RESEARCH ARTICLE | APRIL 07 2025

AI-driven model for optimized pulse programming of memristive devices

Special Collection: Neuromorphic Technologies for Novel Hardware AI

Benjamin Spetzler ; Markus Fritscher ; Seongae Park ; Nayoun Kim ; Christian Wenger ; Martin Ziegler

APL Mach. Learn. 3, 026103 (2025)
https://doi.org/10.1063/5.0251113

AI-driven model for optimized pulse programming of memristive devices

Cite as: APL Mach. Learn. 3, 026103 (2025); doi: 10.1063/5.0251113
Submitted: 29 November 2024 • Accepted: 21 March 2025 •
Published Online: 7 April 2025

Benjamin Spetzler,[1,2,a] Markus Fritscher,[3,4] Seongae Park,[2] Nayoun Kim,[2] Christian Wenger,[3,4] and Martin Ziegler[1,2]

AFFILIATIONS

[1] Energy Materials and Devices, Department of Materials Science, Faculty of Engineering, Kiel University, Kiel, Germany
[2] Micro- and Nanoelectronic Systems, Department of Electrical Engineering and Information Technology, Technische Universität Ilmenau, Ilmenau, Germany
[3] IHP-Leibniz Institute for High Performance Microelectronics, Frankfurt (Oder), Germany
[4] Brandenburg University of Technology Cottbus-Senftenberg, Cottbus, Germany

Note: This paper is part of the APL Machine Learning Special Topic on Neuromorphic Technologies for Novel Hardware AI.
[a] **Author to whom correspondence should be addressed:** besp@tf.uni-kiel.de

ABSTRACT

Next-generation artificial intelligence (AI) hardware based on memristive devices offers a promising approach to reducing the increasingly large energy consumption of AI applications. However, programming memristive AI hardware to achieve a desired synaptic weight configuration remains challenging because it requires accurate and energy-efficient algorithms for selecting the optimal weight-update pulses. Here, we present a computationally efficient AI model for predicting the weight update of memristive devices and guiding device programming. The synaptic weight-update behavior of bilayer HfO_2/TiO_2 memristive devices is characterized over a range of pulse parameters to provide experimental data for the AI model. Three different artificial neural network (ANN) configurations are trained and evaluated regarding the amount of training data required for accurate predictions and the computational costs. Finally, we apply the model to an antipulse weight-update process to demonstrate its performance. The results show that accurate and computationally inexpensive predictions are possible with comparatively few datasets and small ANNs. The normalized weight-update processes are predicted with accuracies comparable with larger model architectures but require only 896 floating point operations and 8.33 nJ per inference. This makes the model a promising candidate for integration into AI-driven device controllers as a precise and energy-efficient solution for memristive device programming.

I. INTRODUCTION

With the dawn of artificial intelligence (AI) as one of the most relevant technologies in the 2020s, the demand for suitable hardware that can handle large amounts of data with low energy and low latencies is rapidly increasing.[1–5] Especially for edge applications, the energy consumption, required memory, and latency of the increasingly large AI models represent a growing challenge.[6–8]

One solution is implementing artificial neural networks (ANNs) in hardware based on memristive devices, i.e., memristive AI accelerators.[7,9–11] Memristive devices are two-terminal microelectronic elements that can combine multilevel, nonvolatile, or (to some extent) volatile memory states with low power consumption and excellent scalability.[12–15] When arranged in crossbar arrays, memristive devices can serve as artificial synapses of the ANN, representing the entries of the synaptic weight matrix via their electrical resistances.[10,11,16,17] During training and inference, the synaptic weight is updated and read out by applying voltage pulses using peripheral control circuits.[18] With this concept, vector–matrix multiplication (VMM) can be performed in a single step and in-memory.[18,19] Conceptionally, such a single-step VMM implementation provides low latency and high energy efficiency because it avoids the data shuttling between memory and processing units, which is the case with conventional von Neumann architectures.[7,11,14,16]

In recent years, remarkable progress has been made in the technology and application of AI accelerators based on memristive crossbar arrays.[18,20–24] Many studies have presented energy-efficient memristive AI-accelerator hardware, e.g., for perceptron networks,[25] long short-term memory ANNs,[26] and reservoir computing.[27] State-of-the-art systems with integrated complementary metal–oxide–semiconductor (CMOS) peripherals demonstrated applications such as image classification,[25,28] speech recognition with near software equivalent accuracy,[29] and more.[21,24,30]

However, transitioning from proof-of-concept demonstrations to real-world applications requires solving challenges beyond the device architecture, from circuit design and integration to algorithm development.[18] For example, one of the main bottlenecks for application is caused by energy-inefficient CMOS peripheral circuits for accurate network programming.[18,31] The programming of memristive devices is still challenging because it requires sophisticated weight-update strategies and peripheral circuits to precisely map synaptic weights to the resistive states of the memristive devices.[32–34] It demands the reproducibility of individual devices and the reliable prediction of their (generally) nonlinear weight update responses.[34–36] Here, accurate and energy-efficient models could guide the programming circuitry to ensure effective weight updates.

Data-driven models based on machine learning and artificial neural networks (ANNs) could be an efficient solution by predicting the resistive states of memristive devices. Such models have already demonstrated great potential for modeling and analyzing nonlinear dynamic systems.[37–41] Compared to other device modeling approaches, such as charge transport[42–47] and compact models,[48–54] they do not require knowledge of the physical mechanisms. In particular, ANNs can effectively interpolate multidimensional datasets based on their generalization capability, i.e., they are not limited by predefined interpolation schemes or analytical expressions.[55,56] Therefore, data-driven models could be promising components of AI-enhanced controllers for programming and prototyping memristive devices,[35] as illustrated in Fig. 1.

However, published reports about ANNs for modeling memristive devices are rare. Physics-informed neural networks (PINNs) have recently been applied as surrogates to integrate memristive compact models into Verilog-A circuit simulations.[57,58] A different approach used convolutional neural networks to improve the fitting procedure of compact models to the current–voltage curves of memristive devices.[59] However, these models still rely on the physics-inspired formulation of compact models and their limitations. Furthermore, a data-driven model was presented for the current–voltage characteristics of nanofiber memristive devices.[60] Data-driven models for predicting the pulse programming of memristive devices are still missing.

Here, we present a fully data-driven model for the programming of memristive devices, which is also computationally inexpensive. The model is based on a multilayer perceptron ANN, which predicts the normalized read current value as a function of the pulse parameters using measured pulse curves for training, testing, and validation. We analyze the number of training datasets required to obtain sufficiently accurate predictions for three different ANN sizes and discuss the networks' energy consumption and computational cost. The results demonstrate that accurate and computationally inexpensive predictions are possible with comparatively few datasets and small networks. This renders the model a potential candidate as a component in an AI-enhanced device controller.

II. MEMRISTIVE DEVICES

A. Device structure

A cross section of the devices used in this work is illustrated in Fig. 2. They are based on a metal–insulator–metal thin-film structure with an in-plane device area of $25 \times 25\ \mu m^2$. A 50-nm-thick Au top electrode and a 50-nm-thick TiN bottom electrode sandwich a sputtered bilayer of HfO_2 (2 nm) and TiO_2 (30 nm). The functional layers are encapsulated with sputtered SiO_2. For the fabrication, we use 4-in. SOI wafer technology with reactive magnetron sputtering, as reported in Ref. 61. For all measurements, the TiN rear-side electrode was grounded.

B. Electrical device characteristics

Current–voltage (I–V) curves were measured by applying a piecewise linear triangular voltage with a maximum value of 3 V and a minimum value of −2 V at the top electrode, while the bottom electrode was grounded. A voltage sweep rate of ≈ 3 V/s was used for $V > 0$ V and ≈ 2 V/s for $V < 0$ V.

Representative example curves measured on five different devices are shown in Fig. 3(a), while Fig. 3(b) presents three consecutively measured I–V curves from a single device. The voltage V as

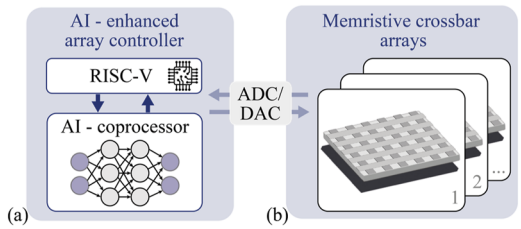

FIG. 1. Illustration of the envisioned programming concept for memristive crossbar arrays. (a) Artificial intelligence (AI)-enhanced array controller comprising a RISC-V microprocessor enhanced by an AI-coprocessor for predicting the weight update. The controller is connected to (b) memristive crossbar arrays via an analog/digital and a digital/analog converter (ADC/DAC) interface for programming and reading out the memristive devices.

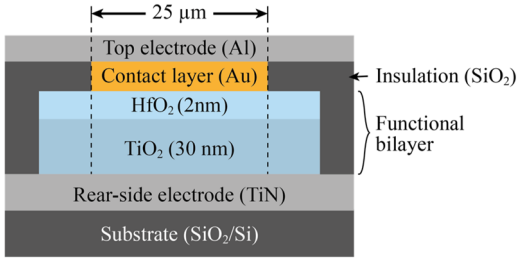

FIG. 2. Illustration of the layer sequence of the memristive thin-film devices used in this work. The functional HfO_2/TiO_2 bilayer is sandwiched by a TiN bottom electrode and an Au layer contacting the top Al electrode. The layer stack is encapsulated by sputtered SiO_2. The layer thicknesses and functionalities are indicated. Further details on the devices can be found in Ref. 61.

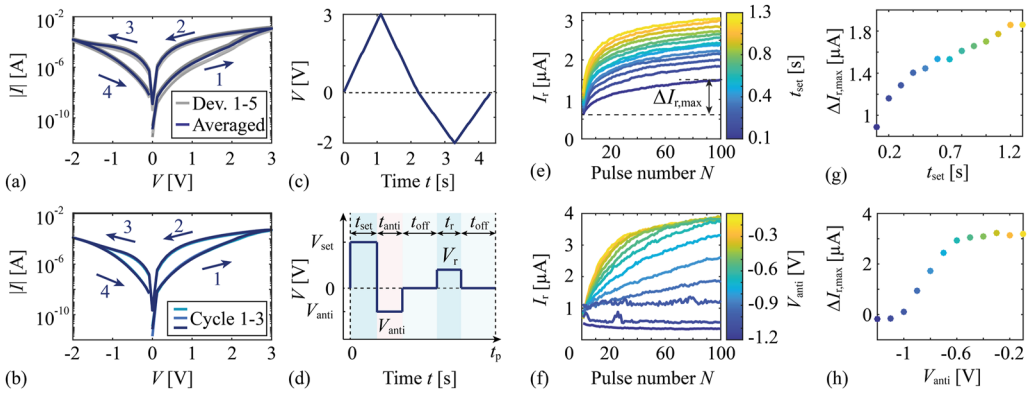

FIG. 3. Representative device characteristics of the memristive bilayer devices used here. (a) Example current–voltage curves of five different devices with a sweep rate of ≈ 3 V/s for $V > 0$ V and ≈ 2 V/s for $V < 0$ V by applying a piecewise linear voltage function. (b) Results from three consecutive I–V curve measurements on one example device. (c) Voltage as a function of time applied during the I–V curve measurements. (d) Illustration of the pulse voltage function within a single set pulse period applied to the devices for characterization including the definitions of the pulse parameters. (e) Example pulse update curve measured over 100 set pulse periods for 13 different values of the set pulse width t_{set}, and (f) ten different antipulse amplitudes V_{anti}. (g) Extracted maximum difference $\Delta I_{r,max} = I_r(N = 100) - I_r(N = 1)$ of the read current I_r between its initial value at the pulse period $N = 1$ and that at the last period $N = 100$ of the data in (e) as a function of the set voltage width t_{set}, and (h) extracted $\Delta I_{r,max}$ from the data in (f) as a function of the antipulse amplitude V_{anti}.

a function of time t is depicted in Fig. 3(c). All I–V curves exhibit consistent qualitative and quantitative behavior, with only minor variations in their maximum currents, differing by approximately a factor of two. The curves exhibit a notable hysteresis oriented counterclockwise in the left and right hysteresis branch (see the arrows from 1 to 4) without discontinuities in the four sections. Such a behavior was previously interpreted as being caused by the continuous drift of oxygen vacancies in the HfO_2 layer and a potential contribution of trap state dynamics.[50,61] However, for the application in neuromorphic systems, the devices are usually operated by applying voltage pulses.

For characterizing the pulse update behavior of the devices, we define a voltage function $V(t)$, which follows the illustration in Fig. 3(d) over one set of pulses of the total period length t_p. Each period starts with a rectangular pulse of duration t_{set} and voltage V_{set}, followed by an antipulse of the duration t_{anti} with a voltage V_{anti} of opposite polarity (i.e., a negative pulse). The antipulse is followed by a pause time with a duration t_{off} and $V = 0$ V, a read pulse with length t_r and voltage V_r, and a second pause time equal to the first one.

Two sets of pulse measurements were performed to demonstrate the weight update behavior of the devices and the influence of the pulse parameters [Figs. 3(e) and 3(f)]. Each measurement set comprises ten pulse measurements with $N = 100$ successive pulse periods. For each of the two measurement sets, a different pulse parameter was varied over the ten pulse measurements, namely, t_{set} for the first measurement set and V_{anti} for the second one. The read pulse parameters $V_r = 0.5$ V, $t_r = 0.5$ ms, and $V_{set} = 2$ V remained identical for all measurements. An overview of the pulse parameters is provided in Table I.

The maximum current during the read pulses (i.e., the read current I_r) is extracted from two different devices and plotted as a function of the pulse number N in Figs. 3(e) and 3(f). Furthermore, we define the maximum difference in the read current as indicated in Fig. 3(e) as $\Delta I_{r,max} = I_r(N = 100) - I_r(N = 1)$ and plot it as a function of the varied pulse parameters in Figs. 3(g) and 3(h).

All pulse curves in Figs. 3(e) and 3(f) show a quasi-continuous update behavior, where I_r continuously increases with N. This increase occurs mainly within approximately the first 20 periods before the slope reduces and I_r reaches its maximum value at $N = 100$. Within the selected parameter range of the first measurement set [Fig. 3(e)], the qualitative shape of the pulse curves remains similar for different parameter values, i.e., with increasing t_{set}, mainly the change in I_r per period increases, which correspondingly increases $\Delta I_{r,max}$ approximately linearly, as shown in Fig. 3(g).

For the second measurement set, where V_{anti} was varied, this trend is different. While the small magnitudes of V_{anti} result in a similar nonlinear conductance increase as before, large negative values of V_{anti} seem to increase the linearity of the set process [Fig. 3(f)] and simultaneously reduce $\Delta I_{r,max}$ [Fig. 3(h)]. Comparing I_r in Fig. 3(e) at $t_{set} = 0.1$ s with the read currents in Fig. 3(f) shows that the maximum read current of the second device is ~2.5 times larger than that of the first device. This difference is consistent with the

TABLE I. Overview of the two pulse parameter sets (sets 1 and 2) used for the measurement series shown in Figs. 3(e) and 3(f). A range of values are provided for the parameters varied between different pulse measurements.

Pulse parameter	Symbol	Set 1	Set 2
Read pulse voltage	V_r	0.5 V	0.5 V
Read pulse length	t_r	0.5 ms	0.5 ms
Set pulse voltage	V_{set}	2 V	2 V
Set pulse length	t_{set}	0.1–1.3 s	0.1 s
Antipulse voltage	V_{anti}	0 V	−1.2 to −0.1 V
Antipulse length	t_{anti}	0 s	0.1 s
Pause length	t_{off}	0 s	0.1 s

device-to-device variability observed in the measured I–V curves in Fig. 3(a).

In addition, the quasi-continuous update behavior of all conductance curves is consistent with the measured I–V curve and the suggested switching mechanism. The increase in I_r with t_{set} is expected because I_r increases with the pulse number N and t_{set} influence the exposure time of the device to the applied voltage. Hence, many short pulses would lead to a similar change in I_r as fewer long pulses. The dependency of I_r on the antipulse amplitude V_{anti} can be explained similarly. The negative antipulse counteracts the initial set pulse of every period and, thereby, reduces the influence of the set pulse on I_r compared to a pulse period without antipulse. This slows down the conductance update process and decreases $\Delta I_{r,max}$, and I_r appears more linear. Consequently, the increased linearity is expected to come at the expense of fewer distinguishable conductance states and a slower weight update process.

III. AI-DRIVEN MODEL

To predict the read current as a function of the pulse number during the set process, we set up a predictive model based on an ANN. In the following, we describe the dataset, its preparation, and the ANN architectures analyzed. We also present example predictions of the model for the weight-update process.

A. Architectures, dataset, and training

We use three different configurations (A_1–A_3) of a fully connected multilayer perceptron ANN, as illustrated in Fig. 4(a). All configurations comprise one input layer, two to four hidden layers, and one output layer. The smallest ANN configuration (A_1) uses two hidden layers with 513 neurons each. The second configuration (A_2) is of intermediate size with three hidden layers and 4673 neurons each, and the third configuration (A_3) is the largest, with four hidden layers and 8833 neurons per layer. All network configurations use rectified linear units as activation functions, and a dropout layer is added after each hidden layer with a dropout rate of 20% to prevent overfitting.

These ANN architectures lead to 513 (A_1), 4673 (A_2), and 8833 (A_3) network parameters that require 896 (A_1), 9088 (A_2), and 17 280 (A_3) floating point operations (FLOPs) per inference. Assuming a hardware implementation of these ANNs based on a Pulpissimo RISC-V core with an energy consumption of 9.3 pJ per FLOP,[62] we estimate an energy consumption of 8.33 nJ (A_1), 84.5 nJ (A_2), and 161 nJ (A_3) per inference. A summary is provided in Table II.

The input vector x of the ANN comprises six input features $x_{i,n}$, which are obtained by normalizing the pulse parameters N, t_{set}, V_{set}, t_{off}, t_r, and V_{anti} of the considered pulse period [Fig. 4(a)], via

$$x_{i,n} = \frac{x_i - \bar{x}_i}{\delta_i}, \quad \text{with } x_i \in \{N, t_{set}, V_{set}, t_{off}, t_r, V_{anti}\}, \tag{1}$$

with the mean $\bar{x}_i$ and the standard deviation δ_i taken over all values of the respective feature. As a target, we select the increment $\Delta I_{r,n}(N) = \Delta I_{r,n}(N+1) - I_{r,n}(N)$ of the scaled read current $I_{r,n}(N)$ per pulse. The scaled read current has values between 0 and 1 and is obtained from the min–max scaling

$$I_{r,n} = \frac{I_r - I_{r,min}}{I_{r,max} - I_{r,min}}, \tag{2}$$

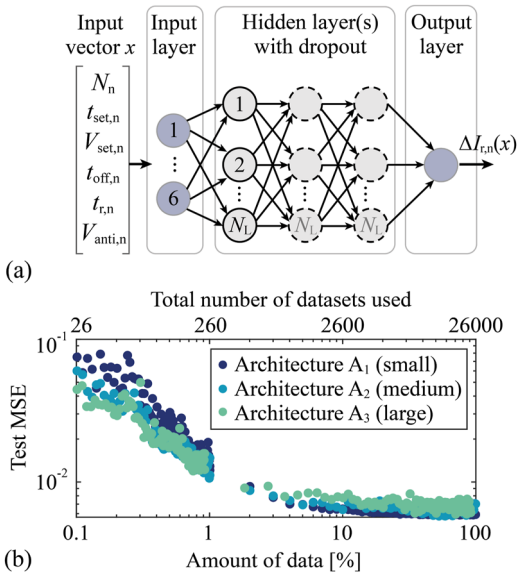

FIG. 4. (a) Illustration of the ANN architecture to predict the increment of the scaled read current $\Delta I_{r,n}(N)$ per pulse, from which the scaled read current $I_{r,n}(N)$ is obtained as a function of the normalized pulse number N_n and five other normalized pulse parameters $t_{set,n}$, $V_{set,n}$, $t_{off,n}$, $t_{r,n}$, and $V_{anti,n}$ [see Eq. (1) and Fig. 3(b)]. Depending on the respective configuration (A_1–A_3) of the ANN, either two, three, or four hidden layers are used with a total of 513, 4673, or 8833 trainable parameters (see Table II). (b) Test mean squared error (MSE) for the three different ANN configurations as a function of the number of datasets used for training and evaluation.

with the read current I_r, minimum read current $I_{r,min}$, and maximum read current $I_{r,max}$ of the considered dataset. Predicting $\Delta I_{r,n}$ then allows us to calculate the entire $I_{r,n}(N)$ curve by summing up the predicted scaled read current increments of a pulse sequence.

Measurements of set pulse trains with 100 set and read pulses were conducted on 13 devices to obtain a total of 260 pulse curves with variations in the pulse parameters, i.e., pulse amplitude and width. This leads to a total of 26 000 individual datasets, which were separated into three subsets used for training, validation, and testing with a ratio of 3:1:1. The training set was used for training the network, the validation set was used for tuning the hyperparameters by testing random variations, and the test dataset was used for

TABLE II. Summary of the implemented ANN architectures (A_1–A_3) including the number of floating-point operations (FLOPs) required and the estimated energy consumption per inference assuming a hardware implementation based on a Pulpissimo RISC-V core with an energy consumption of 9.3 pJ per FLOP.[62]

Architecture	A_1	A_2	A_3
No. of hidden layers	2	3	4
No. of neurons per hidden layer	64	64	64
No. of network parameters	513	4673	8833
FLOPs required per inference	896	9088	17 280
Energy per inference (nJ)	8.33	84.5	161

the final evaluations of the ANN performance. During training, the mean squared error (MSE) is used as a loss function, and the optimization is performed with the Adam optimizer over 1000 epochs with batch sizes of 64 and a learning rate of 10^{-4}.

B. Model evaluation

In the following, we identify the amount of data required for a sufficiently accurate match of the ANN predictions with the measurements. All three ANN configurations are trained and evaluated several times using varying amounts of datasets between 0.1% and 100% of the total available measurement data. This corresponds to ~26 to 26 000 datasets, i.e., 16–1560 training datasets. The resulting test errors are plotted in Fig. 4(b) as a function of the number of datasets used. The number of datasets used for training, test, and validation is provided in percentage relative to the total amount of measurement data on the bottom horizontal axis and in absolute numbers on the top horizontal axis. The test MSE error behaves similarly for all three ANN configurations. For small amounts of data, between ~0.1% and 1% (26–260 datasets), the test error is rapidly decreasing from just below 0.1 down to ~0.01–0.02 depending on the model size. For more data, between 1% and 100% (260–26 000 datasets), the magnitude of the slope reduces for all three model sizes and reaches approximately constant values between 0.006 and 0.008 for dataset numbers ≥30% (7800 datasets). Despite the similarities, slight quantitative differences are apparent between the test errors of the three model sizes. In particular, the relative performance of the three models depends on the amount of data used for training and evaluation. In the few-dataset regime between 0.1% and 1%, the large model (A_3) performs best, followed by the medium-sized model (A_2) and the small model (A_1). This trend is reversed for larger amounts of data (1%–100%). However, the differences in the test errors between the three models are small over the entire dataset size range. For example, at 0.1%, the test errors are 0.047 (A_3), 0.06 (A_2), and 0.07 (A_1), and for 20%, the test errors are 0.0068 (A_3), 0.0065 (A_2), and 0.006 (A_1).

To identify the maximum test error and minimum amount of data required for sufficiently accurate predictions of the weight-update characteristics, we compare the predictions of $I_{r,n}(N)$ with the measurements of the entire weight-update series of 100 pulses each. Each series was measured on a single device, and the individual data points were part of the test datasets. We use the small model (A_1) for predictions, trained and evaluated with the different amounts of data (from Fig. 4), and then calculate the MSE between the measured and predicted $I_{r,n}(N)$ as a quantitative measure of the performance of the model. The representative example curves are shown in Fig. 5(a) for 26 datasets (0.1%), in Fig. 5(b) for 260 datasets (1%), in Fig. 5(c) for 1300 datasets (5%), and in Fig. 5(d) for 2600 datasets (10%).

In the first case, with the smallest amount of available data [0.1%, Fig. 5(a)], the prediction fails to reflect the weight-update characteristics qualitatively with a significant mismatch over the entire pulse range. For 260 datasets [1%, Fig. 5(b)], the qualitative mismatch is reduced to small quantitative deviations, particularly apparent for the first twenty pulses. These deviations are notably reduced in the case of 1300 datasets [5%, Fig. 5(c)] and 2600 datasets [10%, Fig. 5(d)], leading to an excellent match of predictions with measurements. Hence, with the increasing amount of training data,

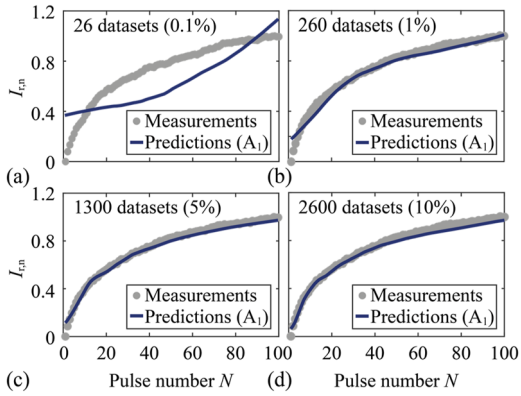

FIG. 5. (a) Comparison of the measured scaled read current $I_{r,n}$ over a series of 100 pulses from the test datasets with predictions from the small model (A_1) using 26 datasets for training, test, and validation (0.1%), (b) using 260 datasets (1%), (c) using 1300 datasets (5%), and (d) using 2600 datasets (10%) of the total available measurement data.

the predictions improve until ~5% of the available data are used. This is consistent with the decreasing test MSE in Fig. 4(b). Correspondingly, the MSE between the measured weight update curve and the prediction decreases from 28.6×10^{-3} (0.1%) to 1.3×10^{-3} (1%), 0.65×10^{-3} (5%), and 0.8×10^{-3} (10%). These results imply that 260–1600 datasets are sufficient to predict the weight-update characteristics of the devices accurately. This corresponds to 160–800 training datasets.

IV. DISCUSSION AND CONCLUSION

This work presents an AI-driven model for the energy-efficient programming of memristive devices. High throughput measurements of HfO_2/TiO_2 memristive devices were performed to characterize the set process and provide data for training various configurations of a multilayer perceptron ANN model. This model is validated and tested with weight-update measurements and analyzed regarding the amount of training data required for making accurate predictions.

The results demonstrate that small model architectures can provide accurate predictions of the weight update characteristic of our devices while using only a small number of datasets between 260 and 2600. Minor deviations between measurements and predictions at the beginning of the pulse programming series reduce significantly until ~1300 datasets are reached. Moreover, comparing the different ANN sizes demonstrates no significant advantage of the larger model configurations over the small ones. While the two larger configurations perform slightly better for small amounts of training data, the number of FLOPs and the energy consumption per inference are estimated to be approximately ten and twenty times larger than for the small model configuration.

For the small model configuration, we estimated an energy consumption of <10 nJ per inference when integrating it into a Pulpissimo RISC-V core. For another recent microcontroller with an integrated AI engine (MAX78000), 0.09 mJ is reported per inference.[31] This controller was optimized for large ANNs with

$\sim 4.5 \times 10^6$ weights, which is order of magnitude larger than the architectures we propose here. Hence, integrating our network into an AI-enhanced controller for memristive devices that can perform online adjustments to a programming algorithm appears feasible.

In conclusion, the results demonstrate that accurate and computationally inexpensive predictions are possible with comparatively few datasets and small networks. This makes the proposed model a potential candidate as a component in an AI-enhanced device controller.

ACKNOWLEDGMENTS

This research was funded by the Carl-Zeiss Foundation via the project Memwerk, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)(Project ID 434434223), SFB 1461 and HYB-RISC (Project No. 536099247), which is part of the SPP MemrisTec (Project No. 422738993), and the Federal Ministry of Education and Research of Germany under Grant Nos. 16KISK026 (6G-RIC) and 16KIS2067K (DI-SIGN-HEP).

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

B.S. and M.F. contributed equally to this work.

Benjamin Spetzler: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Validation (equal); Visualization (lead); Writing – original draft (equal); Writing – review & editing (equal). **Markus Fritscher**: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Validation (equal); Visualization (supporting); Writing – original draft (equal); Writing – review & editing (equal). **Seongae Park**: Conceptualization (supporting); Data curation (equal); Formal analysis (supporting); Investigation (supporting); Methodology (supporting); Writing – review & editing (supporting). **Nayoun Kim**: Conceptualization (supporting); Data curation (equal); Formal analysis (supporting); Investigation (supporting); Methodology (equal); Writing – review & editing (supporting). **Christian Wenger**: Conceptualization (supporting); Project administration (equal); Supervision (equal); Writing – review & editing (equal). **Martin Ziegler**: Conceptualization (supporting); Project administration (equal); Supervision (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are openly available in GitHub, at https://github.com/fritscherm/apl-rram-nn-codes, reference number https://doi.org/10.5281/zenodo.14507591.

REFERENCES

[1] N. Jones, Nature **561**, 163 (2018).
[2] H.-Y. Lin, Computer **54**, 95 (2021).
[3] P. Dhar, Nat. Mach. Intell. **2**, 423 (2020).
[4] B. Bailey, AI power consumption exploding, https://semiengineering.com/ai-power-consumption-exploding.
[5] J. Bughin, J. Seong, J. Manyika, M. Chui, and R. Joshi, Discussion Paper, McKinsey Global Institute, 2018.
[6] M. Rao, H. Tang, J. Wu, W. Song, M. Zhang, W. Yin, Y. Zhuo, F. Kiani, B. Chen, X. Jiang, H. Liu, H.-Y. Chen, R. Midya, F. Ye, H. Jiang, Z. Wang, M. Wu, M. Hu, H. Wang, Q. Xia, N. Ge, J. Li, and J. J. Yang, Nature **615**, 823 (2023).
[7] A. Gebregiorgis, A. Singh, A. Yousefzadeh, D. Wouters, R. Bishnoi, F. Catthoor, and S. Hamdioui, Memories - Mater., Devices, Circuits Syst. **4**, 100025 (2023).
[8] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, Nat. Electron. **1**, 216 (2018).
[9] C. Sung, H. Hwang, and I. K. Yoo, J. Appl. Phys. **124**, 151903 (2018).
[10] S. Choi, J. Yang, and G. Wang, Adv. Mater. **32**, e2004659 (2020).
[11] J. Li, H. Abbas, D. S. Ang, A. Ali, and X. Ju, Nanoscale Horiz. **8**, 1456 (2023).
[12] J. J. Yang, D. B. Strukov, and D. R. Stewart, Nat. Nanotechnol. **8**, 13 (2013).
[13] M. A. Zidan, J. P. Strachan, and W. D. Lu, Nat. Electron. **1**, 22 (2018).
[14] M.-K. Song, J.-H. Kang, X. Zhang, W. Ji, A. Ascoli, I. Messaris, A. S. Demirkol, B. Dong, S. Aggarwal, W. Wan, S.-M. Hong, S. G. Cardwell, I. Boybat, J.-s. Seo, J.-S. Lee, M. Lanza, H. Yeon, M. Onen, J. Li, B. Yildiz, J. A. Del Alamo, S. Kim, S. Choi, G. Milano, C. Ricciardi, L. Alff, Y. Chai, Z. Wang, H. Bhaskaran, M. C. Hersam, D. Strukov, H.-S. P. Wong, I. Valov, B. Gao, H. Wu, R. Tetzlaff, A. Sebastian, W. Lu, L. Chua, J. J. Yang, and J. Kim, ACS Nano **17**, 11994 (2023).
[15] C. Weilenmann, A. N. Ziogas, T. Zellweger, K. Portner, M. Mladenović, M. Kaniselvan, T. Moraitis, M. Luisier, and A. Emboras, Nat. Commun. **15**, 6898 (2024).
[16] S. Yu, Proc. IEEE **106**, 260 (2018).
[17] I. Boybat, M. Le Gallo, S. R. Nandakumar, T. Moraitis, T. Parnell, T. Tuma, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, Nat. Commun. **9**, 2514 (2018).
[18] Y. Huang, T. Ando, A. Sebastian, M.-F. Chang, J. J. Yang, and Q. Xia, Nat. Rev. Electr. Eng. **1**, 286 (2024).
[19] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, and D. B. Strukov, Nature **521**, 61 (2015).
[20] M. Lanza, A. Sebastian, W. D. Lu, M. Le Gallo, M.-F. Chang, D. Akinwande, F. M. Puglisi, H. N. Alshareef, M. Liu, and J. B. Roldan, Science **376**, eabj9979 (2022).
[21] Q. Xia and J. J. Yang, Nat. Mater. **18**, 309 (2019).
[22] F. Aguirre, A. Sebastian, M. Le Gallo, W. Song, T. Wang, J. J. Yang, W. Lu, M.-F. Chang, D. Ielmini, Y. Yang, A. Mehonic, A. Kenyon, M. A. Villena, J. B. Roldán, Y. Wu, H.-H. Hsu, N. Raghavan, J. Suñé, E. Miranda, A. Eltawil, G. Setti, K. Smagulova, K. N. Salama, O. Krestinskaya, X. Yan, K.-W. Ang, S. Jain, S. Li, O. Alharbi, S. Pazos, and M. Lanza, Nat. Commun. **15**, 1974 (2024).
[23] G. C. Adam, A. Khiat, and T. Prodromakis, Nat. Commun. **9**, 5267 (2018).
[24] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, Nat. Nanotechnol. **15**, 529 (2020).
[25] P. Yao, H. Wu, B. Gao, S. B. Eryilmaz, X. Huang, W. Zhang, Q. Zhang, N. Deng, L. Shi, H.-S. P. Wong, and H. Qian, Nat. Commun. **8**, 15199 (2017).
[26] C. Li, Z. Wang, M. Rao, D. Belkin, W. Song, H. Jiang, P. Yan, Y. Li, P. Lin, M. Hu, N. Ge, J. P. Strachan, M. Barnell, Q. Wu, R. S. Williams, J. J. Yang, and Q. Xia, Nat. Mach. Intell. **1**, 49 (2019).
[27] X. Liang, Y. Zhong, J. Tang, Z. Liu, P. Yao, K. Sun, Q. Zhang, B. Gao, H. Heidari, H. Qian, and H. Wu, Nat. Commun. **13**, 1549 (2022).
[28] K. Jeon, J. J. Ryu, S. Im, H. K. Seo, T. Eom, H. Ju, M. K. Yang, D. S. Jeong, and G. H. Kim, Nat. Commun. **15**, 129 (2024).
[29] S. Ambrogio, P. Narayanan, A. Okazaki, A. Fasoli, C. Mackin, K. Hosokawa, A. Nomura, T. Yasuda, A. Chen, A. Friz, M. Ishii, J. Luquin, Y. Kohda, N. Saulnier, K. Brew, S. Choi, I. Ok, T. Philip, V. Chan, C. Silvestre, I. Ahsan, V. Narayanan, H. Tsai, and G. W. Burr, Nature **620**, 768 (2023).
[30] T. Wang, J. Meng, X. Zhou, Y. Liu, Z. He, Q. Han, Q. Li, J. Yu, Z. Li, Y. Liu, H. Zhu, Q. Sun, D. W. Zhang, P. Chen, H. Peng, and L. Chen, Nat. Commun. **13**, 7432 (2022).

[31] M. Giordano, L. Piccinelli, and M. Magno, "Survey and comparison of milli-watts micro controllers for tiny machine learning at the edge," in *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)* (IEEE, 2022), p. 94.
[32] E. J. Fuller, S. T. Keene, A. Melianas, Z. Wang, S. Agarwal, Y. Li, Y. Tuchman, C. D. James, M. J. Marinella, J. J. Yang, A. Salleo, and A. A. Talin, Science **364**, 570 (2019).
[33] S. Tu, J. Li, Y. Ren, Q. Jiang, and S. Xiong, Microelectron. Eng. **280**, 112072 (2023).
[34] E. Perez, M. K. Mahadevaiah, E. P.-B. Quesada, and C. Wenger, IEEE Trans. Electron Devices **68**, 2693 (2021).
[35] M. Fritscher, A. Veronesi, A. Baroni, J. Wen, T. Spätling, M. K. Mahadevaiah, N. Herfurth, E. Perez, M. Ulbricht, M. Reichenbach, A. Hagelauer, and M. Krstic, in *High Performance Computing*, edited by A. Bienz, M. Weiland, M. Baboulin and C. Kruse (Springer Nature, Switzerland, Cham, 2023), Vol. 13999, p. 500.
[36] M. Fritscher, J. Knödtel, M. Mallah, S. Pechmann, E. P.-B. Quesada, T. Rizzi, C. Wenger, and M. Reichenbach, in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, edited by A. Orailoglu, M. Jung and M. Reichenbach (Springer International Publishing, Cham, 2022), Vol. 13227, p. 401.
[37] Y. Elul, E. Rozenberg, A. Boyarski, Y. Yaniv, A. Schuster, and A. M. Bronstein, Commun. Phys. **7**, 141 (2024).
[38] O. Azencot, N. B. Erichson, V. Lin, and M. W. Mahoney, "Forecasting sequential data using consistent Koopman autoencoders," arXiv:2003.02236 (2020).
[39] Y. Zhao, C. Jiang, M. A. Vega, M. D. Todd, and Z. Hu, J. Comput. Inf. Sci. Eng. **23**, 011001 (2023).
[40] B. Lusch, J. N. Kutz, and S. L. Brunton, Nat. Commun. **9**, 4950 (2018).
[41] M. Fritscher, S. Singh, T. Rizzi, A. Baroni, D. Reiser, M. Mallah, D. Hartmann, A. Bende, T. Kempen, M. Uhlmann, G. Kahmen, D. Fey, V. Rana, S. Menzel, M. Reichenbach, M. Krstic, F. Merchant, and C. Wenger, Sci. Rep. **14**, 23695 (2024).
[42] A. Latreche, SN Appl. Sci. **1**, 188 (2019).
[43] B. Spetzler, D. Abdel, F. Schwierz, M. Ziegler, and P. Farrell, Adv. Electron. Mater. **10**, 2300635 (2024).
[44] A. Marchewka, B. Roesgen, K. Skaja, H. Du, C.-L. Jia, J. Mayer, V. Rana, R. Waser, and S. Menzel, Adv. Electron. Mater. **2**, 1500233 (2016).
[45] M. Sivan, J. F. Leong, J. Ghosh, B. Tang, J. Pan, E. Zamburg, and A. V.-Y. Thean, ACS Nano **16**, 14308 (2022).
[46] A. K. Parit, M. S. Yadav, A. K. Gupta, A. Mikhaylov, and B. Rawat, Chaos, Solitons Fractals **145**, 110818 (2021).
[47] B. Spetzler, V. K. Sangwan, M. C. Hersam, and M. Ziegler, npj 2D Mater. Appl. **9**, 17 (2025).
[48] J. Jiménez-León, L. A. Sarmiento-Reyes, and P. Rosales-Quintero, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **41**, 4851 (2022).
[49] M. Saludes-Tapia, M. B. Gonzalez, F. Campabadal, J. Suñé, and E. Miranda, Solid-State Electron. **185**, 108083 (2021).
[50] S. Park, B. Spetzler, T. Ivanov, and M. Ziegler, Sci. Rep. **12**, 18266 (2022).
[51] B. Spetzler, Z. Geng, K. Rossnagel, M. Ziegler, and F. Schwierz, "Lateral 2D TMDC memristors—Experiment and modeling," in *2022 IEEE 16th International Conference on Solid-State & Integrated Circuit Technology (ICSICT)* (IEEE, 2022), p. 1.
[52] J. Singh and B. Raj, Eng. Sci. Technol., Int. J. **21**, 862 (2018).
[53] A. Malik, C. Papavassiliou, and S. Stathopoulos, "A stochastic compact model describing memristor plasticity and volatility," in *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)* (IEEE, 2021), p. 1.
[54] C. La Torre, A. F. Zurhelle, T. Breuer, R. Waser, and S. Menzel, IEEE Trans. Electron Devices **66**, 1268 (2019).
[55] R. González-García, R. Rico-Martínez, and I. G. Kevrekidis, Comput. Chem. Eng. **22**, S965–S968 (1998).
[56] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering*, 2nd ed. (Cambridge University Press, Cambridge, 2022).
[57] Y. Sha, J. Lan, Y. Li, and Q. Chen, Electronics **12**, 2906 (2023).
[58] Y. Lee, K. Kim, and J. Lee, Micromachines **15**, 253 (2024).
[59] F. L. Aguirre, E. Piros, N. Kaiser, T. Vogel, S. Petzold, J. Gehrunger, T. Oster, C. Hochberger, J. Suñé, L. Alff, and E. Miranda, Micromachines **13**, 2002 (2022).
[60] S. Saha, M. C. Kodand Reddy, T. S. Nikhil, K. Burugupally, S. DebRoy, A. Salimath, V. Mattela, S. S. Dan, and P. Sahatiya, Chip **2**, 100075 (2023).
[61] S. Park, S. Klett, T. Ivanov, A. Knauer, J. Doell, and M. Ziegler, Front. Nanotechnol. **3**, 670762 (2021).
[62] F. Schuiki, M. Schaffner, and L. Benini, "NTX: An energy-efficient streaming accelerator for floating-point generalized reduction workloads in 22 nm FD-SOI," in *Proceedings of the 2019 Design, Automation & Test in Europe (DATE). 25–29 March 2019, Florence, Italy* (IEEE, Piscataway, NJ, 2019), p. 662.