Accelerating Machine Learning with Non-Volatile Memory: exploring device and circuit tradeoffs

Alessandro Fumarola*‡, Pritish Narayanan*, Lucas L. Sanches*, Severin Sidler*‡, Junwoo Jang*†,
Kibong Moon†, Robert M. Shelby*, Hyunsang Hwang†, and Geoffrey W. Burr*

*IBM Research–Almaden, 650 Harry Road, San Jose, CA 95120, Tel: (408) 927–1512, Email: *gwburr@us.ibm.com*
†Department of Material Science and Engineering, Pohang University of Science and Technology, Pohang 790-784, Korea
‡EPFL, Lausanne, CH–1015 Switzerland

***Abstract*—Large arrays of the same nonvolatile memories (NVM) being developed for Storage-Class Memory (SCM) – such as Phase Change Memory (PCM) and Resistance RAM (ReRAM) – can also be used in non-Von Neumann neuromorphic computational schemes, with device conductance serving as synaptic "weight." This allows the all-important multiply-accumulate operation within these algorithms to be performed efficiently at the weight data.**

In contrast to other groups working on Spike-Timing Dependent Plasticity (STDP), we have been exploring the use of NVM and other inherently-analog devices for Artificial Neural Networks (ANN) trained with the backpropagation algorithm. We recently showed a large-scale (165,000 two-PCM synapses) hardware-software demo (IEDM 2014, [1], [2]) and analyzed the potential speed and power advantages over GPU-based training (IEDM 2015, [3]).

In this paper, we extend this work in several useful directions. We assess the impact of undesired, time-varying conductance change, including drift in PCM and leakage of analog CMOS capacitors. We investigate the use of non-filamentary, bidirectional ReRAM devices based on PrCaMnO, with an eye to developing material variants that provide suitably linear conductance change. And finally, we explore tradeoffs in designing peripheral circuitry, balancing simplicity and area-efficiency against the impact on ANN performance.

I. Introduction

By performing computation at the location of data, non-Von Neumann (non–VN) computing *ought* to provide significant power and speed benefits (Fig. 1) on specific and assumably important tasks. For one such non–VN approach — on-chip training of large-scale ANN using NVM-based synapses [1]–[4] — viability will require several things. First, despite the inherent imperfections of NVM devices such as Phase Change Memory (PCM) [1], [2] or Resistive RAM (RRAM) [4], such NVM-based networks must achieve competitive performance levels (e.g., classification accuracies) when compared to ANN trained using CPUs or GPUs. Second, the benefits of performing computation at the data (Fig. 2) must confer a decided advantage in either training power or speed (or preferably, both). And finally, any on-chip accelerator should be applicable towards networks of different types (fully–connected "Deep" NN or Convolutional NN) and/or be reconfigurable for networks of different shapes (wide, with many neurons, or deep, with many layers).

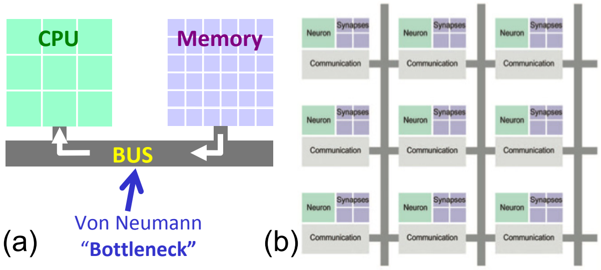


Fig. 1. In the Von Neumann architecture (a), data (both operations and operands) must move to and from the dedicated Central Processing Unit (CPU) along a bus. In contrast, in a Non–Von Neumann architecture, distributed computations take place at the location of the data, reducing the time and energy spent moving data around [1].

We briefly review our work [1]–[4] in assessing the accuracy, speed and power potential of on-chip NVM–based ML.

A. Comparative analysis of speed and power

We have previously assessed the potential advantages, in terms of speed and power, of on-chip machine learning (ML) of large-scale artificial neural networks (ANN) using Non-Volatile Memory (NVM)-based synapses, in comparison to conventional GPU–based hardware [3].

Under moderately-aggressive assumptions for parallel–read and –write speed, **PCM-based on-chip machine learning can**

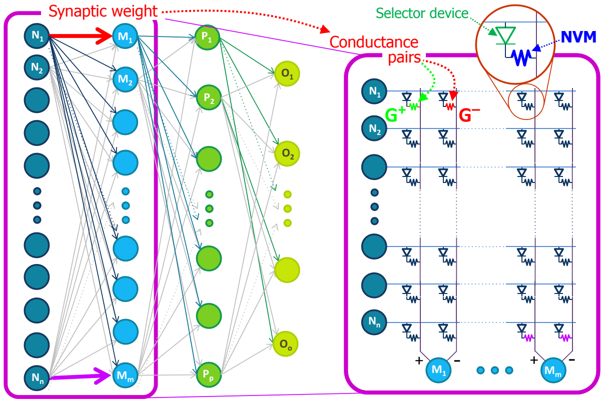


Fig. 2. Neuro-inspired non-Von Neumann computing [1]–[4], in which neurons activate each other through dense networks of programmable synaptic weights, can be implemented using dense crossbar arrays of nonvolatile memory (NVM) and selector device-pairs [1].

978-1-5090-1370-8/16/$31.00 © 2016 IEEE

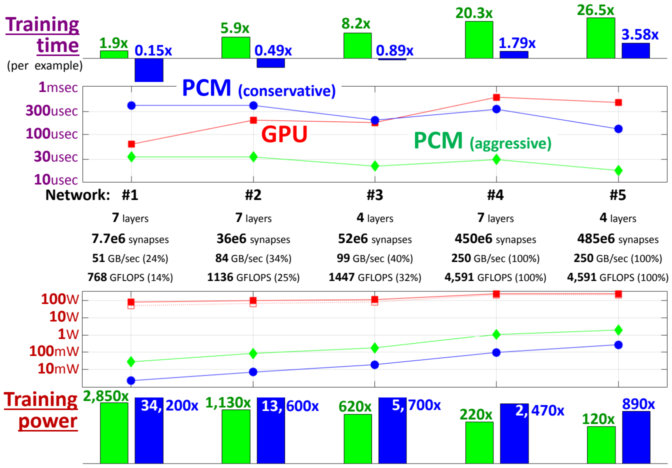


Fig. 3. Predicted training time (per ANN example) and power for 5 ANNs, ranging from 0.2GB to nearly 6GB [3]. Under moderately-aggressive assumptions for parallel–read and –write speed, PCM-based on-chip machine learning can offer lower power and faster training for both large and small networks [3].

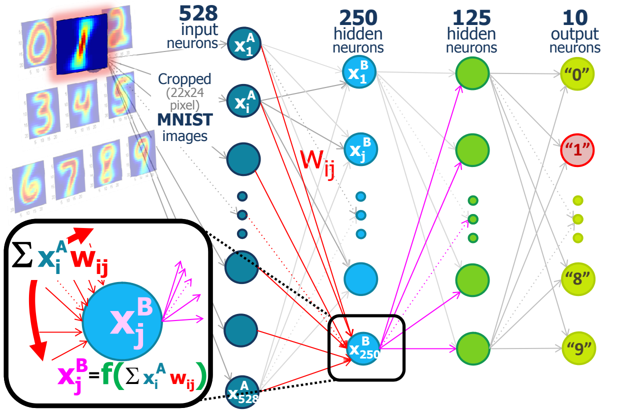


Fig. 4. In forward evaluation of a multilayer perceptron, each layer's neurons drive the next layer through weights w_{ij} and a nonlinearity $f()$. Input neurons are driven by input (for instance, pixels from successive MNIST images (cropped to 22×24)); the 10 output neurons classify which digit was presented [1].

potentially offer lower power and faster training (per ANN example) than GPU-based training for both large and small networks (Fig. 3), even with the time and energy required for occasional RESET (forced by the large asymmetry between gentle partial-SET and abrupt RESET in PCM). Critical here is the design of area-efficient read/write circuitry, so that many copies of this circuitry operate in parallel (each handling a small number of columns (rows), $\mathbf{c}_s$).

B. Potential for competitive classification accuracies

Using 2 phase-change memory (PCM) devices per synapse, we demonstrated a 3–layer perceptron (fully-connected ANN) with 164,885 synapses [1], trained with backpropagation [5] on a subset (5000 examples) of the MNIST database of handwritten digits [6] (Fig. 4), using a modified weight-update rule compatible with NVM+selector crossbar arrays [1]. We proved that this weight-update modification does not degrade the high "test" (generalization) accuracies such a 3–layer network inherently delivers on this problem when trained in software [1]. However, nonlinearity and asymmetry in PCM conductance response limited both "training" and "test" accuracy in our original, mixed hardware-software experiments to 82–83% [1] (Fig. 5).

Asymmetry (between the gentle conductance increases of PCM partial–SET and the abruptness of PCM RESET) was mitigated by an occasional RESET strategy, which could be both infrequent and inaccurate [1]. While in these initial experiments, network parameters such as learning rate η had to be tuned very carefully, a modified 'LG' algorithm offered wider tolerance to η, higher classification accuracies, and lower training energy [3] (Fig. 6).

Tolerancing results showed that all NVM-based ANN can be expected to be **highly resilient to random effects** (NVM variability, yield, and stochasticity), but **highly sensitive to "gradient" effects that act to steer all synaptic weights** [1]. We showed that a bidirectional NVM with a symmetric, linear conductance response of finite but large dynamic range (e.g., each conductance step is relatively small) can deliver the same high classification accuracies on the MNIST digits as

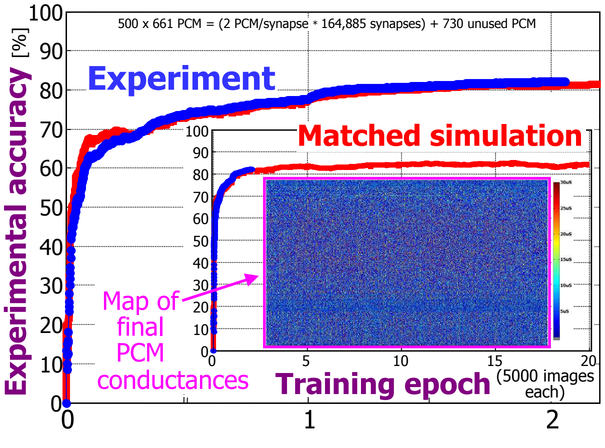


Fig. 5. Training accuracy for a 3–layer perceptron of 164,885 hardware-synapses [1], with all weight operations taking place on a 500 × 661 array of mushroom-cell PCM devices. Also shown is a matched computer simulation of this NN, using parameters extracted from the experiment [1].

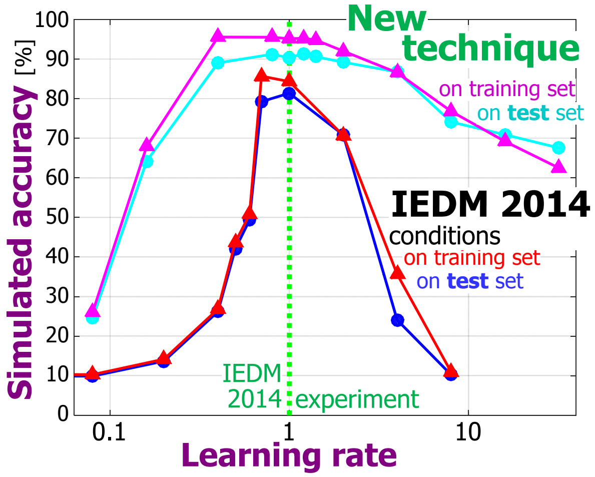


Fig. 6. A large number of synapses tend to "dither," with frequent updates whose aggregate effect *ought* to be zero (but which is non-zero due to the nonlinearity and asymmetry of NVM–based synapses). By suppressing update of such synapses, NN performance can be improved and training energy reduced, while reducing the need to tune the learning rate precisely.

a conventional, software-based implementation (Fig. 7). One key observation is the importance of avoiding constraints on weight magnitude that arise when the two conductances are either both small or both large — e.g., synapses should remain in the center stripe of the "G-diamond" [2].

In this paper, we extend upon these observations and address several different yet useful topics. We assess the impact of undesired, time-varying conductance change, including drift in Phase Change Memory (PCM) and leakage of analog CMOS capacitors. We investigate the use of non-filamentary, bidirectional ReRAM devices based on PrCaMnO (PCMO), with an eye to developing material variants that provide suitably linear conductance change. And finally, we explore tradeoffs in designing peripheral circuitry, balancing simplicity and area-efficiency against the impact on ANN performance.

C. Jump-table concept

A highly useful concept in modeling the behavior of real NVM devices for neuromorphic applications is the concept of a "jump-table." For backpropagation training, where one or more copies of the same programming pulse are applied to the NVM for adjusting the weights [1], we simply need one jump-table for potentiation (SET) and one for depression (RESET).

With a pair of such jump-tables, we can capture the nonlinearity of conductance response as a function of conductance (e.g., the same pulse might create a large "jump" at low conductance, but a much smaller jump at high conductance), the asymmetry between positive (SET) and negative (RESET) conductance changes, and the inherent stochastic nature of each jump. Fig. 8(a) plots median conductance change for potentiation (blue) together with the $\pm 1\sigma$ stochastic variation about this median change (red). Fig. 8(b) shows the jump-table that fully captures this conductance response, plotting the cumulative probability (in color, from 0 to 100%) of any conductance change ΔG at any given initial conductance G. This table is ideal for computer simulation because a random number r (uniform deviate, between 0.0 and 1.0) can be converted to a resulting ΔG produced by a single pulse by scanning along the row associated with the conductance G (of the device before the pulse is applied) to find the point at which the table entry just exceeds r.

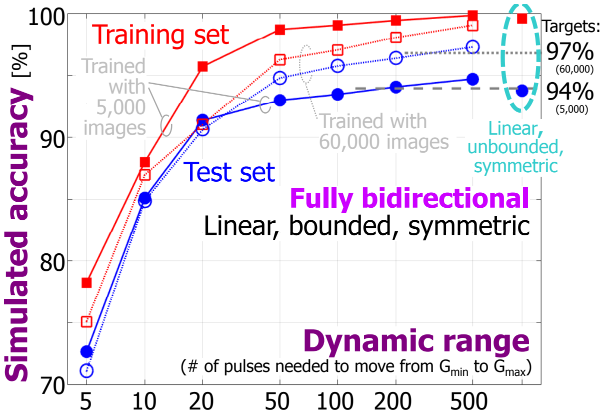


Fig. 7. When the dynamic range of the linear response is large, the classification accuracy can now reach the peak accuracy supported by the original neural network (a *test* accuracy of 94% when trained with 5,000 images; of 97% when trained with all 60,000 images) [2].

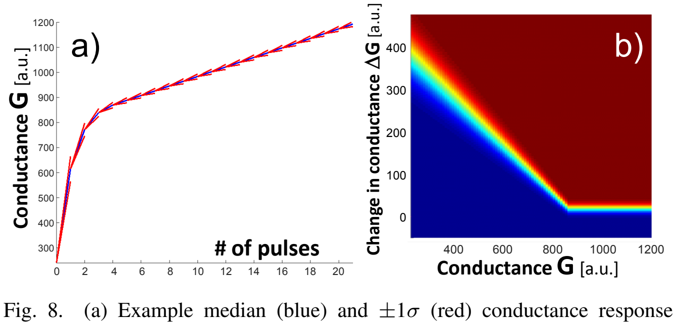


Fig. 8. (a) Example median (blue) and $\pm 1\sigma$ (red) conductance response for potentiation. (b) associated jump-table that fully captures this (artificially constructed in this case) conductance response, with cumulative probability plotted in color (from 0 to 100%) of any conductance change ΔG at any given initial conductance G.

We have previously used a measured jump-table to simulate the SET response of PCM devices [1]. We have recently published a study of various artificially-constructed jump-tables, in order to help develop an intuitive understanding of the impact that various features of such jump-tables have on the classification performance in the ANN application [7].

II. Time-dependent conductance response

One aspect of Phase Change Memory that we did not address in our original tolerancing paper [1] was the role of resistance drift [8], also known as amorphous relaxation. As shown in Fig. 9, after a RESET operation, amorphous relaxation causes conductances to decrease, rapidly at first but then more and more slowly. Here we model this in our Neural Network simulator for the network of Fig. 4, for an otherwise near-perfect PCM device, in which partial-SET conductance increases are gentle and linear (each ∼0.5% of the conductance extent) and Occasional-RESET is performed fairly frequently (every 100 examples) with high precision. The time response for drift starts upon RESET operations, with partial-SET operations assumed only to shift the conductance states without affecting the underlying time-response of the amorphous relaxation.

As expected, as drift coefficients increase dramatically (to the values of $\nu \sim 0.1$ observed for fully amorphous (strong

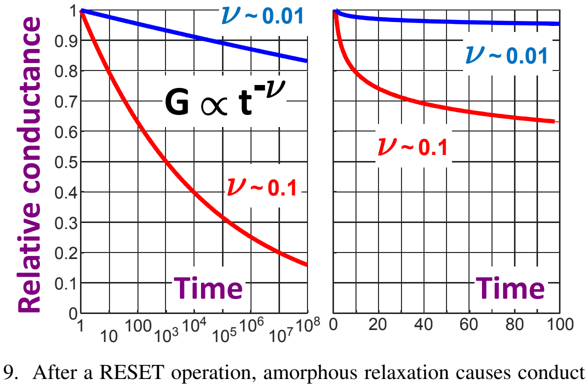


Fig. 9. After a RESET operation, amorphous relaxation causes conductances to decrease, rapidly at first but then more and more slowly. Plots show the same evolution of linear conductance as a function of time on log- (left) and linear- (right) scales, for two different values of drift coeffcient ν.

RESET) states), then accuracy is eventually affected (Fig. 10). However, for the much lower ν values ($\nu \sim 0.005 - 0.01$) associated with SET and the near-SET states relevant to PCM-based implementations of neural networks, accuracy is only minimally affected.

We performed a similar study for the case of fully volatile analog memory elements, such as CMOS capacitors, in which any conductance state immediately begins to decay towards zero after a programming operation. This study was performed with perfectly linear bidirectional conductances with ~0.5% conductance change per pulse, and thus without drift, is identical to the right-hand side of Fig 7, where accuracy becomes extremely high for high synaptic dynamic range.

In this study, we quantify the effective decay constant (the "RC time constant") not in absolute units, but relative to the time required for training of a single data example (e.g., forward evaluation, reverse propagation, and weight update). As shown in Fig. 11, accuracy is strongly affected as soon as the ratio between the RC time-constant and the time-per-example falls below 10,000. However, these initial results revealed an extremely interesting dependence on the choice of learning rate, implying that some further optimization may be possible. Fig. 12 shows that the same global learning rate which is optimal for a truly non-volatile conductance (infinite RC time-constant) is decidedly sub-optimal when the RC time-constant becomes lower. This implies that it is better to either update so many weights that one can counteract the loss of conductance by retraining those weights, or so few that the number of weights being touched (and thus placed into a mode where they will decay rapidly) is much lower.

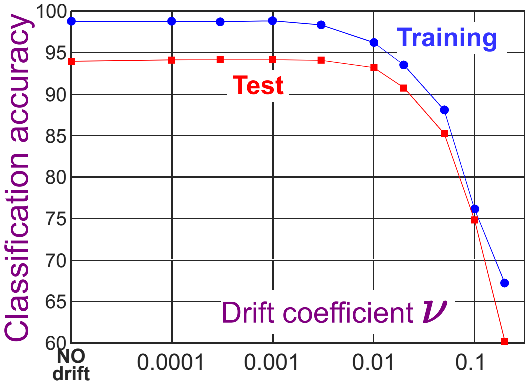


Fig. 10. At the large drift coefficients associated with fully-amorphous RESET phase change memory devices ($\nu \sim 0.1$), neural network accuracy is significantly degraded. At the values of $\nu \sim 0.005$–0.01 relevant to the SET and the near-SET states that dominate the PCM-based implementations of neural networks, accuracy is only slightly degraded. (Results shown for 10 epochs of simulated training on 5000 examples from the MNIST dataset.)

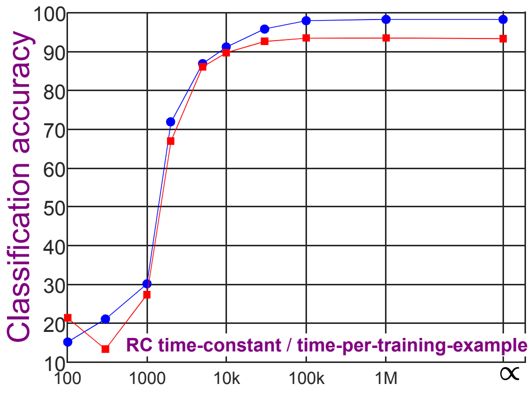


Fig. 11. Neural network accuracy is strongly affected as soon as the RC time-constant becomes less than 10,000× larger than the time needed for each training example. (Results shown for 10 epochs of simulated training on 5000 examples from the MNIST dataset, all at the same global learning rate, $\eta \sim 1$).

III. Impact of measured PCMO conductance response

We have previously studied the impact of the conductance response of PCMO material by fitting a set of functions to the average conductance response [4]. However, this approach is limited by the discrepancy between the real conductance response and the function chosen, and it does not include any stochastic aspect of the conductance response, for scenarios where the conductance response can vary significantly from the average conductance response.

Here, we study the use of measured jump-tables for the non-filamentary RRAM material $\mathrm{Pr}_x\mathrm{Ca}_{1-x}\mathrm{MnO}_3$, also known as PCMO.

A. Analog Bidirectional Switching

Resistive switching in PCMO-based devices is caused by slow and gradual drift of oxygen ions and vacancies in the polycrystalline PCMO layer. Injection (removal) of oxygen ions takes place at the PCMO-oxide (-metal) interface through oxidation (reduction) reactions. Asymmetry in the device structure and the oxidation-reduction reactions contribute to the asymmetry in the switching characteristics, but PCMO-based NVMs show gradual SET and RESET characteristics. Thus, unlike Phase Change Memory (PCM) materials, there is no need to stop training and perform an Occasional–RESET operation [1]. Both the average conductance response and its statistical behavior can be described by a measured jump table (Fig. 13). (Note that unlike non-filamentary RRAM such as

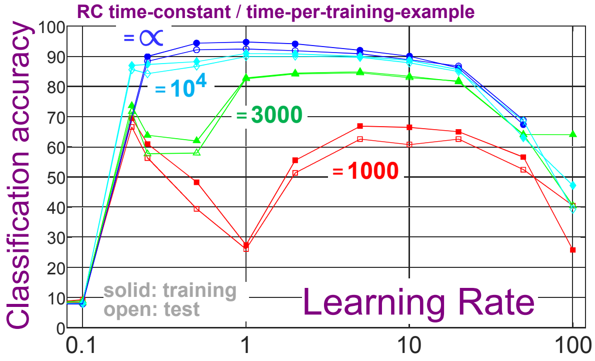


Fig. 12. For truly non-volatile weights (infinite RC time constant), neural network accuracy is optimized by using a global learning rate that is large enough to affect a moderate number of weights, but not so many that chaos ensues. However, as the RC time constant decreases, the volatility of the conductance states favors either a larger learning rate (e.g., we adjust for the decaying weights by retraining many more of them) or, curiously, *lower* learning rates (assumably reducing the number of recently-touched weights that cannot be trusted not to move without being actively programmed).

PCMO, a filament-based RRAM such as HfOx, TaOx, or TiOx exhibits only gradual RESET characteristics, meaning that such filamentary RRAM devices will likely still require an "Occasional–SET" step just like PCM.)

B. Fabrication process

A 10nm PCMO polycrystalline layer was deposited on a 50-nm-thick Pt layer, which served as bottom electrode. Next, an 100-nm-thick SiNx layer was deposited by plasma-enhanced chemical vapor deposition, and via-holes (from 0.15 to 1.0 μm) were formed by conventional lithography and reactive ion etching. The Al and Mo layers (20nm and 3nm, respectively) and an 50-nm-thick Pt layer (top electrode) were deposited and patterned by conventional lithography. Electrical characteristics of the Al/Mo/PCMO-based resistive memory devices were measured using an Agilent B1500A.

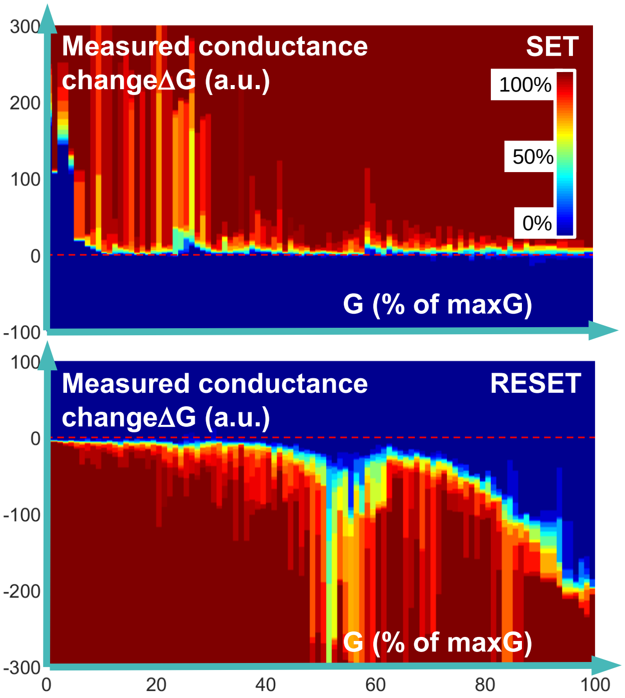


Fig. 13. Jump-table of Al/Mo/PCMO-based RRAM devices for positive (SET) and negative (RESET) conductance changes. Unlike Phase Change Memory (PCM) devices, these materials provide both gradual RESET and gradual SET, enabling truly bidirectional programming. 50000 total SET pulses (-4.0V, 10ms) and RESET pulses (3.5V, 10ms) followed by -1V read pulses were used on three identically-sized (200nm) devices.

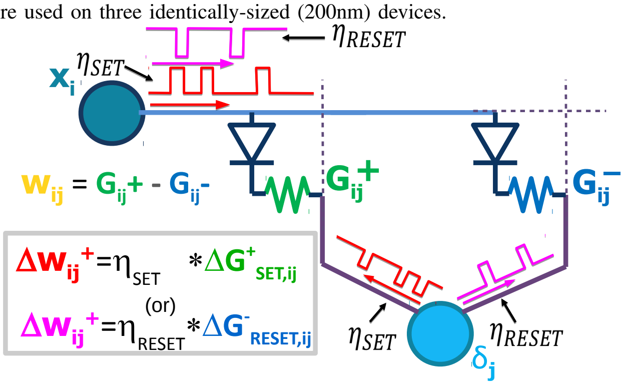

η_{RESET}
η_{SET}
x_i
$w_{ij} = G_{ij}+ - G_{ij}-$
G_{ij}^+
G_{ij}^-
$\Delta w_{ij}^+ = \eta_{SET} * \Delta G^+_{SET,ij}$
(or)
$\Delta w_{ij}^+ = \eta_{RESET} * \Delta G^-_{RESET,ij}$
η_{SET}
η_{RESET}
δ_j

Fig. 14. Schematic showing crossbar-compatible [1] weight-update rule for Analog bidirectional NVMs. Weight increases (decreases) can be implemented either as a SET operation on G^+ (G^-) or a RESET operation on G^- (G^+) devices. Asymmetry in the partial SET and RESET operation is compensated by applying a different learning rate parameter (η_{SET}, η_{RESET}) that modulates the number of pulses fired from the neurons into the array.

C. Simulated performance

A three-layer perceptron with two PCMO-based devices per synapse was simulated performing a classification task on the MNIST database (same network shown in Fig. 4). Fig. 13 plots the modeled conductance response of the resistive switching elements. For average values of conductance G (e.g., the central region of the plot), the response is mostly linear, although somewhat asymmetric, with different average jump values for SET and RESET. In constrast, for extreme values of the conductances (left and right edges of each jump-table), a high degree of non-linearity is observed. However, we have previously observed that when the extent of the non-linear region is sufficiently small, high classification accuracies can still be achieved [7].

The network parameters were tuned to achieve a good performance, with particular focus given to the ratio of $\eta_{\mathrm{SET}}/\eta_{\mathrm{RESET}}$, used to compensate the asymmetry of the jump-table. Fig. 14 shows a schematic version of the crossbar-compaibile weight update rule for backpropagation, in which upstream neurons fire a set of pulses (shown in red) along the horizontal word-lines, based solely on their knowledge of x_i and the global learning rate ($\eta = \eta_{\mathrm{SET}}$) [1]. Simultaneously, the downstream neuron first pulses (shown in magenta) along the vertical bit-lines connected to a large number of G^+ and G^- conductances. These pulses are based only on the downstream neuron's knowledge of δ_j and the global learning rate.

Because these pulses affect all the devices along the shared word-lines and bit-lines, their amplitude and duration cannot be tuned to optimize the programming of any one particular conductance value. This leads to significant problems when conductance response is nonlinear, since the same pulse can cause small conductances to increase much more significantly than conductances that are already large.

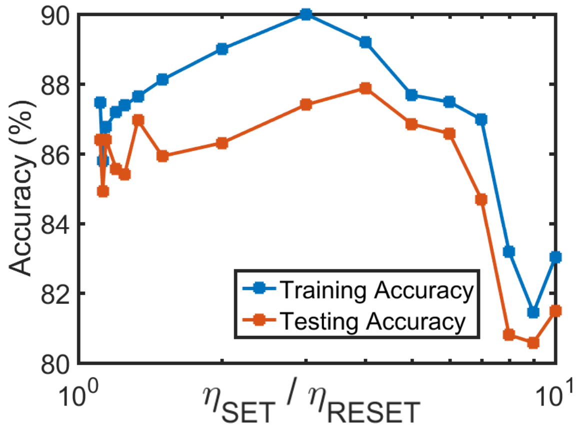
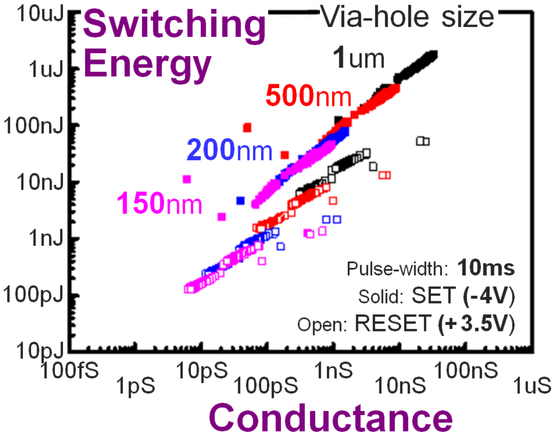


Fig. 15. Simulated training and test accuracy for a three-layer perceptron using PCMO-based devices as synaptic weights. The asymmetry between positive and negative jumps can be compensated by tuning individually the learning rates for SET and RESET (see Fig. 14). The classification accuracy of the network improves as the ratio of SET to RESET learning rate ($\eta_{\mathrm{SET}}/\eta_{\mathrm{RESET}}$) increases.

However, the downstream neuron can easily fire *different* pulse-trains on the separate G^{+} and G^{-} bit-lines, and knowledge of δ_j can be sufficient to identify whether SET or RESET will occur (x_i need only be constrained to be non-negative). Thus it is straightforward to apply a different global learning rate for RESET and for SET, thus leading to more or fewer pulses, and providing a way to compensate for jump-table asymmetry. Fig. 15 shows that classification accuracy can be improved for the Al/Mo/PCMO jump-tables shown in Fig. 13, with an optimal ratio of η_{SET} / η_{RESET} of approximately 3–4.

D. Switching Energy

The switching energy of the devices was measured by integrating the product between the voltage and the current for the duration of a programming pulse ($10ms$). The conductance was measured with read pulses of $-1V$. PCMO-based memory devices (like other non-filamentary switching elements) show a dependence of the programming energy on the active area. Switching energy ranging from sub-nJ to tens of μJ were measured on devices with hole sizes from $0.15nm$to $1\mu m$ (Fig. 16(a). The switching energy was then normalized with respect to the active device area (Fig. 16(b)) to show a good linear dependence between switching current and device hole-size. Following the trend from 150nm down to 25nm, one can anticipate an improvement in switching energy by roughly 35×. If the switching time could potentially be reduced from 10ms down to 10ns, then one would be able to achieve femto-Joule switching energy. Such aggressive scaling of both device area and switching time would be necessary in order to enable highly-parallelized weight update operations.



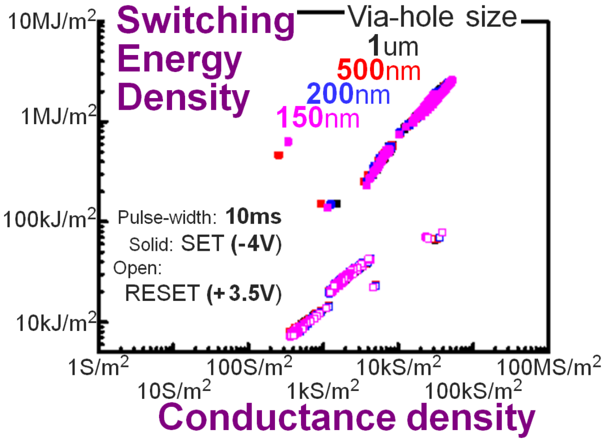


Fig. 16. Switching (a) energy as a function of conductance and (b) energy density as a function of conductance density, measured for Al/Mo/PCMO-based devices with -1V reading voltage.

IV. Circuit Needs

A crossbar-array-based neural network implements the multiply-accumulate operations at the heart of most neural network algorithms extremely efficiently, through Ohm's law followed by current summation (Kirchoff's Current law). However, an important consideration is the design of highly area-efficient neuron circuits that reside at the edges of these arrays enabling read and write of many synaptic rows or columns in parallel. Such high parallelism is essential if we wish to achieve orders of magnitude performance and power benefits over conventional CPU/GPU approaches [3]. Given this need for a large number of distinct copies of neural circuits that can be executed in parallel, it is critical to embrace approximate functionality (for e.g. non-linear squashing functions, calculating and multiplying derivatives etc.) rather than rigorously-precise yet highly area-inefficient functionality.

In this section, we present examples of design choices that simplify the underlying hardware by leveraging the inherent tolerance of ANN algorithms to error. We discuss circuit needs for the forward- and reverse-evaluate operations, including precision/range of the computed neuron activations and backpropagated errors, using piecewise linear (PWL) approximations of non-linear squashing functions, and simplifying the derivatives included during reverse propagation to avoid complex floating-point arithmetic operations. We then demonstrate that these approximations do not significantly degrade classification accuracies as compared to neuron implementations with rigorously-precise functionality.

A. Circuit-Needs for Forward and Reverse Propagate

Forward propagation (Fig. 17) in a fully connected neural network involves the calculation of the neuron activations of a hidden/output layer, based on the neuron activations of the previous layer and the intervening synaptic weights. This is a two-stage process, with the multiply- accumulate operation occurring in the crossbar array, and the non-linear squashing function applied at the periphery. One commonly used function in software implementations is $tanh()$ (the hyperbolic-tangent function), which is difficult to implement exactly unless a large number of transistors are included. However, a piecewise linear implementation of this squashing function would be fairly straightforward to implement (Fig. 17).

A second design choice is the range of distinct neuron activation values that need to be supported by the hardware. In a digital implementation this translates into the number of bits, which would have area implications depending on the amount of local storage required, as well as the resolution of any analog to digital conversion circuits used to convert signals from the crossbar array into those bits. In an analog implementation,

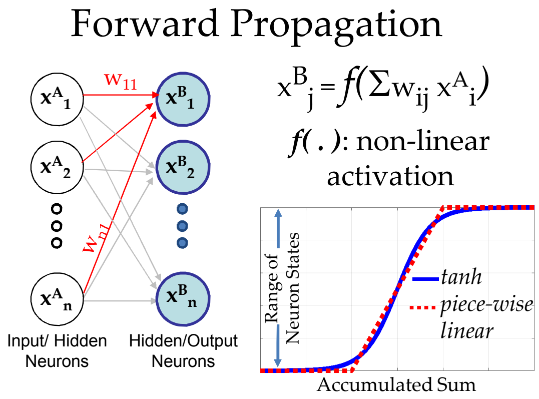


Fig. 17. Forward Propagation operation in a Deep Neural Network. The multiply-accumulate operation occurs on the crossbar array. Neuron circuitry must handle the non-linear squashing function.

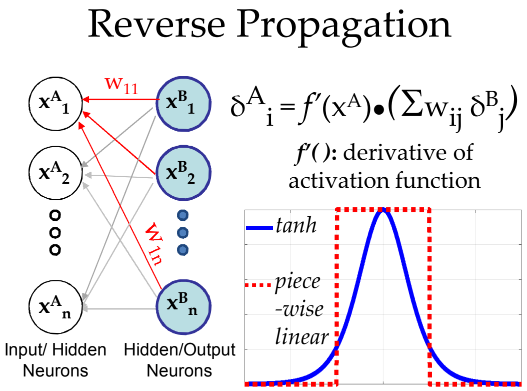


Fig. 18. Reverse Propagation operation in a Deep Neural Network. Multiply-accumulate operation on δ occurs on the crossbar array. Neuron circuitry must handle generation and multiplication of the derivative of the squashing function

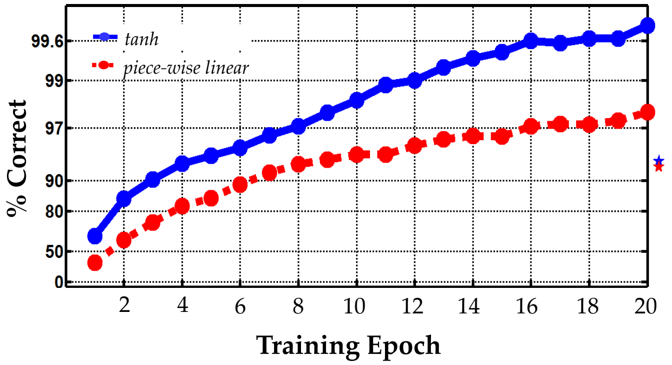


Fig. 19. Training and test accuracies obtained on MNIST with $tanh()$ and piece-wise linear activation functions. PWL achieves test accuracy comparable to $tanh()$.

this would directly translate into the resolution between analog voltage levels and/or time-steps.

Reverse propagation (Fig. 18) is similar to forward propagation, but from output/hidden neurons to preceding hidden neurons. The quantity δ, known as the correction or error, together with the forward-propagated neuron activations, control the weight updates for neural network training (see Fig. 14). An important distinction from forward propagation is that the non-linear squashing function is not applied. Instead, the multiply-accumulated sum (integrated on the crossbar array, but in a direction orthogonal to the integration performed during the forward-propagate step) needs to be scaled by the derivative of the activation function, as evaluated at the neuron activation value. Again, an exact $tanh()$ derivative is not efficient to compute and multiply.

Instead, a step-function derivative with two distinct states can be used. Multiplication by derivative values of zero and one is fairly straightforward to implement in hardware. This corresponds to simply enabling or disabling the transmission of an accumulated sum-of-deltas from any neuron stage to the preceding stage. However, multiplication by arbitrary scale factors may be difficult to achieve since floating-point multipliers are not readily available. The impact of such approximations on neural network training is studied in the next subsection.

B. Results: Circuit Approximations

We explored the impact of the aforementioned circuit approximations on the training and test performance of the MNIST dataset of handwritten digits through simulations. A subset of only 5000 training images from the original dataset of 60000 images is used. Images are cropped to 24× 22 pixels. The same 3-layer neural network (528-250-125-10) is used (Fig. 4. A crossbar-compatible weight update rule [1] is used to emulate how weight updates would be done on a real crossbar array. The baseline training and test accuracies assuming 20 epochs of training, 256 neuron activation states, a $tanh()$ activation function and exact derivatives were found to be 99.7% and 93.6% respectively (blue curve and star, Fig. 19). Note that, as per Fig. 7, both training and test accuracy increase (to ∼100% and ∼97-98%) when all 60,000 examples are used for training.

Fig. 19 also shows the training and test accuracies using a piece-wise linear (PWL) activation function. On MNIST, one observes that the test accuracy obtained (92.7%) is already comparable to the full $tanh()$ implementation. Further improvements in test accuracy can be obtained by optimizing the low value of the derivative. This is akin to the intentional implemention of 'leaky' derivatives in some conventional machine learning techniques, especially in the case of Rectified Linear Units (ReLU). A leaky derivative ensures that some contribution from the downstream neuron gets passed on to earlier stages, thereby participating in the programming of those weights.

Fig. 20 shows that the test accuracy can be further improved to 93.2% when the derivative of the piecewise-linear squashing function at extreme values is made non-zero. However, the multiplication operation is non-trivial. In a digital implementation, one might be able to do bit-shift operations (restricting derivative values to powers of 2). An analog implementation can offer more freedom, since we need only enable one of two non-zero scale factors when transmitting accumulated analog voltages to preceding stages.

In addition to the squashing function and its derivative, the impact of the number of distinct neuron activation and error states on the test accuracy was analyzed. Values from 8

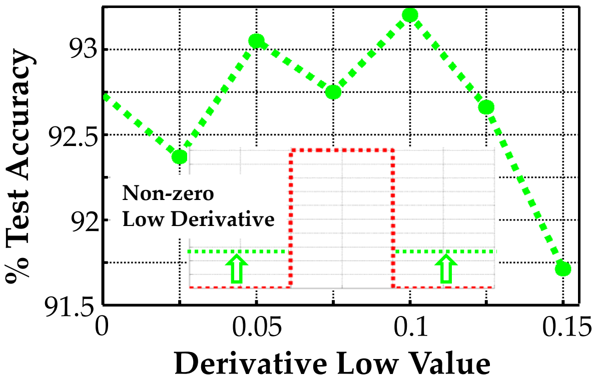


Fig. 20. Optimizing the low derivative value enables further improvements in test accuracy, yet requires some circuit complexity to implement an approximate multiplication function.

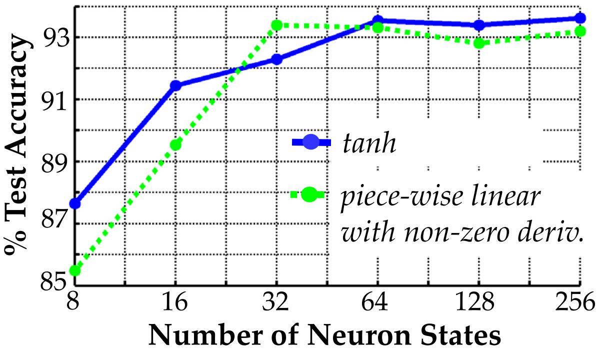


Fig. 21. If the number of distinct neuron activation and error states is lower than 32, then test accuracy degrades. However, reducing the total number of neuron states can help enable significantly more area-efficient peripheral circuitry.

to 256 were considered (Fig. 21). High test accuracies are maintained down to 32 distinct neuron states for both the $tanh()$ and piece-wise linear implementations. Reducing the total number of neuron states can be extremely beneficial in area-efficient circuit design. In a digital implementation, this allows a reduction in the total number of latches or flip-flops. In an analog implementation, it permits a wider separation of analog voltage levels, relaxing noise constraints and enabling simpler circuits.

V. Conclusion

We have studies several aspects of system design when Non-Volatile Memory (NVM) devices are employed as the synaptic weight element for on-chip acceleration of the backpropagation training of large-scale artificial neural networks (ANN).

We have assessed the impact of undesired, time-varying conductance change, including drift in Phase Change Memory (PCM) devices and leakage of analog CMOS capacitors. We have investigated the use of non-filamentary, bidirectional ReRAM devices based on PrCaMnO, which can be considered a promising material variant that could potentially provide both gradual conductance increase *and* conductance decrease. And finally, we have explored some of the tradeoffs in designing peripheral circuitry, balancing simplicity and area-efficiency against the impact on ANN performance for the nonlinear squashing function, the evaluation of its derivation, and the number of resolvable levels when integrating both x (forward-propagate) and δ (reverse-propagate) values.

We briefly reviewed our previous work towards achieving competitive performance (classification accuracies) for such ANN with both Phase-Change Memory [1], [2] and non-filamentary ReRAM based on PrCaMnO (PCMO) [4], and towards assessing the potential advantages for ML training over GPU–based hardware in terms of speed (up to 25× faster) and power (from 120–2850× lower power) [3]. We discussed the "jump-table" concept, previously introduced to model real-world NVM such as PCM [1] or PCMO, to describe the full cumulative distribution function (CDF) of resulting conductance-change at each possible conductance value, for both potentiation (SET) and depression (RESET).

While the 'LG' algorithm, together with other approaches, should help a nonlinear, asymmetric NVM (such as PCM) act more like an ideal linear, bidirectional NVM, the identification of NVM devices and/or pulse-schemes that can offer a conductance response that is at least partly linear, using circuitry that can be highly area-efficient (and thus massively-parallel), will help significantly in achieving equally-high classification accuracies while offering faster and lower-power training than conventional GPUs and CPUs.

References

[1] G. W. Burr, R. M. Shelby, C. di Nolfo, J. W. Jang, R. S. Shenoy, P. Narayanan, K. Virwani, E. U. Giacometti, B. Kurdi, and H. Hwang, "Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element," in *IEDM*, 2014, p. 29.5.

[2] G. W. Burr, R. M. Shelby, S. Sidler, C. di Nolfo, J. Jang, I. Boybat, R. S. Shenoy, P. Narayanan, K. Virwani, E. U. Giacometti, B. Kurdi, and H. Hwang, "Experimental demonstration and tolerancing of a large–scale neural network (165,000 synapses), using phase–change memory as the synaptic weight element," *IEEE Trans. Electr. Dev.*, vol. 62, no. 11, pp. 3498–3507, 2015.

[3] G. W. Burr, P.Narayanan, R. M. Shelby, S. Sidler, I. Boybat, C. di Nolfo, and Y. Leblebici, "Large–scale neural networks implemented with nonvolatile memory as the synaptic weight element: comparative performance analysis (accuracy, speed, and power)," in *IEDM Technical Digest*, 2015, p. 4.4.

[4] J.-W. Jang, S. Park, G. W. Burr, H. Hwang, and Y.-H. Jeong, "Optimization of conductance change in $Pr_{1-x}Ca_xMnO_3$–based synaptic devices for neuromorphic systems," *IEEE Electron Device Letters*, vol. 36, no. 5, pp. 457–459, 2015.

[5] D. Rumelhart, G. E. Hinton, and J. L. McClelland, "A general framework for parallel distributed processing," in *Parallel Distributed Processing*. MIT Press, 1986.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, p. 2278, 1998.

[7] S. Sidler, I. Boybat, R. M. Shelby, P. Narayanan, J. Jang, A. Fumarola, K. Moon, Y. Leblebici, H. Hwang, and G. W. Burr, "Large-scale neural networks implemented with non-volatile memory as the synaptic weight element: impact of conductance response," in *ESSDERC 2016*, 2016, p. to appear.

[8] A. Pirovano, A. L. Lacaita, F. Pellizzer, S. A. Kostylev, A. Benvenuti, and R. Bez, "Low–field amorphous state resistance and threshold voltage drift in chalcogenide materials," *IEEE Trans. Electr. Dev.*, vol. 51, no. 5, pp. 714–719, 2004.