A low-power convolutional neural network implemented in 40-nm CMOS technology for bearing fault diagnosis

Yu-Pei Liang, *Member*, *IEEE*, Hui-Hsuan Chang, and Ching-Che Chung, *Senior Member*, *IEEE*
Department of Computer Science and Information Engineering
National Chung Cheng University
No. 168, University Rd., Min-Hsiung, Chia-Yi, Taiwan
Email: ypliang@cs.ccu.edu.tw, wildwolf@cs.ccu.edu.tw

2024 International VLSI Symposium on Technology, Systems and Applications (VLSI TSA) | 979-8-3503-6034-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/VLSITSA60681.2024.10546454

***Abstract*— This paper presents a machine learning-based method for diagnosing bearing faults in electric motors using a Convolutional Neural Network (CNN) that processes motor current signals. This approach, aiming for real-time detection with cost-effective hardware, employs DoReFa-Net for parameter quantization, reducing memory and computational needs, thus enhancing efficiency and affordability. The developed CNN hardware accelerator, implemented in 40-nm CMOS and incorporating power-gating technology, operates efficiently at 100 MHz with just 11.18 mW power consumption, achieving an impressive 95.93% accuracy in bearing health condition assessment. This innovation offers automatic, precise monitoring of motors, significantly cutting staffing costs and preventing failures due to severe faults.**

Keywords—current signal analysis, bearing fault diagnosis, DoReFa-Net, convolutional neural network (CNN), low power.

I. INTRODUCTION

In the industrial era, electric motors and their bearings are crucial for machinery operation, with bearing health being essential for consistent motor performance. Traditional methods like vibration signal analysis require extra sensors like accelerometers, which increases costs. Measuring temperature or acoustic signals also involves additional sensors. However, current signal analysis is more cost-effective as it doesn't need extra sensors and can be integrated into the existing system. With AI's rise, machine learning, especially Convolutional Neural Networks (CNNs), has become prominent in bearing condition analysis, offering enhanced feature learning capabilities. But CNNs, being resource-intensive, demand considerable memory and computational power, leading to high power consumption.

To mitigate this, model compression and acceleration strategies aim to reduce hardware resource usage while preserving accuracy. Commonly employed quantization techniques include Deep Compression [1], Integer-Arithmetic-Only (IAO) [2], Binary Neural Network (BNN) [3], Ternary Weight Networks (TWN) [4], and DoReFa-Net [5]. These methods enable the execution of CNN models on mobile devices or embedded systems with limited hardware capabilities. Therefore, model quantization becomes an essential step in the development of a CNN hardware accelerator, balancing performance with resource constraints.

Conversely, numerous prior studies have explored combining spectrum analysis with neural networks to enhance fault diagnosis accuracy in bearings. In these approaches, time-domain raw data is first transformed into the frequency domain before applying machine learning techniques. For instance, Wang [6] utilizes wavelet transform to convert vibration signals into a multiscale spectrogram image. This transformed data is then processed through a CNN for feature extraction, aiding in identifying the bearing's health status. Another study, mentioned as [7], integrates time-frequency techniques with neural networks for fault diagnosis. However, this method involves complex data preprocessing, rendering it less suitable for real-time recognition applications. Such complexities highlight the ongoing challenge of balancing the need for detailed, accurate analysis with the practical constraints of real-time processing in industrial settings.

To address the challenges previously discussed, this study introduces a novel approach utilizing a CNN to diagnose bearing faults via current signal analysis. This work's key focus is minimizing the hardware implementation costs, particularly in terms of memory usage. To achieve this, the DoReFa-Net quantization method is employed. Furthermore, the proposed hardware accelerator solution is implemented using 40-nm CMOS technology, enhancing its practicality and efficiency. A significant feature of this method is the use of power gating technology. This technology selectively deactivates unnecessary components, thereby substantially reducing the energy consumption of the proposed hardware.

The structure of this paper is methodically organized. Section II details the proposed architecture, while Section III delves into the specifics of the hardware implementation. Experimental results are presented in Section IV. Finally, the paper concludes with Section V, summarizing the study's findings and implications.

II. THE PROPOSED CNN ARCHITECTURE

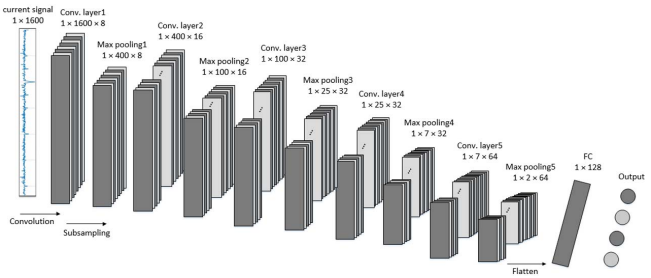

Fig 1. The overview of the proposed 1-D CNN architecture.

The input sensor data of the propsoed one-dimensional (1-D) CNN model consists of 1×1600 raw data points for each image. The model architecture is illustrated in Fig. 1, and is composed of five layers specifically designed for convolution (Conv.). In each of these convolutional layers, there are three main stages: the Convolution stage, where 1×7 kernels are used; the Batch Normalization (BN) stage; and the ReLU stage. Following each of these layers, a max-pooling technique is applied. Lastly, the architecture includes one fully connected (FC) layer, which functions as a classifier in the final layer.

The paper utilizes a bearing dataset from the University of Paderborn's bearing data center to train the proposed model. This dataset includes four types of faults: normal, outer race fault (ORF), inner race fault (IRF), and a compound fault involving both the inner and outer race (OIRF). Labels 0 to 3 are assigned to represent the normal, ORF, IRF, and OIRF

This work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST-111-2221-E-194-049-.

types, respectively. Additionally, the dataset specifies that the sampling rate of the current signal is 64 kHz, and the rotational speed is categorized into two groups: 900 and 1,500 revolutions per minute (rpm).

This study's data preprocessing includes two main steps: down-sampling and fixed-point digitizing. Down-sampling is utilized to lower sampling rates while maintaining the accuracy of the CNN, especially beneficial for datasets with high initial sampling rates. Uniquely in this approach, the total number of data points per image is kept constant, leading to an extended raw data time duration. This extension permits the features of bearing rotation to be represented multiple times within a single image, thereby improving the accuracy of the CNN model. The experimental results indicate that the sampling rate of data points can be effectively reduced by a factor of 10 without causing a significant decrease in accuracy.

The fixed-point format in this study employs a set number of digits for representing numbers, which includes both integer bits and decimal digits. Due to the high computational complexity associated with floating-point calculations, fixed-point arithmetic serves as an approximation of floating-point arithmetic. While fixed-point operations might somewhat reduce the model's accuracy, they also significantly lower computational complexity and enhance computational speed. In the dataset, the integer bits are allocated 4 bits, with a range from -8 to 7. Numerous experiments were conducted to ascertain the optimal number of decimal bits. These experiments revealed that with a down-sampling factor of 10, the test accuracy of the model stabilizes when the number of decimal bits exceeds five.

TABLE I. THE TEST ACCURACY WITH DIFFERENT CHANNELS & LAYERS

Method	Conv. layer (in_channel/out_channel)					FLOPs	**Acc.**
	L1	L2	L3	L4	L5		
1	1/8	8/32	32/64	64/64	-	5.99M	0.98867
2	1/8	8/32	32/64	64/128	-	7.44M	0.99060
3	1/8	8/16	16/32	32/32	32/32	2.11M	0.98708
4	**1/8**	**8/16**	**16/32**	**32/32**	**32/64**	**2.22M**	**0.99010**

The kernel size in the convolution process significantly influences the accuracy of the model. Due to limitations in hardware resources, it is impractical to select a very large kernel size. In this research, the kernel size is set at 7, a decision made to balance between test accuracy and the manageable size of the convolution kernel. Moreover, the number of channels plays a pivotal role in determining the computational effort and the parameters involved. A series of experiments were conducted to establish the optimal number of channels for the architecture. The findings, presented in TABLE I, indicate that an increase in the number of channels is necessary to achieve acceptable accuracy when the number of layers is fixed at 4. Consequently, as detailed in TABLE I, the one-dimensional CNN model developed in this study adopts Method 4.

Once the number of channels and the kernel size for the convolution are established, the parameters of the proposed one-dimensional CNN model can be computed by the software. The batch normalization process involves four parameters. As detailed in this paper, the total count of parameters for the one-dimensional CNN on the software platform amounts to approximately 27,000, as demonstrated in TABLE II. Within this total, the parameters related to convolution constitute 26,000. Consequently, to efficiently minimize the memory demands for storing these model parameters, quantizing the convolution parameters emerges as the most viable approach.

In neural networks, where most parameters consist of weights, it is standard practice to quantize these weights to lessen memory requirements. This paper employs the DoReFa-Net [5] for the quantization process. By converting weights to a lower bit width, it significantly reduces memory usage. The one-dimensional CNN model proposed here, after application of the DoReFa-Net method and subsequent retraining, has its weights transformed from 32-bit floating-point numbers to 5-bit fixed-point numbers. The DoReFa-Net approach not only effectively diminishes the memory needed for the weights but also converts the convolution's floating-point operations into fixed-point operations, thereby reducing computational complexity.

TABLE II. THE NUMBER OF PARAMETERS OF EACH OPERATION.

Operation	**# parameters**	**Sum of parameters**
Convolution 1	1×7×8	56
Batch normalization 1	4×8	32
Convolution 2	8×7×16	896
Batch normalization 2	4×16	64
Convolution 3	16×7×32	3,584
Batch normalization 3	4×32	128
Convolution 4	32×7×32	7,168
Batch normalization 4	4×32	128
Convolution 5	32×7×64	14,336
Batch normalization 5	4×64	256
Total parameters:		27,160

Furthermore, this paper introduces a quantization of the activation using a modified DoReFa-Net approach, as delineated in Eqs. 1 and 2. Eq. 1 is employed as the constrained activation function. A scaling factor of 0.125 is selected to compute this function without requiring an additional multiplier in the hardware. Additionally, a clip function is utilized to maintain the range of input activation between 0 and 1. Given that the bounded activation function effectively scales the activation, it is only necessary to directly quantize the output result of h(x).

$$r = h(x) = clip(x \times 0.125, 0, 1) \quad (1)$$

$$f_a^k(r) = \frac{1}{2^k - 1} round\left((2^k - 1)r\right) \quad (2)$$

Eq. 2 is utilized to carry out the activation quantization, resulting in a k-bit fixed-point number ranging between 0 and 1. A series of experiments were conducted to compare the test accuracy achieved with varying bit widths of activation using the DoReFa-Net method. The experimental outcomes demonstrate that when the activation is quantized to a bit width of 6 bits, the test accuracy reaches a precise convergence.

Implementing Eq. 2 for activation quantization necessitates an additional multiplier in the hardware, which significantly escalates computational demands during model inference. To address this, the activation quantization function employed in this paper is outlined in Eq. 3. This approach allows for direct shifting in hardware calculations. However, the use of Eq. 3 for activation quantization results in the number of activation bits increasing to k+1 bits. Consequently, the final bit-width for activation in this model is 6+1 bits.

$$f_a^k(r) = \frac{1}{2^k} round\left((2^k)r\right) \quad (3)$$

In this paper, due to the inability to retrain the quantization of batch normalization parameters, emphasis is placed on the quantization of weights and activation. Following the quantization of activation, the batch normalization parameters are then quantized. Within the structure of the proposed one-dimensional CNN model, the 608 batch normalization parameters constitute only a minor portion of the overall model parameters. To streamline the number of parameters involved in batch normalization calculations, this paper presents a simplification of the batch normalization parameters from four to two. This simplification is achieved using equations presented as Eqs. 4 and 5, where μ_B represents the mini-batch mean and α_B^2 the mini-batch variance. Subsequently, the batch normalization process is conducted as detailed in Eq. 6, reflecting this simplification.

$$\gamma' = \frac{1}{\sqrt{\alpha_B^2}}\gamma \tag{4}$$

$$\beta' = -(\mu_B\gamma') + \beta \tag{5}$$

$$y_i = \gamma' x_i + \beta' \tag{6}$$

Following the simplification of batch normalization parameters, both γ' and β' are converted from floating-point to fixed-point numbers. A series of experiments have determined the optimal representation for these parameters. The final γ' achieves an accuracy of 0.967 when represented with 5 bits for the integer part (ranging from -16 to 15) and 9 bits for the decimal part. In contrast, β' reaches an accuracy of 0.966 using 6 bits for the integer part (with a range of -32 to 31) and 10 bits for the decimal part. This conversion and representation strategy effectively balances accuracy with computational efficiency.

III. Hardware Implementation

The hardware architecture of the proposed one-dimensional CNN model is depicted in Fig. 2. In this setup, the weights are stored in read-only memory (ROM), accommodating 5 convolutional layers and a single fully-connected layer. To enhance storage efficiency, the convolution weights of the first and second layers share the same ROM (rom_c1c2). A register file (RF) is employed to handle the read/write operations of input data and feature maps. The convolution block, integral to the model, is designed for zero-fill, multiplication, and accumulation (MAC) operations and includes a multiplier. Batch normalization is performed using a multiplier for MAC operations, with parameters being retrieved through a lookup table. The ReLU block is responsible for managing activation and quantization, while the max-pooling block serves to reduce the size of the feature map. The hardware process is completed with a fully connected operation that produces the classification output. This hardware design adeptly balances the need for storage efficiency with the requirements of computational operations in the CNN model.

TABLE III shows the memory requirements for storing the feature maps of each layer. The term feature map here refers to the memory size required post-activation and pooling operations, with each word in the memory comprising 7 bits. The register files, labeled as rf1-1, rf1-2, rf1-3, and rf1-4, are designated for the writing and reading of feature maps from the odd layers of the network. Notably, the feature map of the first level requires the largest amount of memory for storage. Combined, the memory size for rf1-1, rf1-2, rf1-3, and rf1-4 totals 22.4 kb.

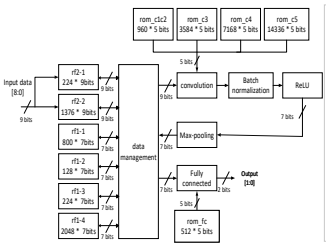

Fig. 2. The proposed hardware architecture.

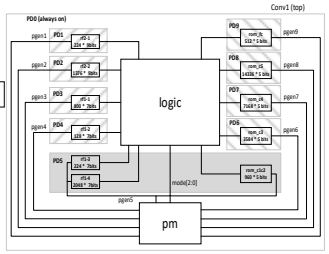

Fig 3. Design with power gating

In contrast, the input data and the feature maps of the even layers are managed by two other register files, rf2-1 and rf2-2. Given that the size of the input data is 1 × 1600 and each word is represented by 9 bits, the total memory requirement for rf2-1 and rf2-2 comes to 14.4 kb. This allocation and organization of memory storage in the CNN model ensure efficient handling of data at various stages of the network's operation.

TABLE III. The memory usage required for the feature maps

Stored data	Feature map size (W×H×C×bit-width)	Total size(bits)	Data management
Feature maps of layer 1	1 × 400 × 8 × 7	22,400	rf1-1, rf1-2, rf1-3 and rf1-4
Feature maps of layer 2	1 × 100 × 16 × 7	11,200	rf2-1, rf2-2
Feature maps of layer 3	1 × 25 × 32 × 7	5,600	rf1-1
Feature maps of layer 4	1 × 7 × 32 × 7	1,568	rf2-1
Feature maps of layer 5	1 × 2 × 64 × 7	896	rf1-2

The proposed hardware architecture incorporates power-gating control in memory components, aiming for minimal power usage. This approach involves decreasing power utilization by deactivating the power supply to idle modules. Within the 40nm CMOS memory compiler, this compiler presents options for selectable power structures, allowing the integration of a power gating mode into the developed memory.

Fig. 3 illustrates the power gating control in the proposed design. Each power domain (PD), from PD0 to PD9, possesses its distinct power switch control signal, ranging from pgen1 to pgen9. PD0 stands as the primary power domain, with a continuous power supply to its modules. The activation of each memory's power switch is governed by the power-gating enable (pgen). Generation of these power gating control signals is the responsibility of the power mode module (PM). Depending on various modes, the power supply for each domain is either activated or deactivated.

IV. Experimental Results

This paper realizes the proposed CNN hardware architecture using TSMC 40-nm CMOS technology, attaining a peak operating frequency of 100 MHz in post-layout simulations. Incorporating power-gating into this CNN hardware design, the power consumption is recorded at 11.18 mW at 100 MHz. A reduction of approximately 18.66% in power usage compared to the same architecture without power-gating control. Additionally, even without employing power gating control, but instead using the chip enable pin (cen) to deactivate the memory chip, the power

consumption of the proposed design can still be lowered to 12.05 mW. Therefore, it is evident that power gating control in memory significantly contributes to reducing overall power consumption. Moreover, the chip area including I/O pads is$1440 \times 1440\ \mu m^2$.

In the hardware implementation phase, as indicated in TABLE IV, the accuracy of the proposed CNN model diminishes due to the quantization of model parameters and the use of fixed-point operations. Specifically, following activation quantization, the CNN model's accuracy drops by approximately 2%. Consequently, there's a 3% to 4% reduction in accuracy transitioning from the software model to the Verilog RTL simulation. This decline in model accuracy, a result of hardware implementation, is an inevitable aspect. Balancing the model's accuracy on the software platform requires a trade-off with hardware resource consumption and prediction loss to optimize outcomes.

TABLE IV. THE TEST ACCURACY WITH DIFFERENT OPERATIONS.

Operation of each stage	Test accuracy
Convert input data to the 9 bits fixed-point	0.99010
Weight quantization to 5 bits by DoReFa-Net	0.98165
Activation quantization to 7 bits	0.96710
Convert γ' in BN to the 14 bits fixed-point	0.96661
Convert β' in BN to the 16 bits fixed-point	0.96575
Verilog Register Transfer Level (RTL)	0.95934

The paper concludes with a discussion on practical application considerations. As the electric motor's current signal is continuously sampled, additional memory becomes necessary to store these sampled signals during the inference phase of the CNN model. This arrangement enables real-time detection without data point loss. It is also crucial to consider not only the time taken by the CNN hardware design for inference operations, but also the input data sample rate of the current signal. The process from inputting the current signal into the CNN hardware design to producing the identification result demands a total of 1,086,540 clock cycles. Moreover, the sampling period for the current signal, which is the interval between consecutive data points, is 156,250 nanoseconds. Thus, additional 69 data points need to be stored in the memory at 100 MHz.

TABLE V. COMPARISON WITH OTHER SOFTWARE METHODS.

	[8]	[9]	[10]	[11]	our work
Architecture	2D CNN	1D CNN	LSTM	1D CNN	1D CNN
Type of damage	Real damage	Artificial damage	Real damage	Mixed damage	Mixed damage
Type of signal	Current/ Vibration	Current	Current	Current	Current/ Vibration
Data preprocessing	Gray image	N/A	Wavelet packet	N/A	Downsampling & fixed-point
Input data size	80×80	1×1200	N/A	1×1800	1×1600
#parameters	25,810	80,600	N/A	40,448	27,160
labels	3	4	3	3	4
Accuracy	98.3%(Current) / 99.47% (Vibration)	99.36 %	96.4%	97.78%	99.01% (Current) / 99.52% (Vibration)

Table V offers a comparative analysis of prior studies utilizing the same dataset. The method proposed in this research is benchmarked against other models on software platforms prior to quantization. In reference [8], both current and vibration signals were employed to determine the health condition of the bearing. It was observed that classification accuracy is higher with vibration signals. However, acquiring vibration signals necessitates extra sensors, leading to increased costs. Although the accuracy of the current work is marginally lower than that in [9], the number of parameters in the proposed model is significantly fewer than in [9], making it more suitable for hardware implementation. In references [10] and [11], which also use the current signal as the input, the accuracy is found to be lower than that of the proposed model.

V. CONCLUSION

This paper introduces a one-dimensional CNN model tailored for bearing fault diagnosis. It optimizes the hardware architecture by deepening layers and minimizing output channels. The approach conserves hardware resources by applying weight and activation quantization through DoReFa-Net. Furthermore, it simplifies batch normalization parameters, converting them into fixed-point numbers. The hardware design's power efficiency is enhanced through power gating control. When implemented in 40nm CMOS technology, this design runs at a 100 MHz operating frequency, consumes 11.18 mW of power, and demonstrates a 95.93% classification accuracy in post-layout gate-level simulation.

REFERENCES

[1] Song Han, et al., "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," *in Proceedings of International Conference on Learning Representations (ICLR)*, Feb. 2016.

[2] Benoit Jacob, et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," *in Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.

[3] Matthieu Courbariaux, et al., "Binarized neural networks: training neural networks with weights and activations constrained to +1 or −1, arXiv:1602.02830 [cs.LG]," *arXiv.org*, Feb. 2016.

[4] Fengfu Li, Bo Zhang, and Bin Liu, "Ternary weight networks," in Proceedings of 30th Conference on Neural Information Processing Systems (NIPS), Nov. 2016.

[5] Shuchang Zhou, et al., "DoReFa-Net: training low bitwidth convolutional neural networks with low bitwidth gradients, arXiv:1606.06160 [cs.NE]," *arXiv.org*, Jul. 2016.

[6] Jinjiang Wang, et al., "A multiscale convolution neural network for featureless fault diagnosis," *in Proceedings of International Symposium on Flexible Automation*, Aug. 2016, pp. 65-70.

[7] Long Wen, Liang Gao, Xinyu Li, Lihui Wang, and Jichu Zhu, "A jointed signal analysis and convolutional neural network method for fault diagnosis," *in Proceedings of CIRP Conference on Manufacturing Systems*, May 2018, pp. 1084-1087.

[8] Duy Tang Hoang and Hee Jun Kang, "A motor current signal-based bearing fault diagnosis using deep learning and information fusion," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 6, pp. 3325-3333, Jun. 2020.

[9] Xun Dong, et al., "Deep cost adaptive convolutional network: a classification method for imbalanced mechanical data," IEEE Access, vol. 8, pp. 71486-71496, Apr. 2020.

[10] Russell Sabir, Daniele Rosato, Sven Hartmann, and Clemens Gühmann, "LSTM based bearing fault diagnosis of electrical machines using motor current signal," *in Proceedings of 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, Dec. 2019, pp. 613–618.

[11] Guangyu Jiang, Zhixiang Xu, and Shouyan Guan, "An intelligent bearing fault diagnosis method with transfer learning from artificial damage to real damage," *in Proceedings of 2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, Dec. 2019, pp. 464-469.