MDM: Manhattan Distance Mapping of DNN Weights for Parasitic-Resistance-Resilient Memristive Crossbars

Matheus Farias
Harvard University
matheusfarias@g.harvard.edu

Wanghley Martins
Duke University
wanghley.soares.martins@duke.edu

H. T. Kung
Harvard University
kung@harvard.edu

***Abstract*—*Manhattan Distance Mapping* (MDM) is a post-training deep neural network (DNN) weight mapping technique for memristive bit-sliced compute-in-memory (CIM) crossbars that reduces parasitic resistance (PR) nonidealities.**

PR limits crossbar efficiency by mapping DNN matrices into small crossbar tiles, reducing CIM-based speedup. Each crossbar executes one tile, requiring digital synchronization before the next layer. At this granularity, designers either deploy many small crossbars in parallel or reuse a few sequentially—both increasing analog-to-digital conversions, latency, I/O pressure, and chip area.

MDM alleviates PR effects by optimizing active-memristor placement. Exploiting bit-level structured sparsity, it feeds activations from the denser low-order side and reorders rows according to the Manhattan distance, relocating active cells toward regions less affected by PR and thus lowering the nonideality factor (NF).

Applied to DNN models on ImageNet-1k, MDM reduces NF by up to 46% and improves accuracy under analog distortion by an average of 3.6% in ResNets. Overall, it provides a lightweight, spatially informed method for scaling CIM DNN accelerators.

***Index Terms*—memristive crossbars, compute-in-memory, crossbar nonidealities, parasitic resistance.**

I. Introduction

Compute-in-memory (CIM) architectures integrate storage and computation within the same physical fabric, offering energy-efficient deep neural network (DNN) acceleration by reducing data movements [1]–[4]. However, their scalability remains limited by nonidealities—such as sneak paths [5]–[8], process-voltage-temperature variations [9], [10], stuck-at faults [11]–[14], conductance drift [15], [16], and parasitic resistance (PR) [17]–[19]—which degrade inference accuracy and restrict computational parallelism in large-scale workloads [20], [21].

PR is a key scalability bottleneck in CIM accelerators, caused by resistive interconnects within crossbars. We hypothesize that the resulting voltage drops grow proportionally with the Manhattan distance from the I/O rails—an effect we term *the Manhattan Hypothesis*. This model enables analytical estimation of PR impact without requiring circuit-level simulations.

DNN weights are typically mapped across crossbar rows, with each column representing a fractional bit [22]–[25]. Because weights follow a bell-shaped distribution centered near zero [22], [23], [26]–[28], high-order columns that encode large magnitudes are sparse, while lower-order columns are more frequently active. This structured imbalance drives current through deeper paths, leading to amplified PR effects.

This spatial nonideality constrains crossbar size. Large arrays amplify PR deviations, forcing DNN partitioning into smaller crossbar tiles to preserve accuracy. However, smaller tiles demand additional digital synchronization and I/O bandwidth between computing phases, mitigating CIM intra-parallelism. Consequently, PR simultaneously degrades model accuracy and undermines system-level throughput, posing a fundamental obstacle to scaling CIM-based DNN accelerators.

To address this limitation, we propose the *Manhattan Distance Mapping* (MDM) algorithm, a post-training spatial remapping strategy that reduces PR distortion without altering crossbar computation. MDM operates in three stages. First, it reverses the dataflow so that denser, lower-order bit regions—where active memristors are concentrated—align with shorter conduction paths, reducing cumulative voltage drops. Second, it assigns each row a Manhattan-based score that quantifies the distance of its active cells from the I/O rails, reflecting their relative exposure to parasitic effects. Finally, rows are reordered in ascending order of this score, relocating dense regions toward areas less affected by resistance buildup. This spatial reorganization reduces the nonideality factor (NF)—the deviation of the measured output from its ideal value—while preserving all arithmetic semantics, requiring neither retraining nor hardware modification, and integrating seamlessly into existing deployments [22], [23]. See Figure 1 for a summary of the approach.

The main contributions of this paper are:

- A theoretical foundation for MDM, built upon (1) the *Manhattan Hypothesis*, which shows that voltage drops accumulate proportionally to the Manhattan distance from the I/O rails, and (2) a mathematical proof of structured bit-level sparsity in DNN weight distributions;
- A post-training weight mapping that reverses dataflow and reorders rows to place active cells in regions less affected by PR accumulation, requiring no retraining nor hardware modification;
- A framework that models PR by injecting spatially dependent noise into DNN weights, enabling analog distortion assessment on PyTorch models.

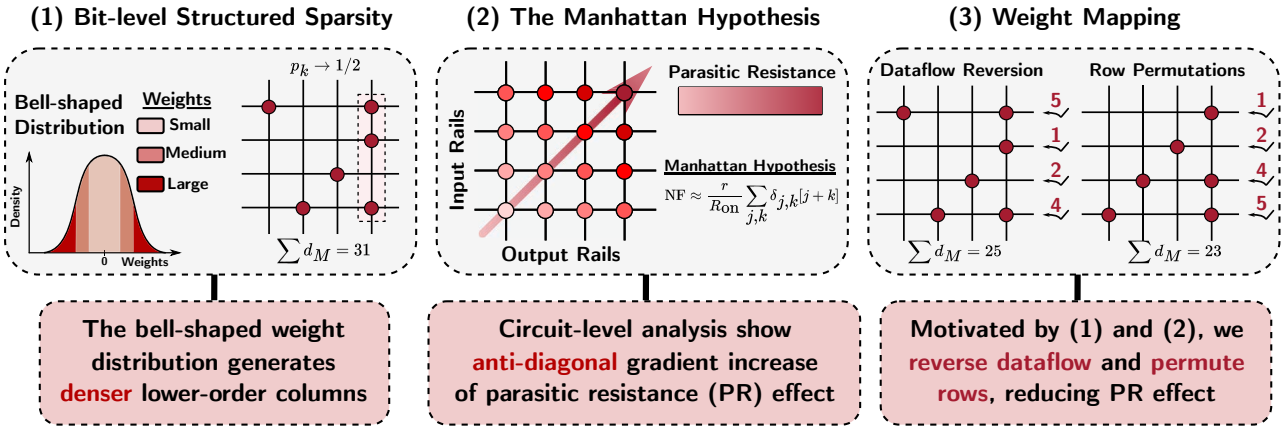

Fig. 1: Summary of the Manhattan Distance Mapping (MDM).

II. Background

A. Memristive Crossbar Assumptions

We adopt bit-sliced crossbars [22], [23], where each row encodes a weight and columns represent power-of-two scaling factors. Higher-order columns near the inputs correspond to larger factors (e.g. $2^0, 2^{-1}, 2^{-2}$), while lower-order columns farther away encode smaller ones (e.g. $2^{-5}, 2^{-6}, 2^{-7}$). For a 128×128 crossbar with 16 multipliers, each row stores eight different weight values (since $128/16 = 8$).

This hierarchy produces structured sparsity according to the bell-shaped distribution of DNN weights (see Section III-A).

B. Nonideality Measurement

Crossbar nonidealities are quantified by the NF [29]–[31],

$$\mathrm{NF} = \left| \frac{\Delta i}{i_0} \right|, \tag{1}$$

where i_0 is the expected output and Δi is the amount of current that was deviated due to nonidealities.

Each cell can be identified by its position (j, k) corresponding to its row and column indices seeing from the I/O interface. The Manhattan distance, $d_{\mathrm{M}}(j, k)$, of a cell is defined as the sum of its horizontal and vertical distances from the I/O rails,

$$d_{\mathrm{M}}(j, k) = j + k. \tag{2}$$

As current propagates along the resistive mesh, voltage drops accumulate with increasing distance from the I/O rails, causing farther cells to contribute less accurately to the overall output. Section III-B uses Kirchhoff's law to hypothesize that the NF grows proportionally to the Manhattan distance of active cells.

III. Theoretical Framework

This section proves that (1) bit-sliced crossbars exhibits a structured bit-level sparsity pattern, and propose that (2) the NF scales with the Manhattan distance from the I/O rails—two properties that underpin the MDM method.

A. Bit-level Structured Sparsity

To characterize the bit-level distribution in a (J, K) crossbar, we apply Theorem 1, derived from the DNN bell-shaped weight distribution [22], [23], [26]–[28], [32]. Each weight w_j is mapped across K fractional-bit columns as $w_j = \sum_{k \le K} b_{j,k}(w_j) 2^{-k}$, where lower-order bits exhibit higher activation probability $(b_k = 1)$[1], yielding denser columns.

Theorem 1. *Let W be a nonnegative random variable with probability density function $f : [0, \infty[\to [0, \infty[$ such that:*

1) *f is continuous on $[0, \infty[$ and strictly decreasing on $]0, \infty[$;*
2) *$f(0) < \infty$ and $\lim_{w \to \infty} f(w) = 0$.*

Let

$$p_k := \mathbb{P}(b_k = 1) = \int_0^\infty f(w) b_k(w) dw,$$

where $\mathbb{P}(b_k = 1)$ is the probability of $b_k = 1$. Then

$$\boxed{\left| p_k - \tfrac{1}{2} \right| \le \frac{1}{2^{2+k}} f(0).}$$

In particular, $p_k < 1/2$ for every k and $p_k \to 1/2$ as $k \to \infty$.

Proof. For $k \le K$, set $L := 2^{-k}$ and define the k-th fractional-bit indicator

$$b_k(w) = \begin{cases} 0, & w \in [mL,\ mL + \frac{L}{2}[, \\ 1, & w \in [mL + \frac{L}{2},\ (m+1)L[, \end{cases} \qquad m = 0, 1, 2, \ldots$$

Let

$$\Delta_k := \mathbb{P}(b_k = 0) - \mathbb{P}(b_k = 1). \tag{3}$$

Then,

$$\Delta_k = \sum_{m=0}^{\infty} \int_0^{L/2} \left[f(mL + u) - f(mL + u + L/2) \right] du. \tag{4}$$

By the Fundamental Theorem of Calculus,

$$f(mL+u) - f(mL+L/2+u) = -\int_0^{L/2} f'(mL+u+\theta)\, d\theta. \tag{5}$$

[1]We supress the row index when the statement is row-independent.

Changing variables to $s = u + \theta \in [0, L]$ gives

$$\Delta_k = \sum_{m=0}^{\infty} \int_0^L \big[-f'(mL + s) \big] A_L(s)\, ds, \tag{6}$$

with $A_L(s) := \min\{s, L - s\}$ on $[0, L]$. Since $A_L(s) \le L/2$,

$$\Delta_k \le \frac{L}{2} \sum_{m=0}^{\infty} \int_0^L \big[-f'(mL + s) \big]\, ds \tag{7}$$

using the Fundamental Theorem of Calculus again

$$\Delta_k \le \frac{L}{2} \sum_{m=0}^{\infty} \big[f(mL) - f((m+1)L) \big]. \tag{8}$$

The series telescopes and $f((m+1)L) \to 0$ (statement 2 of the theorem), hence

$$\Delta_k \le \frac{L}{2} f(0). \tag{9}$$

Noting that $p_k = \frac{1}{2}(1 - \Delta_k)$ and $L = 2^{-k}$, we obtain

$$\big| p_k - \tfrac{1}{2} \big| \le \frac{1}{2^{2+k}} f(0). \tag{10}$$

Moreover, $\Delta_k > 0$ because f is strictly decreasing on sets of positive measure (statement 1 of the theorem), hence $p_k < \frac{1}{2}$. Finally, since $L = 2^{-k} \to 0$, the bound forces $p_k \to \frac{1}{2}$. □

Theorem 1 motivates dataflow reversion by injecting inputs from the denser side to minimize PR along conductive paths.

B. The Manhattan Hypothesis

We consider a (J, K) crossbar whose interconnects have parasitic resistance r, and where each active cell at (j, k) exhibits resistance R_{on}. The array is driven from V_{in} along the rows and sensed at the grounded column outputs.

A single active memristor ℓ cells farther from the input rail satisfies the Kirchhoff's law:

$$\frac{V - V_{\text{in}}}{\ell r} + \frac{V}{R_{\text{on}}} = 0. \tag{11}$$

The memristor current, including PR effects, is

$$i = i_0 + \Delta i = \frac{V}{R_{\text{on}}}, \tag{12}$$

where $i_0 = V_0 / R_{\text{on}}$ is the ideal current (for $r = 0$).

Solving Equation (11) for V under the practical assumption $\ell r \ll R_{\text{on}}$ gives the first-order approximation

$$V \approx V_0 \left[1 - \frac{\ell r}{R_{\text{on}}} \right]. \tag{13}$$

Substituting Equation (13) into (12) and normalizing by i_0:

$$\text{NF} = \left| \frac{\Delta i}{i_0} \right| = \frac{|V - V_0|}{V_0} \approx \ell \frac{r}{R_{\text{on}}}. \tag{14}$$

Equation (14) shows that the deviation increases linearly with the distance between the device and its I/O rail. For an active memristor located j segments from the input rail and k segments from the output rail, the combined contribution is

$$\boxed{\text{NF} \approx \frac{r}{R_{\text{on}}} [j + k]}. \tag{15}$$

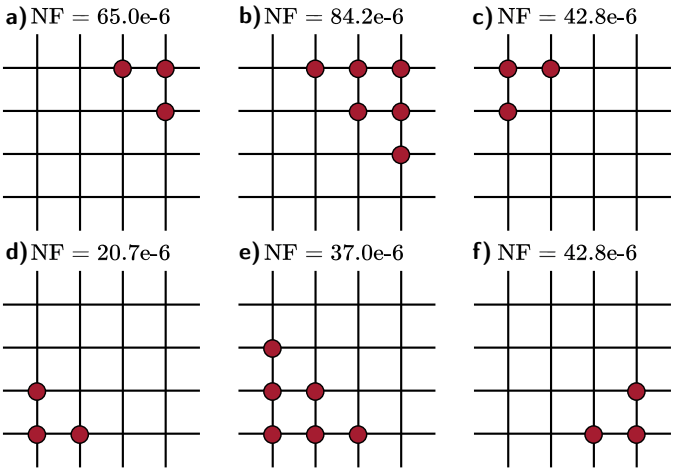

Fig. 2: Circuit-level simulations in SPICE shows anti-diagonal symmetry for $r = 2.5\ \Omega$, $R_{\text{on}} = 300\ \text{k}\Omega$, and $R_{\text{off}} = 3\ \text{M}\Omega$ (values within range suggested in the literature [31], [33], [34].)

Extending for multiple active memristors and sensing the current at each column end, we obtain the Manhattan Hypothesis

$$\boxed{\text{NF} \approx \frac{r}{R_{\text{on}}} \sum_{j,k} \delta_{j,k} [j + k]} \quad \text{(Manhattan Hypothesis)} \tag{16}$$

where $\delta_{j,k} = 1$ if the crosspoint (j, k) is active and 0 otherwise.

This result shows that the NF scales proportionally with the aggregate Manhattan distance of active cells, following a gradient of increase from the bottom-left to the top-right (anti-diagonal) of the array. Consequently, crossbars exhibit identical NF values under anti-diagonal symmetric configurations—a behavior corroborated by SPICE circuit-level simulations (see Figure 2). This linear relationship isolates PR as the sole source of nonideality. Other effects, such as sneak-path currents, are not captured by this first-order model. To decouple these phenomena, we consider the sparse regime of bit-sliced crossbars for DNN workloads. In such configurations, sneak paths are more likely to be suppressed [7], [8].

IV. Manhattan Distance Weight Mapping

The MDM algorithm reduces crossbar PR effect by reorganizing weights to minimize the Manhattan distance of active memristors from the I/O rails in three steps.

First, the dataflow is reversed so that denser, lower-order bits align with shorter conduction paths, thereby reducing the PR impact. Second, a Manhattan-based score is computed for each row, quantifying the distance of its active memristors from the input. Finally, rows are sorted according to this score, positioning denser rows closer to I/O (see Figure 3).

This spatial remapping minimizes NF without modifying the DNN model. It operates as a post-training transformation that can be seamlessly integrated into existing CIM deployments.

From a system-level perspective, row permutations and reversed dataflow require buffer drivers and multiplexing circuitry already present in state-of-the-art CIM implementations [22],

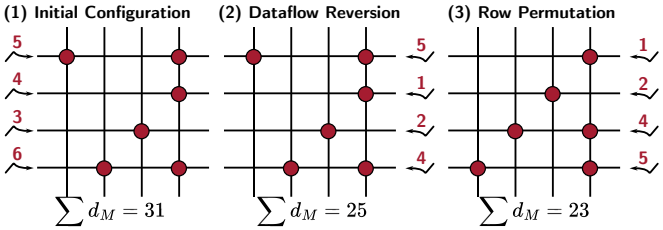

Fig. 3: MDM example. Arrows on the left/right indicate dataflow and numbers on top of each arrow indicate row score.

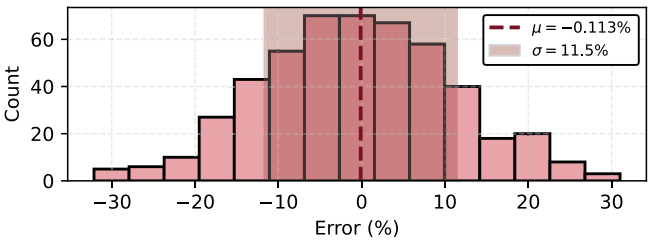

Fig. 4: Error distribution of the Manhattan Hypothesis linear fit has mean $\mu = -0.113\%$ and standard deviation $\sigma = 11.5\%$.

[23]. The approach extends these architectures by modeling and reducing crossbar nonidealities with a novel mapping policy.

V. Experiments

We assess (1) how accurate the Manhattan Hypothesis is and (2) NF reduction and (3) model accuracy drop considering PR effects before/after MDM, benchmarking multiple DNNs. Crossbar computations were simulated in SPICE and PyTorch on ImageNet-1K [35] on all model layers (ResNets, VGGs, ViTs and DeITs from native PyTorch models), trained in 32-bit floating point. The simulations used 128x10 crossbars in 64x64 tiles with the same resistance values as in Section III-B.

A. The Manhattan Hypothesis Accuracy

We evaluate the Manhattan Hypothesis in three stages: (1) we generate 500 randomized crossbar tiles with approximately 80% sparsity, matching the lower bound observed across the evaluated models. Since the least sparse model, DeiT-Base, exhibits 76% sparsity, this level ensures consistency with all architectures, whose sparsity is at least 80%; (2) each tile is simulated in SPICE. The circuit-level simulation measures the NF by probing the column outputs for $r = 0$ (expected output) and $r = 2.5\ \Omega$ (actual output affected by PR); (3) we apply least-squares to find the linear map between the measured and calculated NF[2] (see Figure 4).

B. Nonideality Factor Reduction

The Manhattan hypothesis allows fast PyTorch NF evaluation without exhaustive circuit-level simulation of every DNN tile. As illustrated in Figure 5, MDM significantly reduces the NF. By comparing dataflows, we observe that reverted dataflow improves MDM by up to 50% compared to conventional.

[2]We calculate NF from Equation (16) and measure it using SPICE.

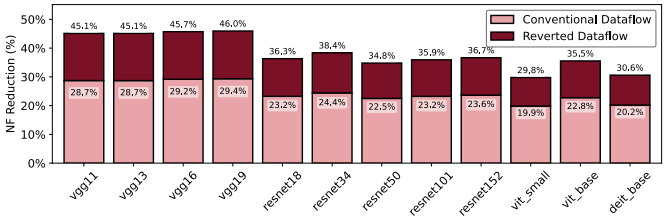

Fig. 5: NF reduction with MDM for different dataflows.

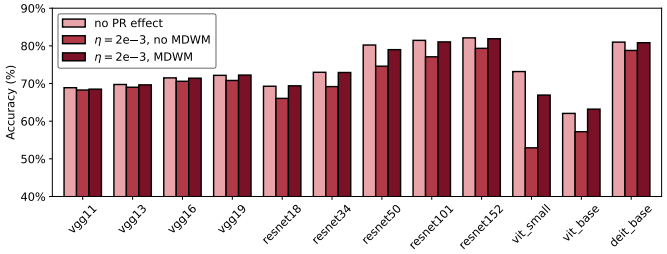

Fig. 6: Model accuracy for different configurations.

C. Model Accuracy Evaluation

Finally, we translate the NF reduction to model accuracy by injecting position-dependent noise in PyTorch, where each weight is modified proportionally to the Manhattan distance:

$$w'_j = \sum_{k \le K} b_{j,k}(w_j) 2^{-k} [1 + \eta \delta_{j,k}], \tag{17}$$

where η is the noise coefficient.

The parameter η is calibrated in SPICE using Equation (17), such that simulations with $r = 2.5\ \Omega$ match the ideal $r = 0$ case. This procedure yields $\eta = 2 \times 10^{-3}$. Figure 6 reports model accuracy under noise injection with and without MDM. Overall, MDM tends to be less effective for transformer models due to their characteristically flatter weight distributions [22], [23], [28], [36]. As a result, their bit-line representations become denser in higher-order columns and sparser in lower-order ones, which diminishes the benefits of MDM.

VI. Conclusion

We introduced the *Manhattan Distance Mapping* (MDM), a spatially informed post-training weight mapping method that reduces parasitic resistance effects in memristive compute-in-memory (CIM) crossbars. By reversing the dataflow and reordering rows according to their cumulative Manhattan distance from the I/O rails, MDM effectively relocates active memristors toward regions less affected by PR voltage drops. The method considerably reduces the nonideality factor (NF).

Through circuit-level and PyTorch-based simulations on ImageNet-1k, we demonstrated that MDM decreases NF by up to 46% and improves inference accuracy under analog distortion by an average of 3.6% in ResNet architectures. These results enable larger crossbars to operate with reduced PR degradation.

By bridging algorithmic and device-level constraints, MDM opens new directions for understanding nonidealities in CIM.

REFERENCES

[1] C.-J. Jhang, C.-X. Xue, J.-M. Hung, F.-C. Chang, and M.-F. Chang, "Challenges and Trends of SRAM-Based Computing-In-Memory for AI Edge Devices," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 5, pp. 1773–1786, 2021.

[2] S. Yu, H. Jiang, S. Huang, X. Peng, and A. Lu, "Compute-in-Memory Chips for Deep Learning: Recent Trends and Prospects," *IEEE Circuits and Systems Magazine*, vol. 21, no. 3, pp. 31–56, 2021.

[3] M. Ali, S. Roy, U. Saxena, T. Sharma, A. Raghunathan, and K. Roy, "Compute-in-Memory Technologies and Architectures for Deep Learning Workloads," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 11, pp. 1615–1630, 2022.

[4] R. Kaur, A. Asad, and F. Mohammadi, "A Comprehensive Review of Processing-in-Memory Architectures for Deep Neural Networks," *Computers*, vol. 13, no. 7, 2024.

[5] Y. Lee, B. Jeon, Y. Cho, J. Kim, W. Shim, and S. Kim, "Recent Progress in Memrsitor Array Structures and Solutions for Sneak Path Current Reduction," *Advanced Materials Technologies*, vol. 10, no. 4, p. 2400585, 2025.

[6] M. Rao, W. Song, F. Kiani, S. Asapu, Y. Zhuo, R. Midya, N. Upadhyay, Q. Wu, M. Barnell, P. Lin, C. Li, Z. Wang, Q. Xia, and J. Joshua Yang, "Timing Selector: Using Transient Switching Dynamics to Solve the Sneak Path Issue of Crossbar Arrays," *Small Science*, vol. 2, no. 1, p. 2100072, 2022.

[7] Y. Cassuto, S. Kvatinsky, and E. Yaakobi, "Sneak-path constraints in memristor crossbar arrays," in *2013 IEEE International Symposium on Information Theory*, pp. 156–160, 2013.

[8] Y. Cassuto, S. Kvatinsky, and E. Yaakobi, "Information-Theoretic Sneak-Path Mitigation in Memristor Crossbar Arrays," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 4801–4813, 2016.

[9] Z. Chen, Z. Wen, W. Wan, A. Reddy Pakala, Y. Zou, W.-C. Wei, Z. Li, Y. Chen, and K. Yang, "PICO-RAM: A PVT-Insensitive Analog Compute-In-Memory SRAM Macro With In Situ Multi-Bit Charge Computing and 6T Thin-Cell-Compatible Layout," *IEEE Journal of Solid-State Circuits*, vol. 60, no. 1, pp. 308–320, 2025.

[10] K.-H. Jo, C.-M. Jung, K.-S. Min, and S.-M. Kang, "Self-adaptive write circuit for low-power and variation-tolerant memristors," *IEEE Transactions on Nanotechnology*, vol. 9, no. 6, pp. 675–678, 2010.

[11] K. You and C. Li, "A Fault-Tolerant Framework for Stuck-at Fault Mitigation in Memristor-Based Ternary Neural Networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2025.

[12] M. Oli-Uz-Zaman, S. A. Khan, W. Oswald, Z. Liao, and J. Wang, "Stuck-at-Fault Immunity Enhancement of Memristor-Based Edge AI Systems," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 12, no. 4, pp. 922–933, 2022.

[13] B. Zhang, N. Uysal, D. Fan, and R. Ewetz, "Handling stuck-at-faults in memristor crossbar arrays using matrix transformations," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, ASPDAC '19, (New York, NY, USA), p. 438–443, Association for Computing Machinery, 2019.

[14] I. Yeo, M. Chu, S.-G. Gi, H. Hwang, and B.-G. Lee, "Stuck-at-Fault Tolerant Schemes for Memristor Crossbar Array-Based Neural Networks," *IEEE Transactions on Electron Devices*, vol. 66, no. 7, pp. 2937–2945, 2019.

[15] I. Muñoz-Martín, S. Bianchi, O. Melnic, A. G. Bonfanti, and D. Ielmini, "A Drift-Resilient Hardware Implementation of Neural Accelerators Based on Phase Change Memory Devices," *IEEE Transactions on Electron Devices*, vol. 68, no. 12, pp. 6076–6081, 2021.

[16] S. Ambrogio, M. Gallot, K. Spoon, H. Tsai, C. Mackin, M. Wesson, S. Kariyappa, P. Narayanan, C.-C. Liu, A. Kumar, A. Chen, and G. W. Burr, "Reducing the Impact of Phase-Change Memory Conductance Drift on the Inference of large-scale Hardware Neural Networks," in *2019 IEEE International Electron Devices Meeting (IEDM)*, pp. 6.1.1–6.1.4, 2019.

[17] P. Xu, L. Zhang, F. Pscheidl, D. Borggreve, F. Vanselow, and R. Brederlow, "A Dynamic Charge-Transfer-Based Crossbar with Low Sensitivity to Parasitic Wire-Resistance," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1397–1401, 2022.

[18] L. Zhang, D. Borggreve, F. Vanselow, and R. Brederlow, "Impact of Parasitic Wire Resistance on Accuracy and Size of Resistive Crossbars," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[19] F. Zhang and M. Hu, "Mitigate parasitic resistance in resistive crossbar-based convolutional neural networks," *J. Emerg. Technol. Comput. Syst.*, vol. 16, May 2020.

[20] M. J. Rasch, C. Mackin, M. Le Gallo, A. Chen, A. Fasoli, F. Odermatt, N. Li, S. R. Nandakumar, P. Narayanan, H. Tsai, G. W. Burr, A. Sebastian, and V. Narayanan, "Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators," *Nature Communications*, vol. 14, p. 5282, Aug. 2023.

[21] A. Ciprut and E. G. Friedman, "Modeling Size Limitations of Resistive Crossbar Array With Cell Selectors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 1, pp. 286–293, 2017.

[22] M. Farias and H. T. Kung, "Sorted Weight Sectioning for Energy-Efficient Unstructured Sparse DNNs on Compute-in-Memory Crossbars," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2025.

[23] M. Farias and H. T. Kung, "Efficient Reprogramming of Memristive Crossbars for DNNs: Weight Sorting and Bit Stucking," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2025.

[24] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pp. 14–26, 2016.

[25] T. Chou, W. Tang, J. Botimer, and Z. Zhang, "CASCADE: Connecting RRAMs to Extend Analog Dataflow In An End-To-End In-Memory Processing Paradigm," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '52, p. 114–125, 2019.

[26] J. Fang, A. Shafiee, H. Abdel-Aziz, D. Thorsley, G. Georgiadis, and J. Hassoun, "Post-Training Piecewise Linear Quantization for Deep Neural Networks," in *The European Conference on Computer Vision (ECCV)*, 2020.

[27] M. Horton, Y. Jin, A. Farhadi, and M. Rastegari, "Layer-Wise Data-Free CNN Compression," in *International Conference on Pattern Recognition (ICPR)*, 2022.

[28] T. Tambe, E.-Y. Yang, Z. Wan, Y. Deng, V. Janapa Reddi, A. Rush, D. Brooks, and G.-Y. Wei, "Algorithm-Hardware Co-Design of Adaptive Floating-Point Encodings for Resilient Deep Learning Inference," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2020.

[29] A. Bhattacharjee, L. Bhatnagar, and P. Panda, "Examining and Mitigating the Impact of Crossbar Non-idealities for Accurate Implementation of Sparse Deep Neural Networks," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1119–1122, 2022.

[30] I. Chakraborty, M. Fayez Ali, D. Eun Kim, A. Ankit, and K. Roy, "GENIEx: A Generalized Approach to Emulating Non-Ideality in Memristive Xbars using Neural Networks," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2020.

[31] I. Chakraborty, M. Ali, A. Ankit, S. Jain, S. Roy, S. Sridharan, A. Agrawal, A. Raghunathan, and K. Roy, "Resistive Crossbars as Approximate Hardware Building Blocks for Machine Learning: Opportunities and Challenges," *Proceedings of the IEEE*, vol. 108, no. 12, pp. 2276–2310, 2020.

[32] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in *International Conference in Learning Representations (ICLR)*, 2016.

[33] T. Cao, W. Yu, Y. Gao, C. Liu, T. Zhang, S. Yan, and W. L. Goh, "Edge PoolFormer: Modeling and Training of PoolFormer Network on RRAM Crossbar for Edge-AI Applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 33, no. 2, pp. 384–394, 2025.

[34] T. Cao, C. Liu, Y. Gao, and W. L. Goh, "Parasitic-Aware Modelling for Neural Networks Implemented with Memristor Crossbar Array," in *2021 IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*, pp. 122–126, 2021.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[36] Y. Bondarenko, M. Nagel, and T. Blankevoort, "Understanding and Overcoming the Challenges of Efficient Transformer Quantization," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Nov. 2021.