Exploring Model Stability of Deep Neural Networks for Reliable RRAM-Based In-Memory Acceleration

Gokul Krishnan, *Student Member, IEEE*, Li Yang, *Student Member, IEEE*, Jingbo Sun, *Student Member, IEEE*, Jubin Hazra, *Student Member, IEEE*, Xiaocong Du, *Member, IEEE*, Maximilian Liehr, *Member, IEEE*, Zheng Li, *Member, IEEE*, Karsten Beckmann, *Member, IEEE*, Rajiv V. Joshi, *Fellow, IEEE*, Nathaniel C. Cady, *Member, IEEE*, Deliang Fan, *Member, IEEE*, and Yu Cao, *Fellow, IEEE*

Abstract—RRAM-based in-memory computing (IMC) effectively accelerates deep neural networks (DNNs). Furthermore, model compression techniques, such as quantization and pruning, are necessary to improve algorithm mapping and hardware performance. However, in the presence of RRAM device variations, low-precision and sparse DNNs suffer from severe post-mapping accuracy loss. To address this, in this work, we investigate a new metric, *model stability*, from the loss landscape to help shed light on accuracy loss under variations and model compression, which guides an algorithmic solution to maximize model stability and mitigate accuracy loss. Based on statistical data from a CMOS/RRAM 1T1R test chip at 65nm, we characterize wafer-level RRAM variations and develop a cross-layer benchmark tool that incorporates quantization, pruning, device variations, model stability, and IMC architecture parameters to assess post-mapping accuracy and hardware performance. Leveraging this tool, we show that a loss-landscape-based DNN model selection for stability effectively tolerates device variations and achieves a post-mapping accuracy higher than that with 50% lower RRAM variations. Moreover, we quantitatively interpret why model pruning increases the sensitivity to variations, while a lower-precision model has better tolerance to variations. Finally, we propose a novel variation-aware training method to improve model stability, in which there exists the most stable model for the best post-mapping accuracy of compressed DNNs. Experimental evaluation of the method shows up to 19%, 21%, and 11% post-mapping accuracy improvement for our 65nm RRAM device, across various precision and sparsity, on CIFAR-10, CIFAR-100, and SVHN datasets, respectively.

Index Terms—In-memory computing, RRAM, model stability, deep neural networks, reliability, pruning, quantization

1 Introduction

Deep neural networks (DNNs) outperform humans for a variety of applications, such as computer vision and natural language processing. Higher accuracy comes at the cost of increased computational complexity and model size, posing great challenges to traditional architectures [1]. In addition, limited on-chip memory capacity leads to a significant amount of communication with off-chip memory, whose energy consumption is 1,000$\times$ higher than that of computations [2].

RRAM-based IMC accelerators provide a dense and parallel structure to achieve high performance and energy efficiency [3], [4]. Prior works with RRAM-based crossbar architectures have shown up to 1,000$\times$ improvement in energy efficiency as compared to CPUs/GPUs [3], [4], [5], [6], [7]. The increased energy-efficiency is attributed to a full-custom design following the assumption; all weights are stored on-chip [3], [4], [5]. However, RRAM-based IMC architectures incur a significant area overhead, especially when the DNN model size is rapidly increasing. Hence, model compression (e.g., pruning and quantization) is necessary for RRAM-based in-memory acceleration of DNNs.

In reality, RRAM suffers from statistical variations such as quantization error, device-to-device write variations, stuck-at-faults, and limited R_{off}/R_{on} ratio, posing a significant challenge to designing reliable RRAM-based IMC architectures [10], [11], [12]. The statistical variations in RRAM cause deviation in the programmed resistance leading to a significant loss in post-mapping accuracy (i.e., accuracy in the presence of RRAM variations) for DNNs. To mitigate the post-mapping accuracy loss in DNNs, variation-aware training (VAT) is employed [9], [10], [12], [13],

- *Gokul Krishnan, Li Yang, Jingbo Sun, Xiaocong Du, Zheng Li, Deliang Fan, and Yu Cao are with the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ 85287 USA. E-mail: {gkrish19, lyang166, jsun127, xiaocong, zhengl11, dfan, Yu.Cao}@asu.edu.*
- *Jubin Hazra, Maximilian Liehr, Karsten Beckmann, and Nathaniel C. Cady are with the State University of New York Polytechnic, Albany, NY 12246 USA. E-mail: {JHazra, mliehr2, kbeckmann, ncady}@sunypoly.edu.*
- *Rajiv V. Joshi is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA. E-mail: rvjoshi@us.ibm.com.*

Manuscript received 1 Nov. 2021; revised 10 Mar. 2022; accepted 3 Apr. 2022. Date of publication 12 May 2022; date of current version 10 Oct. 2022.
This work was supported by C-BRIC, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, and the Air Force Research Laboratory under Grant #FA8750-19-1-0014.
(Corresponding author: Gokul Krishnan.)
Digital Object Identifier no. 10.1109/TC.2022.3174585

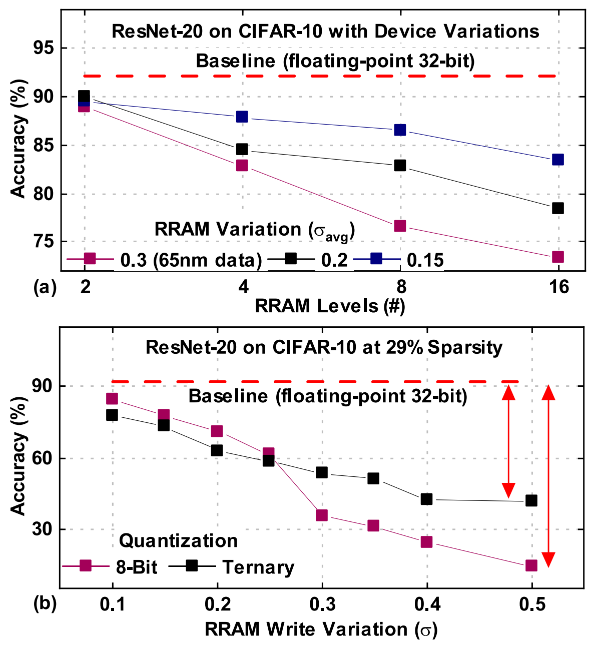


Fig. 1. Post-mapping accuracy for ResNet-20 on CIFAR-10 (a) across different RRAM levels and average RRAM variations. For a given R_{off}/R_{on} ratio, higher RRAM levels suffer from variations and thus lower accuracy. Simultaneously, a lower average RRAM variation results in higher accuracy, (b) at 8-bit and ternary-bit precision, with 29% sparsity [8]. Model is trained and tested with the same variation (σ) [9], [10]. The 8-bit model has more accuracy loss than the ternary model as σ increases.

[14]. VAT exploits the inherent redundancy in DNN by embedding the device variations (σ), based on a log-normal or normal distribution model, into the training process to achieve a variation-tolerant model, with no need of re-training for each individual RRAM chip [9], [10], [13]. The conventional VAT techniques train and test the DNN model at the same level of variation.

Fig. 1a shows the post-mapping accuracy for ResNet-20 [15] on CIFAR-10 dataset for different RRAM levels across various average RRAM variations. The baseline accuracy for the floating-point 32bit (FP-32) model is shown in red (dash line). The red curve shows the variation of pre-mapping accuracy with RRAM levels for our 65nm RRAM data with an average variation (σ_{avg}) of 0.3. For a given R_{off}/R_{on} ratio, a higher number of RRAM levels leads to higher variation and lower accuracy, and vice-versa. A higher variation for a higher number of RRAM levels arises from the increased HRS state utilization. Further, we analyze the scenario with a reduced average variation of 0.2 and 0.15. A reduction in RRAM variation through improved process control improves the pre-mapping accuracy. But, though the reduction in RRAM variation improves the accuracy, it does not achieve the same accuracy as FP-32. Hence, we establish that the reduction in RRAM variation does not get the pre-mapping accuracy back to the FP-32 level (baseline).

Next, we analyze the effect of RRAM variations on a sparse and quantized DNN. Fig. 1b shows the accuracy of ResNet-20 for CIFAR-10 at 29% sparsity for different RRAM write variations and data precision using the conventional VAT method [9]. Conventional VAT proves ineffective under pruning and quantization, resulting in reduced post-mapping accuracy. Furthermore, a lower precision (ternary) helps improve the post-mapping accuracy, as shown in Fig. 1b. Hence, there is a need for a more systematic solution for reliable RRAM-based in-memory computing for dense, sparse, and quantized DNNs.

To address this, in this work, we propose a new metric, *model stability*, from the loss landscape to help shed light on accuracy under variations and model compression and guide an algorithmic solution that mitigates the loss. The model stability is visualized by the loss landscape and evaluated by the roughness score [16]. A lower roughness score indicates a smoother loss landscape and a more stable model. Through this, we select the more stable model that can withstand the variations better. The proposed model stability-based model selection effectively tolerates device variations and achieves a post-mapping accuracy higher than that with 50% lower RRAM variations. Next, we propose a novel variation-aware training (VAT) method for best model stability in compressed DNNs. The proposed method utilizes VAT to train the compressed DNN with different scales of device variations (σ) to search for the most stable model and improve post-mapping accuracy. For a given DNN model, higher model stability implies better tolerance of variations and thus, higher post-mapping accuracy. We utilize a structured pruning method [8] and model quantization [17], [18] to compress DNN. The pruning method considers the mapping of the DNN onto the RRAM crossbar for best IMC performance. We show that pruning results in a less stable model, while quantization improves the model stability. We demonstrate that the proposed method achieves up to 19%, 21%, and 11% improvement in post-mapping accuracy on CIFAR-10, CIFAR-100, and SVHN datasets, respectively. The major contributions of this work are as follows:

- We propose a new metric, *model stability*, using the loss landscape to mitigate the accuracy loss of dense and compressed DNNs in the presence of RRAM variations,
- Using model stability as a metric to choose the more stable model results in similar accuracy improvement as that for 50% lower RRAM variations through costly process control,
- We further propose a model stability-based VAT method for compressed DNNs, which searches the most stable model under variations to achieve the best post-mapping accuracy, without knowing the exact amount of RRAM testing variations upfront,
- Finally, we show that the model-stability-based VAT method achieves up to 19%, 21%, and 11% improvement in accuracy for compressed DNNs on CIFAR-10, CIFAR-100, and SVHN datasets, respectively.

2 DNN Model Stability

Given a trained DNN model, its model stability is an intrinsic property to withstand perturbations, such as variations in model weights and input noise. Model stability of a DNN, i.e., the DNN generalization capability,is directly related to the contour of the loss function [16], [19], [20], [21]. A flatter contour of the loss function leads to a larger region of acceptable minima, which allows the DNN model to better tolerate variations in both weights and inputs.Vice versa, a steeper contour of the loss function leads to a

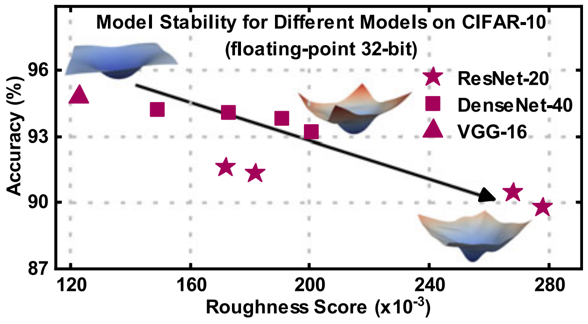


Fig. 2. A lower roughness score leads to a smoother loss landscape, higher stability, and thus, higher model accuracy.

smaller region of acceptable minima [19], which implies that any perturbations to the weights or inputs will lead to appreciable movement of the minima point and thus, reduce model accuracy. In this work, we utilize DNN model stability as a metric to guide reliable RRAM-based IMC acceleration.

2.1 Landscape Visualization

In order to quantitatively understand model stability, we utilize the landscape visualization method [21] to visualize the minima of the loss function. In [21], filter normalization is applied to remove the scaling effect of injected noise, and a 3-dimension matrix is generated with x, y, and z coordinates, where x and y represent the scale of two random perturbations injected into the model and z is the loss function. Essentially this matrix plots the fluctuation of the loss function under the local perturbation around the local minimum.

2.2 Roughness Score

We calculate the smoothness of the loss function, defined as roughness score, to quantify the loss landscape's stability further. We fit the 3-dimensional data from the landscape using quadratic linear regression and obtain the mean square error (MSE) of the fitting model, as shown below:

$$\hat{z}_j = w_{j4}x_j^2 + w_{j3}y_j^2 + w_{j2}x_j + w_{j1}y_j + w_{j0}, \tag{1}$$

$$\hat{w} = \underset{w_j}{\operatorname{argmin}} \frac{1}{n}\sum_{j=0}^{n}(z_j - \hat{z}_j)^2, \tag{2}$$

where w_j represents the fitted coefficients. We denote the stability or roughness score of the DNN model as $\mathrm{MSE}(z; x^2, y^2, x, y; \hat{w})$. A smaller MSE arises from a flat and smooth landscape and vice-versa. Note that such a method was previously used to improve the accuracy in continual learning [16], while it did not consider device variations, sparsity, and quantization. *To the best of our knowledge, this is the first time that model stability has been employedto provide systematic guidanceto improve the post-mapping accuracy for the acceleration of dense and compressed DNNs using RRAM-based IMC architectures.*

2.3 Roughness Score and DNN Accuracy

Fig. 2 shows the accuracy and roughness score for different DNNs. We generate different versions for a DNN model by utilizing different weight initialization, which leads to different roughness scores and corresponding accuracy. VGG-16 for CIFAR-10 achieves 94.2% accuracy and the lowest roughness score of 118x10^{-3}. At the same time, a ResNet-20 version achieves the lowest accuracy of 89.5% with a roughness score of 278x10^{-3}. To further understand the relationship between the roughness score, loss landscape, and DNN accuracy, we visualize the loss landscape using the method detailed in [21]. VGG-16 has a shallow and smooth loss landscape while the lowest accuracy ResNet-20 variant has a rough loss landscape. Through these examples we establish that a lower roughness score leads to a smoother loss landscape, more acceptable local minima for the loss function, and a higher DNN accuracy.

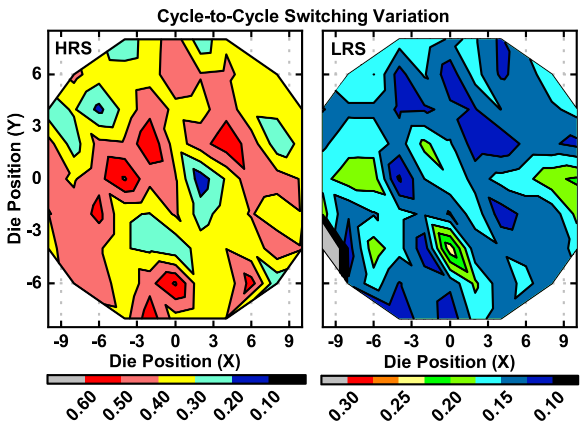


Fig. 3. HRS (left) and LRS (right) cycle-to-cycle switching variation across the 300mm wafer at 65nm. HRS state has a higher variation than LRS.

3 65NM 1T1R RRAM DEVICE

To accurately model the RRAM device properties, RRAM data is collected from a fully integrated 1T1R structure on a 300mm wafer, using a custom RRAM module within the SUNY Polytechnic Institute's 65nm process. The size of each RRAM device is 100nmx100nm. The RRAM device stack is comprised of a 6nm HfO_2 mem-resistive switching layer, a 6nm PVD Ti oxygen exchange layer (OEL), and TiN electrodes (top and bottom). Electrical characterization is performed using a pulse-based approach having a magnitude 1V – 1.2V and width of 10μs for the set and reset operation of RRAM devices.

Fig. 3 shows the wafer-level cycle-to-cycle switching variations for the 65nm RRAM device measured using the pulse-based switching technique. The high-resistance state (HRS) has a higher variation up to 0.6, while the low-resistance state (LRS) has a lower variation up to 0.1. The average variation (σ_{avg}) for the entire range of HRS and LRS amounts to 0.3.

Fig. 4 shows the normalized variation for different $\mathrm{R_{off}/R_{on}}$ ratios across 1110 1T1R RRAM devices at 65nm. As the ratio of $\mathrm{R_{off}/R_{on}}$ goes up its HRS state utilization increases. A higher HRS state utilization results in higher device variations and an increase in overall average variation, as shown in Fig. 3. Overall, a lower $\mathrm{R_{off}/R_{on}}$ ratio results in lower average device variation at the cost of lower usable resistance levels and hardware performance, and vice-versa.

Fig. 5 shows the retention of both HRS and LRS for 10^5 seconds at 100°C. The grey region overlaid on the plot

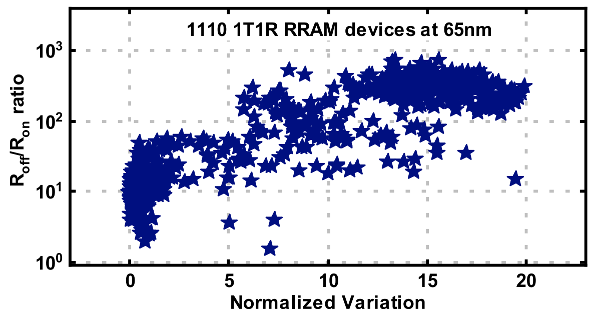


Fig. 4. Normalized variation for different R_{off}/R_{on} ratios for 1110 1T1R RRAM devices at 65nm. A higher R_{off}/R_{on} ratio leads to higher variation.

shows the static variation from the RRAM device. The 65nm 1T1R RRAM device shows low retention variation, as shown in Fig. 5. Table 1 shows the endurance data for both HRS and LRS states up to 1 billion cycles. Both the HRS and LRS states show high endurance with distinction between the two states up to 1 billion cycles. Since static write variations are dominant, in this work, we do not consider the effects of retention or endurance. Table 2 summarizes the device models used in the cross-layer simulation framework described in Section 4. In this work, we use the 65nm RRAM models for all our experiments to provide realistic results.

4 Cross-Layer Simulation Framework

In this work, we develop an in-house simulator to perform system-level benchmarking of the RRAM-based IMC architecture. Fig. 6 shows the block diagram of the cross-layer benchmarking tool utilized.

The simulator incorporates device, circuits, architecture, and algorithm under a single roof to perform system-level benchmarking. The simulator provides post-mapping accuracy (hardware accuracy), the overall hardware performance, roofline model, and model stability. The inputs to the simulator include the DNN structure, data precision, target sparsity, technology node, bits per RRAM cell, IMC crossbar size, ADC resolution, read-out method, frequency, NoC topology, and NoC size, among others. Table 3 shows the comparison between different popular IMC simulators. Compared to prior works [22], [23], [24], the proposed simulator provides supports for both SPICE and behavioral-based computation fabric estimation, supports both accuracy and hardware performance estimation, NoC-mesh interconnect support, estimation of roofline model and model stability of the DNN, and support sparsity and quantization.

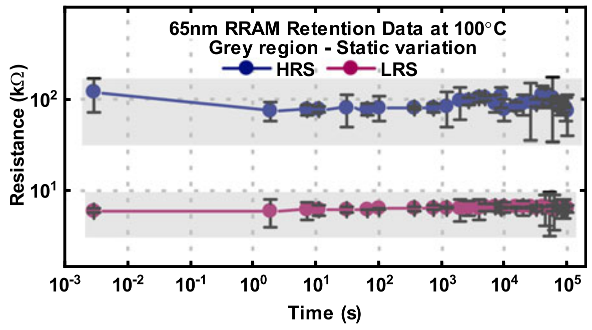


Fig. 5. The 65nm RRAM device achieves good retention for up to 10^5 seconds. The grey region shows the dominating effect of static variations for the 65nm 1T1R RRAM device.

TABLE 1
Endurance Measurement up to 1 Billion Cycles

Cycles	10	10^2	10^3	10^5	10^7	10^9
LRS (KΩ)	5.7	5.5	5.7	5.6	5.8	5.9
HRS (KΩ)	109	231	157	43.3	113.8	30.8

Furthermore, the supported sparsity follows a hardware-friendly structure as detailed in Section 4.4.1.

4.1 Device Models

The tool incorporates device models from the 65nm 1T1R RRAM device, as shown in Table 2. The RRAM device variations are modeled using the log-normal distribution [10], [25]. The R_{off}/R_{on} ratio of the RRAM device ranges between 2 and 650. We assume that the discrete resistance levels used to represent the weights are within the limited R_{off}/R_{on} ratio. The maximum number of resistance levels that a single RRAM cell can handle is 16, limiting the weights to be mapped to a single cell to 4-bits.

4.2 Circuit Estimator

The circuit estimator performs the estimation of the IMC, peripheral circuits, and digital modules within the architecture. We benchmark the overall circuit estimator with SIAM [26]. The circuit estimator performs the mapping of the DNN onto the RRAM-based IMC crossbar architecture. The mapping utilized within the estimator follows that in [26]. The IMC circuit components include the crossbar, wordline (WL) driver, level shifters, bitline (BL) and selectline (SL) multiplexers (MUX), column MUX, analog-to-digital converter (ADC), and shift and add circuit. In addition, the circuit estimator benchmarks the accumulator, pooling unit, and buffer circuits in the architecture. The estimator utilizes the device models and the transistor properties to perform the overall estimation.

4.3 Architecture

Fig. 6 shows the overall architecture utilized in this work. The architecture utilized is similar to that in [4] with a homogeneous crossbar size. The top-level consists of an array of IMC tiles interconnected by an NoC-mesh. Each tile consists of an array of PEs connected by an H-Tree interconnect. The PE consists of an RRAM-based IMC crossbar and

TABLE 2
Device Models from 65Nm 1T1R RRAM Device

Parameter	**Value**
R_{off}/R_{on}	2-650
Write Variation[1]	$r' \leftarrow r.e^{\theta}; \theta \sim \mathcal{N}(0, \sigma^2)$
Average Variation (σ_{avg})	0.3
RRAM levels (#)	16
Aging Variation (σ_{age})	0.1, 0.02 (HRS, LRS)

[1]r is the ideal resistance to be programmed and r′ is the real value in RRAM. $0.1 \leq \sigma \leq 0.5$.

[1]*r is the ideal resistance to be programmed and r′ is the real value in RRAM.* $0.1 \leq \sigma \leq 0.5$.

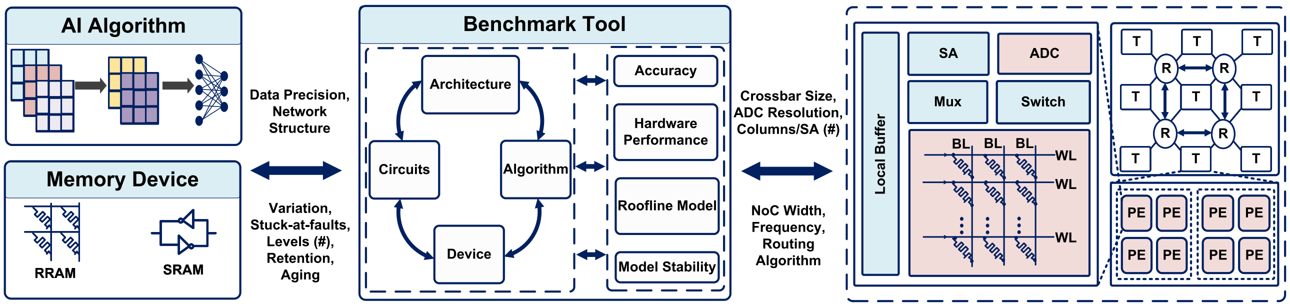

AI Algorithm
Memory Device
RRAM
SRAM
Data Precision, Network Structure
Variation, Stuck-at-faults, Levels (#), Retention, Aging
Benchmark Tool
Architecture
Circuits
Algorithm
Device
Accuracy
Hardware Performance
Roofline Model
Model Stability
Crossbar Size, ADC Resolution, Columns/SA (#)
NoC Width, Frequency, Routing Algorithm
Local Buffer
SA
ADC
Mux
Switch
BL
WL
T
R
PE

Fig. 6. The benchmarking tool incorporates algorithm (pruning and quantization), RRAM IMC architecture properties, circuit models, and the 65nm 1T1R RRAM device models. The tool outputs post-mapping accuracy, hardware performance, roofline model, and model stability.

associated peripheral circuits. The peripheral circuits include an ADC, column MUX, switch matrix, WL decoder and driver, level shifters, and SL and BL MUX. In addition, the PE consists of a local buffer that is utilized for the movement of activations and partial sums into and out of the PE.

The architecture utilizes an NoC-mesh to perform the on-chip data movement at the tile-level. Each tile is associated with an NoC router that performs the packet scheduling and routing. The NoC utilizes an X–Y routing mechanism. To benchmark the NoC interconnect, we utilize a cycle-accurate simulator that is benchmarked against the NoC module within SIAM [26]. To perform the estimation, we generate traces for the packets communicated between the tiles similar to that detailed in [26]. These traces are then utilized as the input to the NoC estimator to evaluate the cost of on-chip communication.

4.4 Algorithm

The algorithm component of the simulator performs the DNN training, pruning, quantization, variation-aware-training (VAT), evaluation of model stability, and hardware-aware training. The VAT training performed in this work utilizes the device models detailed in Table 2. We utilize the log-normal distribution to add the variations for each weight. The variations are added such that the variations depend on the weight value, thus having one-to-one correlation to the real hardware. In addition, for the hardware-aware training we include the effect of the limited precision of the RRAM, the ADC, and the accumulator within the shift and add circuit. Finally, the algorithm component of the simulator evaluates the model stability of the DNN after training. To achieve this, we evaluate the roughness score of the loss function and visualize the loss landscape as detailed Section 2. In the following sections we detail the pruning and quantization methodologies utilized in this work.

4.4.1 *Pruning*

In this work, we adopt the structured pruning method in [8]. It utilizes a weight-penalty clipping with a self-adapting threshold, as shown below:

$$\hat{\mathcal{L}} = \mathcal{L}(f(\mathbf{x};\{\mathbf{W}_l\}_{l=1}^{L}),\mathbf{t}) + \lambda\sum_{l=1}^{L}\sum_{i=1}^{G_i}\min(\|W_{l,i}\|_2;\delta_l) \tag{3}$$

$$\delta_l = \alpha.\frac{1}{G_l}\sum_{i=1}^{G_i}\|\mathbf{W}_{l,i}\|_2, \tag{4}$$

where δ_l denotes the layer-wise self-adapting clipping threshold, L is the number of layers, G_l is the number of groups in the l-th layer, λ is the hyper-parameter to be tuned based on the dataset, and α is the scaling coefficient. The pruning is conducted group-wise along the output channel dimension: for-layer l with weight matrix $W_l \in \mathbb{R}^{N_{of}\times N_{if}\times K_x\times K_y}$, we choose a group of size N_g along the N_{if} dimension, where N_g is determined by the crossbar size. Groups of $K_x \times K_y \times N_g$ weights are pruned across output channels to favor the IMC.

4.4.2 *Quantization*

The pruned model is further compressed by applying quantization. For 4-bits or higher precision, we employ uniform in-training quantization [18]. Furthermore, for ternary bit precision, we follow the ternarization method in [17]. For both ternary and higher bit precision, we employ the straight-through-estimator (STE) method in the backward-propagation to counteract the non-differential issues of the discrete quantization function. In this work, we focus on 8-bit and ternary weight quantization.

4.4.3 *Convergence Analysis*

There exist several works to analyze the convergence of DNN when performing pruning and quantization[27], [28], [29]. First, for the weight quantization, [27] finds that if

TABLE 3
Comparison Between Different IMC Simulators

Simulator	Computation Fabric	Communication Fabric	Accuracy Estimation Support	Hardware Performance Estimation	Roofline & Model Stability Estimation	Sparsity Support
NeuroSim [22]	SPICE-based	P2P (H-Tree)	Yes	Yes	No	No
Zhang et al. [23]	NA	NA	Yes	No	No	No
MNSIM 2.0 [24]	Behavioral Model	NoC-mesh	Yes	Yes	No	No
Ours	**SPICE & Behavioral Model**	**NoC-mesh**	**Yes**	**Yes**	**Yes**	**Yes**

assuming the loss function is L-Lipschitz smooth and the gradients are bounded, the loss of the quantized model will converge to an error related to the weight quantization resolution Δw and weight dimension d as shown below:

$$\frac{R(T)}{T} \leq O\left(\frac{d}{\sqrt{T}}\right) + LD\left(\sqrt{D^2 + \frac{d\alpha^2\Delta_w^2}{4}}\right), \tag{5}$$

$R(T)$ is the evaluated regret, which is equal to $\sum_{t=1}^{T} f(\mathbf{W}_t) - f(\mathbf{W}_t^*)$, where $\mathbf{W}_t^*$ is the best model parameter, T is the number of iterations, and D is the finite diameter in the domain. It can be seen that the smaller the Δw or d, the smaller is the error. In addition, [29] provides the convergence analysis of the group Lasso-based pruning method. By setting suitable smoothing parameters, [29] proves the weak and strong convergence of the training process for the smoothing of neural networks, respectively. In practice, they show that weak convergence indicates that the norm of the gradients of the smooth cost function goes to zero, and the strong convergence implies that the weight sequence tends to a fixed point $\mathbf{W}^f$. Moreover, they demonstrate that such a convergence result of smoothing of the network is consistent with the original non-smoothing one with group-lasso penalty. The consistency between the smooth error function $\hat{E}(\mathbf{W}_t)$ and original error function $E(\mathbf{W}_t)$ is shown below:

$$\lim_{t\to\infty} \hat{E}(\mathbf{W}_t) = \lim_{t\to\infty} E(\mathbf{W}_t) = E(\mathbf{W}^f). \tag{6}$$

5 Model Stability for RRAM-Based IMC

In this section, we detail the algorithm utilized to evaluate the model stability of DNN under the presence of RRAM variations, sparsity, and quantization.

Algorithm 1 details the methodology utilized in this work to evaluate the model stability. First, for each DNN model we perform training to generate the floating-point model A. We perform inference with model A to calculate the inference accuracy. Next, we quantize the DNN model to fixed-point weights and activations across all layers of the DNN, to generate model B. Uniform quantization is employed to maximize the hardware performance for the RRAM-based IMC architecture. The quantized model is then pruned to generate the structured sparse DNN model C. Thereafter, we add the RRAM variations (σ_{train}) and perform hardware-aware training for the quantized model to generate model D. In addition to adding the RRAM variations, hardware-aware training involves breaking the convolution or fully-connected (FC) layer into partial convolutions and FC layer operations based on the size of the crossbar and adding the ADC quantization for the column sum from each crossbar. We then perform inference for the hardware-aware trained model D in the presence of the RRAM testing variations (σ_{test}) to generate the realistic hardware accuracy. Next, to evaluate the model stability for each model, we plot the loss landscape as defined in Section 2.1. Finally, we evaluate the roughness score of the loss landscape (models A, B, C, and D) to quantify the stability of the DNN model. In this work, we propose to use model stability as a metric for reliable RRAM-based IMC accelerations. To achieve this, we propose two directions, first, a model stability-based model selection, and second, a model stability-based VAT method.

Algorithm 1. Model Stability

```
1: Input: DNN, weight precision ($W$), activation precision ($A$),
   RRAM training variations ($\sigma_{train}$), RRAM testing variations
   ($\sigma_{test}$), crossbar size, and ADC precision
2: Output: Roughness Score ($R_s$) and loss landscape
3: for DNN Model do
4:    Perform DNN training
          /* Model A */
5:    if Quantize then
6:    Perform DNN quantization (Section 4.4.2)
          /* Model B */
7:    end
8:    if Pruning then
9:       Perform pruning for DNN Model as in Equation 3 and
             Equation 4
             /* Model C */
10:   end
11:    Add RRAM variations ($\sigma_{train}$) and perform hardware-
       aware training
          /* Model D */
12:   Plot loss landscape using tool in [21] for models A, B, C,
      and D
13:   Calculate roughness score of loss landscape for models
      A, B, C, and D
14: end
```

Given a dataset, Fig. 2 illustrates that the choice of the DNN model significantly affects the accuracy. Based on this observation, we propose a novel loss landscape-based model selection for stability that tolerates RRAM device variations and achieves higher post-mapping accuracy. Such an observation is attributed to the DNN model stability. Model stability of a trained DNN is the intrinsic property to withstand perturbations such as variations and noise. A more stable model with higher model stability will be more robust under RRAM variations and have higher post-mapping accuracy.

Model stability-based model selection provides a viable solution when there is a choice for the target DNN. But, if the DNN cannot be changed and is compressed, model selection cannot be utilized. To address this, in this work, we propose a novel model stability-based VAT to improve the post-mapping accuracy of DNNs under sparsity and quantization. Previous VAT approaches focus on non-pruned DNNs and require the precise knowledge of RRAM testing variations and apply that to the training [9], [10]. Distinct from that, we first train the sparse and quantized DNN model with different scales of device variations (σ_{train}), without knowing the exact amount of RRAM testing variations. The range of σ_{train} is from 0.1 to 0.5, as suggested by the 65nm 1T1R RRAM data.Furthermore, we evaluate the loss landscape and the roughness score for each of the different VAT variants to help identify the optimal model with the highest model stability (Algorithm 1). A higher model stability from the optimal scale of training variation leads to higher post-mapping accuracy.

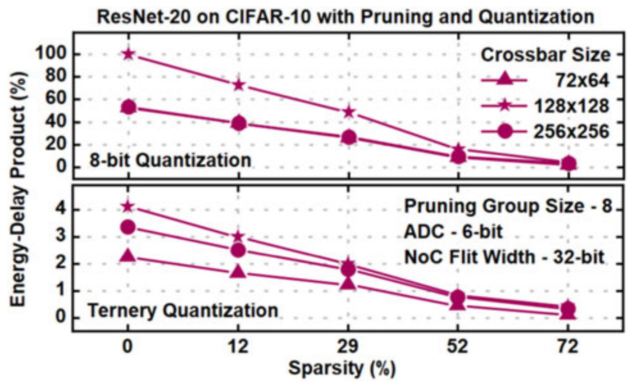


Fig. 7. Energy-delay product (EDP) normalized to the largest value (at 0% sparsity with 128×128 crossbar size) across different sparsity, quantization (8-bit and ternary), and crossbar sizes for ResNet-20 on CIFAR-10.

6 Experiments and Results

6.1 Experimental Setup

We evaluate the proposed model stability metric for reliable RRAM-based IMC acceleration using two main methods. First, we demonstrate a DNN model selection for higher model stability which improves the overall DNN accuracy. We evaluate the proposed method for ResNet-20 on CIFAR-10, DenseNet-40 for CIFAR-10 and CIFAR-100, and ResNet-32 for CIFAR-100. All experiments are done for a crossbar size of 256×256 with a 5-bit ADC at the periphery. We evaluate for different RRAM levels ranging from 2 to 16.

Next, we extend the model stability metric to present a novel model stability-based VAT method to mitigate the accuracy degradation in RRAM-based IMC architectures. We evaluate the proposed VAT method for ResNet-20 on CIFAR-10, VGG-16 on CIFAR-10, ResNet-32 on SVHN, and ResNet-56 for CIFAR-100. We evaluate the VAT method in the presence of RRAM variations that range from 0.1 to 0.5, ternary and 8-bit quantization, and different levels of structured sparsity generated using pruning method detailed in Section 4.4.1. The pruning group size is chosen equal to the crossbar size for maximum hardware inference performance. We utilize the same crossbar size of 256×256 with a 5-bit ADC at the periphery.

6.2 Pruning and Quantization

6.2.1 Effect of Pruning and Quantization on RRAM IMC

We follow the mapping as in [26]. In the pruning method, we set the group size in accordance with the number of rows of the crossbar. For example, for a crossbar of size 72×64 and kernel size of 3x3, we set the group size to be 8. Hence, we prune groups of 3×3×8 weights along the output feature dimension. Therefore, we are able to skip the mapping of 3×3×8 weights along the column dimension of the RRAM crossbar while maintaining high utilization. In this work, we set the group size to be 8 and the crossbar size as 72×64. Fig. 7 shows the energy-delay product for ResNet-20 on CIFAR-10 across different sparsity, quantization (8-bit and ternary), and crossbar sizes. A higher sparsity and lower quantization leads to lower EDP and vice-versa. At higher rates of sparsity, the EDP reduces exponentially across different grades of quantization, thus increasing the hardware performance.

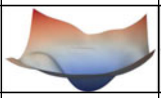
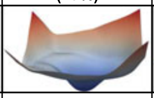
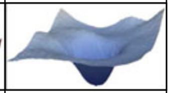
Model	Baseline (32-bit)	Pruning Only (29%)	Pruning (29%) & 8-bit
Landscape			
Roughness Score ($x10^{-3}$)	172	192	130
Accuracy (%)	92.35	92.16	92.62

Fig. 8. Loss landscape for floating-point 32-bit, pruned only (29%), and pruned and quantized (29% and 8-bit) variants of ResNet-20 on CIFAR-10. Model pruning results in a model with lower stability and accuracy, while quantization to a low-precision model improves the stability and the roughness score, leading to higher accuracy.

Fig. 8 shows the loss landscape, roughness score, and accuracy for the floating-point 32-bit (FP-32), pruning only (29%), and pruned and quantized (29% and 8-bit) versions of ResNet-20 on CIFAR-10. Pruning and quantization follow the methodology detailed in Sections 4.4.1 and 4.4.2, respectively. The pruned model has a roughness score higher than the FP-32 model, resulting in a rougher loss landscape, less stability, andlower accuracy. At the same time, the addition of quantization to the pruned model results in a reduced roughness score, making it more stable with a smoother loss landscape and a higher accuracy. Hence, we quantitatively establish through the roughness score and loss landscape that quantization helps improve the model stability and is a necessary step for reliable RRAM-based IMC acceleration of sparse DNNs.

6.3 System-Level IMC Analysis

6.3.1 Precision and Variation

The inherent variations with the RRAM device result in significant accuracy degradation. Section 1 details the effect of RRAM variation for ResNet-20 on CIFAR-10 with full precision and pruned and quantized models (8-bit and ternary). Through this, we establish that higher RRAM levels lead to higher variation and degradation in post-mapping accuracy.

Next, we evaluate the effect of ADC precision on post-mapping accuracy. Fig. 9 shows the total inference energy breakdown for ResNet-20 on CIFAR-10 at two ADC precisions, 8-bit and 5-bit. We divide the total energy into ADC, buffer, and other (accumulator, NoC, crossbar, ReLU, pooling, etc.) components. It can be seen that a higher ADC precision leads to higher energy dominated by the ADC and higher post-mapping accuracy. At the same time, a lower precision reduces the ADC component resulting in reduced total energy. At higher RRAM levels, the ADC cost reduces due to reduced crossbars and associated peripherals. Considering the dramatic design challenge and cost, a low-power and high throughput, and high-precision ADC may not be practical soon for RRAM IMC.

6.3.2 Roofline Model

In this section, we develop a roofline model that comprises of the number of RRAM levels, RRAM variation, stuck-at-faults, R_{off}/R_{on} ratio, and ADC precision. We evaluate the post-mapping accuracy for two DNNs, DenseNet-40 and ResNet-32 for the CIFAR-100 dataset, for our fabricated HfO_2 based RRAM device.

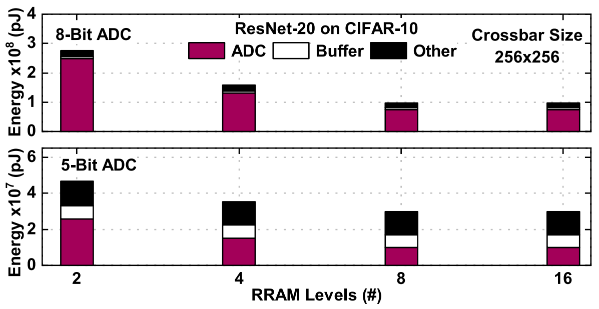


Fig. 9. ADC dominates the energy consumption, especially under high ADC precision. With higher RRAM levels, its portion reduces due to reduced number of crossbars and associated peripherals.

Fig. 10 shows the roofline model for the two DNNs, DenseNet-40 and ResNet-32, on the CIFAR-100 dataset. The black curve shows the post-mapping accuracy for different RRAM levels with (black) and without considering the ADC precision (grey). A higher number of RRAM levels leads to lower post-mapping accuracy and vice-versa. A similar trend is seen for both ResNet-32 and DenseNet-40. Next, we consider the ADC precision in the accuracy estimation and evaluate the post-mapping accuracy. The red curve shows the maximum achievable post-mapping accuracy with a 5-bit ADC. Hence, the RRAM-based IMC accuracy at lower RRAM levels is limited by the ADC precision, while at higher RRAM levels, the RRAM device limits the accuracy. Finally, for our 16-level 65nm 1T1R RRAM devices, only 4-6 levels are useful to achieve the best performance due to the ADC precision, RRAM variations, and algorithm limits.

6.4 Model Stability-Based Model Selection

In this section, we show the efficacy of the model stability-based model selection method. Here the training and testing RRAM variations are the same (σ) [10]. Table 4 shows the post-mapping accuracy of various DNNs for CIFAR-10 and CIFAR-100 datasets. We evaluate ResNet-20 and DenseNet-40 for CIFAR-10. For our 65nm RRAM device with a device variation (σ) of 0.3, ResNet-20 achieves 68.83% accuracy. A 50% reduction in RRAM variation (σ = 0.15) through process control results in 72.04% accuracy, a 4% increment. At the same time, DenseNet-40, which has higher model stability, achieves a higher accuracy of 72.34% at a σ of 0.3, a 0.3% and 4% improvement over ResNet-20 with 0.15 and 0.3 σ, respectively. For CIFAR-100 dataset, we evaluate ResNet-32 and

TABLE 4
Roughness Score and Post-Mapping Accuracy for Different DNN Models under RRAM Variation and a 5-Bit ADC

Model Size (Size)	Variation (σ)	Roughness Score (x10^{-3})	Accuracy (%)
CIFAR-10			
ResNet-20 (0.3M)	0.3*	278	68.83
	0.15		72.04
DenseNet-40 (0.2M)	0.3*	173	72.34
CIFAR-100			
ResNet-32 (0.5M)	0.3*	130	36.76
	0.15		43.41
DenseNet-40 (0.2M)	0.3*	121	44.68

A more stable model (lower roughness score) effectively improves the accuracy, more than that by reducing RRAM variation by 50% only.

DenseNet-40. For our 65nm RRAM device, ResNet-32 achieves 36.76% and 43.41% post-mapping accuracy at σ of 0.3 and 0.15, respectively. DenseNet-40 (more stable model) achieves 44.68% post-mapping accuracy at a σ of 0.3, a 1.27% improvement over ResNet-32 at of 0.15 (7.9% higher than ResNet-32 at σ of 0.3). We note that the more stable model has a smaller model size and attributes to improved hardware performance. The improved accuracy is attributed to the lower roughness score and higher model stability from the proposed loss landscape-based model selection. Through this, we establish that a loss landscape-based model selection achieves higher post-mapping accuracy than a 50% reduction in RRAM device variation through process control.

6.5 Model Stability-Based VAT

In this section, we detail the efficacy of the proposed model stability-based VAT method. Fig. 11 shows the loss landscape, roughness score, and post-mapping accuracy for ResNet-20 at 29% sparsity and 8-bit quantization for a testing variation (σ) of 0.1 on the CIFAR-10 dataset. The model is trained with different scales of variations (σ) from 0.1 to 0.3 to generate each of the VAT models. The most stable VAT model, σ equal to 0.15, has the lowest roughness score of 91×10^{-3}, resulting in a smoother loss landscape, higher model stability, and post-mapping accuracy. As the roughness score increases, the loss landscape becomes more rough, and the post-mapping accuracy reduces. Fig. 12 shows the detailed result. The optimal scale of training variation is different for each testing variation and is chosen based on the model

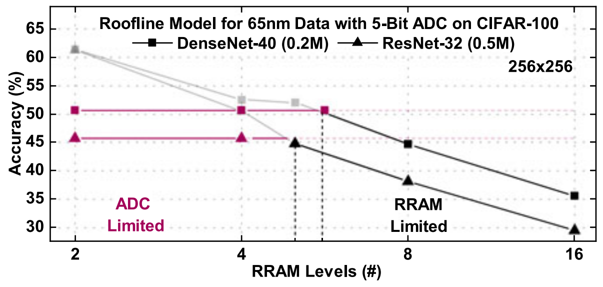


Fig. 10. ADC precision and RRAM device variations govern the maximum and realistic post-mapping accuracy. A more stable DNN model improves the realistic post-mapping accuracy.

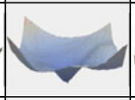
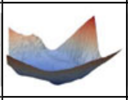
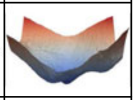
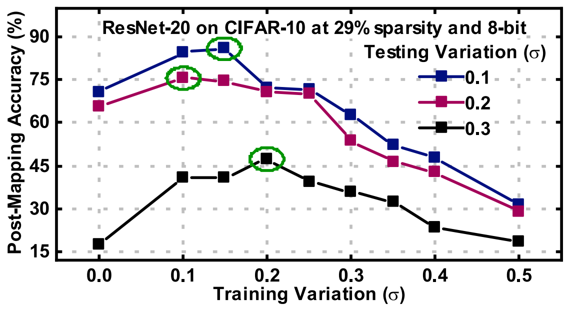
Training Variation (σ)	0.1	0.15	0.2	0.3
Landscape				
Roughness Score (x10^{-3})	94	91	120	139
Post-mapping Accuracy (%)	84.7	86.0	72.3	62.8

Fig. 11. Loss landscape for ResNet-20 on CIFAR-10 trained with different device variations at 29% sparsity and 8-bit quantization, tested at 0.1 variation. The optimal scale of training variation (σ) of 0.15 has the lowest roughness score and smoother loss landscape and hence, higher model stability and post-mapping accuracy.



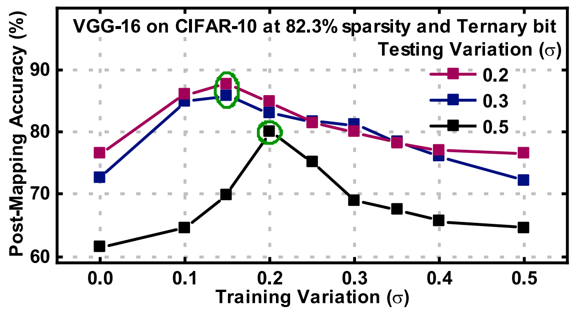
Conventional Method: Same Training and Testing Variation

Testing Variation (σ)	0.10	0.20	0.30
Optimal Training Variation (σ)	0.15	0.10	0.20
Accuracy Improvement over Conventional method (%)	1.3	5.1	11.5

Fig. 12. Post-mapping accuracy for ResNet-20 on CIFAR-10 at 29% sparsity and 8-bit precision for three different RRAM variations under test. The optimal scale of training variation (green circle) has the lowest roughness score and highest post-mapping accuracy.

stability. *Thus, the proposed method is also applicable to situations with unknown precise RRAM testing variation.*

We repeat the same experiment for VGG-16 on CIFAR-10 with ternary quantization and 83% sparsity as shown in Fig. 13. Table 5 shows the detailed results for VGG-16 on CIFAR-10 across the entire range of testing variations. Conventional methods refer to using the same scale of variation for both training and testing [9], [10]. In contrast, in this work, we show that an optimal scale for training variation results in higher model stability and post-mapping accuracy. Furthermore, at a higher range of testing variations, the proposed method provides greater improvement in post-mapping accuracy. Hence, the systematic model stability-based VAT method is effective in choosing the optimal VAT model at different precision and sparsity for best accuracy across a range of DNN models.



Conventional Method: Same Training and Testing Variation

Testing Variation (σ)	0.20	0.30	0.50
Optimal Training Variation (σ)	0.15	0.15	0.20
Roughness Score (x10^{-3}) (Conventional, Optimal)	(172, 147)	(584, 454)	(750, 665)
Accuracy Improvement over Conventional method (%)	2.9	4.6	15.4

Fig. 13. Post-mapping accuracy for VGG-16 on CIFAR-10 at 82.3% sparsity and ternary precision for three different RRAM testing variations (σ). The optimal model (green circle) has the lowest roughness score resulting in higher accuracy.

TABLE 5
Post-Mapping Accuracy and Improvement for VGG-16 on CIFAR-10 (82.3% Sparsity and Ternary Precision) Conventional Method: Same Training and Testing Variation

Testing Variation	Optimal Training Variation (σ)	Post-Mapping Accuracy (%)		Accuracy Improvement (%)
		Conventional [9, 10]	Optimal	
0.1	0.15	87.9	88.5	**0.6**
0.2	0.15	84.8	87.7	**2.9**
0.3	0.15	81.2	85.8	**4.6**
0.4	0.10	64.9	83.9	**19.0**
0.5	0.20	64.6	80.0	**15.4**

6.5.1 *Overall Results With Proposed VAT*

Table 6 shows the overall results across different models and datasets. All post-mapping accuracy is compared to that of conventional VAT, where the training and testing variations are the same. ResNet-20 on CIFAR-10 at 29% sparsity and 8-bit precision shows 11.5% improvement in post-mapping accuracy with the proposed method at 0.3 testing variation. At the same time, VGG-16 at 88.3% and 82.3% sparsity and ternary bit quantization achieve 2.2% and 19% improvement post-mapping accuracy for 0.2 and 0.4 testing variations (σ), respectively. We evaluate ResNet-32 on SVHN dataset at ternary quantization and two sparsity (48.4% and 71.5%) and achieve up to 11.2% improvement in post-mapping accuracy. Finally, for ResNet-56 for CIFAR-100, at 17.8% sparsity, 8-bit quantization, and testing variation of 0.15, we achieve a 21.1% improvement in post-mapping accuracy.

6.5.2 *Comparison With Other Work*

We compare the proposed model-stability-based VAT method with the state-of-the-art method as proposed in [9]. Table 7 shows the post-mapping accuracy for VGG-16 on CIFAR-10. The proposed method achieves a 5.1% higher average improvement (as defined in [9]) in post-mapping accuracy as compared to [9]. Furthermore, the proposed method provides an improvement in the presence of structured sparsity, which reduces the model stability due to the presence of more sensitive weights. Finally, the proposed method does not require precise prior knowledge of the testing variations (instead requires only the expected range), hence providing a more generic solution for reliable RRAM-based IMC acceleration

7 Related Work

7.1 RRAM-Based IMC Architectures

IMC-based hardware architectures have emerged as a promising alternative to conventional von-Neumann architectures. The crossbar-based IMC structure efficiently combines both memory access and analog-domain computation into a single unit for the acceleration of DNNs. RRAM-based IMC architectures provide a promising alternative to conventional von-Neumann architectures [3], [4], [5], [6], [7], [26], [30]. Authors of ISAAC [5] proposed an RRAM-based IMC architecture for DNN inference. The architecture utilizes a crossbar of size 128×128 to perform the multiply-and-accumulate

TABLE 6
Comprehensive Results Using the Proposed Model Stability-Based VAT Method Conventional Method: Same Training and Testing Variation

Dataset	Network (Size)	Quantization	Sparsity (%)	Testing Variation	Optimal Training Variation (σ)	Post-Mapping Accuracy (%)		Accuracy Improvement (%)
						Conventional [9, 10]	Optimal	
CIFAR-10	ResNet-20 (0.27M)	8-Bit	29.0	0.30	0.20	35.7	47.2	**11.5**
	VGG-16 (15M)	Ternary	82.3	0.40	0.10	64.9	83.9	**19.0**
			88.3	0.20	0.10	84.7	86.9	**2.2**
SVHN	ResNet-32 (0.45M)	Ternary	48.4	0.15	0.20	91.0	91.6	**0.6**
			71.5	0.30	0.15	75.1	86.3	**11.2**
CIFAR-100	ResNet-56 (0.85M)	8-Bit	17.8	0.15	0.10	30.5	51.6	**21.1**

(MAC) operations. A parallel read-out method is utilized to accelerate the MAC computations in the analog domain. In addition, a digital-to-analog converter (DAC) and an ADC are employed to switch between digital and analog domains. The array of RRAM-based IMC tiles is interconnected by an NoC (c-mesh). In contrast, [3] utilized spike-based computation to perform MAC operations in the time domain. Such an architecture does not need DAC and ADC units. An atomic computation-based RRAM IMC architecture for both training and inference of DNNs is proposed in [31]. Authors in [30] proposed a systolic array-based RRAM IMC design for DNN inference. Authors in [4] proposed a methodology to optimize the IMC utilization, area and energy by utilizing a heterogeneous IMC structure and a custom NoC router-to-tile mapping and scheduling. But, prior works do not focus on the non-idealities associated with an RRAM-based IMC. To address this, in this work, utilizing model stability as a metric, we propose a novel model selection and VAT method to improve the post mapping accuracy of RRAM-based IMC architecture. The proposed methods incorporate the effect of architectural and circuit properties into the accuracy estimation while ensuring the best performance.

7.2 Pruning and Quantization

Pruning has been an effective method to reduce the DNN model size. Pruning can be classified into element-wise pruning, kernel-wise or channel-wise pruning for structured sparsity, and group-wise pruning using the underlying hardware execution flow. Element-wise pruning [32], [33] prunes weights of DNN in a random manner, while structured pruning techniques [34], [35] produce a structured sparse DNN. Such sparsity is more hardware-friendly to achieve better performance as compared to random pruning. Other works have focused on crossbar-aware pruning and FPGA-aware pruning by incorporating the crossbar structure [36] and the underlying FPGA execution flow [37]. Simultaneously, quantization provides an efficient method to reduce the bit precision of the weights and activations within the DNN. A quantized DNN model results in improved hardware performance while reducing the overall inference accuracy. Prior works have focused on weight and activation quantization using uniform and non-uniform methods [17], [18], [38].

TABLE 7
Post-Mapping Accuracy of VGG-16 on CIFAR-10

Method	Sparsity (%)	Quantization	Accuracy (%)	
			Baseline	Post-Mapping (Average)
DFP+DVA [9]	-	8-Bit	93.85	80.1
Ours	**83.2**	**Ternary**		**85.2**

In contrast, in this work, we utilize group-wise structured sparsity along the output channel dimension, where the size of the group is determined by the number of rows in the crossbar. Such a pruning requires no additional hardware overhead or special mapping techniques. Furthermore, we combine pruning with quantization to generate a structured sparse low-precision DNN model. We utilize both uniform (4-bits and higher) [18] and ternary quantization [17]. Furthermore, we analyze the effect of sparsity and quantization on the DNN model by utilizing the model stability and the loss landscape visualization. Through our results, we show that pruning reduces the model stability while quantization improves the stability. Finally, we analyze the effect of non-idealities associated with RRAM-based IMC architecture in the presence of lower precision and sparsity for the DNN.

7.3 Mitigation of Post-Mapping Accuracy Loss

Several VAT methods have been proposed to mitigate the post-mapping accuracy loss due to RRAM variations. Closed-Loop-on-Device (CLD) and Open-Loop-off-Device (OLD) perform iterative read-verify-write (R-V-W) operations at the RRAM device level until the resistance converges to the desired value [39], [40]. Other approaches, such as [41], involve VAT based on known device variation (σ) characterized from devices, while [9] combines VAT with dynamic precision quantization to mitigate the post-mapping accuracy loss. These approaches partially recover the accuracy but fail to consider the effect of sparsity and quantization on the DNN model. Furthermore, these works do not provide a systematic metric to provide a reliable DNN model. In addition, these methods assume a known RRAM device variation and utilize the same scale of variation in training and testing. Hence, precise variations models need to be extracted by mapping the weights onto the fabricated RRAM device.

To address these drawbacks, in this work, we propose model stability as a metric for reliable RRAM-based in-memory computing. Utilizing the roughness score and loss landscape, as defined in Section 2, we show that a lower roughness score leads to higher model stability and higher accuracy. Second, we propose a model stability-based model selection where the choice of the model is driven by the inherent model stability in the presence of the variations within the RRAM-based IMC architecture. Finally, based on model stability, we provide a systematic VAT method that searches for an optimal scale of variation during training for the best accuracy. The proposed VAT method is agnostic to the actual variations within the RRAM device and provides a generic solution for reliable RRAM-based IMC acceleration.

8 Conclusion

In this work, we explore the model stability of DNNs as a metric for reliable RRAM-based in-memory acceleration. Utilizing the loss landscape and roughness score, we show that a more stable model has a lower roughness score, a smoother loss landscape, and higher accuracy under variations. To provide realistic evaluation, we measured statistical variations from a 65nm 1T1R RRAM test chip and integrated them into a cross-layer benchmark tool to access model accuracy and other performance metrics under variations. Based on the model stability of DNNs, we propose two methods to achieve reliable RRAM-based in-memory acceleration. First, a novel model stability-based model selection that effectively tolerates RRAM device variations and achieves higher accuracy than that with 50% lower RRAM variations for both CIFAR-10 and CIFAR-100 datasets. Second, we propose a variation-aware training (VAT) method to mitigate the post-mapping accuracy loss in sparse and quantized DNNs. We conclude that quantization improves the stability under variations, leading to higher accuracy, but pruning reduces the model stability. The proposed VAT method searches for the most stable model to mitigate the post-mapping accuracy loss without pre-knowledge of testing RRAM variations and no re-training during mapping. Experimental evaluation shows up to 19%, 21%, and 11% improvement in post-mapping accuracy at different sparsity, quantization, and device variations on CIFAR-10, CIFAR-100, and SVHN datasets, respectively.

References

[1] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, “Eyeriss V2: A flexible accelerator for emerging deep neural networks on mobile devices,” *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 9, no. 2, pp. 292–308, Jun. 2019.

[2] M. Horowitz, “1.1 computing’s energy problem (and what we can do about it),” in *Proc. IEEE Int. Solid-Statist. Circuits Conf. Dig. Tech. Papers*, 2014, pp. 10–14.

[3] L. Song, X. Qian, H. Li, and Y. Chen, “PipeLayer: A pipelined reram-based accelerator for deep learning,” in *Proc. IEEE Int. Symp. High Perform. Comput. Architecture*, 2017, pp. 541–552.

[4] G. Krishnan *et al.*, “Interconnect-aware area and energy optimization for in-memory acceleration of DNNs,” *IEEE Des. Test*, vol. 37, no. 6, pp. 79–87, Dec. 2020.

[5] A. Shafiee *et al.*, “ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars,” in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Architecture*, 2016, pp. 14–26.

[6] S. K. Mandal *et al.*, “A latency-optimized reconfigurable NoC for in-memory acceleration of DNNs,” *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 10, no. 3, pp. 362–375, Mar. 2020.

[7] M. Imani, S. Gupta, Y. Kim, and T. Rosing, “FloatPIM: In-memory acceleration of deep neural network training with high precision,” in *Proc. ACM/IEEE 46th Annu. Int. Symp. Comput. Architecture*, 2019, pp. 802–815.

[8] L. Yang, Z. He, and D. Fan, “Harmonious coexistence of structured weight pruning and ternarization for deep neural networks,” *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, pp. 6623–6630, 2020.

[9] Y. Long, X. She, and S. Mukhopadhyay, “Design of reliable DNN accelerator with un-reliable ReRAM,” in *Proc. Des., Automat. Test Eur. Conf. Exhib.*, 2019, pp. 1769–1774.

[10] G. Charan *et al.*, “Accurate inference with inaccurate RRAM devices: Statistical data, model transfer, and on-line adaptation,” in *Proc. 57th ACM/IEEE Des. Automat. Conf.*, 2020, pp. 1–6.

[11] C.-Y. Chen *et al.*, “RRAM defect modeling and failure analysis based on march test and a novel squeeze-search scheme,” *IEEE Trans. Comput.*, vol. 64, no. 1, pp. 180–190, Jan. 2015.

[12] I. Chakraborty, M. F. Ali, D. E. Kim, A. Ankit, and K. Roy, “GENIEx: A generalized approach to emulating non-ideality in memristive xbars using neural networks,” in *Proc. 57th ACM/IEEE Des. Automat. Conf.*, 2020, pp. 1–6.

[13] Z. He *et al.*, “Noise injection adaption: End-to-end reram crossbar non-ideal effect adaption for neural network mapping,” in *Proc. 56th ACM/IEEE Des. Automat. Conf.*, 2019, pp. 1–6.

[14] C. Ma *et al.*, “Go unary: A novel synapse coding and mapping scheme for reliable ReRAM-based neuromorphic computing,” in *Proc. IEEE Des. Automat. Test Eur. Conf. Exhib.*, 2020, pp. 1432–1437.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[16] X. Du *et al.*, “Noise-based selection of robust inherited model for accurate continual learning,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 983–988.

[17] Z. He, B. Gong, and D. Fan, “Optimize deep convolutional neural network with ternarized weights and high accuracy,” in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2019, pp. 913–921.

[18] S. Zhou *et al.*, “DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients,” 2016, *arXiv:1606.06160*.

[19] S. Hochreiter and J. Schmidhuber, “Flat minima,” *Neural Computation*, vol. 9, no. 1, pp. 1–42, 1997.

[20] N. S. Keskar, J. Nocedal, P. T. P. Tang, D. Mudigere, and M. Smelyanskiy, “On large-batch training for deep learning: Generalization gap and sharp minima,” in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[21] H. Li *et al.*, “Visualizing the loss landscape of neural nets,” in *Proc. Adv. Neural Informat. Process. Syst.*, 2018, pp. 6391–6401.

[22] X. Peng, S. Huang, Y. Luo, X. Sun, and S. Yu, “DNN+ NeuroSim: An end-to-end benchmarking framework for compute-in-memory accelerators with versatile device technologies,” in *Proc. IEEE Int. Electron Devices Meeting*, 2019, pp. 32.5.1–32.5.4.

[23] W. Zhang *et al.*, “Design guidelines of RRAM based neural-processing-unit: A joint device-circuit-algorithm analysis,” in *Proc. 56th ACM/IEEE Des. Automat. Conf.*, 2019, pp. 1–6.

[24] Z. Zhu *et al.*, “Mnsim 2.0: A behavior-level modeling tool for memristor-based neuromorphic computing systems,” in *Proc. Great Lakes Symp. VLSI*, 2020, pp. 83–88.

[25] G. Krishnan *et al.*, “Robust RRAM-based in-memory computing in light of model stability,” in *Proc. IEEE Int. Rel. Phys. Symp.*, 2021, pp. 1–5.

[26] G. Krishnanetal, “SIAM: Chiplet-based scalable in-memory acceleration with mesh for deep neural networks,” *ACM Trans. Embedded Comput. Syst.*, vol. 20, no. 5s, pp. 1–24, 2021.

[27] L. Hou, R. Zhang, and J. T. Kwok, “Analysis of quantized models,” in *Proc. Int. Conf. Learn. Representations*, 2018.

[28] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein, “Training quantized nets: A deeper understanding,” *Proc. Adv. Neural Informat. Process. Syst.*, vol. 30, 2017.

[29] J. Wang, Q. Cai, Q. Chang, and J. M. Zurada, “Convergence analyses on sparse feedforward neural networks via group lasso regularization,” *Informat. Sci.*, vol. 381, pp. 250–269, 2017.

[30] M. Mao *et al.*, “MAX2: An ReRAM-based neural network accelerator that maximizes data reuse and area utilization,” *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 9, no. 2, pp. 398–410, Jun. 2019.

[31] X. Qiao, X. Cao, H. Yang, L. Song, and H. Li, “AtomLayer: A universal ReRAM-based CNN accelerator with atomic layer computation,” in *Proc. 55th Annu. Des. Automat. Conf.*, 2018, pp. 1–6.

[32] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in *Proc. Adv. Neural Informat. Process. Syst.*, 2015, pp. 1135–1143.

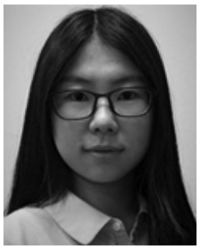
[33] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” in *Proc. Int. Conf. Learn. Representations*, 2019.
[34] Z. Liu *et al.*, “Learning efficient convolutional networks through network slimming,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2755–2763.
[35] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, “Learning structured sparsity in deep neural networks,” in *Proc. Adv. Neural Informat. Process. Syst.*, 2016, pp. 2082–2090.
[36] L. Liang *et al.*, “Crossbar-aware neural network pruning,” *IEEE Access*, vol. 6, pp. 58324–58337, 2018.
[37] G. Krishnan, Y. Ma, and Y. Cao, “Small-world-based structural pruning for efficient FPGA inference of deep neural networks,” in *Proc. IEEE 15th Int. Conf. Solid-Statist. Integr. Circuit Technol.*, 2020, pp. 1–5.
[38] M. Lin *et al.*, “Rotated binary neural network,” in *Proc. Adv. Neural Informat. Process. Syst.*, 2020, pp. 7474–7485.
[39] M. Hu, H. Li, Y. Chen, Q. Wu, and G. S. Rose, “BSB training scheme implementation on memristor-based circuit,” in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, 2013, pp. 80–87.
[40] B. Liu *et al.*, “Reduction and IR-drop compensations techniques for reliable neuromorphic computing systems,” in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2014, pp. 63–70.
[41] L. Chen *et al.*, “Accelerator-friendly neural-network training: Learning variations and defects in RRAM crossbar,” in *Proc. Des. Automat. Test Eur. Conf. Exhib.*, 2017, pp. 19–24.

Gokul Krishnan (Student Member, IEEE) received the BTech degree in electronics and communication engineering from Govt. Model Engineering College, Kochi, India, in 2016. He is currently working towards the PhD degree in electrical engineering with Arizona State University, Tempe, Arizona. His current research interests include neuromorphic hardware and microarchitecture design for deep learning, chiplet-based hardware architecture for deep learning acceleration, joint algorithm-architecture design for learning on a chip, model compression of DNNs, and performance modeling for CMOS and post-CMOS-based hardware architectures. He is the recipient of the Richard Newton Fellowship at DAC 2020, the Joseph A. Barkson Fellowship for 2020-21, and the GPSA Outstanding Research Award winner in 2021.

Li Yang (Student Member, IEEE) received the the BS degree in control engineering from Northeastern University, Qinhuangdao, China, in 2014, and the MS degree in electrical engineering from the University of Central Florida, Orlando, Florida, in 2018. He is currently working toward the PhD degree with Arizona State University, advised by Dr. Deliang Fan. His research interests include primarily focus on hardware-aware deep learning networks compression and optimization, and FPGA-based Deep Neural Network Accelerator design. During his Ph.D. study, he has been the author of ten publications on IEEE/ACM top-tier journal and conference (e.g., CVPR, AAAI, DAC, ASP-DAC, ISVLSI, GLSVLSI etc) in the aforementioned areas.

Jingbo Sun (Student Member, IEEE) received the BE degree in communication engineering from the Tianjin University (TJU), Tianjin, China, in 2008, and the MS degree in computer science from the University of Southern California (USC), in 2020. He is currently working toward the PhD degree in computer engineering from Arizona State University. Prior to that, he was a senior hardware engineer with Oracle, Santa Clara, where he worked on the physical design of various SPARC Microprocessors. His research interests include machine learning algorithms for dynamic systems, such as novelty detection, continual learning, and graph-based perception.

Jubin Hazra (Student Member, IEEE) received the master’s degree from the University of Southern California, Los Angeles, in 2014, and the engineer’s degree in electrical engineering from the University of Southern California, Los Angeles in 2016. His research was mostly based on design, fabrication and characterization of III-V compound semiconductor electronic and optoelectronic devices on Si substrates. He also has one year working experience in ASML US Inc in AZ, US as a senior optics engineer prior to joining SUNY Poly as a full time PhD student in Jan 2018. He joined Cady lab in Aug 2018 and currently his research is focused on hardware implementation of neuromorphic chips using memristor-CMOS hybrid technology and improving the ferroelectric performance in HZO films. He has authored several papers in reputed journals and conference proceedings.



Xiaocong Du (Member, IEEE) received the BS degree in control engineering from Shandong University, Jinan, China, in 2014, and the MS degree in electrical and computer engineering from the University of Pittsburgh, Pittsburgh, Pennsylvania, in 2016, and the PhD degree from the Electrical Engineering Department, Arizona State University, Tempe, Arizona, in 2020. Her research interests include efficient algorithm and hardware co-design for deep learning, neural architecture search, continual learning, and neuromorphic computing. In 2020, she worked as a research intern at Facebook, Menlo Park, CA. Now, she is working as Machine Learning Engineer in Facebook, Inc, Menlo Park.

Maximilian Liehr (Member, IEEE) received the BA and MEng degrees from Rensselaer Polytechnic Institute in Troy, New York. He is currently working toward the PhD degree in nanoscale engineering from the Colleges of Nanoscale Science and Engineering, SUNY Polytechnic Institute. He has active research interests include development of non-volatile nanoelectronics including resistive random access memory (ReRAM) devices, and neuromorphic computing.

Zheng Li (Member, IEEE) received the BS degree in electronics and information engineering from Beihang University, Beijing, China, in 2014, and the MS degree in electrical and computer engineering from the University of Pittsburgh, Pittsburgh, Pennsylvania, in 2017, and the PhD degree in computer engineering from Arizona State University, Tempe, Arizona, in 2020. He worked as a summer intern in Machine Learning with MobaiTech, Inc, in 2018 and 2019, and with Facebook in 2020. His current research interests include algorithm design and optimization for computer vision tasks, such as object detection, and autonomous driving. Currently, he is a research scientist with Facebook, Inc, Menlo Park, CA.

Karsten Beckmann (Member, IEEE) received the BSc and MSc degrees in electrical engineering and information technology form the Technical University of Darmstadt, Germany, in 2011 and 2013, respectively, and the PhD degree in nanoscale engineering from the Colleges of Nanoscale Science and Engineering (CNSE), University at Albany (now SUNY Polytechnic Institute). He is currently a senior process integration engineer with NY CREATES / SUNY Poly working on emerging memory solution with a focus on resistive devices and selector technology. His strong background spans interdisciplinary fields combining Material Science, Nanofabrication and Electrical Engineering with a focus in electrical testing (DC, RF and automated testing), module integration into a 65 nm technology and materials development (ReRAM, Selector, Electrodes).

Rajiv V. Joshi (Fellow, IEEE) received the BTech degree from the Indian Institute of Technology, Bombay, India, the MS degree from the Massachusetts Institute of Technology, and the Dr. Eng. Sc. degree from Columbia University. He is a research staff member and key technical lead with T. J. Watson research center, IBM. He has led successfully predictive failure analytic techniques for yield prediction and the technology-driven SRAM with IBM Server Group. He developed and commercialized novel memory designs which are universally accepted. He received three Outstanding Technical Achievement (OTAs), three highest Corporate Patent Portfolio awards for licensing contributions, holds 60 invention plateaus and has more than 260 US patents and more than 400 including international patents. He has authored and co-authored more than 210 papers. He received NY IP Law association "Inventor of the year" Award in Feb 2020, the IEEE Daniel Noble Award for 2018 and Industrial Pioneer Award, the Best Editor Award from IEEE TVLSI journal, recipient of 2015 BMM Award, and inducted into New Jersey Inventor Hall of Fame in 2014.

Nathaniel C. Cady (Member, IEEE) received the BA and PhD degrees from Cornell University, in Ithaca, New York. He is currently an empire innovation professor in nanobioscience with the Colleges of Nanoscale Science and Engineering, SUNY Polytechnic Institute. He has active research interests include development of novel biosensor technologies and biology-inspired nanoelectronics, including novel hardware for neuromorphic computing.

Deliang Fan (Member, IEEE) is currently an assistant professor with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, Arizona. His primary research interests include energy efficient and high performance Big Data processing-in-memory circuit, architecture and algorithm, with applications in deep neural network, data encryption, graph processing and bioinformatics acceleration-in-memory system, hardware-aware deep learning optimization, brain-inspired (Neuromorphic) computing, AI security. He has authored and co-authored more than 130 peer-reviewed international journal/conference papers in above area. He is the receipt of best paper award of 2019 ACM Great Lakes Symposium on VLSI, 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), and 2017 IEEE ISVLSI. He is also the technical area chair of DAC 2021, GLSVLSI 2019/2020/2021, ISQED 2019/2020/2021, and the financial chair of ISVLSI 2019. Please refer to https://dfan.engineering.asu.edu/ for more details.

Yu Cao (Fellow, IEEE) received the BS degree in physics from Peking University, in 1996, the MA degree in biophysics and PhD degree in electrical engineering from the University of California, Berkeley, in 1999 and 2002, respectively. He is now a professor in electrical engineering with Arizona State University, Tempe, Arizona. He has published numerous articles and two books on nano-CMOS modeling and physical design. His research interests include neural-inspired computing, hardware design for on-chip learning, and reliable integration of nanoelectronics. He served as associate editor of *IEEE Transactions on CAD*, and on the technical program committee of many conferences.