Tolerating Noise Effects in Processing-in-Memory Systems for Neural Networks: A Hardware–Software Codesign Perspective

Xiaoxuan Yang,* Changming Wu, Mo Li, and Yiran Chen*

Neural networks have been widely used for advanced tasks from image recognition to natural language processing. Many recent works focus on improving the efficiency of executing neural networks in diverse applications. Researchers have advocated processing-in-memory (PIM) architecture as a promising candidate for training and testing neural networks because PIM design can reduce the communication cost between storage and computing units. However, there exist noises in the PIM system generated from the intrinsic physical properties of both memory devices and the peripheral circuits. The noises introduce challenges in stably training the systems and achieving high test performance, e.g., accuracy in classification tasks. This review discusses the current approaches to tolerating noise effects for both training and inference in PIM systems and provides an analysis from a hardware–software codesign perspective. Noise-tolerant strategies for PIM systems based on resistive random-access memory (ReRAM), including circuit-level, algorithm-level, and system-level solutions are explained. In addition, we also present some selected noise-tolerate cases in PIM systems for generative adversarial networks and physical neural networks.

1. Introduction

The development of deep learning has inspired innovations in a variety of popular applications, including computer vision, natural language processing, and autonomous driving. We have seen successful deep learning models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), generative adversarial networks (GANs), and attention-based networks. Since the data that need to be processed for deep learning models is increasing dramatically, the efficiency of training and implementing the networks is an important issue. In addition, it has been noticed that the data movement between the storage component and the computing resource dominates the total power consumption in modern computing architecture, such as von Neumann architecture.[1] Therefore, improving efficiency and eliminating frequent data movement are critical for designing hardware systems for neural networks.

Processing-in-memory (PIM) designs, also known as computing-in-memory designs, can reduce the data communication cost because the computation can be implemented within the memory device.[2] The comparison between von Neumann architecture and in-memory computing can be seen in Figure 1. For instance, if one neural network is implemented in PIM design, the weight can be programmed to the memory device, and fetching weight data of each layer from the cache can be eliminated. Currently, nanoscale memory devices are being explored and studied, and representative devices include static random-access memory (SRAM),[3] flash memory,[4] phase-change memory (PCM),[5] ferroelectric field-effect transistor (FeFET),[6] and resistive random-access memory (ReRAM).[7] These memory candidates and memory-based PIM designs feature high performance and energy efficiency. In this article, we mainly discuss the ReRAM-based PIM design because of the advantages of ReRAM, such as its nonvolatility, multilevel cell representations, low access energy, and long endurance of up to $10^{12}$ cycles for binary states.[8] Neural networks for cutting-edge applications like image categorization,[7] image generation,[9] and natural language translation,[10] can benefit from ReRAM-based PIM designs.

Crossbar-based ReRAM, owing to the advantages in physically executing vector–matrix multiplication, enables highly power-efficient and area-efficient neuromorphic system designs. Therefore, memristor-based designs have become indispensable for implementing neural network architectures in the analog domain. However, memristors have nonideal properties due to programming inaccuracies and device imperfections from fabrication, which lead to errors or mismatches in computed...
In this article, we discuss circuit-level, algorithm-level, and system-level solutions to build reliable memristor-based designs for neural network training and inference in the condition of the noise in the PIM system. Inferencing with the PIM system indicates that the forward path of the neural network is implemented in the PIM system. The noise will influence the inferencing accuracy, but the backward path and weight update are noise-free.\textsuperscript{[11]} However, training with PIM systems is a more complex topic, and thus a small mismatch in the system can lead to poor training results.\textsuperscript{[12]} Therefore, the on-chip training algorithm should adapt to the system imperfections.\textsuperscript{[13,14]}

In addition, we look into the latest work on noise-tolerating in PIM systems. For instance, the photonic PIM system has the advantage of high parallelism and high energy efficiency, but the device suffers from programming noise. Likewise, physical systems have to deal with the device’s and peripheral circuits’ nonideality. Therefore, we show representative noise-tolerant cases in these systems.

2. Background

In this section, we want to explain the ReRAM-based processing-in-memory designs and analyze the noise in PIM systems. Besides, we include the general approaches towards noise in neural networks.

2.1. ReRAM-Based Processing-in-Memory Designs

ReRAM has been viewed as one of the most outstanding techniques in neural network acceleration. The ReRAM cell is made up of a metal–oxide material layer inserted between two metal electrode layers. ReRAM has the advantages of nonvolatility, small size, multilevel cell representation, fast access time as well as low access energy. ReRAM is energy efficient because the matrix–vector multiplication can be performed using the memristive array structure. Beyond that, novel designs such as PRIME,\textsuperscript{[13]} ISAAC,\textsuperscript{[16]} and PipeLayer,\textsuperscript{[7]} are exploring the improvements in energy efficiency.

The benefit of utilizing ReRAM as a neural network accelerator lies in the crossbar structure because it can complete the matrix-vector calculation. The horizontal wordlines (WL) and the vertical bitlines (BL) can be treated as matrix rows and columns, and the conductance value in each intersection can be viewed as the element in the corresponding matrix position. The conductance is preprogrammed according to the weight values. Figure 2 illustrates the crossbar array structure. Let $G$ be the conductance matrix of the ReRAM crossbar. When the voltage $V_I$ is applied to the crossbar wordlines, the output current generated through the bitlines is $I_O = V_I G$ according to Kirchhoff’s current law and Ohm’s law.

We would use GAN as an example to explain the advantages of ReRAM-based PIM design. In a GAN model, two neural network models (i.e., generator and discriminator) are simultaneously trained and therefore, consume high computation and storage resources. Moreover, the generator in GAN uses a novel transposed convolutional (TCONV) layer, which inserts many zeros.
into its input feature maps before performing traditional convolution operations, leading to significant resource underutilization. To address the challenges, ReRAM-based PIM designs are proposed to reduce the resource requirement and improve the resource utilization rate. For example, ReGAN,[9] the ReRAM-based GAN accelerator, is proposed to efficiently pipeline the training to on-chip memory access and reduce the training latency. In addition, spatial parallelism and computation sharing are proposed to further improve performance. Another work, ZARA, addresses the resource underutilization problem with a novel computation deformation technique that can skip zero-insertions in TCONV.[17] By pre-classifying the kernels into sub-kernels, the proposed design can get rid of the zero-insertion step in TCONV. Most importantly, this deformation design in the forward phase can be extended into the error backpropagation phase and weight update phase, and thus the training efficiency can be further improved with this ReRAM-based design.

2.2. Noise in ReRAM-Based PIM Systems

The variation in ReRAM-based neural network computing refers to parameter variation and process variation. The resistance status (including high resistance state or low resistance state) of the ReRAM cell changes according to the voltage applied to the electrode layers. If there is a voltage applied to the ReRAM cell, there is a possibility that a conductive filament consisting of oxygen vacancies is formed. The randomness of generating oxygen vacancies is the major reason for the process variation.[18] Given the same level of voltage and equivalent conditions, the shape of oxygen vacancies is unpredictable, which leads to the difference in resistance levels among cycles. Hsu et al. studied the variation in the ReRAM resistance and showed the fitted distribution for high-resistance state and low-resistance state.[19]

For the ReRAM-based PIM system inferencing, different kinds of noise should be carefully studied. Thermal noise is generated due to the thermal agitation of the charge carriers inside the conductor and depends on temperature and operating frequency. In ReRAM devices, this thermal noise affects the output current. Therefore, the thermal noise can be modeled as a current source in parallel with a passive resistor.[20] Shot noise originates from the discrete charges in the current flow and is related to the operating frequency.[21] Random telegraph noise (also known as RTN) is generated in semiconductor and ultrathin oxide films. Programming noise is introduced when writing to ReRAM devices.

To summarize, the potential challenges of noise in the PIM system lie in the following aspects: First, when there is a high operating frequency or high temperature, the thermal noise and shot noise increase and lead to the high-amplitude noise. Accuracy degradation happens when the operating frequency or temperature increases.[22] Second, the noise is aggregated when considering the combination of noise sources. Third, the tolerable noise margin is reduced if the ReRAM cell resolution is high, such as 8 bit. Therefore, the study of tolerating the noise effect in the PIM system should consider these challenges and improve the system’s reliability under model perturbation.

2.3. General Approaches Towards Noise in Neural Networks

Before moving to the PIM-specific solution, it is important to determine the general approaches toward noise in neural networks. The elements involved in the software-based neural network training process are in floating-point format. However, when implementing the neural network with analog or digital hardware, the data precision of the system should be carefully determined because of these two concerns.

First, reducing weight and activation precisions introduces quantization noise to the process and can lead to test accuracy degradation. This circumstance indicates that the quantization effect is only considered in the forward path. For instance, the deep compression method utilizes a full-precision pretrained model and quantizes the model with finetuning procedure.[23] Recent research shows that post-training quantization that doesn’t involve retraining or fine-tuning the full model can also effectively quantize the model.[24–27]

Another critical issue of training convergence appears when utilizing the quantized version of neural networks during both training and testing. Chen et al. proposed a hardware accelerator for convolutional neural networks and showed that the training precision should be selected as 32 bits instead of 16 bits to guarantee the convergence.[28] To reduce the constraints for training precision, Gupta et al. proposed stochastic rounding for gradient so that the computing units can use 16-bit fixed-point data for both the forward and backward path.[29] However, the step of quantization changes the elements into discrete values, and the performance of the gradient-based method degrades because continuous space is not achievable. The straight-through estimator (STE) is proposed to support the training of quantized neural networks with ultralow precision.[30] In this method, a copy of the full-precision weight is kept, and the quantized weight will be used for the forward path calculation. During the backward path, the gradient of the quantized weight is directly passed to the gradient of the full-precision weight. Consequently, gradient-based optimization can be performed on the full-precision weight. This STE method inspires multiple quantized neural network designs.[31–33] For generalized quantization robustness, recent works, including gradient L1 regularization and Hessian-based regularization, target the various bit widths instead of one specific width and show the effectiveness of the proposed regularization scheme.[34,35]

3. Circuit-Level and Algorithm-Level Solutions

In this section, we aim to analyze the circuit-level and algorithm-level approaches for tolerating noise effects in PIM designs. We first go over an example of in situ learning of a single-layer perceptron classifier, as shown in Figure 3.[36] Training set and weight initialization are the inputs for the training system, and the crossbar (shown in the gray-shaded box) is the place for the matrix-vector multiplication and gradient update. Besides, the Delta learning rule and the Manhattan learning rule are adopted for hardware-based in situ learning, and the equations for learning rules can be found in Figure 3b.[37,38] Here, the Manhattan learning rule is more hardware-friendly and efficient because it only updates the sign information of the result from
the Delta learning rule. Considering the switching dynamics of memristive hardware devices, the variable-amplitude writing scheme is proposed to make the training algorithm more compatible with the circuit design.\cite{43}

When the design scale increases to multilayer perceptron, the noise problem becomes more severe because the disturbed output from the first layer serves as the activation for the second layer. Thus, the impact of noise in different layers may accumulate. In a circuit-level work,\cite{40} the weight disturbances of memristor cells under different read operation directions are analyzed, and preferred read schemes are proposed to reduce the impact of memristance drift and improve the system reliability. To combat the possible noise effect in the system, the closed-loop circuit provides a feedback control from each layer’s output to the weight. This design takes advantage of the feedback loop to stabilize the weight. In this work, the learning algorithm introduces the hardware inferencing result to the update rule. Results show that the closed-loop design can maintain good performance for \(14.95 \times 10^3\) service time compared with the open-loop design.

There are some solutions targeting the effect of one specific noise in the ReRAM-based PIM system. For example, a design strategy is to use the hardware training method to minimize problems caused by process variations, where the parameter tuning strategy is based on the difference between the target value and the output value with noise.\cite{41} Taking programming noise as another example, we can analyze the effect of programming noise and the possible solutions. Multiple memristor cells can be integrated to represent a single weight if the memristor cell’s resolution is smaller than the weight’s resolution, which is often the case for a ReRAM-based PIM system.\cite{42} Thus, programming noise that relates to the more significant cells will be multiplied by a scalar during integration, and the programming noise in the most significant cell is of the greatest concern. Long et al. proposed a dynamic fixed-point representation to reduce the unused most significant bit of memristor cells.\cite{43} Besides, the noise-aware training algorithm also manipulates the disturbed weight in the forward path while keeping the original weight copy for the backward phase, which is similar to the quantization-aware training method. The result shows that the combination of dynamic fixed-point data format and the noise-aware training facilitates finding a more smooth local optimal surface in the loss space.

Besides, the analysis of weight elements’ importance in the neural network is also leveraged in hardware design. Beigi et al. analyze the importance of weight elements and identify the most effective rows in a weight matrix. In the thermal-aware mapping method, they map the most effective rows to cold rows and the most ineffective rows to hot rows.\cite{44} Furthermore, Shin et al. propose a thermal-aware optimization framework to reduce the average temperature and temperature variance among memristor arrays.\cite{45}

4. System-Level Solutions

To achieve the goal of designing noise-tolerant, high-performance, energy-efficient PIM systems, system-level approaches are necessary. Besides, the overhead of such a powerful system-level approach should be minimized. One important feature of system-level solutions is that the framework includes the whole design space into consideration. For instance, Xia et al. propose an exploration flow of system parameter selection to overcome the impact of variations.\cite{46} Moreover, they design a training algorithm to leverage the inherent self-healing capability of a neural network to further prevent large weights from being mapped to a large variation memristor. For reliability optimization, a simulation framework which enables reliable ReRAM-based accelerators for deep learning (DL-RSIM) can be used to explore the design space of ReRAM-based deep learning accelerators.\cite{47} DL-RSIM proposes a workload-dependent sensing strategy to show how to leverage the framework to build reliability optimization techniques. The simulation findings can assist chip designers in selecting a superior design alternative during the early stages of development.

In recent work, Yang et al. consider different existing noise sources in PIM systems and target at achieving robustness and effectiveness with minimal design exploration cost.\cite{48} They design a ReRAM-based stochastic-noise-aware training method (ReSNA), and include three major design analyses. First, the distribution of noise under frequency and temperature settings is analyzed. Thermal noise, shot noise, and programming noise is modeled as Gaussian distributed noise, while

![Flowchart of one epoch for batch- and stochastic-mode training algorithms](image-url)
RTN is modeled as Poisson-distributed noise. Programming noise scales linearly with the conductance levels, while thermal and shot noise is related to the square root of the conductance values.\textsuperscript{[20]} Second, the relative conductance change is included in the ReRAM cell during training. Note that there is a larger relative impact for lower conductance levels, where most weights of DNN are distributed. Besides, thermal and shot noise dominates at small conductance, which is troublesome under high frequency. Then the hardware settings, such as the resolutions of the analog-to-digital converter and digital to analog converter, are also included in the ReSNA algorithm.

For achieving robust and efficient ReRAM-based designs, the goal of the design space exploration is to find Pareto-optimal ReRAM-based system with minimal cost. However, the challenge here is that the number of available design configurations is up to $10^7$, and each evaluation with ReSNA will consume a large computation cost. The solution here is continuous-fidelity max-value entropy search for multi-objective optimization (CF-MESMO) with the following key steps: Gaussian processes are employed as the choice of the statistical model due to their superior uncertainty quantification ability. Then, sample the Pareto Front with a surrogate model and calculate the information gain per unit cost with the formula derived from the properties of Gaussian processes and Monte Carlo sampling. Afterward, the next candidate ReRAM design and fidelity pair can be selected with the information gain formula. The key idea is to find the ReRAM design and fidelity pair that maximizes the information gain per unit cost about the Pareto front of the highest fidelity. According to the evaluation of optimization efficiency, CF-MESMO achieves a 90.91% reduction in computation cost compared to the popular multi-objective optimization algorithm NSGA-II when reaching the best performance of NSGA-II.

Generally, system-level solutions for tolerating noise effects in PIM systems can successfully explore the design space for the optimal design choice while maintaining the test performance under stochastic noise.

5. Representative Noise-Tolerant Case Study

In this section, we want to present the latest work on noise-tolerating designs in PIM systems, including the photonic system and physical system. Other than these cases, SRAM-based PIM design and FeFET-based PIM design can also tolerate the noise effects through different design techniques.\textsuperscript{[47,48]}

5.1. Photonic Systems: Harnessing Noise Effect in Generative Adversarial Network

In a photonics-based generative adversarial network design (shown in Figure 4), Wu et al. propose an efficient weight-compensatory training method to harness the noise effect to generate the diverse image pattern successfully.\textsuperscript{[49]} Compared with previously mentioned designs, the uniqueness of this design is that generative adversarial networks take stochastic random noise as an input. Therefore, the noise in the computing kernel will be added to the noise from the input vector. Besides, the network itself takes advantage of the noise effect to produce different patterns to fool the discriminator. The photonic computation kernel consists of an array of programmable phase-change mode converters (PMMC). During the programming of the kernel weight, the noise distribution can be estimated as the Gaussian distribution. During the training process, the noise of a specific standard derivation is included in the forward path, as shown in Figure 4b. In each layer, the input signals pass through the photonic tensor core and are converted to the electrical domain by photodetectors (PDs). After postprocessing, the data are converted back into the optical domain and transferred to the next layer. This data transformation process can also introduce noise to the result. The experimental result shows that the noise-aware training can outperform the noiseless training with respect to image quality and diversity. The results of the noise impact show that the weight-compensatory GAN built on photonic hardware with practical noise outperforms the noiseless hardware in inference. The inference accuracy of a discriminative network, on the other hand, inevitably decreases as the hardware becomes noisier.\textsuperscript{[50]}

Despite the unavoidable optoelectronic disturbances and faults, this unexpected increase in performance reveals photonic neural networks’ promise in generative models.

However, the convergence of the photonic system with complementary metal-oxide semiconductor (CMOS) so far is still far from mature. Specifically, the processing-in-memory photonic systems require the incorporating working between the photonic circuit and the electronic circuit to perform high-fidelity optical signal processing with a variety of optoelectronic functions. Monolithic integration of the photonic chip in the proximity of the electronic circuits thus is necessary. One challenge toward this monolithic integration is the lack of a suitable material for photonic functions either in bulk CMOS or state-of-the-art CMOS technics below 28 nm transistor node.\textsuperscript{[51]} Another challenge is that several necessary photonic building blocks such as on-chip circulators, isolators, and amplifiers have still not proliferated to the extent that they have in bulk fiber-optics technologies.\textsuperscript{[52–55]} Despite this, it is worth noting that efforts toward monolithic photonic–electronic systems have been demonstrated and paved the way to integrate photonics with state-of-the-art CMOS technics.\textsuperscript{[51]}

5.2. Physical Systems: Redesigning Backpropagation to Tolerate Noisy

To eliminate the effect of device asymmetry, an alternative stochastic gradient descent method called the Tiki-taka algorithm is proposed and validated with various neural networks.\textsuperscript{[56]} The Tiki-taka algorithm can simultaneously minimize the training loss and the hidden cost introduced by the asymmetric update. Therefore, the adaptive learning algorithm can be beneficial for the training process with real devices.

In recent work, a physics-aware training method (PAT) has been proposed to support the backpropagation in physical neural networks, as shown in Figure 5.\textsuperscript{[57]} The physical neural network includes the mechanical neural network, optical neural network, and electrical neural network. PAT is capable of creating accurate hierarchical classifiers that take advantage of each system’s unique physical transformations and intrinsically reduce each system’s noisy processes and flaws. The difference between PAT and previously mentioned noise-aware training methods
**Figure 5.** Physics-aware training method (PAT) is a hybrid in situ–in silico algorithm to apply backpropagation to train controllable physical parameters so that physical systems perform machine-learning tasks accurately even in the presence of modeling errors and physical noise. Instead of performing the training solely within a digital model (in silico), PAT uses the physical systems to compute forward passes. Although only one layer is depicted in (a), PAT generalizes naturally to multiple layers. Reproduced with permission.[57] Copyright 2022, Springer Nature.

**Figure 4.** Photonic generative adversarial network (GAN) network with optoelectronic noises. a) A GAN architecture is composed of two subnetwork models: a generator and a discriminator. The generator competes with the discriminator during training and produces new instances after it is trained. b) The offline noise-aware training and inference processes flow of the generator. The process of mapping the trained weight to the hardware during implementation inevitably introduces optoelectronic noise. c) Decomposition of the generator into individual layers. EOM, electro-optic modulator. d) Optical microscopic image of the photonic tensor core consisting of four input channels. The optical RNG is input to the photonic tensor core through O/E and E/O conversion in our experiment. DEMUX, demultiplexers. e) The false-colored scanning electron microscope (SEM) image of the photonic tensor core. The Si$_3$N$_4$ waveguide, the GST metasurface, and the Al$_2$O$_3$ protection layer are colored green, red, and blue, respectively. Scale bar, 10 μm. Inset: The zoomed-in SEM image of the phase-gradient metasurface on the waveguide. Scale bar, 2 μm. Reproduced with permission.[49] Copyright 2022, AAAS.
Conflict of Interest

The authors declare no conflict of interest.

Keywords

hardware–software co-design, noise, processing-in-memory, resistive random-access memory

Acknowledgements

This work is supported in part by NSF 1955196, ARO W911NF-19-2-0107, and ONR MURI N00014-17-1-2661.

1. J. Gómez-Luna, I. E. Hajj, I. Fernandez, C. Giannoulia, G. F. Oliveira, O. Mutlu, in 12th Int. Green and Sustain. Computing Conf., IEEE, Piscataway, NJ 2021 pp. 1–7.
2. B. Yan, B. Li, X. Qiao, C. Xue, M. Chang, Y. Chen, H. H. Li, Adv. Intell. Syst. 2019, 1, 1900068.
3. J. Zhang, Z. Wang, N. Verma, IEEE. J. Solid-State Circuits 2017, 52, 915.
4. F. Merrikh-Bayat, X. Guo, M. Klachko, M. Prezioso, K. K. Likharev, D. B. Strukov, IEEE Trans. Neural Netw. Learn. Syst. 2017, 29, 4782.
5. M. Le Gallo, A. Sebastian, R. Mathis, M. Manica, H. Giefers, T. Tuma, C. Bekas, A. Curioni, E. Eleftheriou, Nat. Electron. 2018, 1, 246.
6. Y. Luo, P. Wang, X. Peng, X. Sun, S. Yu, IEEE J. Exploratory Solid-State Comput. Dev. Circuits 2019, 5, 142.
7. L. Song, X. Qian, H. Li, Y. Chen, in 2017 IEEE Int. Symp. On High Performance Computer Architecture, IEEE Computer Society, Piscataway, NJ 2017 pp. 541–552.
8. C.-W. Hsu, I.-T. Wang, C.-L. Lo, M.-C. Chiang, W.-Y. Jang, C.-H. Lin, T.-H. Hou, 2013 Symposium On VLSI Technology, IEEE, Piscataway, NJ 2013 pp. T166–T167.
9. F. Chen, L. Song, Y. Chen, in 23rd Asia and South Pacific Design Automation Conf., IEEE, Piscataway, NJ 2018, pp. 178–183.
10. X. Yang, B. Yan, H. Li, Y. Chen, in IEEE/ACM Int. Conf. On Computer Aided Design,IEEE, Piscataway, NJ 2020 pp. 92:1–92:9.
11. D. Kwon, S. Lim, J.-H. Bae, S.-T. Lee, H. Kinn, Y.-T. Seo, S. Oh, J. Kim, K. Yeom, B.-G. Park, J.-H. Lee, Front. Neurosci. 2020, 423.
12. O. Krestinskaya, K. N. Salama, A. P. James, IEEE Trans. Circuits and Systems I: Regular Papers 2018, 66, 719.
13. C. Li, D. Belkin, Y. Li, P. Yan, M. Hu, N. Ge, H. Jiang, E. Montgomery, P. Lin, Z. Wang, W. Song, Nat. Commun. 2018, 9, 1.
14. P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, H. Qian, Nature 2020, 577, 641.
15. P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, Y. Xie, in 43rd ACM/IEEE Annual Int. Symp. On Computer Architecture, IEEE Computer Society, Piscataway, NJ 2016 pp. 27–39.
16. A. Shafee, A. Nag, M. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, V. Srikumar, in 43rd ACM/IEEE Annual Int. Symp. On Computer Architecture, IEEE Computer Society, Piscataway, NJ 2016, pp. 14–26.
17. F. Chen, L. Song, H. H. Li, Y. Chen, in Proc. of the 56th Annual Design Automation ConfACM, New York, NY 2019, p. 133.
18. M. Lin, H. Cheng, W. Lin, T. Yang, I. Tseng, C. Yang, H. Hu, H. Chang, H. Li, M. Chang, in Proc. of the Int. Conf. On Computer-Aided Design, ACM, New York, NY 2018, p. 31.
19. K. Hsieh, C. Lu, in SSDM, Japan 2015, pp. 1168–1169.
20. B. Feinberg, S. Wang, E. Ipek, in IEEE Int. Symp. On High Performance Computer Architecture, IEEE Computer Society, Piscataway, NJ 2018 pp. 52–65.
21. T. Gokmen, Y. Vlasov, Front. Neurosci. 2016, 10, 333.
Xiaoxuan Yang received her B.S. degree in electrical engineering from Tsinghua University in 2016 and M.S. degree in electrical engineering from the University of California, Los Angeles (UCLA) in 2018. She is currently pursuing Ph.D. degree in the Department of Electrical and Computer Engineering, Duke University. Her research interests include emerging nonvolatile memory technologies and hardware accelerators for deep learning applications.
Changming Wu received his B.S. degree in physics from the University of Science and Technology of China (USTC) in 2014 and M.S. degree from the Hong Kong University of Science and Technology (HKUST) in 2017. He is currently pursuing a Ph.D. degree at the University of Washington. His main research interests include integrated photonics for machine learning and optical computing using nonvolatile phase-change memory.

Mo Li is a professor in the UW Department of Electrical and Computer Engineering and the UW Physics Department. He received his Ph.D. degree in applied physics from Caltech in 2007, B.S. degree in Physics from the University of Science and Technology of China (USTC) in 2001. Dr. Li is the principal investigator of the Laboratory of Photonic Devices at UW. His main research interests include integrated photonics, optoelectronics, optomechanics, NEMS/MEMS, spintronics, and quantum photonics. He has published over 80 peer-reviewed journal papers and has been granted 6 USA patents.

Yiran Chen received his B.S (1998) and M.S. degrees (2001) from Tsinghua University and Ph.D. degree (2005) from Purdue University. He is now the professor at the Department of Electrical and Computer Engineering at Duke University and serves as the director of the NSF AI Institute for Edge Computing Leveraging the Next-generation Networks (Athena) and the NSF Industry–University Cooperative Research Center (IUCRC) for Alternative Sustainable and Intelligent Computing (ASIC), and the co-director of Duke Center for Computational Evolutionary Intelligence (CEI). Dr. Chen has published 1 book and about 500 technical publications and has been granted 96 USA patents.