MLP+NeuroSimV3.0: Improving On-chip Learning Performance with Device to Algorithm Optimizations

Yandong Luo, Xiaochen Peng, and Shimeng Yu†
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332, United States
{ yandongluo, xpeng76 }@gatech.edu, † shimeng.yu@ece.gatech.edu

ABSTRACT

On-chip learning with compute-in-memory (CIM) paradigm has become popular in machine learning hardware design in the recent years. However, it is hard to achieve high on-chip learning accuracy due to the high nonlinearity in the weight update curve of emerging nonvolatile memory (eNVM) based analog synapse devices. Although digital synapse devices offer good learning accuracy, the row-by-row partial sum accumulation leads to high latency. In this paper, the methods to solve the aforementioned issues are presented with a device-to-algorithm level optimization. For analog synapses, novel hybrid precision synapses with good linearity and more advanced training algorithms are introduced to increase the on-chip learning accuracy. The latency issue for digital synapses can be solved by using parallel partial sum read-out scheme. All these features are included into the recently released MLP + NeuroSimV3.0, which is an in-house developed device-to-system evaluation framework for neuro-inspired accelerators based on CIM paradigm.

CCS CONCEPTS

• Computer systems organization • Neural networks • Hardware • Analysis and design of emerging devices and systems

KEYWORDS

Benchmark simulator, neuromorphic computing, NeuroSim, non-volatile memory, on-chip learning, synaptic devices.

ACM Reference format:

Yandong Luo, Xiaochen Peng and Shimeng Yu. 2019. MLP+ NeuroSimV3.0: improving on-chip learning performance with device to algorithm level optimizations. In *Proceedings of Proceedings of ACM International Conference on Neuromorphic Systems (ICONS'19). ACM, Knoxville, Tennessee, USA.*

ICONS '19, July 23–25, 2019, Knoxville, TN, USA

ACM ISBN 978-1-4503-7680-8/19/07…$15.00
https://doi.org/10.1145/3354265.3354266

1) Introduction

Compute-In-Memory Paradigm In the past decade, neural network based machine learning algorithms have witnessed rapid development and achieved commercial success in image/speech recognition. However, the intensive matrix multiplication operations and massive data movement between computation units and memory make it inefficient to implement neural network based algorithms on traditional computation platform (e.g. CPU), which is based on von-Neumann architecture. The reasons can be explained by the lack of parallelism during the matrix multiplication and the so-called memory bottleneck. To overcome these challenges, GPU is used to speed up the training process of the neural networks due to its highly parallel computation mechanisms. However, the high power consumption of GPU (~100W scale) prevents it from being applied to the edge devices where the power budget is limited.

To alleviate the memory bottleneck, which is caused by the insufficient memory bandwidth during the frequent data transfer between memory and computing unit when doing matrix multiplication, one promising solution is to perform computation inside the memory, which is referred to as compute-in-memory (CIM) [1]-[6]. The basic idea of CIM is to map the elements of a matrix into the conductance matrix of memory cells, which is termed as synaptic array and each memory cell is called a synapse. The vector is input at each row of the memory array as voltage level and the multiplication results are represented by the partial sum current at each column, as shown in Figure 1.

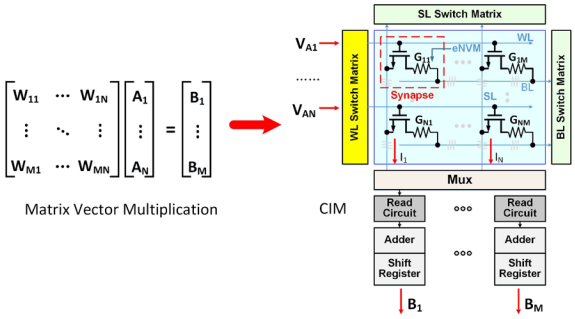


Figure 1: Implementation of vector-matrix multiplication with compute-in-memory paradigm.

MLP + NeuroSim Simulator CIM paradigm is facilitated by recent research breakthroughs in emerging non-volatile memory (eNVM). Four promising eNVM candidates: resistive RAM (RRAM), phase change memory (PCM), spin-transfer-torque magnetic RAM (STT-MRAM) and ferroelectric field-effect transistor (FeFET) are featured of their small cell size, short programming time, good endurance and data retention, which is beneficial for CIM paradigm. To help evaluate the impact of memory device properties on the performance of CIM based neural network accelerator, an in-house device-to-system simulation framework MLP+NeuroSim was developed [7],[8], which is publically available at GitHub [9]. The overall architecture of this simulator is shown in Figure 2. In general, the framework consists of two parts: the MLP simulator and NeuroSim simulator. The MLP simulator helps evaluate the on-chip learning or inference accuracy with a 2-layer MLP network. The default network topology is 400-100-10 for MNIST dataset. Device properties such as nonlinearity, cycle-to-cycle variations, device-to-device variations and number of conductance levels are considered as input parameters to build the synaptic arrays. On the other hand, the circuit-level performance metrics including chip area, read/write latency, dynamic energy consumption and leakage power are estimated by the NeuroSim simulator. The performance metrics of periphery circuit modules (e.g. sense amplifiers, mux, decoders et. al) and synaptic arrays are evaluated by built-in analytical models and user defined values at specific technology node. More details about the methodologies used in the framework can be found in our previous papers [7] [8].

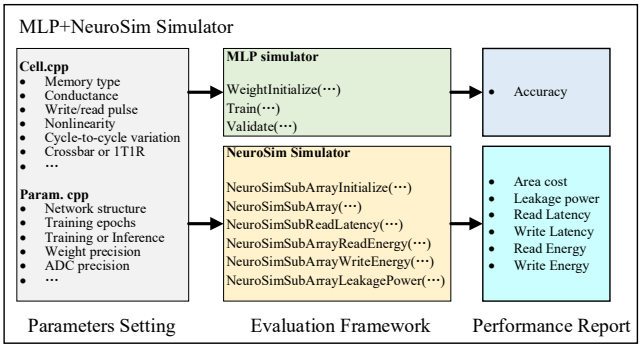


Figure 2: Overview of the simulator architecture of MLP + NeuroSim framework.

So far, Version 1.0 and 2.0 of the MLP + NeuroSim framework have been released in 2017 and 2018, respectively. In these two versions, both analog and digital synaptic arrays with their periphery circuit modules are supported. Various devices such as RRAM, PCM, STT-MRAM, FeFET, and SRAM are benchmarked for their performance in on-chip learning and on-chip inference. While these devices achieve good performance for on-chip inference, challenges occurs for on-chip learning. The main reason is that the nonlinear and asymmetric weight update curve of analog synapses prevent it from achieving high learning accuracy because of the inaccurate weight update. Although digital synapses based on eNVMs or SRAM offers good learning accuracy, the drawbacks of them are also obvious. For eNVMs based digital synaptic arrays, the row-by-row read is time consuming for partial sum calculation. For SRAM based synaptic arrays, although it features of high read/write speed, the high leakage power, volatility and high area cost limit its applications in large scale neural network accelerators.

In this paper, device-to-algorithm level optimizations are pursued to address the aforementioned challenges. In section 2, the methods to improve the on-chip learning accuracy of the analog synapses based synaptic array are illustrated. In general, hybrid precision synapses and advanced training algorithms such as adaptive momentum estimation (Adam) are utilized. In section 3, the high read latency of digital synaptic arrays is reduced by parallelizing the partial sum read-out. In section 4, the new features of MLP+NeuroSimV3.0 are summarized and a benchmark table of state-of-the-art synaptic devices is presented. The key factors to achieve high on-chip learning accuracy is discussed. In section 5, conclusions are drawn and a blueprint for the future design automation tool development plan is presented.

2) Improving Online Learning Accuracy for Analog Synapse Based Accelerator

In analog synapse based accelerators, the multi-bit weight is stored in one memory cell, where RRAMs and PCMs with either crossbar or 1-transistor and 1-resistor (1T1R) cell structure are usually used. With analog synapses, each partial sum value is obtained in parallel from one column of the array. It is more time-efficient compared with digital synapses, where the partial sums corresponding to different weight bits need to be sequentially added up with an adder at the edge of the array. However, as described previously, the on-chip learning accuracy is degraded due to the nonlinear/asymmetric weight update curve of eNVMs based analog synapses. In this section, hybrid precision synapse and advanced learning algorithms are applied to improve the on-chip learning accuracy.

2.1 Hybrid Precision Synapse

As is known, the nonlinear and asymmetric weight update curve (conductance vs. # programming pulse) prevents the analog synapses from achieving high on-chip learning accuracy. Besides, the relatively longer programming pulse width (~tens of nanoseconds or above) of eNVMs limits the training speed. Recently, capacitor based analog synapse such as 3-transistor-1-capacitor (3T1C) cell [10] is proposed to alleviate these problems. Although capacitor based analog synapse offers good linearity and fast programming speed, it suffers from the volatility and small dynamic range.

To leverage the good linearity, fast programming speed of capacitor and the non-volatility, abundant conductance states of eNVMs, the weight stored in a synapse can be divided into 2 parts and stored into different devices. The first part of the weight has a lower numerical significance and is stored in the capacitor devices, which

is termed low significance weight (LSW, W_{LSW}). The other part of the weight with higher numerical significance is stored in the eNVMs, which is termed high significance weight (HSW, W_{HSW}). During training, only the LSW is frequently updated due to the fast programming speed of capacitor. A significance factor F is defined to represent the numerical significance of the HSW. Therefore, the weight stored can be represented as $W = F \times W_{HSW} + W_{LSW}$, as shown in Figure 3. A synapse that is capable to store the volatile LSW and non-volatile HSW is termed as hybrid precision synapse in this paper.

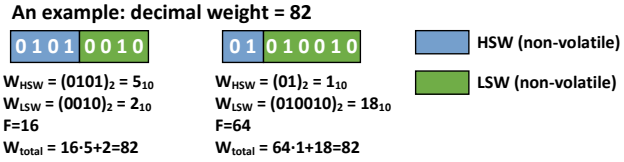
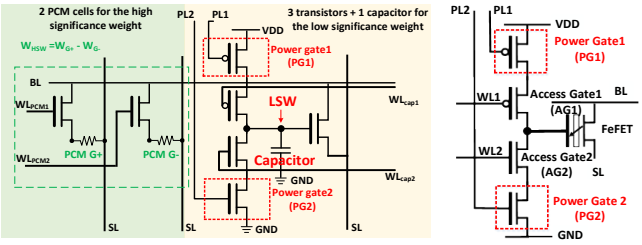


Figure 3: an illustration of the weight storage in a hybrid precision synapse

At present, two types of hybrid precision synapses are proposed, as shown in Figure 4. One is based on PCMs and capacitor [11], which is called 3T1C +2PCM synapse since in the original design there are 3 transistors, 1 capacitor and 2 PCMs (with 2 selection transistors) in a synapse. The other one is based on the ferroelectric transistor [12], which is called 2T1F synapse since in the original design, there are 2 CMOS transistors and 1 ferroelectric transistor in a synapse.

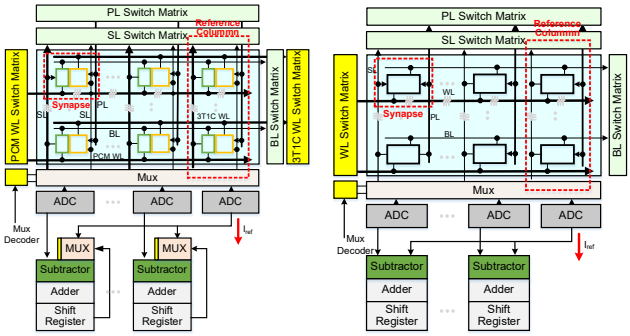
In the 3T1C + 2PCM synapse, the LSW is stored in a MOS capacitor, which tunes the gate voltage of the NMOS transistor connected. Two access transistors (AG) are used to control the charging and discharging of the capacitor. The HSW is represented by the weight difference between a PCM pair, symbolled as G_+ and G_-, respectively. To increase the HSW, the G_+ is programmed by a series of set pulses. To decrease the HSW, the G_- is programmed by a series of set pulses. For the 2T1F synapse, similarly, the LSW is stored at the gate capacitor of the FeFET, which is tuned by charging and discharging the gate node. The HSW is programmed by the multi-domain polarization switching in the ferroelectric gate dielectrics [13], which is non-volatile. It should be noted that the LSW and HSW are both stored as the channel conductance of the FeFET but with different programming mechanisms. In such hybrid synapses, the LSW is volatile and may be lost due to the leakage in the off-state transistors.



(a) (b)

Figure 4: Schematics of hybrid precision synapses (a). 3T1C+2PCM synapse [11]. (b). 2T1F synapse [12].

Array Design To enable the control of individual synapse in an array, we add two more transistors to control the access to VDD and GND in a synapse, respectively, as shown in the red dash box in Figure 4. These two transistors are called power gate (PG) in this paper. The array level architecture design is shown in Figure 5 (a) and (b) respectively. In both designs, switch matrices are used to manipulate the control lines. A reference column is added to represent negative partial sum values, which is obtained by subtracting the partial sum digits of the reference column from the partial sum digits of regular columns. In the periphery circuits, analog to digital converter (ADC) based on multilevel sense amplifiers is used to convert the analog partial sum current to digital values. Specifically, for the synaptic array with 3T1C + 2 PCM synapse, it is assumed that the partial sum current corresponding to LSW and HSW are first converted to digital values and then added up to obtain the total weight, although in [11], an analog summation of the partial sum current is conducted and followed by digital conversion. To reduce the overhead of periphery circuits, a mux is used to share periphery circuit modules among different columns.



(a) (b)

Figure 5: synaptic arrays with (a) 3T1C+2PCM synapse and (b) 2T1F synapse

Partial Sum Read The partial sum read operation in 3T1C + 2PCM is conducted in 3 steps as follows.

1. Read the partial sum corresponding to HSW (HSW Psum). It is a 2-step process. First, the partial sum corresponding to the G_+ synapses (G_+ Psum) is read-out. Then, it is subtracted by the partial sum corresponding to G_- synapses (G_- Psum). The 2-1 mux will forward the G_+ Psum to subtractor from register during subtraction.
2. The HSW Psum is shifted to left by log2(F), where the significance factor F is defined as an integer power of 2 to make the shift amount integer.
3. Read the partial sum corresponding to the LSW (LSW psum) and then adding it up with the HSW Psum. To get LSW psum, the 2-1 mux will be configured to deliver the partial sum of the reference column to the subtractor.

The partial sum read in 2T1F based synaptic cell is similar except that it does not need to read the LSW and HSW separately as they are both encoded as the channel conductance of FeFET.

LSW Write In both synapses, the weight update is conducted by programming the capacitor row-by-row during training.

1. Turn on the PGs of the selected synapse
2. If weight update $\Delta W > 0$, pulses with high voltage are applied to AG1 to charge the gate node. Otherwise, if $\Delta W < 0$, pulses with low voltage are applied to AG2 to discharge the gate node.

HSW Write (Weight Transfer) For the capacitor node, if its voltage is too low or too high, the NMOS may operate out of linear region, and also the gate voltage may decay over time due to leakage current. Therefore, after certain amount of training batch, the LSW is read out and transferred to HSW by programming eNVMs. The weight transfer for 3T1C+ 2PCM based synapse can be conducted as follows.

1. Read out LSW weight W_{LSW} row by row.
2. Calculate the amount of weight to be transferred to HSW by $\Delta W_{HSW} = W_{LSW}/F$.
3. Program the G_+ cell by applying set pulses to its BL if $\Delta W_{HSW}>0$. Otherwise, program the G_- PCM cell if $\Delta W_{HSW}<0$.
4. After weight transfer, program the LSW to an intermediate level and therefore the original LSW is discarded.

The weight transfer for 2T1F synapse based array is slightly different. The weight of the whole synapse (W) is read-out row-by-row. The HSW is reprogrammed to a new weight level of W/F. Similarly, the LSW is discarded after the weight transfer.

Table 1 list the MLP+NeuroSim benchmark results for synaptic arrays based on these two hybrid precision synapses. The device parameters for PCM and FeFET are obtained from [14] and [13], respectively. An ideal eNVM synapse is selected as the baseline. Due to the improved linearity, the synaptic arrays with hybrid precision synapses can achieve comparable on-chip learning accuracy with ideal device. Slight accuracy degradation is observed due to the weight loss after weight transfer and the cycle-to-cycle variations in a real device. However, a significant area overhead is observed due to two reasons: 1). The multilevel polarization switching in FeFET is only reported at μm scale for FeFET [13] 2). The relatively large capacitance (100fF) is used in the design [11], [12].

Table 1 The Benchmark Results for Hybrid Precision Synapse (32nm Technology Node)

Device	3T1C+2PCM	2T1F	Ideal eNVMs
# of conductance states	PCM: 16 Capacitor: 32 (6 bits in total) F=4	HSW: 2bits LSW: 4 bits (6 bits in total) F=16	6 bits
Nonlinearity (weight increase/decrease)	Capacitor: 0.2/-0.2 PCM: 0.105 (LTP only)	0.5/0.5	0/0
R_{ON}	PCM: 4.71 KΩ Capacitor: 25KΩ	559.28KΩ	200kΩ
ON/OFF ratio	PCM: 19.8 Capacitor: 20	45	50
Weight increase pulse	HSW: 0.7V (avg.) /6μs LSW:1V/300ps	HSW: 2-4V/3μs LSW: 1V/300ps	2V/10ns
Weight decrease pulse	Capacitor:1V/300ps	HSW: 2-4V/3μs LSW: 1V/300ps	2V/10ns
Cycle-to-cycle variation (σ)	PCM: 1.5% Capacitor: 0.5%	1.5%	0%
Online learning accuracy	93.8%	94.6%	94.7%
Area	330,330 μm^2	334,270μm^2	7477.4μm^2
Latency	3.15s (3.06s for transfer)	0.38s (0.28s for transfer)	1.56s
Energy	16.69 mJ	7.53 mJ	4.37 mJ
Leakage power	1.66 mW	2.94 mW	105.6 μW

2.2 Training Algorithms

In the previous two versions of MLP + NeuroSim framework, stochastic gradient descent (SGD) is used during the backpropagation stage for on-chip learning. In the V3.0, more training algorithms are supported to help increase the on-chip learning accuracy. Here, we briefly talk about the options of training algorithms in V3.0. More mathematical details about these algorithms can be found in an online tutorial [15].

Stochastic Gradient Descent (SGD) SGD is the training algorithm used in V1.0 and V2.0. It calculates the gradient and conduct weight update after each training image with a pre-defined learning rate. This feature makes SGD suitable for online learning because of the fast execution speed. However, relatively large fluctuation of the learning accuracy is also observed during training because of the fixed learning rate.

Momentum Momentum method is a revised weight update scheme for SGD. The weight update is a linear combination of the gradient at present time t and the weight update at previous time t-1. It alleviates the accuracy fluctuation in SGD and offers faster convergence. Similar to SGD, the learning rate is fixed.

Adaptive Gradient (Adagrad) In Adagrad, the learning rate for weight w_{ij} is divided by its accumulative gradient update during the training, i.e. the learning rate undergoes a monotonic decay during the training and it decays faster if the w_{ij} undergoes a large amount of weight update in its training history. Therefore, Adagrad provides good convergence and alleviates the learning accuracy fluctuation during training.

Root Mean Square Propagation (RMSprop) In RMSprop, the learning rate of weight w_{ij} is divided by the moving average of its recent gradients. Therefore, RMSprop avoid the monotonic decay of learning rate in Adagrad.

Adaptive Moment Estimation (Adam) Adam is a training algorithm that combines RMSprop and the momentum method. The learning rate is self-adaptive based on the moving average of 1st and 2nd moments of the gradient.

To illustrate the effect of different training algorithms on on-chip training, the learning accuracy vs. epoch is plotted in Figure 6. Two devices, Ag:a-Si based RRAM [16] with high non-linearity and EpiRAM [17] with low non-linearity is selected here for

comparison. It is observed that the on-chip learning accuracy can be improved by using more advanced training algorithms. It can be explained that those batch-based training algorithms improve the equivalent weight precision by accumulating the tiny ΔW [18]. Another possible explanation is that more "deep" local optimal points are introduced in the loss function by using devices with high nonlinearity. Training with SGD are easy to be trapped into those local optimal points while algorithms such as Adam, RMSprop can help escape from local optimal points and therefore achieves high training accuracy. For Ag/a-Si, the learning accuracy fluctuation is alleviated by the training algorithms with self-adaptive learning rate, i.e. Adagrad, RMSprop and Adam.

However, since those algorithms are batch based, whether they are still effective when the training images is not abundant in on-chip learning needs further examination. Besides, the hardware overhead (such as buffers) is not negligible to implement these algorithms, which is scheduled in our development plan for future version. In this version, the algorithms are supported in software level to examine their effect on improving the on-chip learning accuracy.

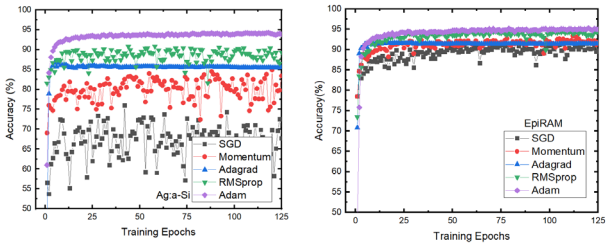


Figure 6: Training accuracy vs. epochs for Ag:a-Si [16] and EpiRAM [17] devices.

3) Improving On-chip Learning Efficiency for Digital Synapse Based Accelerator

Accelerators based on digital synapses feature of the immunity to device nonlinearity/variability as only "0"s and "1"s are stored in each memory cell. However, the partial sum read-out for digital synaptic array (e.g. STT-MRAM, SRAM) is conducted row by row, which leads to high latency. In this V3.0, a parallel read-out architecture for SRAM based synaptic array is proposed in Figure 7 [2],[4]. The WL decoder for the row-by-row read-out array is replaced with a WL switch matrix to turn on all WLs at a time. The partial sum of each column is sensed by the multilevel sense amplifier. A reference generator is used to generate the reference voltage levels for the multilevel sense amplifier. The adder and register to accumulate the partial sum of each row after the S.A. in the row-by-row SRAM array is eliminated.

The benchmark results for row-by-row and parallel read-out SRAM array are listed in Table 2. The area cost for parallel SRAM array is increased due to the multilevel sense amplifiers and the reference generator. The read latency of parallel read-out SRAM array is reduced significantly compared with the row-by-row scheme, which leads to lower total latency. In the parallel SRAM array, the total latency is limited by the write latency, which has to be conducted row-by-row. In Table 2, row-by-row read-out scheme using STT-MRAM is shown for comparison, which suffers from longer write latency/energy but benefits from lower leakage power. The parallel read-out scheme of STT-MRAM is for the future investigation, as the on/off ratio of STT-MRAM is very limited.

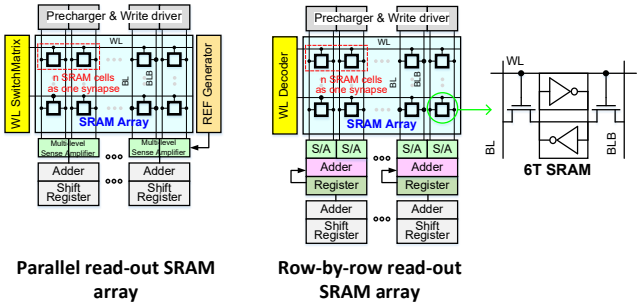


Figure 7: SRAM based digital synaptic array with parallel partial sum read and row-by-row partial sum read

Table 2 Benchmark Results for Digital Synaptic Arrays with Row-by-Row Read and Parallel Read

Device	SRAM (row-by-row)	SRAM (parallel read)	STT-MRAM (row-by-row)
# of conductance states	6 bits	6 bits	6 bits
R_{ON}	--	--	3.5kΩ
ON/OFF ratio	--	--	2.3
Weight increase pulse	--	--	1V/10ns
Weight decrease pulse	--	--	1V/10ns
Online learning accuracy	~94%	~94%	~94%
Area	65,728 μm²	74,699μm²	66,632 μm²
Latency	5.98 s (4.16s for read)	1.73s (0.11s for read)	90.1s (row-by row)
Energy	15.56 mJ	19.1 mJ	146.2 mJ
Leakage power	2.80 mW	2.69 mW	84.0 μW

4) Benchmark Results for State-of-the-art Synaptic Devices

Besides the features mentioned above, in V3.0, the range of algorithm weight is changed from (0,1) to (-1,1), which is more widely in today's neural network algorithms. To represent the negative weight, a reference column is added in the array, which represents the level "0". In the forward stage, the partial sum current of a regular column is first read out and converted to digits by ADC. At the same time, the partial sum digit of the reference column is obtained. Then, the partial sum of the reference column is subtracted from that of the regular column by subtractor.

A new benchmark table in V3.0 is presented in Table 3 while the benchmark results in V2.0 [19] is presented in Table 4 for comparison. Both tables are obtained at 32nm node. From the benchmark table, a few remarks can be made.

1. In V3.0, both the latency and energy consumption are reduced for synaptic devices with relatively good linearity but long

write pulse width (e.g. EpiRAM, TaO_x/HfO_x, PCM). It can be explained by the weight range change from (0,1) to (-1,1), where the number of programming pulses is reduced by half to change the weight by ΔW in V3.0. As a result, both the write latency and energy consumption due to synapse programming decrease.

2. For PCMO, which has poor linearity, both latency and energy consumption are increased since its learning accuracy is slightly increased from 10% to 20%, which leads to more write pulses applied.
3. The area cost for all the devices is increased due to the hardware overhead to support the negative weight.
4. In V3.0, for digital synaptic arrays, row-by-row read-out SRAM shows much lower latency and energy consumption than STT-MRAM due to the relatively large write pulse width and the large write current for STT-MRAM. By parallelizing the partial sum current read-out, the latency of SRAM can be further reduced.
5. In general, analog synapses provides better area efficiency but they suffers from low learning accuracy due to the non-ideal weight update. Both the eNVM-based digital synapses and hybrid precision synapses shows good learning accuracy with additional area cost. SRAM-based digital synapses is good for on-chip learning due to its low latency and high learning accuracy. However, due to its volatility, the weights are stored off-chip and weight load is needed before inference.

From the benchmark table, the key factors to achieve high on-chip learning accuracy can be concluded. First, good linearity and symmetry in the weight update curve is required. Relatively low cycle-to-cycle variation is also necessary. For example, even though the linearity of TaO_x/HfO_x is comparable with EpiRAM, it shows lower on-chip learning accuracy due to its large cycle-to-cycle variation. Besides, for analog synapses, abundant conductance levels are also needed to provide sufficient precision for weight update.

Conclusions

In this paper, new features of the MLP + NeuroSimV3.0 are introduced. For analog synapses, hybrid precision synapse and advanced learning algorithms are added to increase the on-chip learning accuracy. For digital synapses, the parallel partial sum read-out scheme is supported to reduce the read latency for SRAM array. For future versions, the spiking neural network is to be supported. The hardware modules to support on-chip learning will be added.

ACKNOWLEDGMENTS

This work is supported by NSF-CCF-1903951, NSF/SRC E2CDA program, and ASCENT, one of SRC/DRAPA JUMP centers.

REFERENCES

1) S. Yu, "Neuro-inspired computing with emerging nonvolatile memorys," in Proceedings of the IEEE, vol. 106, no. 2, pp. 260-285, Feb. 2018.
2) X. Si, J.-J. Chen, Y.-N. Tu, W.-H. Huang, J.-H. Wang, W.-C. Wei, S.-Y. Wu, X. Sun, R. Liu, S. Yu, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, Q. Li, M.-F. Chang, "A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning," IEEE International Solid-State Circuits Conference (ISSCC) 2019
3) W. Chen, K. Li, W. Lin, K. Hsu, P. Li, C. Yang, C. Xue, E. Yang, Y. Chen, Y. Chang, T. Hsu, Y. King, C. Lin, R. Liu, C. Hsieh, K. Tang and M. Chang., "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," 2018 IEEE International Solid - State Circuits Conference - (ISSCC), San Francisco, CA, 2018, pp. 494-496.
4) R. Liu, X. Peng, X. Sun, W.S. Khwa, X. Si, J. J. Chen and S. Yu, "Parallelizing SRAM arrays with customized bit-cell for binary neural networks", In Proceedings of the 55th Annual Design Automation Conference (p. 21). ACM, Jun. 2018.
5) L. Song, X. Qian, H. Li and Y. Chen, "PipeLayer: A Pipelined ReRAM-Based Accelerator for Deep Learning," 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA), Austin, TX, 2017, pp. 541-552.
6) H. Ji, L. Song, L. Jiang, H. H. Li and Y. Chen, "ReCom: An efficient resistive accelerator for compressed deep neural networks," 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2018, pp. 237-240.
7) P. Chen, X. Peng and S. Yu, "NeuroSim: A Circuit-Level Macro Model for Benchmarking Neuro-Inspired Architectures in Online Learning," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 37, no. 12, pp. 3067-3080, Dec. 2018.
8) P. Chen, X. Peng and S. Yu, "NeuroSim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," 2017 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, 2017, pp. 6.1.1-6.1.4.
9) https://github.com/neurosim/MLP_NeuroSim_V3.0
10) Y. Li, S. Kim, X.Sun, P. Solomon, T. Gokman, H. Tsai, S. Koswatta, Z. Ren, R. Mo, C. Yeh, W. Haensch and E. Leobandung, "Capacitor-based Cross-point Array for Analog Neural Network with Record Symmetry and Linearity," 2018 IEEE Symposium on VLSI Technology, Honolulu, HI, 2018, pp. 25-26.
11) S. Ambrogio, P. Narayanan, H. Tsai, R. M. Shelby, I. Boybat, C. Nolfo, S. Sidler, M. Giordano, M. Bodini, N. C. Farinha, B. Killeen, C. Cheng, Y. Jaoudi and G. W. Burr, "Equivalent-accuracy accelerated neural-network training using analogue memory". Nature, 558(7708), p.60, 2018
12) X. Sun, P. Wang, K. Ni, S. Datta and S. Yu, "Exploiting Hybrid Precision for Training and Inference: A 2T-1FeFET Based Analog Synaptic Weight Cell," 2018 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, 2018, pp. 3.1.1-3.1.4.
13) M. Jerry P. Chen, J. Zhang, P. Sharma, K. Ni, S.Yu and S. Datta, "Ferroelectric FET analog synapse for acceleration of deep neural network training," 2017 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, 2017, pp. 6.2.1-6.2.4.
14) D. Kuzum, R. Jeyasingh, B. Lee and H.-S.P.Wong, "Nanoelectronic programmable synapses based on phase change materials for brain-inspired computing," Nano Letters, 12.5 (2011): 2179-2186.
15) http://ruder.io/optimizing-gradient descent/index.html#momentum
16) S. H. Jo, T. Chang, I. Ebong, B. Bhavitavya, P. Mazumder and W. Lu. "Nanoscale memristor device as synapse in neuromorphic systems." Nano Letters, 10.4 (2010): 1297-1301.
17) S. Choi, S. Tan, Z. Li, Y. Kim, C. Choi, P. Chen, H. Yeon, S. Yu and J. Kim. "SiGe epitaxial memory for neuromorphic computing with reproducible high performance based on engineered dislocations." Nature Materials, 17.4 (2018): 335-340.
18) S. Wu, G. Li, F. Chen and L. Shi, "Training and inference with integers in deep neural networks," arXiv preprint (2018):1802:04680.
19) P.-Y. Chen, S. Yu, "Technological benchmark of analog synaptic devices for neuro-inspired architectures," IEEE Design & Test, 2019
20) W. Wu, H. Wu, B. Gao, P. Yao, X. Zhang, X. Peng, S. Yu and H. Qiang. "A methodology to improve linearity of analog RRAM for neuromorphic computing." IEEE Symposium on VLSI Technology, 2018.
21) S. Park, A. Sheri, J. Kim, J. Noh, J. Jang, M. Jeon, B. Lee, B. R. Lee, B. H. Lee, and H. Hwang, "Neuromorphic speech systems using advanced ReRAM-based synapse." IEEE International Electron Devices Meeting, 2013.
22) J. Woo, K. Moon, J. Song, S. Lee, M. Kwak, J. Park and H. Hwang, "Improved synaptic behavior under identical pulses using AlOx/HfO2 bilayer RRAM array for neuromorphic systems." IEEE Electron Device Letters, 37.8 (2016): 994-997.

`

Table 3. The Benchmark Table in NeuroSimV3.0 (at 32nm node)

	Analog eNVM synapses						
Device type	**Ag:a-Si [16]**	**TaO_x/HfO_x [20]**	**PCMO [21]**	**AlO_x/HfO_2 [22]**	**GST PCM [14]**	**EpiRAM (Ag:SiGe) [17]**	**HZO FeFET [13]**
# of conductance states	97	128	50	40	100-120	64	32
Nonlinearity (weight increase/decrease)	2.4/-4.88	0.04/-0.63	3.68/-6.76	1.94/-0.61	0.105/2.4	0.5/-0.5	1.75/1.46
R_{ON}	26 MΩ	100 KΩ	23 MΩ	16.9 KΩ	4.71 KΩ	81 KΩ	559.28 KΩ
ON/OFF ratio	12.5	10	6.84	4.43	19.8	50.2	45
Weight increase pulse	3.2V/300µs	1.6V/50ns	-2V/1ms	0.9V/100µs	0.7V (avg.)/ 6µs	5V/5µs	3.65V (avg.)/ 75ns
Weight decrease pulse	-2.8V/300µs	1.6V/50ns	2V/1ms	-1V/100µs	3V (avg.)/ 125ns	-3V/5µs	-2.95V (avg.)/ 75ns
Cycle-to-cycle variation (σ)	3.5%	3.7%	<1%	5%	1.5%	2%	<0.5%
Online learning accuracy	~72%	~80%	~33%	~20%	89%	92%	88%
Area	6292.3 μm^2	8663.1 μm^2	6292.4 μm^2	21,760 μm^2	46,565μm^2	9144.3μm^2	7032.6μm^2
Latency (optimized)	31997s	10.15s	12218s	470.42s	203.0s	229.6s	2.73s
Energy (optimized)	13.44mJ	4.01mJ	2.53mJ	15.26mJ	35.0mJ	31.01mJ	1.9mJ
Leakage power	105.65µW	105.65µW	105.65µW	105.65µW	105.65µW	105.65µW	105.65µW

	Digital synapse			Hybrid precision synapse	
Device type	**6-bit SRAM (row-by-row)**	**6-bit SRAM (parallel)**	**6-bit STT-MRAM (row-by-row)**	**3T1C+2PCM [11]**	**2T1F [12]**
# of conductance states	--		--	PCM: 16 Capacitor: 32 (6 bits in total) F=4	HSW: 2bits LSW: 4 bits (6 bits in total) F=16
Nonlinearity (weight increase/decrease)	--		--	Cap.: 0.2/-0.2; PCM: 0.105 (LTP only)	0.5/0.5
R_{ON}	--		3.5 KΩ	PCM: 4.71 KΩ; Cap. : 25KΩ	559.28KΩ
ON/OFF ratio	--		2.3	PCM: 19.8; Cap. : 20	45
Weight increase pulse	--		1V/10ns	HSW: 0.7V (avg.) /6µs LSW:1V/300ps	HSW: 2-4V/3µs LSW: 1V/300ps
Weight decrease pulse	--		1V/10ns	Capacitor:1V/300ps	LSW: 1V/300ps HSW: 2-4V/3µs
Cycle-to-cycle variation (σ)	--		--	PCM: 1.5%, Cap.: 0.5%	1.5%
Online learning accuracy	~94%	~94%	~94%	93.8%	94.6%
Area	65,728 μm^2	74,699μm^2	66,632 μm^2	330,330 μm^2	334,270μm^2
Latency (optimized)	5.98 s	1.73s (0.11s for read)	90.1s	3.15s (3.06s for transfer)	0.38s (0.28s for transfer)
Energy (optimized)	15.56 mJ	19.1 mJ	146.2 mJ	16.69 mJ	7.53 mJ
Leakage power	2.80 mW	2.69 mW	84.0 µW	1.66 mW	2.94 mW

Table 4. The Benchmark Table in NeuroSimV2.0 (at 32nm node)

	Analog eNVM synapses							Digital synapse
Device type	**Ag:a-Si**	**TaO_x/HfO_x**	**PCMO**	**AlO_x/HfO_2**	**GST PCM**	**EpiRAM**	**HZO FeFET**	**6-bit SRAM (row-by-row)**
# of conductance states	97	128	50	40	100-120	64	32	
Nonlinearity (weight increase/decrease)	2.4/-4.88	0.04/-0.63	3.68/-6.76	1.94/-0.61	0.105/2.4	0.5/-0.5	1.75/1.46	
R_{ON}	26 MΩ	100 KΩ	23 MΩ	16.9 kΩ	4.71 kΩ	81 kΩ	559.28 kΩ	
ON/OFF ratio	12.5	10	6.84	4.43	19.8	50.2	45	
Weight increase pulse	3.2V/300µs	1.6V/50ns	-2V/1ms	0.9V/100µs	0.7V (avg.)/ 6µs	5V/5µs	3.65V (avg.)/ 75ns	
Weight decrease pulse	-2.8V/300µs	1.6V/50ns	2V/1ms	-1V/100µs	3V (avg.)/ 125ns	-3V/5µs	-2.95V (avg.)/ 75ns	
Cycle-to-cycle variation (σ)	3.5%	3.7%	<1%	5%	1.5%	2%	<1%	
Online learning accuracy	76.2%	73.7%	10.1%	19.5%	87%	92.6%	90%	94.5%
Area	5594.6μm^2	7965.3μm^2	5594.6μm^2	14834μm^2	45203μm^2	8445.9μm^2	6334.8μm^2	53944μm^2
Latency (optimized)	68327s	33.3s	1.78s	3161.2s	386.0s	294.36s	3.11s	4.19s
Energy (optimized)	27.65mJ	9.72mJ	0.89mJ	136.03mJ	83.86mJ	45.95mJ	1.19mJ	9.48mJ
Leakage power	93.74µW	93.74µW	93.74µW	93.74µW	93.74µW	93.74µW	93.74µW	2.65mJ