An Efficient Unstructured Sparse Convolutional Neural Network Accelerator for Wearable ECG Classification Device

Jiahao Lu, Dongsheng Liu, *Senior Member, IEEE*, Xuan Cheng, Lai Wei, Ang Hu, and Xuecheng Zou

***Abstract*— Convolution neural network (CNN) with pruning techniques has shown remarkable prospects in electrocardiogram (ECG) classification. However, efficiently deploying the existing pruned neural network to wearable devices for ECG classification is a great challenge due to the limited hardware resource and randomly distributed sparse weights. To address this issue, an efficient unstructured sparse CNN accelerator is proposed in this paper. A tile-first dataflow with compressed data storage format is presented to skip zero weight multiplications and increase the computing efficiency during inference of small-scale model with large sparsity. The two-level weight index matching structure in the dataflow exploits shifting operation to select valid data pairs and maintain the fully-pipelined calculation process. A configurable processing element (PE) array with 32-bit instruction control is proposed to increase the flexibility of the accelerator. Verified in FPGA and post-synthesis simulations in SMIC 40nm process, the proposed sparse CNN accelerator consumes 3.93μ J/classification at 2MHz clock frequency and it achieves an averaged ECG classification accuracy of 98.99%. A computing efficiency of 118.75% is realized which is improved by 48% compared to the dense baseline. In brief, the proposed efficient CNN accelerator is especially suitable for wearable ECG classification device.**

***Index Terms*— Wearable ECG classification, unstructured sparse, convolution neural network, computing efficiency.**

I. Introduction

CONVOLUTIONAL neural networks (CNNs) have been widely used for electrocardiogram (ECG) classification [1]–[4]. The customized neural network with well-trained parameters can achieve a remarkable performance in detecting different kinds of ECG beats for heart disease diagnosis, which even surpasses the manually annotation. In general, constructing a deep network [5], [6] or hybrid network [7], [8] are the two most utilized methods to extract detailed features and increase the diagnostic accuracy. Therefore, many researches are dedicated to design a complicated neural network for pursuing higher classification accuracy.

However, the paroxysmal and occult characteristics of heart disease require wearable devices for long-term continuous monitoring. The limitation of hardware resource makes the exiting large-scale neural networks with high computational complexity hard to deploy on the wearable devices. Many recent works have focused on this challenge and are dedicated to designing a wearable ECG classification device. Some works [9]–[11] used a compact neural network structure to ECG classification. But an extra preprocessing method such as specially designed filter or discrete wavelet transform (DWT) is usually needed for a high classification accuracy, which will improve the complexity of overall system. There are also some other works [12] that utilized a complicated neural network which can take the original ECG beat as input for classification. However, the massive arithmetic operations in complicated neural network structures make the power consumption much larger. In order to reduce the scale of neural network with tolerable accuracy loss, pruning [13] is one of the most extensively applied compression techniques [14]–[16]. However, by turning unimportant data into zero during training, pruning technology leads to significant sparsity of weights, which brings great challenge to parallel hardware implementation.

To make full use of the irregular distribution of weights, many previous works [17]–[21] have already focused on the sparsity of neural networks. Reference [22]–[24] optimizes the pruning techniques, which prevents kernels from completely random distribution and makes it friendly to hardware implementation. Hybrid pruning methods are proposed in these works to make kernels maintain a certain regularity, which can be used to increase the computing efficiency. But it is generally difficult to achieve a high compression rate and high inference accuracy with this kind of method in resource limited wearable devices. And there are many other researches concentrating on sparse hardware implementation for the common pruning method. Reference [25] proposed a sparsity aware CNN accelerator using a candidate pool architecture to select the randomly needed activations according to the nonzero weights. Reference [26] presented a weight-oriented dataflow to handle the irregularity of convolutional layers. And a tile look-up table (TLUT) is designed to accomplish the index matching process, which relieves the runtime decoding overhead. However, the existing optimization methods are mostly aiming for accelerating the large-scale 2-D sparse CNN.

Manuscript received 10 February 2022; revised 4 May 2022; accepted 21 July 2022. Date of publication 16 August 2022; date of current version 26 October 2022. This work was supported in part by the Key Research and Development Project of Hubei Province under Grant YFYB2020000413; in part by the National Natural Science Foundation of China under Grant 61874163, Grant 62104076, and Grant 62134002; in part by the National Key Research and Development Program of China under Grant 2021YFA0715502; in part by the Introduced Innovative Research and Development Team of Dongguan under Grant 201760712600139; and in part by the Laboratory Open Fund of Beijing Smart-Chip Microelectronics Technology Company Ltd. This article was recommended by Associate Editor X. Zhang. *(Corresponding author: Dongsheng Liu.)*

The authors are with the School of Optical and Electronic Information, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: dsliu@mail.hust.edu.cn; ang_hu@hust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2022.3194636.

Digital Object Identifier 10.1109/TCSI.2022.3194636

The calculation in 2-D convolutional layer has six nested loops [27] while 1-D convolutional layer only has four. The sliding operation along width direction does not exist in 1-D CNN. The non-zero data reuse between two adjacent convolutions in width direction dose not needs to be taken into consideration. Directly applying the hardware architecture of 2-D sparse CNN to the ECG classification in wearable devices will introduce a large decrease on hardware resource efficiency and computing efficiency.

In this paper, an efficient and flexible processor aiming at accelerating the unstructured pruning 1-D CNN is proposed for ECG classification in wearable devices. The main contributions of this paper are summarized as follows:

1) A tile-first dataflow with specially designed data compression format is proposed. Processing in tiles with index array directly indicating the position of valid weights can effectively increases the computing efficiency and it is friendly to resource limited wearable devices.
2) A resource efficient two-level weight index matching structure is presented. The shifting operation used to select the valid weight-activation pairs largely reduces the hardware resource consumption.
3) A configurable processing element (PE) array is proposed. With the designed 32-bit instructions, it supports a flexible pipelined processing across multiple pooling types, kernel sizes and number of feature maps.
4) The proposed accelerator is implemented on both Xilinx ZC706 platform and SMIC 40nm process. It achieves an averaged ECG classification accuracy of 98.99% with a 70% sparsity 1-D CNN. And a computing efficiency of 118.75% with 3.93μJ/classification energy efficiency is realized, which is superior to dense baseline and much more suitable for the resource limited wearable devices.

The rest of this paper is organized as follows: Section II introduces the fundamental of 1-D CNN and different kinds of pruning techniques. Section III presents our tile-first dataflow and the data storage format for the sparse weights. In Section IV, the overall hardware architecture of the sparse CNN accelerator is described in detail. Section V shows the experimental results of ECG classification in different target sparsity, followed by the comparison and analysis of hardware implementation. Finally, Section VI concludes this article.

II. Background

A. Convolutional Neural Network

One-dimensional CNN (1-D CNN) is mainly designed to process time domain data such as ECG signals. A typical 1-D CNN structure shown in Fig. 1 is formed by several different layers, including convolution (CONV) layers, pooling layers and fully-connected (FC) layers. It takes the pre-processed ECG signal as input and followed by the feature extraction in CONV layers. The FC layers are usually used for classification.

With the trend of avoiding FC layers, many methods of decreasing the scale of FC layers have occurred such as global average pooling (GAP) and 1×1 convolution operation. This makes the proportion of computation in CONV layers become larger. So accelerating convolution operation while properly

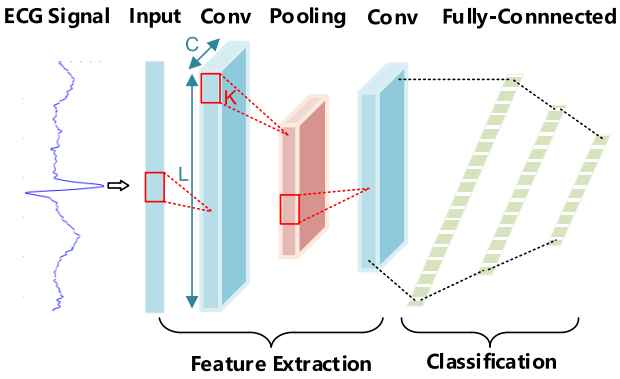


Fig. 1. Typical 1-D CNN structure for ECG classification.

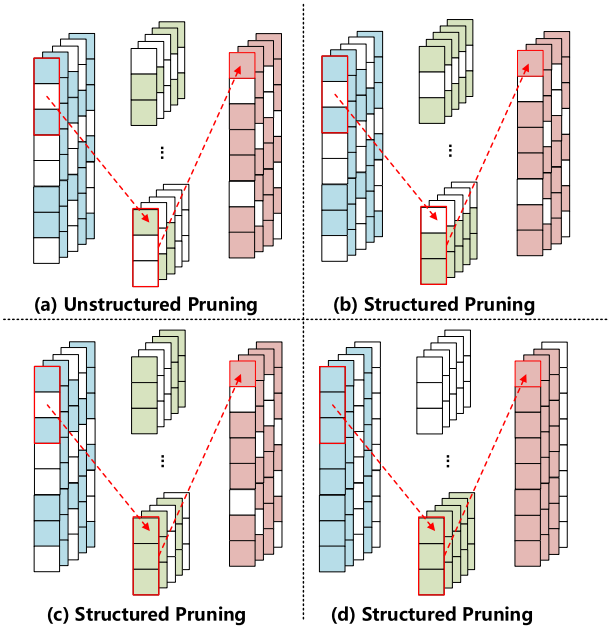


Fig. 2. (a) Fine-grained sparsity. (b) Point-level sparsity. (c) Channel-level sparsity. (d) Filter-lever sparsity.

compatible with matrix multiplication in FC layers is the main target of hardware architecture design. The basic convolution operation is given by the following formula:

$$sum\,[n]\,[x] = \sum_{c=0}^{C} \sum_{k=0}^{K} In\,[x+k]\,[c] \times w\,[n]\,[k]\,[c] \quad (1)$$

where C is the number of input channel and K is the kernel size.

B. Pruning Technique and Sparsity in CNN

Pruning has been proven to be one of the most effective techniques of accelerating the calculation of CNN. It can reduce the computing scale of CNN by dropping some of the connection between neurons according to their contribution to the neural network, which is benefit for resource limited wearable devices.

As shown in Fig. 2, there are four common pruning techniques, including fine-grained, point-level, channel-level and filter-level. The regularity and granularity of the four pruning techniques increase sequentially. The fine-grained sparsity, also named as unstructured pruning, removes all

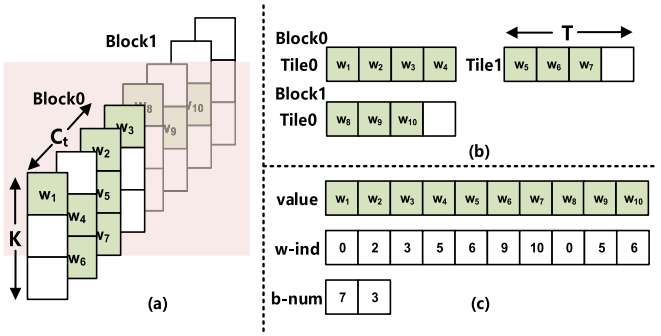


Fig. 3. Tile-first data compression format for CONV layers. (a) weight data in dense format. (b) uncompressed weight tiles. (c) compressed data format.

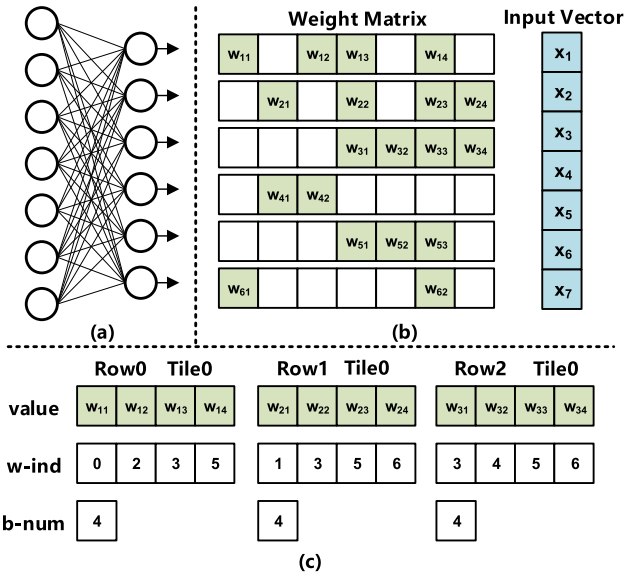


Fig. 4. Tile-first data compression format for fully-connected layers. (a) weight connection in dense format. (b) uncompressed weight matrix. (c) compressed data format.

the unimportant weights whose eigenvalue is less than the manually chosen threshold. The weight eigenvalue is basically pre-defined and magnitude-based such as L1-norm and L2-norm [28], [29]. The other pruning methods can be classified into structured pruning as they group weights which are in the same dimension and decide whether to remove according to the group overall significance [30], [31].

The structured pruning technique can make the sparsity of weight matrix regular, which is friendly to hardware implementation. It is easier to map the pruned neural network into general hardware platform. However, this advantage is at the cost of accuracy loss. The unstructured pruning method can achieve higher model compression rate with negligible inference accuracy loss. The experiment executed in [32] shows that unstructured pruned models (large-sparse) outperforms structured pruned models (small-dense) in accuracy and can achieve up to 10x reduction in number of parameters. The irregularity of computation in unstructured pruning can be solved by designing application-specific hardware architecture. The characteristic of unstructured pruning technique makes it suitable for wearable ECG classification device.

III. Tile-First Dataflow and Compression Format

Skipping zero-operation and power gating are two common methods of sparse computation optimization and they can be applied on both weight (W) and activation (A). Skipping zero-operation is able to save both clock cycles and power consumption with the cost of large hardware resource on searching for valid W-A pair, while power gating needs less resource but can only reduce power consumption. Due to the limited hardware resource in wearable ECG classification device, the proposed tile-first dataflow supports skipping zero-operation on weight and power gating on activation.

A. Data Compression Format

In the tile-first dataflow, kernels are first divided into several blocks along the input channel dimension in CONV layers. Each block of weights is then tiled up and the non-zero weights of each tile are ordered and processed in the following computation. The sparse weight matrices in FC layers are calculated in rows and each row can be viewed as a block of weight kernel for compatibility with CONV layers. Fig. 3 and Fig. 4 show an example of 3×1 CONV layer and FC layer respectively.

In Fig. 3(a), a dense format kernel can be divided into $\left\lceil \frac{C}{C_t} \right\rceil$ weight blocks with C_t channels in each one. And then each weight block is flatten into $\left\lceil \frac{C_t \times K \times (1-\alpha)}{T} \right\rceil$ tiles in channel-first order as shown in Fig. 3(b), where α is the sparsity of the weight block defined as $\frac{\#numofzeros}{\#numoftotal}$ and the tile length T is pre-defined and equal to the number of PE array so that the non-zero weights in a tile can be loaded to PE array at a time. As illustrated in Fig. 3(c), tiles are stored in the compressed data format with all zero values excluded, which consists of a data value array that includes non-zero weights, a weight index (w-ind) array that stores the position index of the corresponding weight in a block and a block number (b-num) array that indicates the total number of non-zero weights in a block.

In order to reuse computing resources, the compression format of FC layers needs to be consistent with CONV layers. Each Row of the weight matrix can be regarded as a flattened weight kernel. So, the non-zero weights in a row can be squeezed into tiles and stored in the same format of value array, w-ind array and b-num array as shown in Fig. 4(c). Weight tiles are provided to PE array in order for parallel processing.

In this work, the bit-width of non-zero weights are set to 16-bit for high precision. The bit-width of w-ind and b-num are the same and only need 5-bit to represent the maximum value of a 1-D CNN. So, the compression rate (CR) of the tile-first data format for kernels in CONV layer and weight matrices (nxm) in FC layer can be calculated by the following formula (2), (3) respectively:

$$CR_1 = \frac{(1-\alpha)\,KC\,(16bit+5bit) + \left\lceil \frac{C}{C_t} \right\rceil \times 5bit}{KC \times 16bit} = \frac{5\left\lceil \frac{C}{C_t} \right\rceil}{16KC} + \frac{21}{16} - \frac{21}{16}\alpha \quad (2)$$

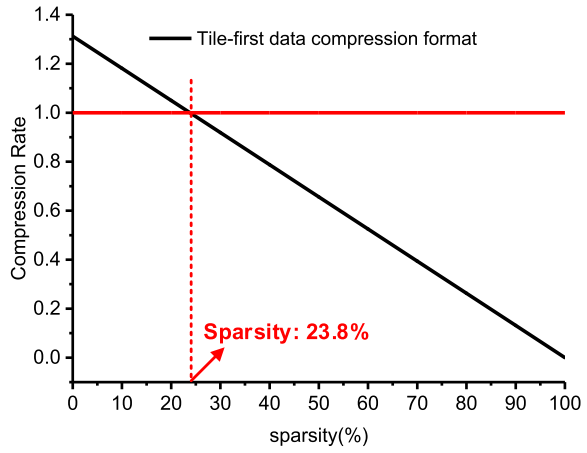


Fig. 5. The compression rate with different sparsity.

$$CR_2 = \frac{(1-\alpha)\,nm\,(16bit+5bit)+n\left\lceil \frac{m}{KC_t}\right\rceil \times 5bit}{nm\times 16bit} = \frac{5\left\lceil \frac{m}{KC_t}\right\rceil}{16m}+\frac{21}{16}-\frac{21}{16}\alpha \tag{3}$$

Since the $\frac{5\left\lceil \frac{C}{C_t}\right\rceil}{16KC}$ and $\frac{5\left\lceil \frac{m}{KC_t}\right\rceil}{16m}$ are both much smaller than $\frac{21}{16}$ when the maximum supported dimension becomes larger, the overall CR can be simplified as formula (4). As shown in Fig. 5, the tile-first data format can achieve an effective compression if the sparsity is larger than 23.8%. And the w-ind array can be directly used for weight-activation pairs matching without extra calculation and hardware resource consumption.

$$CR \approx \frac{21}{16}-\frac{21}{16}\alpha \tag{4}$$

B. Tile-First Dataflow

As the skipping zero-operation is applied on weight optimization, w-ind array of the non-zero weights is the only factor the needs to be taken into consideration in index matching, which reduces the complexity of hardware implementation. The optimization of input activations is directly performed on PE array by dynamically powering off according to the zero activations. This method reduces the power consumption at the cost of little resource overhead, which is benefit for the wearable ECG classification devices.

The tile-first dataflow is composed of three parts as shown in Fig. 6. In the front weight index matching part, the input activations are streamed in channel-first order. And the corresponding activations are selected through two levels of matching module according to the w-ind array. Channel selection module chooses the corresponding channels from the input C_t channels according to the lower bits of w-ind. The higher bits of w-ind are used to point out the valid data position that is the start of sliding operation in position matching module. The PE array in computing module receives the selected input activation and performs the calculation in weight tiles. The non-zero weights in a tile are distributed to different PEs before starting the calculation and they will remain unchanged until finishing computing with the entire length of corresponding input activations. The b-num is used to

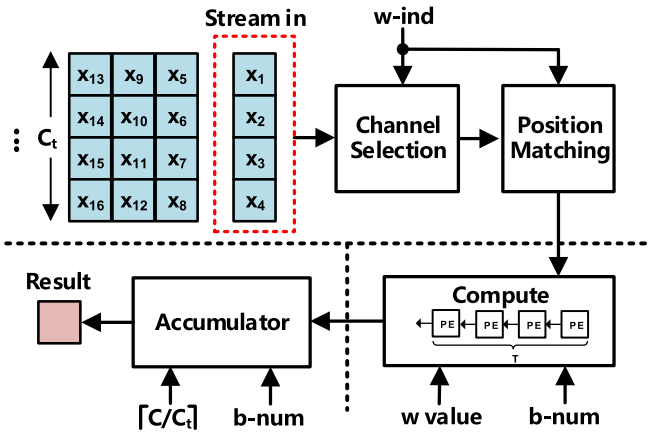


Fig. 6. Tile-first dataflow.

indicate the total number of enabled PEs and it will be reduced by T after each weight tile of computing until be updated when less than T. The actual working PEs are activated from all the enabled PEs according to corresponding zero values of input activations. The partial sum output from PE array is then accumulated in the last part of tile-first dataflow to generate the final results. The accumulator module takes b-num array and the number of weight blocks as input to indicate the accumulation times.

Different from the sliding operation, the matrix multiplication in fully connected layers requires specific operations to be compatible with tile-first dataflow. The input vector is folded by C_t to form the stream in data. And the enabled PEs are activated in sequence instead of working in pipeline.

IV. The Overall System Architecture

The overall system architecture of the CNN accelerator for ECG classification is shown in Fig. 7. The configuration instructions, weights and activations are transferred into corresponding register files and RAMs from external memory before the inference gets started. The FSM in central controller fetches the instruction and provides control and configuration signals for the whole system according to instruction decoder. The corresponding activations are then fetched into PE array for computation after selected from index matching. PE array consists of four 12-stage cascade PE structures, which perform multiplication and addition operations to generates partial sums (psums) of a tile. These psums are then accumulated after adding bias to get the calculation results. Pooling module can be configured to execute four kinds of operations in the calculation results including none, maxpooling, average pooling and global average pooling (GAP). A 16-bit data quantization is applied on the network and it has been proved that 16-bit is sufficient for ECG classification with negligible accuracy loss [35]. The results output from pooling module are cut to specific 16-bit based on the data scale of each layer in the truncator before write back.

A. Configurable PE Array

The configurable PE array is composed of four 12-stage cascade PE structures as shown in Fig. 8. Each PE can accomplish a multiplication and addition operation in one cycle. The psum output from a PE is connected to the next one

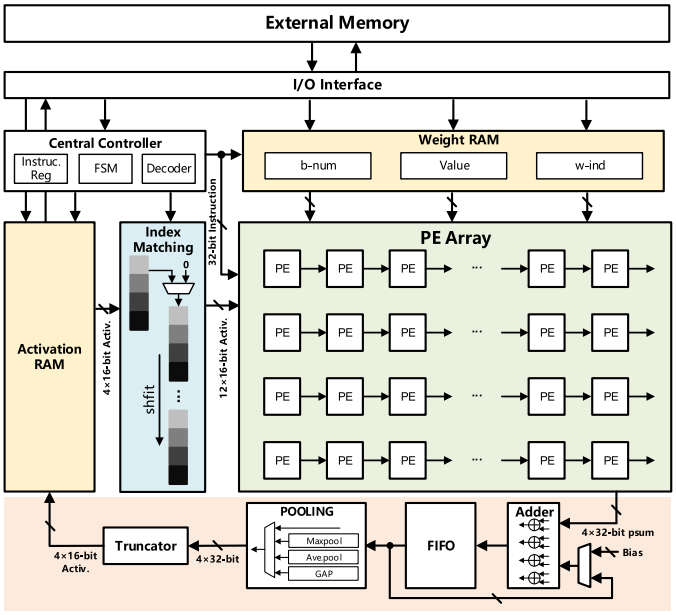


Fig. 7. The overall system architecture of the CNN accelerator.

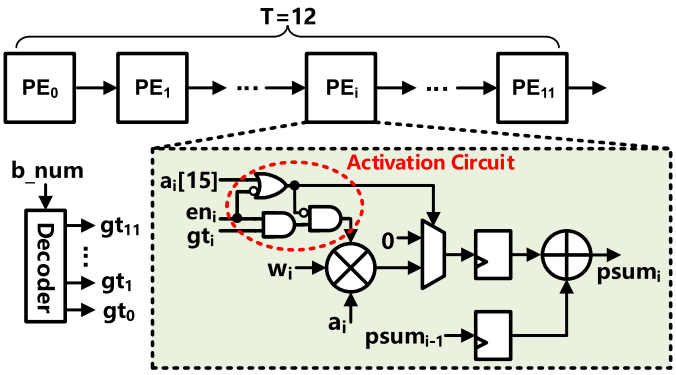


Fig. 8. The 12-stage cascade PE structure.

for addition. The number of cascade stages is equal to the tile length T so that the calculation in a tile can be performed in one cascade PE structure. In this work, four 12-stage cascade PE structures are implemented. They share the same stream in activations and work independently to perform the calculation in different kernels.

The activation circuit contained in each PE is the key innovative part for configuration. It is capable of dynamically activating each PE and determining the working mode of the cascade structure. Three signals including the highest bit of input activation a_i [15], enable signal en_i and gate control signal gt_i are input to the activation circuit to control the execution of multiplication. The gate control signal gt_i is generated from a priority decoder which can transform the b_num into a 12-bit signal. The 12-bit gate control signal will stay unchanged during the calculation of a tile and it is set to all "1" when b_num is large than 12. Weights and activations in computation are represented in 16-bit signed format. The highest bit of input activation a_i [15] which is also the sign bit is used to avoid the redundant zero multiplication randomly caused by ReLU function and bypass the multiplication data path from the previous stage, which makes

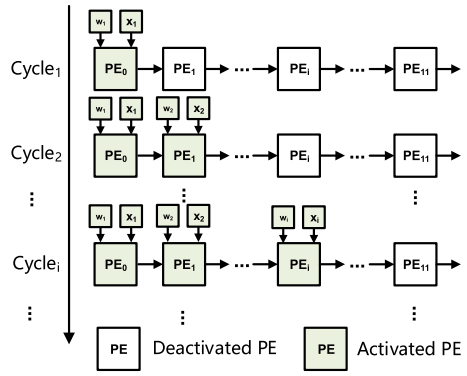


Fig. 9. Pipelined working mode for CONV layers.

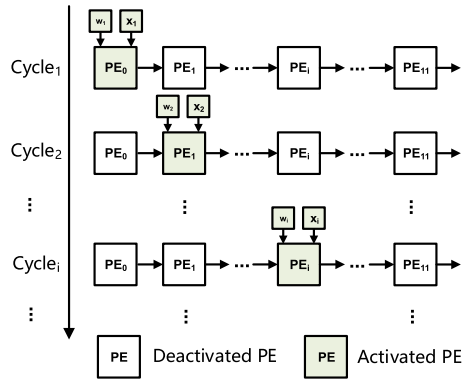


Fig. 10. Serial working mode for FC layers.

the physical implementation of ReLU function unnecessary. The different working modes are controlled by the enable signal en_i to compatible with the calculation in CONV layers and FC layers.

The different working modes of PE array in CONV layers and FC layers are as shown in Fig. 9 and Fig. 10 respectively. Sliding operation in CONV layers requires the cascade PEs to be activated in pipeline and weights are temporally reused during the calculation. Matrix multiplication in FC layers needs them to be activated stage by stage. Only one multiplication and addition inside a PE is executed in each clock cycle. Although only input activations adopt spatial reused strategy during the calculation of FC layers, the low PE utilization of serial working mode does not influence the overall hardware performance since the GAP layer largely reduces the scale in FC layers.

B. Two-Level Weight Index Matching

Due to the sparsity caused by unstructured pruning is only occurred in weights, the main target of weight index matching module is to efficiently handle the irregular weights in hardware by accurately selecting the valid activations to participate in computation according to the w-ind array. With the limited hardware resource in wearable ECG classification, the design of weight index matching module requires to build a non-blocking data path with least resource overhead.

In this work, the proposed weight index matching module is based on shift registers (SRs). As shown in Fig. 11, each

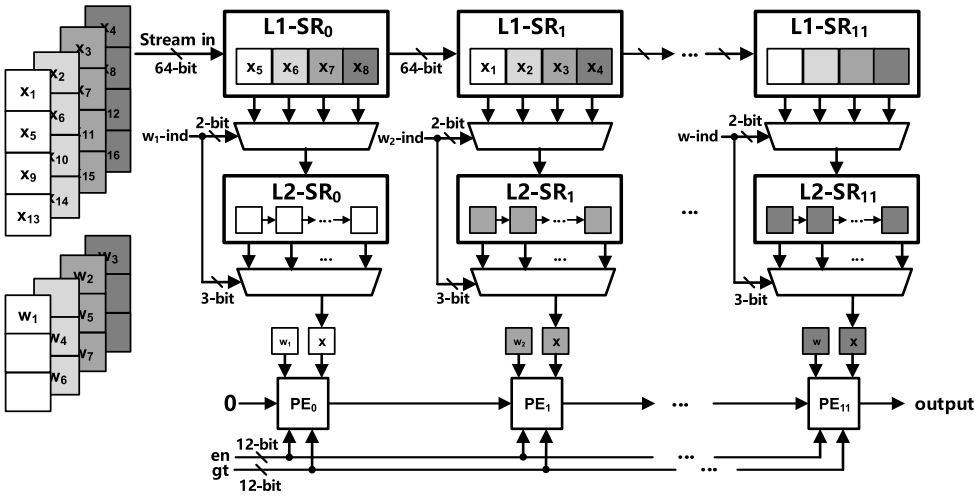


Fig. 11. Two-level weight index matching module.

PE has two-level of SRs for activation selection. The first level of SR (L1-SR) contains 64-bit registers to store the four channels of stream in activations, which will then be input to a MUX controlled by the lower 2-bit of w-ind to choose the corresponding channel. The four-channel data is shifted between L1-SRs for temporal reuse. The second level of SR (L2-SR) consists of 7×16-bit registers, which can cover the input data length corresponding to a maximum 7×1 CONV kernel. The selected channel data from L1-SR is shifted between 16-bit registers inside the L2-SR to imitate the sliding operation of CONV layers. The valid activation for calculation is picked out according to the higher 3-bit of w-ind array when all the 7×16-bit registers are filled with shifted data.

Due to the same compressed data format of CONV layers and FC layers, the proposed shift-based weight index matching module can be reused for the both two kinds of calculation, which is benefit for the wearable ECG classification devices.

C. Storage Policy

In order to improve the flexibility of the proposed accelerator, a well-designed configuration instruction is required. An instruction format that is too fine-grained will become a performance bottleneck due to millions of operations in CNN, while too coarse-grained will result in a loss of flexibility. So, the operation scale of a single instruction needs to be carefully designed. In this work, the operations in one layer of neural networks are packed up in a single instruction. The 1-D CNN is processed layer by layer and the parameters of each layer can be configured by the designed instructions.

The 32-bit instruction format is shown in Fig. 12. The computation of CONV layers and FC layers are distinguished by the highest bit. The followed bits are partitioned to indicate the necessary parameters in the two kinds of layers. FC layers only differ in the number of input and output neurons. As for CONV layers, the instruction supports the 1-D same convolution of input with a maximum length of 1024 and a channel of 128. The kernel size can be set to 1×1, 3×1, 5×1 and 7×1 according to the 2-bit W SIZE part. And the pool type can be configured to none, maxpooling, average pooling and global average pooling (GAP) by the last 2-bit of CONV instruction. The END bit in instruction is to imply

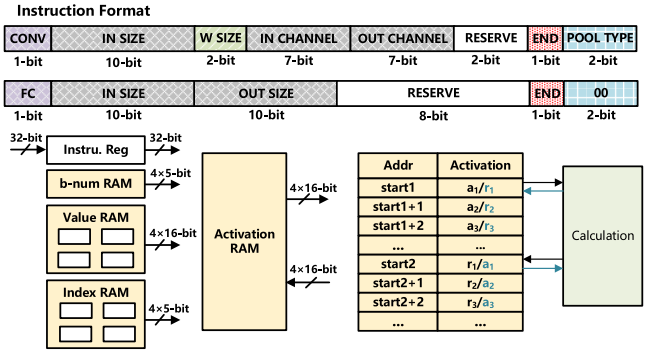


Fig. 12. The storage policy of the CNN accelerator.

the last layer of the network. Multiple 32-bit instructions which can represent a neural network are first stored in instruction register files and they will be read out sequentially for configuration before starting the hardware calculation of each layer.

The three kinds of weight related arrays in compressed format are stored separately. Due to the multiple supported kernel dimensions and C_t channels for block division which is fixed to 4 in this work, b-num and w-ind both need 5-bit data to cover the maximum combination. The four 12-stage cascade PE structures work independently for different kernels so that each one needs a set of RAMs to provide the corresponding b-num, weight value and index. The b_num is first read out for PE activation. The weight value and w_ind are then loaded to PEs stage by stage for calculation.

The activation RAM stores the input and output activations of one layer in sequence. In the activation RAM, four channels of activations can be accessed in one cycle. The storage capacity is divided into two parts with start address predefined. The input activation of a layer is read out from one part and the computation results are written into the other part. The start addresses of read and write operations are exchanged after the calculation of each layer to make the most of storage space.

V. Results and Comparison

In this work, the 1-D CNN model used for ECG classification is described in [35] and the redundant maxpooling before

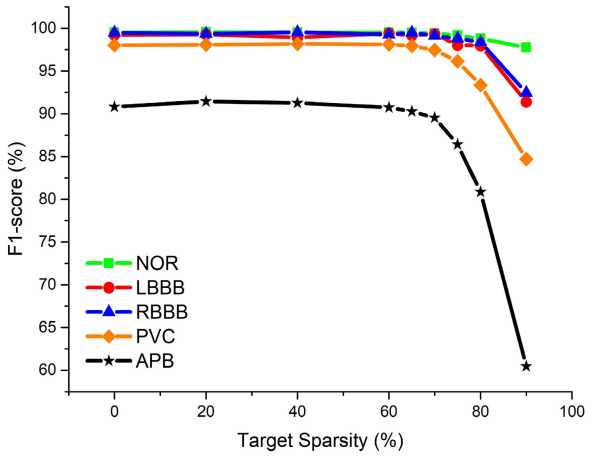


Fig. 13. The F1-score of the five kinds of ECG beats.

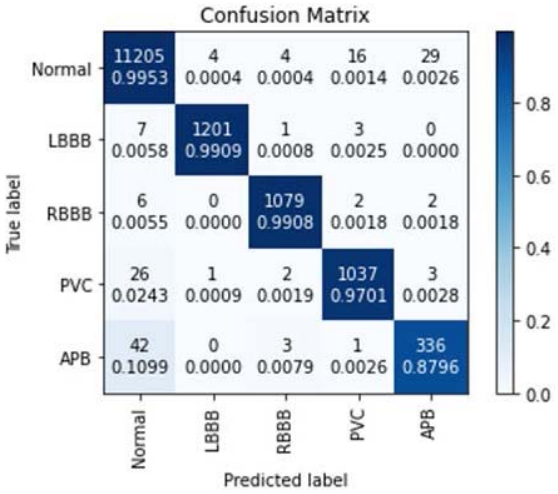


Fig. 14. The normalized confusion matrix of hardware implementation result.

GAP is removed for simplification. The dataset division and model training parameter follow the settings in [35]. Pruning technique makes the specific 1-D CNN more suitable for wearable healthcare device.

A. Pruning Strategy

The implementation of unstructured pruning is based on TensorFlow. The officially provided tensorflow-model-optimization tool is used for pruning in this work. It utilizes a broadly applicable algorithm [32] which can iteratively remove weights according to their magnitude during training. The sparsity of neural network will gradually increase with the execution of the selected pruning schedule until it reaches the manually set final target. In this work, the polynomial decay pruning schedule is applied to make the sparsity smoothly grow.

B. Experimental Results

1) Model Pruning Results: Large target sparsity is benefit for reducing the scale of 1-D CNN in wearable devices. However, it will lead to a decrease of the performance of 1-D CNN model. In order to search for a largest target sparsity which only cause a negligible accuracy loss, the 1-D CNN model is trained with different target sparsity from 0% to 90%. And each trained model is then evaluated by accuracy (ACC), sensitivity (Sen), specificity (Spec) and positive predictive value (Ppv) metrics of five common classes including normal beat (NOR), left bundle branch block beat (LBBB), right bundle branch block beat (RBBB), premature ventricular contraction beat (PVC) and atrial premature beat (APB) in testing set.

The performance of pruned 1-D CNN model with different target sparsity is shown in Table I. The overall accuracy of the five beat classes fluctuates around only 0.2% from 0% to 70% target sparsity. But it declines rapidly during 70% to 90% target sparsity. The F1-score that is the harmonic average of Sen and Ppv is used for comprehensive evaluation. As shown in Fig. 13, 70% target sparsity is a cut-off point. The F1-score changes relatively smoothly before 70% and it starts to have a large decrease between different target sparsity after 70%. The same trend is also reflected in Spec. Therefore, the 70% target sparsity is selected to be the most suitable one for the specific 1-D CNN model in wearable device.

2) FPGA Implementation: The pruned 1-D CNN can be mapped to the configurable PE array with ten instructions to represent the whole network. The trained weights with 70% target sparsity are transformed to the compressed storage format for compatible with the unstructured sparse accelerator. According to the value distribution measured during model performance evaluation in 32-bit float point data format, weights are quantized to 16-bit signed format with 2-bit integer to cover the maximum value. The data range of activations in different layers varies greatly. Therefore, the uniform quantization of activation format across all layers need to be carefully considered. Excessive long integer bit width will result in a decrease of calculation precision, while truncation errors will occur to influence the overall accuracy if the integer bit width is too short. In this work, a 9-bit integer in 16-bit signed format is applied on activations for quantization. And it exhibits a good performance in the following evaluation and comparison. And the computing efficiency [34] used for evaluation in calculated by formula (5). It can be utilized to evaluate the optimization on valid calculations in sparse accelerators.

$$Computing\ Efficiency = \frac{measured\ performance}{theoretical\ peak\ performance} \tag{5}$$

The designed CNN accelerator is implemented on Xilinx Zynq ZC706 FPGA board with a 200MHz clock. The same evaluation performed on 70% target sparsity experiments is executed on the hardware platform. The detailed normalized confusion matrix is shown in Fig. 14. Compared with software inference performance, the hardware implementation only causes a 0.1% loss on overall accuracy as shown in Table III. Most of the other metrics stays the same without significant reduction. The largest decline occurs in Sen of APB which is only 0.52%. The results show that the pruned 1-D CNN achieves a high performance on hardware platform and it is sufficient for the wearable ECG classification device.

TABLE I
THE PERFORMANCE OF THE PRUNED 1-D CNN IN DIFFERENT TARGET SPARSITY

Target Sparsity	ACC (%)	Normal (%)			LBBB (%)			RBBB (%)			PVC (%)			APB (%)		
		Spec	Sen	Ppv	Spec	Sen	Ppv	Spec	Sen	Ppv	Spec	Sen	Ppv	Spec	Sen	Ppv
0%	99.16	97.86	99.72	99.29	99.96	98.93	99.50	99.99	99.17	99.82	99.88	97.57	98.49	99.85	87.96	93.85
20%	99.21	98.18	99.72	99.40	99.95	99.17	99.42	99.99	98.90	99.81	99.88	97.75	98.40	99.84	89.53	93.44
40%	99.17	98.42	99.56	99.48	99.89	99.09	98.77	99.97	99.45	99.63	99.87	98.04	98.31	99.80	90.31	92.25
60%	99.17	97.97	99.70	99.32	99.97	99.17	99.67	99.95	99.17	99.36	99.87	97.94	98.31	99.87	87.17	94.60
65%	99.11	98.31	99.51	99.44	99.96	98.76	99.58	99.96	99.54	99.45	99.86	97.66	98.21	99.72	91.10	89.46
70%	**99.00**	**97.88**	**99.53**	**99.30**	**99.96**	**99.09**	**99.59**	**99.93**	**99.08**	**99.17**	**99.84**	**97.01**	**97.92**	**99.55**	**88.48**	**90.62**
75%	98.53	98.01	99.33	99.02	99.91	98.27	97.78	99.91	98.71	98.90	99.83	95.60	96.69	99.70	83.25	89.83
80%	97.89	95.38	99.22	98.38	99.84	97.69	98.26	99.91	97.89	98.89	99.56	92.52	94.19	99.74	74.61	88.24
90%	95.20	90.17	98.57	96.96	99.10	92.41	90.40	99.29	93.48	91.46	99.16	81.29	88.40	99.68	48.43	80.43

TABLE II
COMPARISON OF HARDWARE PERFORMANCE WITH OTHER WORKS IN FPGA PLATFORM

	[26]	[22]	[24]	[33]	[34]	This Work
CNN Model	VGG16	VGG16	VGG16	VGG16	VGG16	Pruned 1-D CNN
FPGA Platform	Zynq ZCU102	Zynq ZCU102	Zynq ZCU102	Zynq ZCU102	Zynq 7100	Zynq ZC706
Clock (MHz)	200	200	200	200	60	200
DSP Utilization	1144	1352	655	1564	128	48
Resource Utilization (kLUT)	552	390	78.236	203	229	3.238
BRAM Utilization	912	1460	1573	840	386	10.5
Latency/Input (ms)	32.26	21.74	101.326	26.3	2269	45×10^{-3}
Throughput (GOP/s)	309	495.4	231.92	534.2	17.2	22.8
Power (W)	23.6	15.4	24.3	——	——	0.506
Power Efficiency (GOP/s/W)	13.09	32.17	9.544	——	27.4	45.06
Hardware Efficiency (GOP/s/kLUT)	0.56	1.27	2.964	2.632	0.075	7.04
Computing Efficiency	67.53%	91.61%	88.52%	85.4%	114.24%	118.75%

As there are few reports about 1-D CNN accelerator realized in FPGA platform, we compare this work with existing state-of-the-art 2-D sparse CNN processors in which the classic VGG16 model is realized as shown in Table II. All of the listed performance of these works is obtained based on 16-bit precision. The NullHop accelerator in [34] realizes high flexibility and high utilization of the computing resources at the same time. It supports multiple dimensions of kernel size and up to 128 channel calculations. The skipping optimization used in activations makes it achieve a high computing efficiency of 114.24% in VGG16. However, the high flexibility is at cost of hardware resource consumption and the complicated decoding and matching process of the sparse matrix compression scheme based on sparsity map (SM) and nonzero value list (NZVL) further decrease the hardware efficiency. Reference [22] also designs a sparsewise dataflow for skipping the zero weights computation cycles and minimizes power consumption through zero gating technique. But the structured sparse oriented optimization which only includes channel-level and filter-level limits the accelerator to adapt an unstructured network with higher sparsity and higher accuracy. The proposed CNN accelerator in this article outperforms all the other works in terms of hardware and computing efficiency. Benefiting from the shifting based weight index matching and tile-first data compression format, it achieves the highest hardware efficiency of 7.04GOP/s/kLUT and the highest computing efficiency of 118.75%. And the power efficiency of 45.06 GOP/s/W also surpasses other works.

In order to further evaluate the optimization of this work, we execute the experiment proposed in reference [40], in which sparse, dense base and dense equivalent, three cases of model designs are setup for comparison. In this work, the dense base design is implemented by modifying the PE array in reference [35] to $5 \times 3 \times 3$, which makes the total number of computing resource close to the sparse design. Then the DSP mapping is disabled and a $5 \times 4 \times 3$ PE array

TABLE III
COMPARISON BETWEEN SOFTWARE AND HARDWARE PERFORMANCE

	Software (%)				Hardware (%)			
	Acc	Spec	Sen	Ppv	Acc	Spec	Sen	Ppv
NOR	99.00	97.88	99.53	99.30	98.99	97.83	99.53	99.28
LBBB		99.96	99.09	99.59		99.96	99.09	99.59
RBBB		99.93	99.08	99.17		99.93	99.08	99.08
PVC		99.84	97.01	97.92		99.84	97.01	97.92
APB		99.55	88.48	90.62		99.77	87.96	90.81

TABLE IV
EXPERIMENTAL RESULTS WITH THREE SETUPS

	Dense Base	Sparse	Dense Equivalent
DSP Utilization	45	48	60
Resource Utilization* (LUT)	16033	19677	20185
Latency (μs)	84	45	57
Power (W)	0.462	0.506	0.503
Power Efficiency (GOP/s/W)	26.49	45.06	35.86

*Disable DSP mapping for resource utilization evaluation

is implemented to form the dense equivalent design. Table IV shows the comparison results of these three cases. By exploiting the sparse optimization, this work achieves about 1.8× speed up and 1.7× improvement in power efficiency with an extra 22.7% resource overhead compared to the dense base. The dense equivalent design consumes almost the same logic as sparse case in terms of disabling DSP mapping with more computing resources. But the sparse design still surpasses the dense equivalent case in latency and power efficiency. In summary, the overall performance and characteristic of this work is superior to the dense architecture and it is much more suitable for the wearable ECG classification device.

3) ASIC Implementation: The proposed sparse CNN accelerator is also implemented using SMIC 40nm LL RVT process. Fig. 15 shows the corresponding layout with a total area of 2.044 mm^2. The maximum synthesizable clock frequency after placement and routing is 400 MHz. However, high clock frequency will lead to large dynamic power consumption. For low-frequency ECG signals, the operating frequency of 100kHz to 25MHz is sufficient for real-time classification. To explore the optimal operating frequency which costs the lowest energy during each beat classification, the power consumption in different operating frequency is estimated from PrimeTime-PX tool by annotating the backward switching activity information generated from post simulation in Synopsys VCS. The averaged evaluation results of the five types of ECG beats classification in different working frequency are shown in Fig. 16. Dynamic power consumption increases with frequency, while static power consumption is relatively stable at different frequencies. Therefore, static power consumption has a major influence in overall power consumption at low frequencies region. With the frequency increases, dynamic power consumption starts to dominate. Further increase of frequency does not exhibit significant energy efficiency reduction after 2MHz. And the power consumption remains at a low level in 2MHz. Consequently, 2MHz is chosen as the operating frequency.

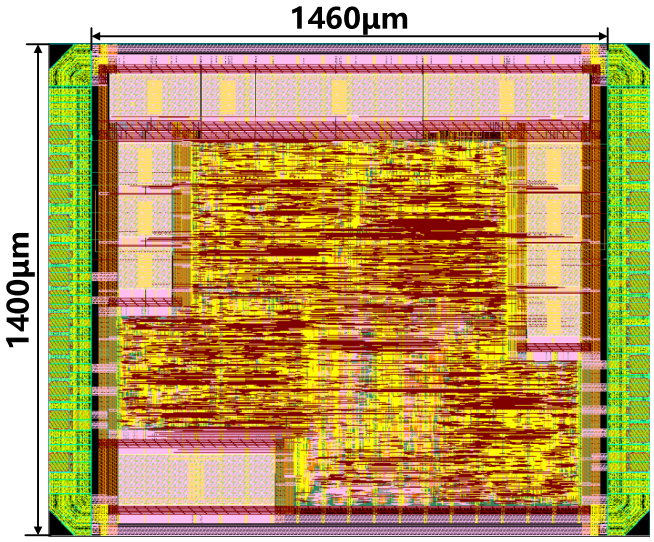


Fig. 15. The layout of the proposed 1-D sparse CNN accelerator.

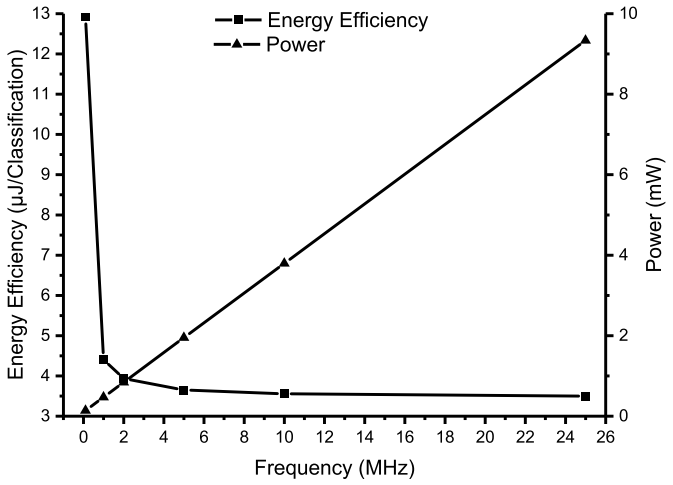


Fig. 16. Energy efficiency and power consumption distribution across different clock frequency.

The performance comparison with other related works in ASIC is shown in Table V. References [36] and [37] are both designed for general purpose. Reference [36] implements a 1-D chain architecture to perform multiple 1-D convolutions. By utilizing the dual-channel PE structure with multiple MUX, a 100% computing efficiency is achieved in this work without any sparsity optimization. Reference [37] exploits the sparsity in both weights and activations. The proposed hierarchical mesh structure effectively improves the computing efficiency and power efficiency. But both of the two works consume too many hardware resources for the high efficiency. As for ECG classification, [10] implements a fixed small ANN with continuous-in-time discrete-in-amplitude (CTDA)preprocessing method and achieves an extremely low energy efficiency of 1.8 μJ/classification.

TABLE V
COMPARISON WITH RELATED WORKS

	[36]	[37]	[38]	[10]	This Work
Technology (nm)	28	65	65	180	40
Supply Voltage (V)	0.9	1	0.75	1.8	1.1
SRAM (KB)	352	246	73	——	48
Frequency (MHz)	700	200	2.5	5	2
PE Number	576	384	25	——	48
Area	3751k gates	2695k gats	1.74mm^2	0.75mm^2	2.044mm^2 384k gates
Configurable NN	Yes	Yes	Yes	No	Yes
Computing Efficiency	100%	265%	——	——	118%
Energy Efficiency (μJ/classification)	——	——	4.36 5.77*	1.8 0.149*	3.93
Power Efficiency (GOP/s/W)	1421 665.9*	962.9 1293.2*	——	——	271
ECG Classification Accuracy	——	——	99.16%	98%	98.99%
Class number	——	——	5	2	5

*Technology scaling to 40nm according to [39]

This kind of work is suitable for specific kinds of ECG beats classification in wearable devices in terms of energy efficiency and area but lack of flexibility. The proposed sparse CNN accelerator has the capability of configuration and it contains 48KB SRAM to cover the maximum support parameter. Under 2MHz clock frequency, it consumes 3.93μJ during the ECG beat classification time of 4.6ms. Compared with the [38], this work achieves a 31% decrease in energy efficiency with a comparable classification accuracy.

VI. CONCLUSION

In this paper, an efficient and flexible unstructured sparse CNN accelerator especially for wearable ECG classification is presented. Aiming at getting benefits from the irregularly distributed weights, a tile-first dataflow with specially designed data compression format is implemented. With this flow, the computing efficiency is effectively improved during the inference of small-scale model suitable for wearable devices. Meanwhile, the shifting based two-level weight index matching structure in the tile-first dataflow is designed to guarantee the fully-pipelined calculation process with minimum extra hardware overhead. Moreover, the proposed configurable PE array can adapt to neural networks with different parameters under the control of the designed 32-bit instructions. The accelerator is implemented based on both FPGA platform and SMIC 40nm process and it achieves an averaged accuracy of 98.99% in the classification of five ECG beats using 70% sparsity well-trained weights. With all the optimization techniques, the experiment results show that the computing efficiency can reach 118.75%, which is increased around 48% over the dense baseline. And it can complete a beat classification in 4.6ms under 2MHz clock frequency with energy efficiency of 3.93μJ/Classification, which is suitable for wearable ECG classification devices.

REFERENCES

[1] A. H. Ribeiro *et al.*, "Automatic diagnosis of the 12-lead ECG using a deep neural network," *Nature Commun.*, vol. 11, no. 1, pp. 1–9, Apr. 2020.

[2] J. Torres-Soto and E. Ashley, "Multi-task deep learning for cardiac rhythm detection in wearable devices," *Digit. Med.*, vol. 3, no. 1, pp. 1–8, 2020.

[3] S. Raghunath *et al.*, "Prediction of mortality from 12-lead electrocardiogram voltage data using a deep neural network," *Nature Med.*, vol. 26, no. 6, pp. 886–891, May 2020.

[4] B. S. Chandra, C. S. Sastry, and S. Jana, "Robust heartbeat detection from multimodal data via CNN-based generalizable information fusion," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 3, pp. 710–717, Mar. 2019.

[5] Y. H. Awni *et al.*, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network," *Nature Med.*, vol. 25, pp. 65–69, Jan. 2019.

[6] H. Chao *et al.*, "Deep learning predicts cardiovascular disease risks from lung cancer screening low dose computed tomography," *Nature Commun.*, vol. 12, no. 1, pp. 1–10, May 2021.

[7] D. Verma and S. Agarwal, "Cardiac arrhythmia detection from single-lead ECG using CNN and LSTM assisted by oversampling," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 14–17.

[8] K. Yamamoto, R. Hiromatsu, and T. Ohtsuki, "ECG signal reconstruction via Doppler sensor by hybrid deep learning model with CNN and LSTM," *IEEE Access*, vol. 8, pp. 130551–130560, 2020.

[9] Y. Zhao, Z. Shang, and Y. Lian, "A 13.34 μW event-driven patient-specific ANN cardiac arrhythmia classifier for wearable ECG sensors," *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 2, pp. 186–197, Apr. 2020.

[10] Q. Cai, X. Xu, Y. Zhao, L. Ying, Y. Li, and Y. Lian, "A 1.3 μW event-driven ANN core for cardiac arrhythmia classification in wearable sensors," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 9, pp. 3123–3127, Sep. 2021.

[11] J. Loh, J. Wen, and T. Gemmeke, "Low-cost DNN hardware accelerator for wearable, high-quality cardiac arrythmia detection," in *Proc. IEEE 31st Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2020, pp. 213–216.

[12] N. Wang, J. Zhou, G. Dai, J. Huang, and Y. Xie, "Energy-efficient intelligent ECG monitoring for wearable devices," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 5, pp. 1112–1121, Oct. 2019.

[13] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," 2015, *arXiv:1506.02626*.

[14] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.

[15] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," 2017, *arXiv:1712.05877*.

[16] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1," 2016, *arXiv:1602.02830*.

[17] S. Zhang *et al.*, "Cambricon-X: An accelerator for sparse neural networks," in *Proc. 49th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2016, pp. 1–12.

[18] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, Jun. 2017, pp. 27–40.

[19] S. Yin *et al.*, "A high energy efficient reconfigurable hybrid neural network processor for deep learning applications," *IEEE J. Solid-State Circuits*, vol. 53, no. 4, pp. 968–982, Apr. 2017.

[20] Z. Yuan *et al.*, "STICKER: An energy-efficient multi-sparsity compatible accelerator for convolutional neural networks in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 55, no. 2, pp. 456–477, Nov. 2019.

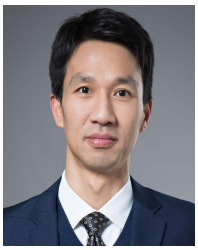
[21] J.-F. Zhang, C.-E. Lee, C. Liu, Y. S. Shao, S. W. Keckler, and Z. Zhang, "SNAP: An efficient sparse neural acceleration processor for unstructured sparse deep neural network inference," *IEEE J. Solid-State Circuits*, vol. 56, no. 2, pp. 636–647, Feb. 2021.

[22] C. Zhu, K. Huang, S. Yang, Z. Zhu, H. Zhang, and H. Shen, "An efficient hardware accelerator for structured sparse convolutional neural networks on FPGAs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 9, pp. 1953–1965, Sep. 2020.

[23] T. Yuan, W. Liu, J. Han, and F. Lombardi, "High performance CNN accelerators based on hardware and algorithm co-optimization," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 1, pp. 250–263, Jan. 2021.

[24] X. Chang, H. Pan, W. Lin, and H. Gao, "A mixed-pruning based framework for embedded convolutional neural network acceleration," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 4, pp. 1706–1715, Apr. 2021.
[25] J. Wen, Y. Ma, and Z. Wang, "An efficient FPGA accelerator optimized for high throughput sparse CNN inference," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Dec. 2020, pp. 165–168.
[26] L. Lu, J. Xie, R. Huang, J. Zhang, W. Lin, and Y. Liang, "An efficient hardware accelerator for sparse convolutional neural networks on FPGAs," in *Proc. IEEE 27th Annu. Int. Symp. Field-Program. Custom Comput. Mach. (FCCM)*, Apr. 2019, pp. 17–25.
[27] X. Yang *et al.*, "A systematic approach to blocking convolutional neural networks," 2016, *arXiv:1606.04209*.
[28] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," 2015, *arXiv:1506.02626*.
[29] A. See, M.-T. Luong, and C. D. Manning, "Compression of neural machine translation models via pruning," 2016, *arXiv:1606.09274*.
[30] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," 2016, *arXiv:1611.06440*.
[31] J. H. Luo, J. Wu, and W. Lin, "ThiNet: A filter level pruning method for deep neural network compression," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 5068–5076.
[32] M. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," 2017, *arXiv:1710.01878*.
[33] M. Qasaimeh, J. Zambreno, and P. H. Jones, "An efficient hardware architecture for sparse convolution using linear feedback shift registers," in *Proc. IEEE 32nd Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2021, pp. 250–257.
[34] A. Aimar *et al.*, "NullHop: A flexible convolutional neural network accelerator based on sparse representations of feature maps," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 644–656, Mar. 2018.
[35] J. Lu *et al.*, "Efficient hardware architecture of convolutional neural network for ECG classification in wearable healthcare device," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 7, pp. 2976–2985, Jul. 2021.
[36] S. Wang, D. Zhou, X. Han, and T. Yoshimura, "Chain-NN: An energy-efficient 1D chain architecture for accelerating deep convolutional neural networks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2017, pp. 1032–1037.
[37] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," *IEEE J. Emerging Sel. Topics Circuits Syst.*, vol. 9, no. 2, pp. 292–308, Jun. 2019.
[38] J. Liu *et al.*, "4.5 BioAIP: A reconfigurable biomedical AI processor with adaptive learning for versatile intelligent health monitoring," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 62–64.
[39] F. Ren and D. Markovic, "18.5 A configurable 12-to-237KS/s 12.8 mW sparse-approximation engine for mobile ExG data aggregation," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2015, pp. 1–3.
[40] A. Moreno, J. Olivito, J. Resano, and H. Mecha, "Analysis of a pipelined architecture for sparse DNNs on embedded systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 9, pp. 1993–2003, Sep. 2020.

Jiahao Lu received the B.S. degree in integrated circuit design and integrated system from the Huazhong University of Science and Technology, Wuhan, China, in 2018, where he is currently pursuing the Ph.D. degree with the School of Optical and Electronic Information. His current research interests include digital integrated circuits and artificial intelligence processor design for wearable healthcare.



Dongsheng Liu (Senior Member, IEEE) received the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, China, in June 2007. In 2013, he was selected into the Wuhan Chenguang Youth Talent Support Program. He is currently a Full Professor with the School of Optical and Electronic Information, Huazhong University of Science and Technology. He has been serving as the team leader for at least ten important projects in last five years, including a sub-project of the National Science and Technology Major Project, the National Natural Science Foundation of China, the National Key Research and Development Program of China, the Key Research and Development Project of Hubei Province, Huawei Hisilicon Cooperation, and other enterprise cooperation projects. He has authored or coauthored more than 50 academic papers and applied for 41 authorized Chinese patents and one authorized American patent. Many of them have been published in the flagship transactions of several IEEE societies, including IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, RFIC, and other international top/high-level journals and conferences. Further, his long-tern dedication on the field of the IoT is well recognized by the industry community. His main research interests include VLSI design, AI processor, RF transceiver, and cryptographic processor.

Xuan Cheng received the B.S. degree in integrated circuit design and integrated system from the Huazhong University of Science and Technology, Wuhan, China, in 2019, where he is currently pursuing the M.S. degree with the School of Optical and Electronic Information. His current research interests include digital integrated circuit and deep learning.

Lai Wei received the B.S. degree in electronic science and technology from the Wuhan University of Technology, Wuhan, China, in 2020. He is currently pursuing the M.E. degree with the School of Optical and Electronic Information, Huazhong University of Science and Technology, Wuhan. His current research interests include digital integrated circuit and artificial intelligence processor design.

Ang Hu received the B.S., M.S., and Ph.D. degrees in integrated circuits engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2014, 2016, and 2020, respectively. He is currently working as a Post-Doctoral Researcher at the School of Optical and Electronic Information, HUST. His current research interests include all-digital frequency synthesizer design and RF SoC design.

Xuecheng Zou received the Ph.D. degree in electronic science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 1993. He is currently a Professor with the School of Optical and Electronic Information, Huazhong University of Science and Technology. His research interests include IC design and the Internet of Things.