Vince Tran[1], Demeng Chen[1], Roman Genov[1], Mostafa Rahimi Azghadi[2], and Amirali Amirsoleimani[3]

[1]Department of Electrical and Computer Engineering, University of Toronto
[2]College of Science and Engineering, James Cook University
[3]Department of Electrical Engineering and Computer Science, York University

March 07, 2024

BITLITE: Light Bit-wise Operative Vector Matrix Multiplication for Low-Resolution Platforms

Vince Tran[1*], Demeng Chen[1*], Roman Genov[1], Mostafa Rahimi Azghadi[3], and Amirali Amirsoleimani[2]
[1]Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada
[2]Department of Electrical Engineering and Computer Science, York University, Toronto, Canada
[3]College of Science and Engineering, James Cook University, Townsville, Australia
Email: vince.tran@mail.utoronto.ca, demeng.chen@mail.utoronto.ca, roman@eecg.utoronto.ca,
mostafa.rahimiazghadi@jcu.edu.au,
amirsol@yorku.ca

***Abstract*—As machine learning (ML) algorithms, particularly neural networks (NN), expand in popularity and capacity, the quest for more efficient computation methods gains momentum. Memristor crossbar technology emerges as a promising alternative to traditional computing units, aiming to address traditional computing challenges. However, conventional matrix-vector multiplication (MVM) methods on these platforms are often plagued by device imperfections and drift. In this work, we introduce an innovative lightweight calculation approach leveraging bit-transformation for MVM, significantly enhancing operation precision and, consequently, the performance of ML algorithms on memristor crossbar platforms. We provide details of the core algorithm and its extensions, furnish digital validation, and simulate its efficacy using an autoencoder (AE) neural network with an extended VTEAM model. Our tests demonstrate an average reconstruction precision improvement of approximately 53.5%. This work's applicability extends beyond NNs, offering a foundational method for conducting more precise analog MVM operations.**

***Index Terms*—Memristor Crossbar, matrix-vector multiplication (MVM) , Deep Learning, Bit-transformation**

I. Introduction

RECENT strides in neural network (NN) driven artificial intelligence (AI) have notably reshaped the contemporary landscape [1]–[3]. However, the computational demand of NNs is intensive, often requiring multiple parallel computing units for efficient computation. This leads to significant energy expenditure. While existing parallel computing architectures such as the central processing unit (CPU) and graphics processing units (GPU) suffice, the intrinsic constraints of the traditional Von Neumann architecture hinder further scalability. A salient manifestation of this is the memory wall challenge [4], [5], born from the physical disjunction of computation and storage units. Owing to the memory wall predicament and hefty power consumption, memristor technology has attracted substantial research attention.

The concept of memristors was initially posited in 1971 and realized in 2008 [6], [7]. The central allure of memristors stems from their conductance modulation in response to external stimuli. Upon stimulation, for example in the Phase Change Memory technology, due to joule heating, the phase-changing material encased between the two electrodes transitions from a structured crystalline form to a high-resistance, amorphous state, elevating the resistance. This characteristic allows for the application of partial stimuli to achieve intermediate conductance stages, offering finer control over the memristor's state [5], [8]–[13]. However, memristors have finite intermediate conductance stages and are susceptible to drift, read noise, and asymmetry due to inherent device imperfections [14]–[16], which in turn, constrain the performance of ML algorithms on crossbar circuits.

In this work, we introduce a novel multiplication algorithm that leverages the pronounced difference in high/low resistance and employs bit-wise long multiplication to supplant the conventional matrix-vector multiplication (MVM) calculation method, which relies on a direct application of Ohm's Law [17], [18]. The paper is structured as follows. First, we will introduce the preliminary, required steps to reach the final algorithm. Then, we will simulate the proposed algorithm, called BITLITE, to study its performance on an autoencoder-type neural network on a $Ge_2Sb_2Te_5$ memristor crossbar circuit. We then demonstrate BITLITE's reconstruction precision improvement against previous works [19], [20].

II. Preliminary

We first start with the classic binary multiplication of two values a, b, expressed with α, β bits respectively. This calculation can be fully expanded into an expression representing the long-multiplication method, which processes the result bit-wise and reflects the traditional shift-and-add algorithm used in modern digital computing.

$$\begin{aligned} a \cdot b &= \sum_{k=1}^{\beta} \sum_{j=1}^{\alpha} (a_j \cdot 2^{\alpha-j})(b_k \cdot 2^{\beta-k}) \\ &= \sum_{k=1}^{\beta} \sum_{j=1}^{\alpha} a_j b_k \cdot 2^{\alpha+\beta-j-k} \end{aligned} \tag{1}$$

In the above equation, a_i and b_i are the i^{th} bit of a and b respectively. Following the same convention, $i, j = 1$ are the most significant bits (MSB), and α, β are the least significant bits (LSB). The term $2^{\alpha+\beta-j-k}$ represents a variable left bit shift depending on the significance of both bits.

*These authors contributed equally to this work

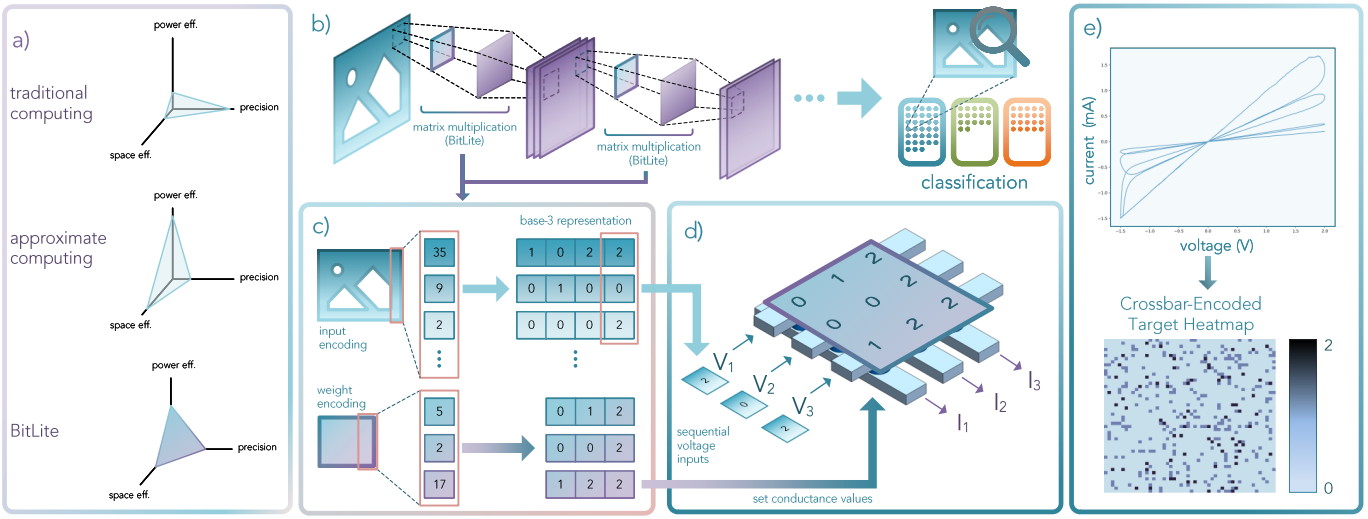

Fig. 1. (a) A brief summary of the intended advantages and trade-offs of BITLITE compared to traditional and approximate computing schemes. (b) A high-level overview of the BitLite algorithm and its application in an example neural network. (c) Pre-processing of input and weights visualized, specifically the extension of vectors into new base-digit matrices. (d) Processed data encoded as memristor crossbar weights and fed as voltage inputs. Here, recurring matrices such as CNN kernels are tuned as memristor conductance values. (e) An example encoding of matrix values. Here, target integer values of the example base-3 system are encoded as specifically tuned resistances.

Extending this expression to a MAC operation over a set of values $a_i \in [a_1, \ \dots \ , a_n]$, $b_i \in [b_1, \ \dots \ , b_n]$, the first index now marks the value's position in the set. Introducing another index for the bits, let a_{ij} be the j^{th} bit of a_i and b_{ik} be the k^{th} bit of b_i. Using the above expression, we can expand the operation to a similar form.

Now, we observe that this binary case is an extension of a more general expression:

$$a \cdot b = \sum_{k=1}^{\beta} \sum_{j=1}^{\alpha} a_j b_k \cdot \theta^{\alpha+\beta-j-k} \tag{2}$$

Here, a, b are represented in any arbitrary base-θ system with no theoretical upper bound over the maximum digit count α, β.

In the simplest case, we can directly apply the above equation to a simple multiply-and-accumulate (MAC) operation, achieving a vector dot product. We extend both vectors into their individual digits and express the operation using a similar form.

$$\sum_{i=1}^{n} a_i b_i = \sum_{k=1}^{\beta} \sum_{j=1}^{\alpha} \sum_{i=1}^{n} a_{ij} b_{ik} \cdot \theta^{\alpha+\beta-j-k} \tag{3}$$

III. Variable-Base Memristor Matrix Multiplication

Using a memristor crossbar, matrix-vector multiplication can easily be performed using Ohm's Law, expressed alternatively using conductance $G = 1/R$.

$$I = VG$$

By taking advantage of the circuit behavior of memristor crossbars, direct MVM can be performed in one step. Here, we state a similar expression in matrix form to reflect the crossbar:

$$\mathbf{G}\vec{V} = \vec{I} \tag{4}$$

Above is the memristor analogue of the classic expression $\mathbf{A}\vec{x} = \vec{b}$. Voltage inputs are fed into a conductance-tuned crossbar array, and the resulting outputs yield the required MVM results.

We now start with the simplest case as presented in section II: some vector $\vec{a}$ to be expressed in extended form over the crossbar and an input vector $\vec{b}$ to perform the dot product $\vec{a} \cdot \vec{b}$. Over some arbitrary base system, the two $1 \times n$ vectors $\vec{a}, \vec{b}$ are separated by digit to form matrices:

$$\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \dots \\ a_{\alpha 1} & \cdots & a_{\alpha n} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_{11} & \cdots & b_{1\beta} \\ \vdots & \ddots & \dots \\ b_{n1} & \cdots & b_{n\beta} \end{bmatrix} \tag{5}$$

In the extended representations above, a_{ij} is the i^{th} digit of the j^{th} entry in the original vector $\vec{a}$, and b_{ij} is the j^{th} digit of the i^{th} entry in the original vector $\vec{b}$. The transposition of the bit and entry indexes is essential for consistent matrix multiplication dimensionality.

By sequentially inputting these voltage vectors and reading the outputs, we achieve a full output matrix. The values of this matrix can be manipulated into the desired MAC result. The full resulting output matrix produced by reading the current outputs is as follows:

$$\mathbf{AB} = \begin{bmatrix} \sum_{i=1}^{n} a_{1i} b_{i1} & \sum_{i=1}^{n} a_{1i} b_{i2} & \cdots & \sum_{i=1}^{n} a_{1i} b_{i\beta} \\ \sum_{i=1}^{n} a_{2i} b_{i1} & \sum_{i=1}^{n} a_{2i} b_{i2} & \cdots & \sum_{i=1}^{n} a_{2i} b_{i\beta} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} a_{\alpha i} b_{i1} & \sum_{i=1}^{n} a_{\alpha i} b_{i2} & \cdots & \sum_{i=1}^{n} a_{\alpha i} b_{i\beta} \end{bmatrix} \tag{6}$$

The entries of the output matrix $\mathbf{A}_{bin}\mathbf{V}_{bin}$ do not include the digit-shift terms, however. Each vector passed into the crossbar holds the digits of a consistent significance; this is likewise for the digits encoded in each column of the crossbar.

Summing results directly from the crossbar would therefore yield an incorrect result in base 10. For some θ-base system,

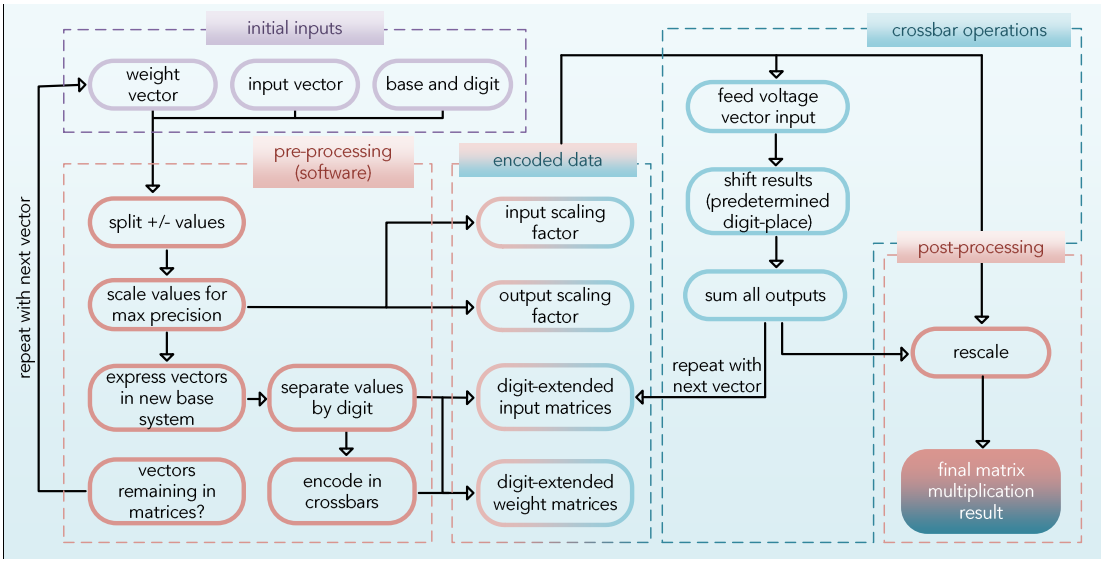

Fig. 2. A full BITLITE algorithm flowchart separated into three major sections: pre-processing, crossbar computations, and post-processing. This top-down view of the entire process also includes key functional features such as scaling and sign splitting.

we must first incorporate these exponential terms, of form $\theta^{\alpha+\beta-i-j}$, which hold variables for the significance of the input vector and the column from which the output current was read. Specifically, i is the significance of the column, and j is the significance of the vector input.

$$\boldsymbol{\theta}_{out} = \begin{bmatrix} \theta^{\alpha+\beta-1-1} & \theta^{\alpha+\beta-1-2} & \dots & \theta^{\alpha+\beta-1-\beta} \\ \theta^{\alpha+\beta-1-2} & \theta^{\alpha+\beta-2-2} & \dots & \theta^{\alpha+\beta-2-\beta} \\ \vdots & \vdots & \ddots & \vdots \\ \theta^{\alpha+\beta-\alpha-1} & \theta^{\alpha+\beta-\alpha-2} & \dots & \theta^{\alpha+\beta-\alpha-\beta} \end{bmatrix} \tag{7}$$

The exponent terms have been left uncombined to explicitly show the digit-shifting behavior of this algorithm. The sum of all the entries of the resulting matrix $\mathbf{AB}\boldsymbol{\theta}_{out}$ gives us the desired dot product result corresponding to equation (3). With this formulation, we have achieved a crossbar-based approach generalized to any base system. This algorithm can further be extended to MVM, where each row of a desired LHS matrix is extended into a 2-dimensional digit representation as shown in equation (5).

IV. Experimental Section

Knowing the algorithm and its derivation, we now analyze BITLITE's performance in a simulated environment. As shown in Fig. 3(a), to better illustrate BITLITE's advantage in improving MVM accuracy, we trained a simple autoencoder on the MNIST dataset to demonstrate the improved reconstruction precision against a pure analog implementation. Due to the bit-expansion nature of the platform, BITLITE is best suited for places where precision is needed and MVM operations are not overly large. Thus, we will utilize BITLITE to implement the MVM operation in the bottleneck layer. For a single vector, the bit expanded matrix will be sized $n \times \beta$, where n is the number of elements in the input vector and β is the bit width. To program the encoding, we interpolate these values and fit them into the appropriate conductance range. To program zeros, we selectively turn off memristors, blocking the incoming current from the input to that column. Finally, we program the expanded matrix of the first row of the weight matrix onto the memristor crossbar array. A sample programming result is shown in Fig. 3(b, c). From here, we can progressively feed the rest of the rows after to obtain the complete MVM result.

TABLE I
Average Precision with Different Methods

Digital	Analog	Analog + BITLITE	BITLITE
0.0054	0.0642	0.0378	0.0299
-	-	41.1%	53.5%

We also round down the results we obtain from the MVM operation since in an ideal world, all MVM results should be integers in BITLITE. Table 1 compares different methods like ideal, pure analog, and BITLITE variants in terms of how precisely they reconstruct data. Using BITLITE just for the critical forward pass increased precision by 41.1% over a fully analog setup. Fig. 3(d) shows examples where improvements range up to 83.9% and average a 53.5% increase in performance.

BITLITE can be used in binary form by using only the lowest resistant state and the off state. In this form, BITLITE does not require expensive reprogramming for each memristor after every MVM operation. Rather, BITLITE simply switches off the required memristors to achieve the zero encoding and leave the rest on for the one encoding, speeding up weight programming drastically. Due to the nature of using memristors as bits, BITLITE brings another useful advantage: it simplifies the design of the input DACs since the conversion range is tiny. In our experiment, we used up to base 8. As a result, we only need 3-bit DACs to effectively apply the input currents to the crossbar circuit.

Moreover, BITLITE is more resistant to device imperfections. We proved this by intentionally increasing noise and drift levels in our tests. Table 2 shows that with a 10% increase in these factors, traditional methods experienced about a 30% drop in performance, while BITLITE's performance decreased by only 10%. Under more extreme conditions, BITLITE was up to 218% more robust than standard MVM methods.

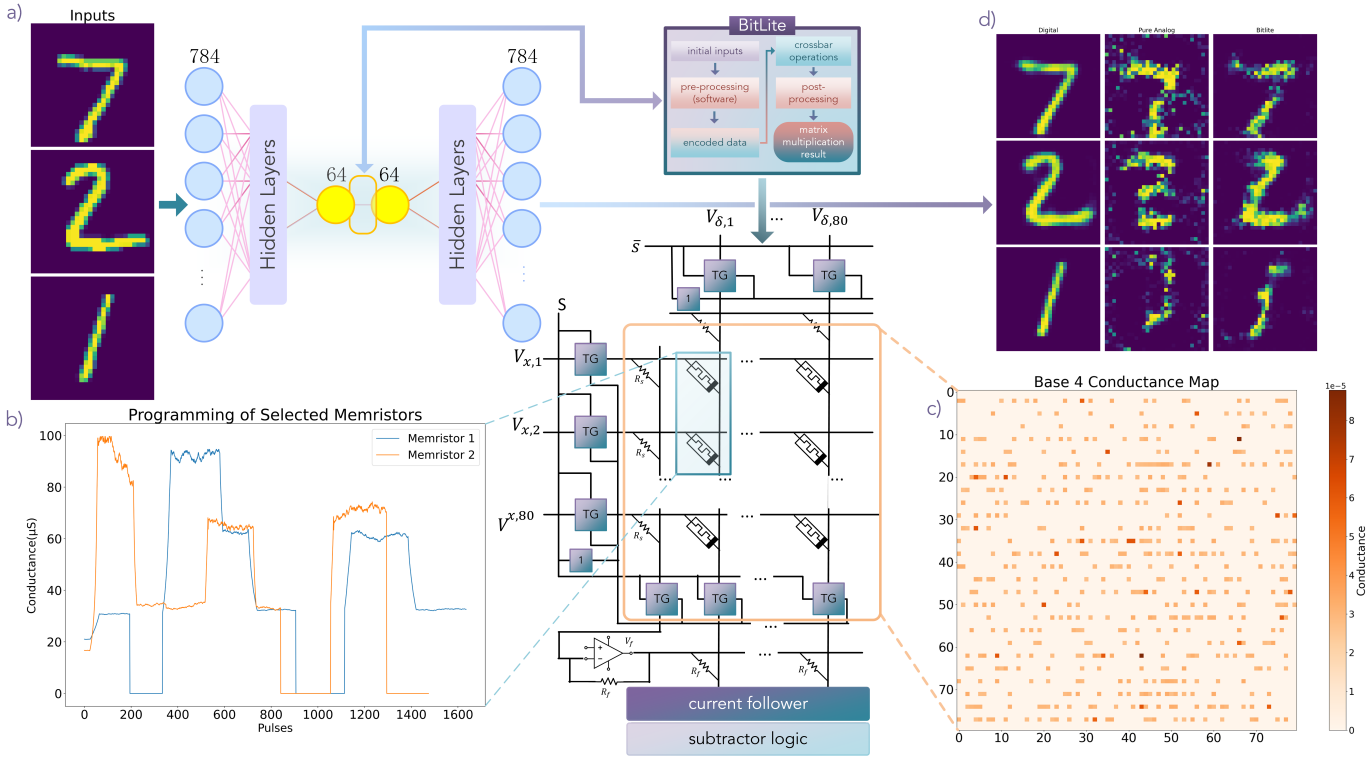

Fig. 3. (a): The inputs are simple handwriting digits from the MNIST dataset. The task is image reconstruction using an autoencoder. We apply BITLITE in the bottleneck layer for better illustration. (b): We randomly select two memristors and observe their conductance state after some read/programming pulses. (c): The conductance map for the first 80 by 80 memristors on the crossbar. (d): The reconstruction result comparison between different calculation methods.

TABLE II
COMPARISON OF MEMRISTOR-BASED MVM TECHNIQUES

$Ge_2Sb_2Te_5$ [23], $1 \times n$ input, $n \times n$ matrix, β-digit input in base-θ			
	Pure Analog	**k Parallel Kernels [22]**	**BITLITE**
Crossbar Footprint	n^2	$k \cdot n^2$	$\beta \cdot n^2$
Max Power Draw† (W)	$n^3 V_{max}^2 R_{LRS}^{-1}$	$n^2 k V_{max}^2 R_{LRS}^{-1}$	$n^2 \beta V_{max}^2 R_{LRS}^{-1}$
Complexity	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(\beta)$
Non-Ideality Impacts	Linear	Linear	Digit-wise Linear
10% Artificial Noise	-28.6%	-	-10.9%
50% Artificial Noise	-79.3%	-	-36.4%

† $V_{max} \rightarrow V_{th} = 1.48$ V, $R_{LRS} = 0.94$ kΩ

Given our experimental results, BITLITE's trade-offs in the approximate MVM space are clear (see Table II). Firstly, at the expense of a sequential input of digit vectors, we attain a vastly superior representation of matrix entries; a maximum positive integer value of $\theta^\beta - 1$ is achieved. Pure analog and other speed-focused approaches built upon a straightforward, single memristor and voltage input encoding scheme fail to replicate this level of precision.

BITLITE is also constrained in performance by the number of digits, which offers practical limitations to an otherwise theoretically unbound formulation. The multiplicative effect of introducing more digits per entry directly impacts the overall footprint of a BITLITE-based architecture, which then has the potential to amplify the effects of non-idealities. An even more pronounced effect on matrix entries is therefore observed. Namely, the drift of memristors responsible for the most significant digits *can* result in significant disparities from an intended value, though our initial observations still present a marked improvement from the analog case. This calls for techniques to mitigate drift, periodically tune conductance values, and create a more robust architecture overall.

Finally, it is important to note that the magnitude of these drawbacks are by design. The variable precision introduced by an optimized digit-base combination minimizes unnecessary precision, thereby lowering the overall footprint and impact of these sources of error. BITLITE is a versatile approximate MVM scheme suited to accelerated applications that require higher precision than is feasible by 1-to-1 encoding.

V. CONCLUSION

In this research, we explored the challenges of traditional MVM methods when implemented on memristor crossbar platforms, particularly issues arising from device imperfections. We proposed a new lightweight calculation technique incorporating bit-transformation, which notably improves the precision of MVM operations. This enhancement directly contributes to better performance in machine learning algorithms running on memristor platforms. We validated our method in a simulated environment and an AE neural network, where we observed an average precision improvement of approximately 53.5%. While our focus was on neural networks, the proposed technique has broader applications. It lays a foundation for future work aiming to refine analog MVM operations on memristor platforms.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, 'ImageNet Classification with Deep Convolutional Neural Networks', in Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, Lake Tahoe, Nevada, 2012, pp. 1097–1105.

[2] R. Yang and Y. Yu, "Artificial convolutional neural network in object detection and semantic segmentation for medical imaging analysis," Frontiers in Oncology, vol. 11, 2021.

[3] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J. & Lowe, R. Training language models to follow instructions with human feedback. (2022)

[4] W. A. Wulf and S. A. McKee, "Hitting the memory wall," ACM SIGARCH Computer Architecture News, vol. 23, no. 1, pp. 20–24, 1995.

[5] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," Nature Nanotechnology, vol. 15, no. 7, pp. 529–544, 2020. doi:10.1038/s41565-020-0655-z

[6] Chua, L. Memristor-The missing circuit element. *IEEE Transactions On Circuit Theory*. **18**, 507-519 (1971)

[7] Strukov, D., Snider, G., Stewart, D. & Williams, R. The missing memristor found. *Nature*. **453**, 80-83 (2008)

[8] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A General Model for Voltage-Controlled Memristors," Circuits and Systems II: Express Briefs, IEEE Transactions on, vol. 62, pp. 786–790, Aug. 2015.

[9] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "TEAM: ThrEshold Adaptive Memristor Model," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 60, no. 1, pp. 211–221, Jan. 2013.

[10] N. Papandreou et al., 'Programming algorithms for multilevel phase-change memory', in 2011 IEEE International Symposium of Circuits and Systems (ISCAS), 2011, pp. 329–332.

[11] M. Hu et al., 'Dot-Product Engine for Neuromorphic Computing: Programming 1T1M Crossbar to Accelerate Matrix-Vector Multiplication', in Proceedings of the 53rd Annual Design Automation Conference, Austin, Texas, 2016.

[12] A. Sebastian et al., "Tutorial: Brain-inspired computing using phase-change memory devices," Journal of Applied Physics, vol. 124, no. 11, p. 111101, 2018. doi:10.1063/1.5042413

[13] N. Papandreou et al., 'Estimation of amorphous fraction in multilevel phase-change memory cells', Solid-State Electronics, vol. 54, no. 9, pp. 991–996, 2010.

[14] A. Chen, "A comprehensive crossbar array model with solutions for line resistance and nonlinear device characteristics," IEEE Transactions on Electron Devices, vol. 60, no. 4, pp. 1318–1326, 2013. doi:10.1109/ted.2013.2246791

[15] G. C. Adam, A. Khiat, and T. Prodromakis, "Challenges hindering memristive neuromorphic hardware from going mainstream," Nature Communications, vol. 9, no. 1, 2018. doi:10.1038/s41467-018-07565-4

[16] W.-Q. Pan et al., "Strategies to improve the accuracy of memristor-based Convolutional Neural Networks," IEEE Transactions on Electron Devices, vol. 67, no. 3, pp. 895–901, 2020. doi:10.1109/ted.2019.2963323

[17] X. Liu and Z. Zeng, 'Memristor crossbar architectures for implementing deep neural networks', Complex & Intelligent Systems, vol. 8, no. 2, pp. 787–802, Apr. 2022.

[18] Z. Sun et al., "Solving matrix equations in one step with cross-point resistive arrays," Proceedings of the National Academy of Sciences, vol. 116, no. 10, pp. 4123–4128, 2019. doi:10.1073/pnas.1815682116

[19] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation." , Parallel Distributed Processing. Vol 1: Foundations. MIT Press, Cambridge, MA, 1986.

[20] S. R. Nandakumar, M. Le Gallo, C. Piveteau, V. Joshi, G. Mariani, I. Boybat, G. Karunaratne, R. Khaddam-Aljameh, U. Egger, A. Petropoulos, T. Antonakopoulos, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Mixed-precision deep learning based on Computational Memory," Frontiers in Neuroscience, vol. 14, 2020.

[21] E. Eleftheriou et al., 'Deep learning acceleration based on in-memory computing', IBM Journal of Research and Development, vol. 63, no. 6, p. 7:1-7:16, 2019.

[22] P. Lin et al., "Three-dimensional memristor circuits as complex neural networks," Nature Electronics, vol. 3, no. 4, pp. 225–232, 2020. doi:10.1038/s41928-020-0397-9

[23] Y. Li et al., "Intrinsic memristance mechanism of crystalline stoichiometric ge2sb2te5," Applied Physics Letters, vol. 103, no. 4, 2013. doi:10.1063/1.4816283