ZEBRA: A Zero-Bit Robust-Accumulation Compute-In-Memory Approach for Neural Network Acceleration Utilizing Different Bitwise Patterns

Yiming Chen[1], Guodong Yin[1], Hongtao Zhong[1], Mingyen Lee[1], Huazhong Yang[1], Sumitha George[2], Vijaykrishnan Narayanan[3], and Xueqing Li[1]
BNRist, EE, Tsinghua University[1], North Dakota State University[2], Pennsylvania State University[3]
Email: xueqingli@tsinghua.edu.cn

***Abstract*—Deploying a lightweight quantized model in compute-in-memory (CIM) might result in significant accuracy degradation due to reduced signal-noise rate (SNR). To address this issue, this paper presents ZEBRA, a zero-bit robust-accumulation CIM approach, which utilizes bitwise zero patterns to compress computation with ultra-high resilience against noise due to circuit non-idealities, etc. First, ZEBRA provides a cross-level design that successfully exploits value-adaptive zero-bit patterns to improve the performance in robust 8-bit quantization dramatically. Second, ZEBRA presents a multi-level local computing unit circuit design to implement the bitwise sparsity pattern, which boosts the area/energy efficiency by 2x-4x compared with existing CIM works. Experiments demonstrate that ZEBRA can achieve <1.0% accuracy loss in CIFAR10/100 with typical noise, while conventional CIM works suffer from > 10% accuracy loss. Such robustness leads to much more stable accuracy for high-parallelism inference on large models in practice.**

***Keywords*—Neural Network, Compute-in-Memory, Robustness Computing**

I. Introduction

In recent years, artificial intelligence (AI) enabled by deep neural networks (DNN) has made significant breakthroughs in various fields, such as computer vision (CV), natural language processing (NLP), and automatic control. It has been proved that large models have strong generalization and robustness [1]. Unfortunately, it also results in the memory wall issue [2].

To overcome the challenges of neural network inference, compute-in-memory (CIM) is proposed as a promising technique that reduces data movement by performing binary-format MAC with the support of in-cell or in-array computing units after clamping and uniformly quantizing trained floating-point weights. CIM has been explored in various memory techniques, including SRAM [3], [4], eDRAM [5], [6], RRAM [7], FeFET [8], and even ROM [9]. Among them, CMOS-based SRAM-CIM and eDRAM-CIM are highlighted for their mature fabrication, high reliability, and high energy efficiency, which effectively mitigates the data movement for many data-intensive applications.

Nevertheless, despite the macro-level energy efficiency advantages of SRAM-CIM, it remains severe challenges to support large models due to the constraints of limited capacity in the edge devices [9]. The advantages of system-level energy efficiency will be diminished as the model size increases. Therefore, to address this issue, several techniques across different levels have been developed to reduce the parameter number in the neural network.

First, a possible approach on the architecture level is to adopt lightweight versions of large models such as MobileNet [10], and ShuffleNet [11]. However, these lightweight model structures have lower redundancy. It could be challenging on analog-based CIM deployment due to low model robustness.

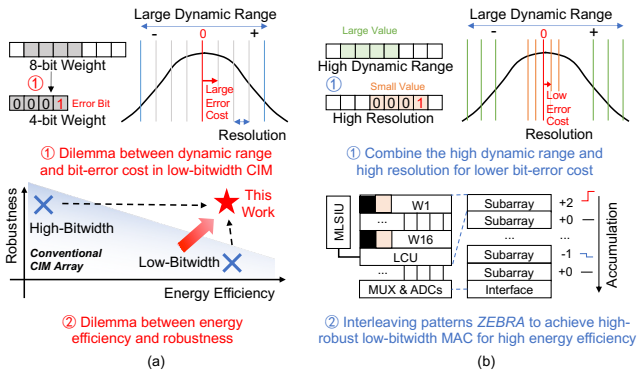

Fig. 1. Challenges and opportunities in (a) conventional CIM and (b) proposed ZEBRA architecture.

Second, another approach is employing layer-wise compression techniques, such as sparse pruning [12]. Sparse pruning involves skipping unnecessary computations for certain layers, channels, and feature maps in a large model. Unstructured sparsity methods [13] perform no accuracy loss, but may not be well-suited for regular CIM arrays. Structured layer-wise and channel-wise sparsity methods [12], [14], on the other hand, could be more feasible for energy-efficient CIM computations with reduced workloads of NNs.

Third, a parameter-level approach is weight quantization. Recently, a promising solution has emerged in the form of a vector-wise quantization [15], which allows for efficient 4-bit quantization over a wide range and delivers significant performance improvement on GPUs. Despite the advantages, due to the irregular quantization scales in matrix-vector multiplication (MVM), it is difficult to be implemented on the CIM macro and PE arrays.

This work provides new insights into low-bit quantization and bitwise structured sparsity in CIM. As shown in Fig. 1(a), low-bit quantization can be modeled as a form of bitwise sparsity based on 8-bit quantization. However, there is a dilemma between the quantization dynamic range and resolution. It is noticed that low-bit quantization also suffers from low redundancy, which results in accuracy degradation under the noise due to hardware non-idealities or malicious injection during MVM computation. Therefore, we ask this inspiring question: *is it possible to introduce a robust zero-bit compression method on CIM to deploy a high-efficient lightweight model?*

This question is answered affirmatively in this paper by the proposed ZEBRA shown in Fig. 1(b) with robust bitwise sparsity utilization and corresponding hardware-software co-optimization. ZEBRA has several highlights:

First, by introducing value-adaptive zero-bit patterns, we prove the key idea that effectively combines the high dynamic range and high resolution to perform a high-robustness 8-bit quantization. This makes it possible to achieve dramatic

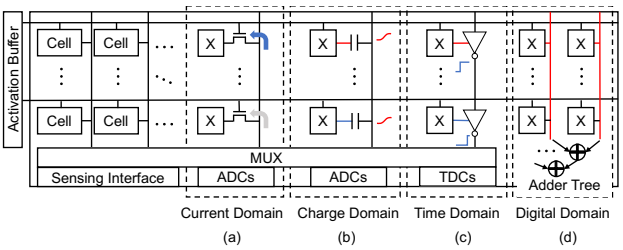

Fig. 2. Different CIM implementation methods in (a) current domain, (b) charge domain, (c) time domain, and (d) digital domain.

improvement by combining the high area/energy efficiency benefits of 4-bit quantization with the high robustness of 8-bit quantization. Notably, both uniform and non-uniform quantization could be utilized for the proposed value-adaptive zero-bit patterns.

Second, we demonstrate a hardware-software co-optimized computing implementation of ZEBRA that achieves the aforementioned versatile robust CIM. This work supports multi-bit signed input-weight multiplication with the proposed multi-level local computing unit (ML-LCU) and the multi-level signed input unit (ML-SIU). With the support of ML-LCU and ML-SIU, ZEBRA shows 2x-4x energy efficiency improvement and 3-3.5x noise tolerance compared with prior CIM works.

The key contributions of this work ZEBRA include:

- **Encoding method**: a robust low-bitwidth data encoding method enabled by value-adaptive zero-bi patterns for the analog CIM. It overcomes the dilemma between the low efficiency of 8-bit quantization and the robust issue of 4-bit quantization.
- **Circuits implementation**: a local computing unit supports multi-level input and weight multiplication. Besides, a signed weight mode by signed binary encoding is adopted to enhance the bitwise sparsity.
- **Experiments and evaluation**: rich results across application, macro, and system levels, showing performance and robustness improvement brought from value-adaptive zero-bit patterns method and multi-level local computing unit.

II. Background

A. Compute-In-Memory (CIM)

Compute-in-memory is an emerging technique raised recently to alleviate the memory access bottleneck between memory cells and computing units.

According to the location of the computing units, CIM could be divided into in-cell [3] and in-array [16] designs. In-cell designs use standard memory cells [17] or add additional transistors to implement bit multiplications. This approach can greatly increase the upper limit of CIM computing parallelism.

According to the principle of accumulation, the analog CIM could be divided into current-domain [3], charge-domain [18], and time-domain [19] designs, as shown in Fig. 2(a)(b)(c), respectively. The current-domain methodology accumulates the output of each computing cell by near-linear leakage current [20]. However, the inherent bottleneck of the current-domain CIM is the PVT robustness, preventing the current-domain CIM from achieving higher parallelism. The time domain methodology accumulates output from cells or local computing cells (LCC) by the cascaded delay chain. It overcomes the read-out interface bottleneck. However, the upper bound of parallelism is still limited by the maximum latency. The charge-domain methodology accumulates output

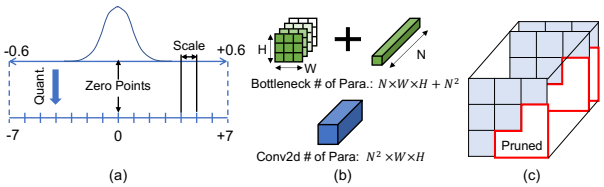

Fig. 3. Existing neural network compression methods: (a) quantization, (b) lightweight model structure, and (c) structured sparsity utilization.

by the principle of charge redistribution. The accuracy of charge-domain CIM depends on the capacitors matching, which usually has a better PVT robustness than transistors discharging. In this case, much higher parallelism of up to one thousand rows [18] could be motivated.

Digital-CIM [21], [22] is recently proposed, as shown in Fig. 2(d). To completely overcome the sensing overhead and the analog noise margin, the analog-mode accumulation modules are replaced with a digital adder tree, leading to higher accuracy. However, digital CIM has difficulty in exploiting bitwise sparsity by the adder trees with high area overhead compared to analog CIM.

B. Quantization and Sparsity Utilization

In addition to hardware exploration, software-hardware co-optimization is also a crucial approach to improving the inference efficiency of large neural network models.

One promising technique is quantization as shown in Fig. 3(a). One quantization method is quantization-aware training (QAT) using fine-tuning. It achieves higher accuracy on lightweight models. However, QAT is vulnerable to noise. On the contrary, post-training quantization is another practical technique without extra training. However, with more sensitive weight bits and activation bits in low-bitwidth quantization, the robustness would be a challenge in noise-sensitive analog-based CIM and noisy environments [23].

Lightweight models also motivate energy-efficient edge smart devices. As shown in Fig. 3(b), the cascaded depthwise convolution and pointwise convolution could perform as alternatives to larger-scale convolution layers. However, this approach poses challenges for other redundancy-dependent compression strategies, such as low-bit quantization on CIM.

Sparsity utilization is another approach to high-efficient inference. Thanks to the normal distribution of the weights and the ReLU activation function, massive weights and activations with value '0' provide an opportunity to alleviate the high computing overhead. There are two kinds of sparsity utilization methods. One is the unstructured sparsity using a sparse matrix multiplication optimization algorithm, and the other is the structured sparsity. Unstructured sparsity poses a significant challenge in regular CIM arrays. To overcome this issue, a structured sparsity [14], as shown in Fig. 3(c), was proposed to prune specific parts of the neural network. By taking advantage of input-wise, layer-wise, and channel-wise structured sparsity, the operation numbers of NNs can be significantly reduced, leading to energy-efficient and high-performance neural network inferences.

III. Proposed Architecture ZEBRA

A. Challenges and Opportunities

Deployment of low-bit quantized neural networks on the charge-domain CIM is a promising technique for achieving high-energy-efficient edge intelligence. As mentioned above, this design space has been well explored. However, there are still challenges and opportunities.

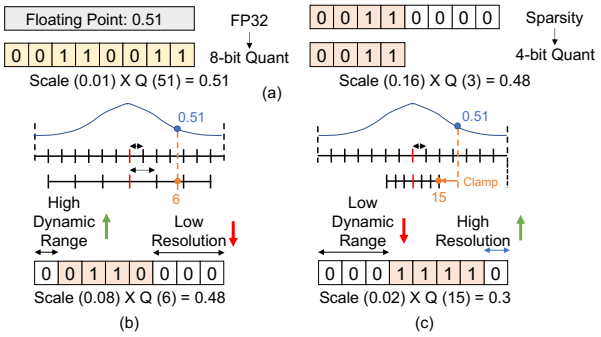

Fig. 4. The insight into (a) low-bit quantization method, and different existing optimization dilemmas towards (b) high dynamic range (but low resolution) and (c) high resolution (but low dynamic range).

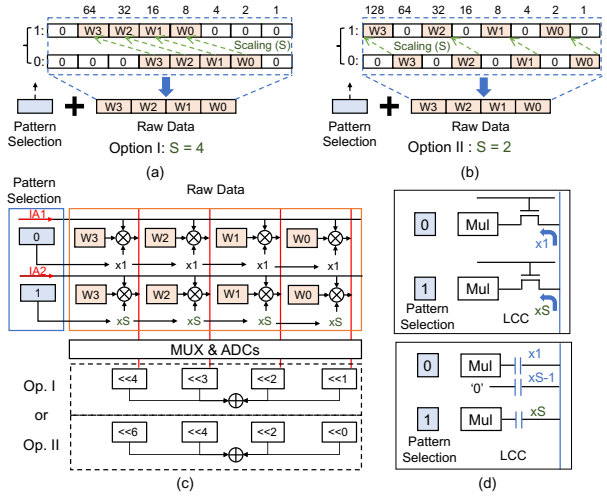

Fig. 5. Proposed value-adaptive sparsity patterns based on 8-bit quantization with (a) Option I: dynamic range and resolution tradeoff, (b) Option II: interleaving pattern; (c) CIM method; (d) Potential solutions.

First, there is a conflict between high dynamic range and high resolution in low-bit quantization for high-efficient inference. Improving quantization accuracy by shrinking the quantization scale without increasing precision will result in an accuracy loss due to dynamic range reduction.

Second, despite the advantages of improved matching, the charge-domain CIM still requires robust computing to maintain accuracy. Non-ideal factors or malicious attackers that inject noise in the analog CIM can result in significant accuracy loss in lightweight NNs due to low redundancy.

B. Robust Value-Adaptive Zero-Bit Pattern

To overcome the challenges above, the proposed ZEBRA first explores opportunities for robust analog CIM by combining high-dynamic and high-resolution patterns. Note that ZEBRA is the first work to utilize zero-bit patterns in the CIM structure. It is orthogonal to other sparsity utilization techniques and non-uniform quantization methods in the design space since they work on a different level.

Concerning zero-bit pattern in neural networks, we present an insight that lower-bit quantization can be viewed as a form of structured sparsity by fixing specific bit locations as '0'. This concept is illustrated in Fig. 4(a). It is noticed that, without altering the dynamic range, a 4-bit quantization can be considered equivalent to sparsing the lower 4-bit of 8-bit data. As shown in Fig. 4(b) and (c), there is a tradeoff between the dynamic range and the resolution of quantized parameters:

At low-bit quantization, it is feasible to reduce the dynamic range by a factor of two, as shown in Fig. 4(b). To maintain a high data resolution, aggressive range reduction is also an option shown in Fig. 4(c), with the overhead of a shrunk dynamic range. Thus, there exists a dilemma between dynamic range and resolution, which has been discussed by various works [15], [24]. The key is to reduce the dynamic range by reducing the amount of data in a single quantization process. An efficient approach is a vector-wise quantization [15]. However, it is a fine-grained dynamic quantization technique that is unfriendly to regular CIM arrays.

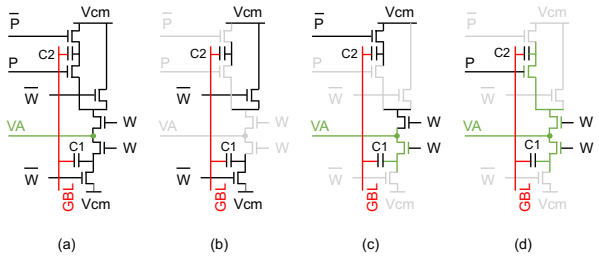

Fig. 6. The operation diagram of (a) the ML-LCU in situations of (b) weight bit (W)=0, (c) W=1, pattern selection bit (P) = 0, and (d) W=1, P=1.

It motivates us: *is it possible to achieve both high dynamic range and high quantized resolution in CIM architecture?*

This paper provides an answer: value-adaptive patterns. Fig. 5(a) and (b) show two implementation options, respectively.

Option I presented in Fig. 5(a) attempts to directly apply the aforementioned insight to represent a value by selecting between a high dynamic range pattern and a high-resolution pattern. It switches between two different sparse patterns with an extra pattern selection bit. One of the patterns fixed the most significant bit (MSB) and the least 3 significant bits to '0'. The other pattern fixed the least significant bit (LSB) and the most 3 significant bits to 0. This approach tends to achieve both a broad dynamic range for large values and higher resolution for small values at different zero-bit patterns while maintaining structured computing.

Option II tries to employ alternating zero-bit patterns, as shown in Fig. 5(b), to merge high dynamic range and high resolution in both patterns. One pattern is fixed to '0' for odd-bit locations, while the other pattern is fixed to '0' for even-bit locations. In this case, the scaling of the same bit is reduced from 4x to 2x.

Additionally, possible circuit implementations of Option I and Option II in the CIM array are discussed in Fig. 5(c) and (d), respectively. Fig. 5(d) shows two potential CIM cell structures that have been designed separately in the current domain and charge domain. The pattern selection bit of each parameter determines the gain of local computing units *S* in the current domain or the size of computing capacitors in the charge domain. In Option I, the size of *S* is 4 while in Option II, it is 2. This work efficiently implements structured CIM arrays, thereby avoiding performance degradation due to unstructured sparsity. However, scaling the LCC presents a challenge in that more accuracy accumulation is required due to a larger *S*, especially in Option I. What's worse, the complex computing units by the proposed bitwise sparsity bring significant area overhead, resulting in a decrease in overall area efficiency. Therefore, it is crucial to perform hardware and software co-optimization for implementing value-adaptive sparsity patterns on the analog CIM.

C. Multi-Level Local Computing Unit

First, it should be discussed how to represent a signed number. There are two possible implementations: two's complement encoding and signed binary encodings. In this

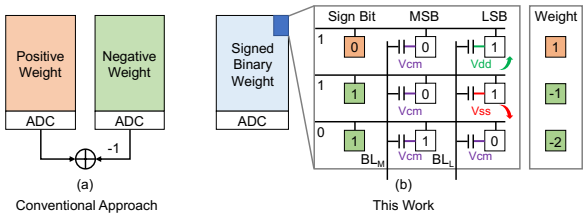

Fig. 7. Two approaches to performing signed binary encoding in the analog CIM: (a) separate positive and negative array and (b) sign bit with negative accumulation.

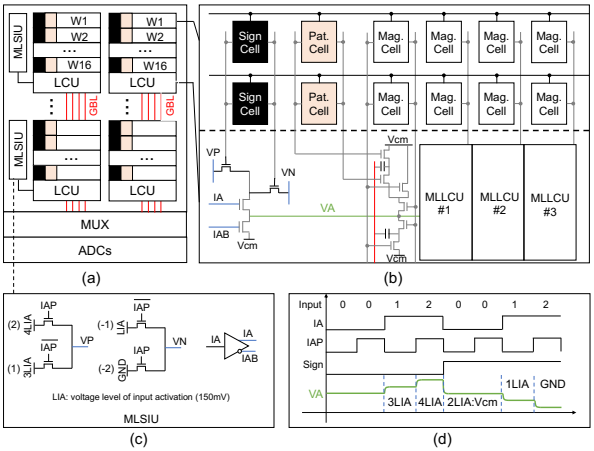

Fig. 8. Proposed multi-level local computing unit (ML-LCU) design in (a) macro level and (b) array level with (c) multi-level signed input unit circuits (ML-SIU) and (d) its waveform diagram.

work, to obtain a bitwise '0' pattern, the signed binary encoding of weights with a sign bit is necessary. As shown in Fig. 7, there are two ways to implement signed binary encoding in CIM. The first approach uses separated positive/negative weight arrays shown in Fig. 7(a). However, it is associated with a severe density overhead due to redundant memory cells. Alternatively, adding a sign bit is more promising, as shown in Fig. 7(b). This option requires a specialized design of the computing unit, but its area overhead is much smaller than that of the first scheme.

Second, to reduce the area overhead introduced by the extra computing units, local computing units that share computing units with multiple rows are more promising in ZEBRA.

Based on the discussions above, this work proposes a Multi-Level Local Computing Unit (ML-LCU) to perform value-adaptive bitwise sparsity patterns for high-robustness analog CIM in the charge domain with low-bit quantization.

As shown in Fig. 8 (a), the computing macro includes a weight array, local computing units (LCUs), multi-level signed input units (ML-SIUs), and the peripherals such as MUXs and ADCs. The input activations are sent in parallel to ML-SIUs for input signal generation. The ML-LCUs read the weights from the array and perform multiplication with the input signal. The results are then accumulated onto the global bitlines (GBL) and sensed by ADCs.

Fig. 8(b) and (c) depict the circuit design of the ML-LCU and the ML-SIU. The ML-SIU handles multi-level activations with value-adaptive zero-bit patterns by generating two pairs of differential signals based on the input activation (IA) and pattern selection bits (IAP). These signals determine the bitwise '0' pattern of the input activation. VP and VN represent the positive and negative values of the input activation bit, respectively. This work implements the signed binary encoding by setting the common-mode voltage level to '0'. The IA and IAB indicate the bit value of input activation.

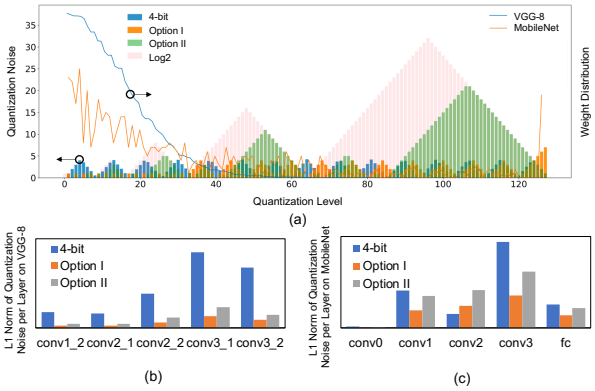

Fig. 9. Low-bit quantization noise study on (a) L1 norm of quantization noise, and different weight distribution on (b) VGG and (c) MobileNetV2.

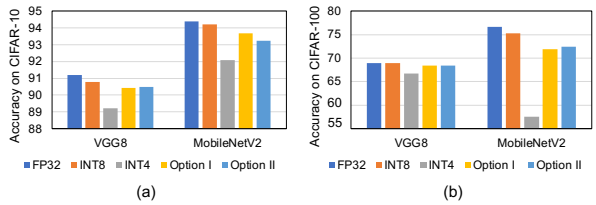

Fig. 10. Accuracy comparison between the proposed ZEBRA and prior low-bit quantization methods on (a) CIFAR10 and (b) CIFAR100.

The waveform diagram of the ML-SIU is shown in Fig. 8(d). ML-LCUs perform multiplication based on the signals from ML-SIU, the pattern selection bit of the weight, and the corresponding data bit. The activation voltage (VA) is selected between VP and VN by the sign bit from the weights array.

Fig. 6(a) presents the detailed structure and operation diagram of the proposed ML-LCU. The weights (W) and pattern selection bit (P) are read from the local bitline. When the weight bit of this column is 0, as shown in Fig. 6(b), the lower plate of computing capacitors is set to the common-mode voltage as value '0'. When the weight bit is 1, as shown in Fig. 6(c) and (d), the lower plate of capacitors will be charged to the input voltage (VA) from the ML-SIU. For example, when using value-adaptive zero-bit pattern Option II, if the pattern selection bit is 0, the weight scaling is 1. In this case, one of the computing capacitors is connected to the VA while another one is connected to the common-mode voltage. Conversely, if the pattern selection bit is 1, the weight scaling is 2, so both computing capacitors should be connected to the VA to perform value-adaptive zero-bit patterns. In pattern II, C2 = C1, while in pattern I, C2 = 3C1.

In conclusion, the proposed ML-LCU and ML-SIU implement low-cost multi-level signed multiplication with only 14T compared with the original 6T LCC. The hardware and software co-optimization enables robust zero-bit patterns to perform lightweight models on the analog CIM.

IV. Experiment

A. Experiment Setup

Software Setup. The robustness study in this work is evaluated on an end-to-end emulator based on PyTorch. The impact on Gaussian noise based on the deviation normalized to LSB of weights as prior works [23], [25] is evaluated in the emulator to get the output of the CIM macro.

Hardware Setup. This work is based on the charge-domain local computing cell [18], [26]. The layout of ZEBRA is implemented in a 28nm CMOS process to evaluate area, power, and latency overhead compared with 4-bit quantization. The macro-level performance is evaluated on SPICE post-

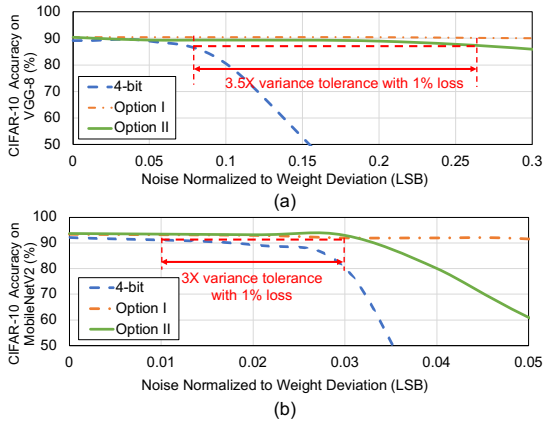

Fig. 11. Robustness enhancement based on weight equivalent noise on CIFAR-10 with (a) VGG-8 (4W-4A) and (b) MobileNet (4W-8A).

simulations. The system-level performance is evaluated by considering computing arrays with the peripheral same as the baseline including read/write IO, control units, and ADCs.

Dataset Setup. The models explored cover different redundancy NNs including VGG-8 and MobileNetV2. The datasets evaluated in this work are CIFAR-10 and CIFAR-100. Two options mentioned in Section III are discussed as follows.

(i) Dynamic range and resolution tradeoff: Four data bits with a pattern selection bit that decides the zero-bit patterns between higher dynamic range and higher resolution.

(ii) Interleaving pattern: Four data bits with a pattern selection bit that decides whether the odd locations are sparse or even locations are sparse.

B. Analysis and Insight

Fig. 9(a) shows the classical weight distributions on one typical convolutional layer of VGG-8 and MobileNetV2 trained on CIFAR-10 and the quantization noise relative to standard 8-bit quantization. The noise introduced by three low-bit quantization methods is presented, including conventional 4-bit quantization ("4-bit" in Fig. 9), logarithmic quantization with a base of 2 ("Log2" in Fig. 9) [24], and proposed ZEBRA methods (Option I and Option II).

Fig. 9(b) and (c) visually reflect the L1 norm quantization noise on each layer of VGG-8 and MobileNetV2. Overall, the comparison of the noise is consistent with the discussion above. However, the conv2 layer in MobileNetV2 is perverse. The reason for this exception is that the conv2 layer is a channel-wise convolutional layer. The weights of these layers do not match the typical distribution.

Fig. 10 shows the accuracy comparison among different methods. It could be found that the low-bit quantization of the proposed bitwise value-adaptive zero-bit pattern achieves <1% accuracy drop in VGG-8 and <3% accuracy drop in MobileNetV2, compared with ~2-3x intolerant accuracy loss of the same bitwidth 4-bit quantization.

Despite the accuracy improvement thanks to lower quantization noise, the advantages of the value-adaptive zero-bit pattern in accuracy may still not motivate the ZEBRA design. Therefore, insight is provided into the following robustness study.

C. Robustness Evaluation

The robustness issue of low-bit quantization is an intrinsic difficulty due to the more sensitive output to disturbances. The lower quantization resolution makes the impact of noise on accuracy greater. Although it is possible to compress the quantization range by clamping to improve resolution, the shrinking dynamic range also results in more outliers, which finally causes an accuracy drop. The proposed value-adaptive bitwise zero-bit patterns combine the larger dynamic range at the large value and the higher resolution at the small value to achieve software accuracy with high robustness.

TABLE I. MACRO-LEVEL PERFORMANCE COMPARISON

Metrics	Charge-domain CIM Methodologies			
	Baseline [a]		*ZEBRA Option I*	*ZEBRA Option II*
Process	28 nm CMOS			
Area of LCC (μm^2)	1.04		3.58	2.17
Macro area (mm^2)	0.32		0.36	0.34
Precision (I-W)	4-4	8-8	8-8	8-8
Parameter density (M/mm^2)	0.97	0.50	0.58	0.61
Energy efficiency[b] (TOPS/W)	65.4	15.9	64.3	64.6
Area efficiency[b] ($GOPS/mm^2$)	390	95.25	346	367
Accuracy[c] under noise	80.1%	94.2%	93.7%	93.3%
SNR Robustness	No	Yes	Yes	Yes
VGG-8 support	😐	☺	☺	☺
MobileNet support	☹	☺	☺	☺
Robustness	☹	☺	☺	☺

[a] Charge-domain local computing cell with 16 SRAM rows [18], [26]
[b] OP is defined with the precision of the same column
[c] Accuracy is obtained based on MobileNetV2 (4-8) under a variance of 0.03LSB

Fig. 11 shows the robustness comparison between the conventional 4-bit quantization and the proposed ZEBRA. Considering the actual analog compute-in-memory hardware, there are many possible error sources, such as non-ideal ADCs, capacitor mismatching, and external noise injection by attackers. To evaluate the robustness of low-bit quantization methods without certain hardware, this work abstracts the magnitude of noise into the standard deviation of equivalent noise [23], [25]. Fig. 11(a) shows the accuracy of VGG-8 with noise injection. It could be found that VGG-8 with higher redundancy could work on the analog CIM. On the contrary, the lightweight MobileNet fails. As shown in Fig. 11(b), ZEBRA improves the deviation tolerance with 1% accuracy loss by 3-5x. The advantage of the proposed ZEBRA is that it enables a lightweight model structure on the analog CIM for higher area/energy efficiency and accuracy.

It is noticed that experiments prove that the robustness improvement of Option I is more significant than Option II. However, Option II performs higher area and energy efficiency than Option I because of the lower capacitor C2 in Option II enabled by the lower scaling factor of the weight bit.

D. Macro Evaluation with Deployment Considerations.

According to the robustness evaluation above, the proposed value-adaptive bitwise zero-bit patterns are proved to have high robustness similar to 8-bit quantization in the analog CIM and achieve almost no accuracy degradation, overcoming the deployment issue of lightweight models in noisy analog CIM. However, the proposed sparsity utilization method could not be directly mapped into conventional analog CIM, such as the SRAM-based charge-domain CIM. Thus, the extra overhead introduced by the ML-LCU and ML-SIU should be discussed.

Table I shows the metrics comparison between the conventional charge-domain LCC of baseline and the proposed ML-LCU and ML-SIU under the SPICE simulation on a 28nm CMOS process. Compared with 8-bit quantization with similar accuracy, ZEBRA achieves 20% higher parameters density, 3.1x higher energy efficiency, and 2.9x

higher area efficiency improvement. The improvement of ZEBRA in area/energy efficiency comes from the low operation numbers by reducing computing bits, which still maintains the accuracy and robustness of the original 8-bit quantization by value-adaptive zero-bit patterns.

E. Overhead Discussion and Future Works

Table I demonstrates that the parameter density of ZEBRA suffers 37% overhead compared with 4-bit quantization. This is because though only 4-bit is used in sparse computing, 2 extra bits are still occupied as a sign bit and a pattern selection bit. Also, the energy efficiency of ML-LCU and ML-SIU deteriorates by 1.8-1.9x compared with the conventional LCC. However, only 2% energy efficiency overhead at the system level because the computing power consumption of the ML-LCU and ML-SIU occupies a small proportion of the overall system power consumption. What's more, thanks to the much higher noise tolerance compared with the 4-bit quantization method, the parallelism of the ZEBRA could be higher to improve the performance or the computing capacitors could be smaller to improve the energy efficiency.

In this work, the number of sparsity patterns in a single scheme is limited to two due to the hardware overhead introduced by the pattern selection bit and the ML-LCU. In addition, using multiple options on demand keeps unexplored. A reconfigurable ZEBRA architecture with multiple options and a hybrid encoding scheme for more patterns will promise better system performance and accuracy.

V. Conclusion

This paper proposes a high-robustness charge-domain compute-in-memory (CIM) architecture ZEBRA by value-adaptive zero-bit patterns to deal with the bottleneck of deploying a low-redundancy neural network on analog-based CIM. Experiments from the macro level to the system level are completed to evaluate the accuracy and the hardware performance. On the one hand, ZEBRA is capable of tolerating noise due to mismatches, non-ideal interfaces, noise injection, etc. On the other hand, ZEBRA shows about 2.9x higher area efficiency and 3.1x higher energy efficiency compared to existing schemes with the same accuracy, thanks to the higher robustness enabled by ZEBRA.

Acknowledgment

This work is supported in part by the National Key R&D Program of China (2019YFA0706100), in part by NSFC (U21B2030, 92264204), and in part by NSF (2008365, 2132918, 2246149).

References

[1] L. Floridi and M. Chiriatti, "GPT-3: Its Nature, Scope, Limits, and Consequences," in *Minds & Machines*, vol. 30, no. 4, pp. 681–694, Dec. 2020.

[2] N. P. Jouppi *et al.*, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *ISCA' 17*, Jun. 2017, pp. 1–12.

[3] X. Si *et al.*, "A Dual-Split 6T SRAM-Based Computing-in-Memory Unit-Macro With Fully Parallel Product-Sum Operation for Binarized DNN Edge Processors," *IEEE Trans. Circuits Syst. I*, vol. 66, no. 11, pp. 4172–4185, Nov. 2019.

[4] A. Biswas and A. P. Chandrakasan, "CONV-SRAM: An Energy-Efficient SRAM With In-Memory Dot-Product Computation for Low-Power Convolutional Neural Networks," *JSSC*, vol. 54, no. 1, pp. 217–230, Jan. 2019.

[5] S. Ha *et al.*, "A 36.2 dB High SNR and PVT/Leakage-Robust eDRAM Computing-In-Memory Macro With Segmented BL and Reference Cell Array," *IEEE Trans. Circuits Syst. II*, vol. 69, no. 5, pp. 2433–2437, May 2022.

[6] S. Xie *et al.*, "16.2 eDRAM-CIM: Compute-In-Memory Design with Reconfigurable Embedded-Dynamic-Memory Array Realizing Adaptive Data Converters and Charge-Domain Computing," in *ISSCC'21*, San Francisco, CA, USA: IEEE, Feb. 2021, pp. 248–250.

[7] L. Xia *et al.*, "Technological Exploration of RRAM Crossbar Array for Matrix-Vector Multiplication," *J. Comput. Sci. Technol.*, vol. 31, no. 1, pp. 3–19, Jan. 2016.

[8] T. Soliman *et al.*, "Ultra-Low Power Flexible Precision FeFET Based Analog In-Memory Computing," in *IEDM' 20*, San Francisco, CA, USA: IEEE, Dec. 2020, p. 29.2.1-29.2.4.

[9] Y. Chen *et al.*, "YOLoC: deploy large-scale neural network by ROM-based computing-in-memory using residual branch on a chip," in *DAC '22*, ACM, Jul. 2022, pp. 1093–1098.

[10] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv:1704.04861 [cs]*, Apr. 2017. Accessed: Oct. 18, 2021.

[11] X. Zhang *et al.*, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *CVPR '16*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 6848–6856.

[12] G. Li *et al.*, "SCWC: Structured channel weight sharing to compress convolutional neural networks," *Information Sciences*, vol. 587, pp. 82–96, Mar. 2022.

[13] C. Hong *et al.*, "Adaptive sparse tiling for sparse matrix multiplication," in *PPoPP' 19*, pp. 300–314.

[14] W. Wen *et al.*, "Learning Structured Sparsity in Deep Neural Networks," *arXiv: 1608.03665*, Oct. 2016.

[15] S. Dai *et al.*, "VS-Quant: Per-vector Scaled Quantization for Accurate Low-Precision Neural Network Inference".

[16] C. Eckert *et al.*, "Neural Cache: Bit-Serial In-Cache Acceleration of Deep Neural Networks," in *ISCA'18*, Los Angeles, CA: IEEE, Jun. 2018, pp. 383–396.

[17] Jintao Zhang *et al.*, "A machine-learning classifier implemented in a standard 6T SRAM array," in *ISVLSI '16*, Honolulu, HI, USA, Jun. 2016, pp. 1–2.

[18] H. Jia *et al.*, "A Programmable Heterogeneous Microprocessor Based on Bit-Scalable In-Memory Computing," *IEEE J. Solid-State Circuits*, vol. 55, no. 9, pp. 2609–2621, Sep. 2020.

[19] J. Song *et al.*, "TD-SRAM: Time-Domain-Based In-Memory Computing Macro for Binary Neural Networks," *IEEE Trans. Circuits Syst. I*, pp. 1–11, 2021.

[20] C.-J. Jhang *et al.*, "Challenges and Trends of SRAM-Based Computing-In-Memory for AI Edge Devices," *IEEE Trans. Circuits Syst. I*, vol. 68, no. 5, pp. 1773–1786, May 2021.

[21] Y.-D. Chih *et al.*, "16.4 An 89TOPS/W and 16.3TOPS/mm ^{2} All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," in *ISSCC '21*, San Francisco, CA, USA, Feb. 2021, pp. 252–254.

[22] F. Tu *et al.*, "A 28nm 29.2TFLOPS/W BF16 and 36.5TOPS/W INT8 Reconfigurable Digital CIM Processor with Unified FP/INT Pipeline and Bitwise In-Memory Booth Multiplication for Cloud Deep Learning Acceleration," in *ISSCC 2022*, San Francisco, CA, USA: IEEE, Feb. 2022, pp. 1–3.

[23] S. K. Gonugondla *et al.*, "Fundamental limits on the precision of in-memory architectures," in *ICCAD' 20*, Virtual Event USA, Nov. 2020, pp. 1–9.

[24] S.-E. Chang *et al.*, "Mix and Match: A Novel FPGA-Centric Deep Neural Network Quantization Framework," in *HPCA' 21*, Seoul, Korea (South): IEEE, Feb. 2021, pp. 208–220.

[25] A. S. Rekhi *et al.*, "Analog/Mixed-Signal Hardware Error Modeling for Deep Learning Inference," in *DAC' 19*, Las Vegas NV USA: ACM, Jun. 2019, pp. 1–6.

[26] X. Si *et al.*, "A Local Computing Cell and 6T SRAM-Based Computing-in-Memory Macro With 8-b MAC Operation for Edge AI Chips," *IEEE J. Solid-State Circuits*, vol. 56, no. 9, pp. 2817–2831, Sep. 2021.