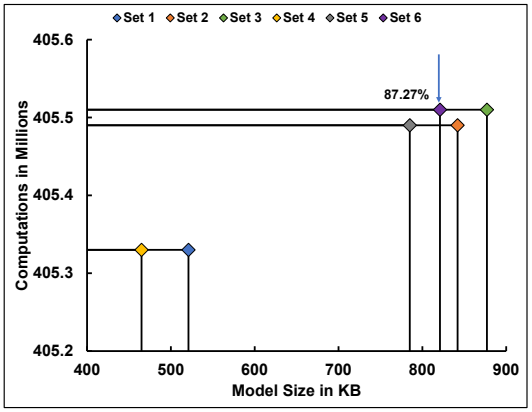


Fig. 4: Trend for computations and model size (KB) with 6 different network configurations for dyspnea detection architecture.

convolution, max-pooling, and fully connected layers using the data and address flow control unit. Based on the network model, convolution or fully connected layer is chosen through the multiplexer and the corresponding addresses is sent to weight memories and feature map memory. The weight memories and feature map memory in turn provide the data for P.E. required for computations. In the P.E., output from the add block is accumulated and tanh activation function is activated for the accumulated output. The whole design is parameterized in the sense that once the layers have been aligned, we can easily change the number of filters and kernel shapes to accommodate different architectures. Along with this, the implementation has provisions for multiple P.E. in terms of parallel processing. For a comprehensive analysis of the hardware performance, we set up our design to have 4 and 8 P.E. configuration for both case studies.

We implemented our design for Xilinx Artix-7 100t FPGA. Both models have a sampling rate of 44.1 kHz and a window size of 1s and 5s respectively for cough and dyspnea detection. With this information, we decided to set a latency deadline of 1s for both model architectures to allow serial operation in a real-time setting. Another important factor that was taken into account during the design was the bit width of the data and the weights. In this work, we considered 32-bit fixed-point operations of the data instead

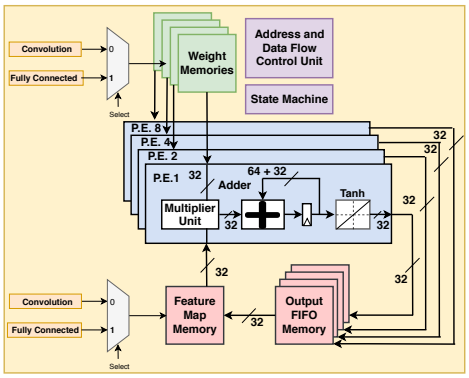


Fig. 5: Proposed parameterized hardware architecture that implements both cough detection as well as dyspnea detection, through a control unit that generate layers, memory addresses and maintains data flow. The hardware can be configured up to 8 P.E. for higher throughput and energy efficiency.

of floating-point operations. This is precisely done because floating-point arithmetic is cumbersome to implement and does not necessarily increase the performance of the Register Transfer Level (RTL) design.

The module takes input from the feature map and weights memory to feed them to the multiplier unit (MU) which in return creates a 64-bit output. This output goes through an adder and the adder truncates the 64-bit data to 32-bit while also considering the ReLU logic for activation. This whole setup acts as the Multiply-Accumulation Unit (MAC) for the design. With a parallel setup, different P.E. work on separate parts of the data and save the output sequentially on to the output FIFO (First In First Out) memory. Layer creation and process flow depend on the address and data flow control unit. It generates addresses depending on the functionality of the layer (i.e. convolution, max-pooling, or fully connected operation) and the state machine regulates the data flow in between layers to ensure efficient operation.

VIII. Hardware Implementation and Results

Table V contains the hardware implementation results for different P.E. configurations. The objective of our design is to meet the latency deadline while consuming low power. For both cough and dyspnea case studies, 8 P.E. configuration produces the best result in terms of power and energy efficiency. To provide a thorough analysis, our implementation is compared to other convolution-based hardware frameworks. Since these are different convolution architectures, our primary point of comparison is based on energy efficiency. The cough model implementation with 8 P.E. configuration meets the 1s latency deadline required and is 7.6× more energy-efficient than [26]. Also, the dyspnea model achieves an energy efficiency of 3.57 with 8 P.E. which is 71.4×, 8.5×, and 2× higher than [26], [27], and [3], respectively.

IX. Conclusion

This paper introduces a low power hardware framework for early detection of cough and dyspnea symptoms. The architecture attains a detection accuracy of 87.5% for cough

TABLE V: Hardware Implementation Results and Comparison with Relevant Architectures

Architecture	Cough Detection Model		Dyspnea Detection Model		[26]	[27]	[3]
Input Dimension	44100x1	**44100x1**	220500x1	**220500x1**	32x32x3	32x32x3	64x40x1
Fixed Point Precision	32 bit	**32 bit**	32 bit	**32 bit**	16 bit	16 bit	16 bit
#P.E Configuration	4	**8**	4	**8**	1	1	8
Frequency (MHz)	41.7	**47.6**	41.7	**50**	180	100	100
Latency (s)	2.2	**1**	1.2	**0.55**	0.028	0.021	0.002
Throughput (labels/s)	0.45	**1**	0.78	**1.82**	35.7	46.7	491
BRAM	84	**84**	84	**84**	N/A	N/A	35
Total Power (mW)	192	**211**	193	**207**	$>$ 17,587	2,944	175
Energy (mJ)	422	**211**	247	**114**	$>$ 500	63	0.35
Performance (GOPS)	0.03	**0.08**	0.32	**0.74**	0.94	1.23	0.3
Efficiency (GOPS/W)	0.18	**0.38**	1.66	**3.57**	$<$ 0.05	0.42	1.71

and 87.27% for dyspnea with a compressed model size of 960 KB and 821 KB respectively. Our optimization process resulted in a $193\times$ compressed model compared to the one in [11]. Along with this, the proposed hardware framework meets the latency deadline of 1s required for efficient hardware operation while also being energy efficient when compared to related works. The future expansion of this work includes implementation on the ASIC and NVIDIA Jetson TX2 GPU platform for a more reformed evaluation of the design and the addition of quantization to lower bit widths for achieving higher hardware performance.

REFERENCES

[1] A. Imran *et al.*, "Ai4covid-19: Ai enabled preliminary diagnosis for covid-19 from cough samples via an app," *arXiv preprint arXiv:2004.01275*, 2020.

[2] P. G. Gibson *et al.*, "Cicada: Cough in children and adults: Diagnosis and assessment. australian cough guidelines summary statement," *Medical Journal of Australia*, vol. 192, no. 5, pp. 265–271, 2010.

[3] A. Jafari *et al.*, "Sensornet: A scalable and low-power deep convolutional neural network for multimodal data classification," *IEEE Transactions on Circuits and Systems I: Regular Papers*, no. 99, pp. 1–14, 2018.

[4] C. Shea, A. Page, and T. Mohsenin, "Scalenet: a scalable low power accelerator for real-time embedded deep neural networks," in *ACM Proceedings of the 28th Edition of the Great Lakes Symposium on VLSI (GLSVLSI)*. ACM, 2018.

[5] H.-A. Rashid, N. K. Manjunath, H. Paneliya, M. Hosseini, and T. Mohsenin, "A low-power lstm processor for multi-channel brain eeg artifact detection," in *2020 21th International Symposium on Quality Electronic Design (ISQED), Accepted*. IEEE, 2020.

[6] A. N. Mazumder, H.-A. Rashid, and T. Mohsenin, "An Energy-Efficient low power LSTM processor for human activity monitoring," in *2020 IEEE 33rd International System-on-Chip Conference (SOCC) (SOCC 2020)*, Sep. 2020, in press.

[7] O. Abdel-Hamid *et al.*, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[8] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.

[9] M.Hosseini *et al.*, "Neural networks for pulmonary disease diagnosis using auditory and demographic information," in *epiDAMIK 2020: 3rd epiDAMIK ACM SIGKDD International Workshop on Epidemiology meets Data Mining and Knowledge Discovery*. ACM, 2020, pp. 1–5, in press.

[10] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[11] Y. Tokozume and T. Harada, "Learning environmental sounds with end-to-end convolutional neural network," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2721–2725.

[12] C. Bales *et al.*, "Can machine learning be used to recognize and diagnose coughs?" *arXiv preprint arXiv:2004.01495*, 2020.

[13] S. Birring *et al.*, "The leicester cough monitor: preliminary validation of an automated cough detection system in chronic cough," *European Respiratory Journal*, vol. 31, no. 5, pp. 1013–1018, 2008.

[14] K. McGuinness, K. Holt, R. Dockry, and J. Smith, "P159 validation of the vitalojak™ 24 hour ambulatory cough monitor," *Thorax*, vol. 67, no. Suppl 2, pp. A131–A131, 2012.

[15] K. McGuinness, A. Morice, A. Woodcock, and J. Smith, "The leicester cough monitor: a semi-automated, semi-validated cough detection system?" *European Respiratory Journal*, vol. 32, no. 2, pp. 529–530, 2008.

[16] J. Amoh and K. Odame, "Deep neural networks for identifying cough sounds," *IEEE transactions on biomedical circuits and systems*, vol. 10, no. 5, pp. 1003–1011, 2016.

[17] J. Amoh and K. Odame, "Deepcough: A deep convolutional neural network in a wearable cough detection system," in *2015 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2015, pp. 1–4.

[18] P. Kadambi *et al.*, "Towards a wearable cough detector based on neural networks," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2161–2165.

[19] S. Khomsay, R. Vanijjirattikhan, and J. Suwatthikul, "Cough detection using pca and deep learning," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, 2019, pp. 101–106.

[20] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[21] S. Boelders, V. S. Nallanthighal, V. Menkovski, and A. Härmä, "Detection of mild dyspnea from pairs of speech recordings," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 4102–4106.

[22] O. Egorow, T. Mrech, N. Weißkirchen, and A. Wendemuth, "Employing bottleneck and convolutional features for speech-based physical load detection on limited data amounts," *Proc. Interspeech 2019*, pp. 1666–1670, 2019.

[23] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. ACM Press, pp. 1015–1018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2733373.2806390

[24] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[25] B. McFee *et al.*, "librosa: Audio and music signal analysis in python," 2015.

[26] G. Hegde, N. Ramasamy, N. Kapre *et al.*, "Caffepresso: an optimized library for deep learning on embedded accelerator-based platforms," in *2016 International Conference on Compliers, Architectures, and Sythesis of Embedded Systems (CASES)*. IEEE, 2016, pp. 1–10.

[27] Y. Wang, J. Xu, Y. Han, H. Li, and X. Li, "Deepburning: Automatic generation of fpga-based learning accelerators for the neural network family," in *Proceedings of the 53rd Annual Design Automation Conference*, ser. DAC '16. New York, NY, USA: Association for Computing Machinery, 2016.