Memory Organization for Energy-Efficient Learning and Inference in Digital Neuromorphic Accelerators

Clemens JS Schaefer*, Patrick Faley*, Emre O Neftci† and Siddharth Joshi*
*Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA
†Department of Cognitive Sciences and Department of Computer Science, UC Irvine, Irvine, CA, USA
Email: {cschaef6, pfaley, sjoshi2}@nd.edu, eneftci@uci.edu

***Abstract*—The energy efficiency of neuromorphic hardware is greatly affected by the energy of storing, accessing, and updating synaptic parameters. Various methods of memory organisation targeting energy-efficient digital accelerators have been investigated in the past, however, they do not completely encapsulate the energy costs at a system level. To address this shortcoming and to account for various overheads, we synthesize the controller and memory for different encoding schemes and extract the energy costs from these synthesized blocks. Additionally, we introduce functional encoding for structured connectivity such as the connectivity in convolutional layers. Functional encoding offers a 58% reduction in the energy to implement a backward pass and weight update in such layers compared to existing index-based solutions. We show that for a 2 layer spiking neural network trained to retain a spatio-temporal pattern, bitmap (PB-BMP) based organization can encode the sparser networks more efficiently. This form of encoding delivers a 1.37× improvement in energy efficiency coming at the cost of a 4% degradation in network retention accuracy as measured by the van Rossum distance.**

I. Introduction

Biological organisms operate autonomously with extreme energy efficiency, learning continuously amid unreliable and noisy environmental stimuli. Designing artificial autonomous systems that can embody these traits remains a grand challenge in engineering. Taking inspiration from biology, autonomous systems using biologically inspired computational and sensory systems have been able to deliver remarkable results over the past few decades [1]–[4]. Neuromorphic computing aims to develop hardware and algorithms that embody the principles upon which biology operates [5]–[7], crucially, while maximizing the energy efficiency of learning.

Neuromorphic hardware platforms designed to efficiently run large-scale spiking neural networks generally consist of a neuro-synaptic core and a communication fabric to connect multiple such cores. The neurosynaptic core, in turn, is generally composed of some form of a neuron subsystem which implements spike accumulation and all calculations associated with membrane potential dynamics; a synapse subsystem which stores associated weights and implements synaptic dynamics; and if learning is supported, then additional circuitry to implement synaptic plasticity [5], [6], [8]–[10].

In this paper, we focus on representations of synaptic connectivity in digital memories for energy-efficient inference and learning in neuromorphic spiking architectures. Specifically, we examine the energetic impact of implementing backpropagation-through-time (BPTT) using a spike-based learning rule together with a surrogate gradient (SG). This learning rule minimizes a global loss function, together with the SG, BPTT can enable supervised gradient based learning despite the discontinuity induced by spiking non-linearities encountered in Spiking Neural Networks (SNNs) [11]. Over the course of this paper, we examine how different storage schemes impact the energy required to implement BPTT.

In digital memories implemented in large-scale neuromorphic systems, the organization and representation of synaptic parameters in memory impacts the energy efficiency of learning [12] and inference [13]. This is exemplified in the difference in forward and reverse memory access patterns in a crossbar memory and an index-based memory [14]. In a crossbar memory, since both the synaptic weight matrix and its transpose are immediately accessible [15], given a pre-synaptic neuron all post-synaptic neurons connected to it can be determined and with equal ease, given a post-synaptic neuron, all neurons pre-synaptic to it can be determined. However, index-based storage schemes often sacrifice ease of reverse access for storage efficiency. This has led to investigations into various formats for storage efficiency [12], [16], [17]. Different structures for forward access storage efficiency are examined in [16], which argue for a bitmap based approach. This was countered by investigations in [17] which determined that storage and complexity of access are both crucial, arguing for an indexed-list based approach. However, rather than directly studying energy efficiency, prior work has examined this question through the lens of number of bits needed for storage or the number of operations required to access the weights. This paper explores the design space of memory organization against the more direct metric of access energy in order to re-examine the results of prior work in light of both algorithmic and circuit considerations.

The central contributions of this paper are two-fold: first, we introduce a functional approach to storing regular and structured connectivity such as the connectivity in convolutional layers; second, rather than taking access efficiency or storage as proxies for energy as done in [16], [17], we directly compare the energy cost induced by the different data encoding schemes. In order to accomplish this energy comparison, we obtain the energy of the different schemes by synthesizing a datapath, a controller, and the associated memories in a 40 nm CMOS technology. The memory results were verified against

978-1-7281-3320-1/20/$31.00 ©2020 IEEE

CACTI [18] (a SPICE accurate architectural memory model) to ensure correctness.

II. Background

A. Surrogate Gradient Learning

The model used in this paper consists of networks of plastic integrate-and-fire neurons, expressed here in discrete time:

$$U_i^{(l)}[n] = \sum_j W_{ij}^{(l)} P_j[n] - \delta R_i[n], \tag{1}$$

$$\begin{aligned} S_i^{(l)}[n] &= \Theta(U_i^{(l)}[n] - \vartheta) \\ Q_j[n+1] &= \alpha Q_j[n] + S_j^{(l-1)}[n], \\ P_j[n+1] &= \beta P_j[n] + Q_j[n], \\ R_i[n+1] &= \gamma R_i[n] + S_i[n]. \end{aligned}$$

where $U_i^{(l)}[n]$ is the membrane potential of neuron i at layer l at time step n, W is synaptic weight matrix, ϑ is the firing threshold and $S_i^{(l)}$ is the spiking output of this neuron. The function Θ is the step function, *i.e.* $S_i^{(l)}[n] = 1$ when $U_i^{(l)}[n] = 0$. The constants β, γ and α capture the decay dynamics of the membrane potential U_i, the refractory (resetting) state R_i and the synaptic state Q_i and can be related to time constants in leaky integrate-and-fire neurons. States P and Q describe the traces of the membrane and the current-based synapse, respectively. R is a refractory state that resets and inhibits the neuron after it has emitted a spike, and δ is the constant that controls its magnitude. Note that (1) is equivalent to a discrete-time version of the Spike Response Model $(\text{SRM})_0$ with linear filters [19]. This SNN and the ensuing learning dynamics can be transformed into a standard binary neural network by setting all $\alpha = 0$, replacing all $P_j[n]$ with $S_j^{(l)}[n-1]$ and dropping Q and R.

Assuming a global cost function $\mathcal{L}$, the gradients with respect to the weights in layer l can be computed using backpropagation-through-time [11]. Θ is non-differentiable but following a surrogate gradient learning, Θ's derivative can be replaced by a smooth sigmoidal or piecewise constant function for optimization purposes [11]. Our experiments make use of the normalized negative part of a fast sigmoid function and a Van Rossum distance [20].

B. Quantization

Efficient implementations of learning on-chip entail learning with quantized weights, gradients, and membrane voltage dynamics. To accurately model the effect of quantizing the weight and gradient values we follow the procedures outlined in [21] and [22], as shown in Fig. 1. Weights are quantized by restricting them to a feasible weight range defined by ($\min = -1 + \sigma(b_w)$ and $\max = +1 - \sigma(b_w)$), where b_w is the number of bits encoding the weight and $\sigma(b) = 2^{1-b}$.

To prevent overflow, weights in each layer are scaled by η where:

$$\eta = 2^{\text{round}\left[\log_2\left(\frac{(\frac{1}{\sigma(b_w)} - 0.5)\cdot\sigma(b_w)}{\sqrt{\frac{3}{\text{fan in}}}}\right)\right]}, \tag{2}$$

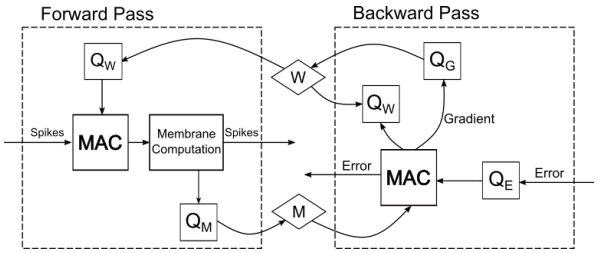

Fig. 1. Quantization schematic illustrating the quantization process in forward and backward pass. Note the membrane potential (M) needs to be stored for the backward pass. Squares indicate operations (i.e. Q_E quantize error) and diamonds stored values (i.e. membrane potential and weights).

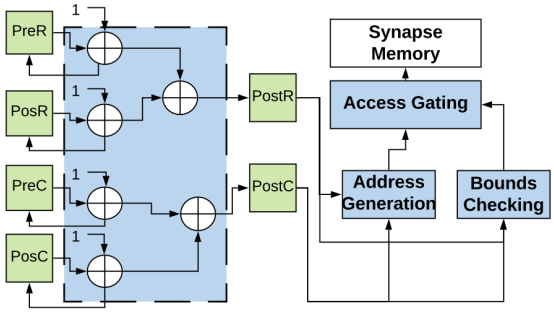

Fig. 2. The schematic of the logic used to generate the convolutional sparsity pattern. The blue-shaded elements are combinational logic while the green shaded elements are registers. Precomputed parameters are used to determine if the read address is valid, preventing spurious reads to minimize energy consumption.

and *fan in* represents the number of connections into a layer. In the backward pass the error is first normalized by its greatest absolute value and then clipped and quantized to ensure precision is maintained. After computing and normalizing the gradients, stochastic rounding is applied to increase the gradient precision when learning over multiple epochs.

C. Encoding Connectivity and Weights

Index based and bitmap based representations can improve the efficiency with which different synaptic connectivity patterns are stored in digital memories [13], [16], [17], [23]. Here, we very briefly introduce the terminology used in this paper, but refer the readers to [16], [17] for a more in-depth review of the topic. Crossbar (CB) based storage schemes sequentially store all potential synaptic parameters between input (pre-synaptic neurons) and outputs (post-synaptic neurons) in the memory. This offers constant-time access to any synapse based on the post and pre-synaptic neuron address [12], [24].

Alternatively, index based methods are better suited to storing sparse connections. Such methods only store the nonzero connections and an additional set of pointers, obviating the need to store any absent synapses. Two sparse representation schemes have been introduced in [16], [17] within the context of weight storage; these are the compressed sparse row (PB-CSR) and pointer based bitmap (PB-BMP). Due to the flexibility afforded by different index-based sparse storage schemes

they are also employed in Intel Loihi [8]. In addition to these general representation schemes, this paper also introduces a functional scheme to better represent regular and patterned connectivity. Through functional encoding, the connectivity information can be derived through run-time reconfigurable combinational logic shown in Fig. 2. Hence only the synaptic parameters that exist need be stored. Zero-weight synapses don't require explicit storage and the connectivity can be computed through the function. Since, the connectivity is computed rather than stored, this saves on the memory size as well as the energy required per memory access. This saves on both the size of the memory and the number of reads to the memory which are more expensive than evaluation combinational logic.

Encoding the connectivity pattern induced by convolutions into a function (see (3), (4)) incurs a few integer additions and subtractions while iterating over the size of the convolutional kernel. The latency and energy cost of this is minimal in comparison to a register file look-up. Further, parts of this can be executed in parallel to enable multi-bank memory access if required. The function that governs the forward lookup is written as a two-dimensional function given by:

$$\begin{aligned} f(PreR, PosR) &= PreR + PosR \\ f(PreC, PosC) &= PreC + PosC \end{aligned} \tag{3}$$

where, *Pre* and *Post*, represent the pre and post-synaptic neurons respectively, *R* and *C* represent the row and column in a 2D kernel, and *pos* represents an iterator to generate all fanouts for a given presynaptic neuron. Similarly, the inverse function is given by:

$$\begin{aligned} f^{-1}(PostR, PosR) &= PostR - PosR \\ f^{-1}(PostC, PosC) &= PostC - PosC. \end{aligned} \tag{4}$$

III. Energy Impact of Connectivity Encoding

In order to study the interplay between memory storage, access efficiency, synaptic parameter quantization, and energy efficiency we implement these storage schemes at the RTL level. We synthesized controllers and the memory and datapaths, for the different encoding strategies, in 40 nm CMOS using a standard Synopsys flow. Memory read and write energies extracted from this are verified against a CACTI model for the same technology-node. In all the experiments reported below, unless otherwise mentioned, only the active energy of this synthesized system is reported. These energies are reported for one entire forward or backward pass of a layer in an SNN. In each case the indirection tables and the weight tables are split into different memories, to minimize the energy-cost of indirection. Figure 3 shows the effect of weight precision on the energy efficiency for different encoding schemes for both the convolution (bottom) and the fully connected (top) layers. The energy for the forward pass across an entire layer is shown on the left, while the energy for the backward pass for an entire layer is shown on the right. We examine the energy required to implement a forward

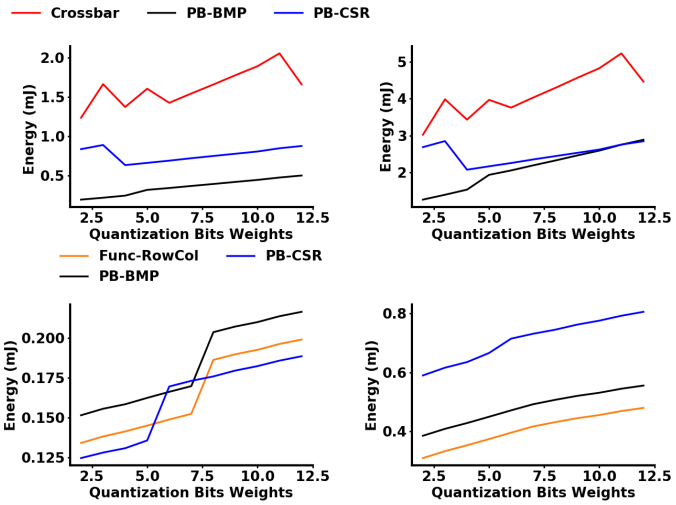

Fig. 3. Top: Energy trade-offs for different levels of weight quantization for a fully connected layer (input 728, output 128) given a sparsity level of 25%. The left graph shows the energy cost associated with memory accesses during a forward pass and right graph shows the energy cost associated with memory access and writes during a backward pass. Bottom: Energy trade-offs for different levels of weight quantization for a convolutional layer (input 28×28, size 3×3, 32 in channels, 32 out channels). Left graph shows energy cost incurred during a forward pass and right graph shows the energy cost incurred a backward pass.

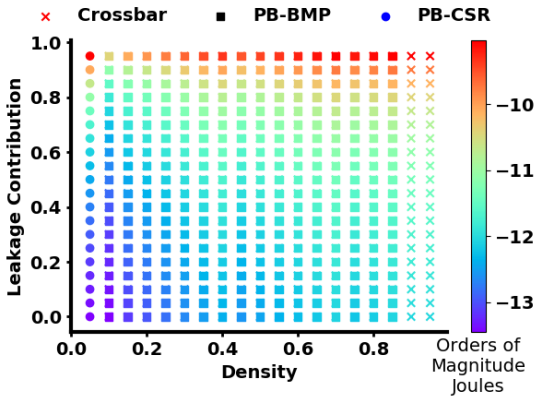

Fig. 4. Order of magnitude of the minimal energy consumption for a fully connected layer (input 728, output 128) with 8-bit weight precision. For the range of leakage energy contribution and the density nonzeroes in the layer, we denote the energy through the color and the most efficient storage scheme through the symbol. The energy is calculated for a combination of the forward and backward pass.

and a backward pass for a fully connected layer of size 728×128 with a density of 75% (sparsity of 25%). For the FC layer, in the forward pass, the PB-BMP structure delivers the lowest energy over a range of weight quantization values. This is due to the more compact representation leading to smaller memories which are not penalized as much during write operations. For the convolutional layer, we implement convolution over an input of size 28×28 with a filter of size 3×3. While the energy cost of calculating the connectivity through a function is comparable to that of the index based

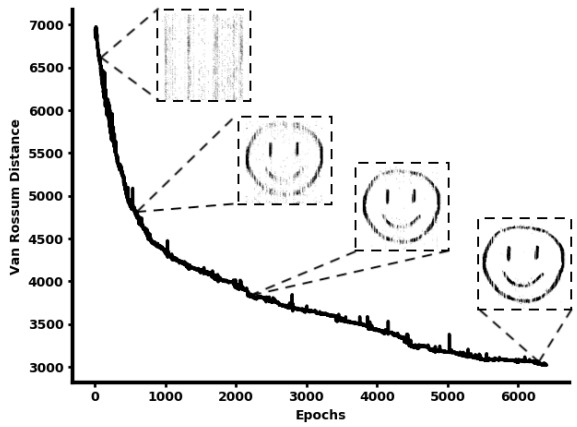

Fig. 5. A learning curve with an exemplary output of the SNNs after 60, 600, 2000 and 6400 epochs, we use the Van Rossum distance as a measure of the SNNs performance.

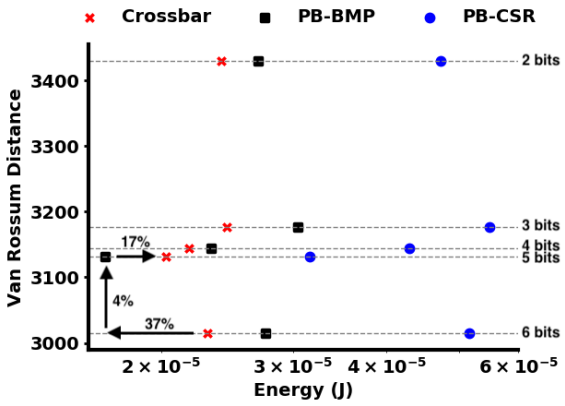

Fig. 6. The accuracy and energy tradeoff for a 700-400-250 neural network at different weight precisions. We annotate the accuracy achieved by a network with the respective quantization levels: 2-6 bits. The weights in the network are encoded using CB, PB-BMP, and PB-CSR memory access schemes, since lower precision also results in a larger number of zeros and thus greater sparsity, this impacts the efficiency with which the that network can be processed.

PB-CSR, the energy consumed in the backward pass when using our function is much lower due to far fewer memory accesses.

Figure 4 shows the performance of the different connectivity encoding schemes when connection density is varied over the range 5% to 100% while simultaneously varying the sparsity of input activity. Since the frequency of activity corresponds to the frequency with which a forward and reverse pass through the memory occur, sparse activity would correspond to leakage power from the memory dominating. Interestingly, the results contradict general assumption from previous work [17] where the number of memory accesses dominated all energy. This is because the size of the memory has an out-sized impact on the access and write energy as well as the leakage energy.

IV. System Design Considerations

We analyze the effect of different encoding schemes and varying resolution on the accuracy of a spiking network trained as outlined in Section II-A. The network, trained on a spatio-temporal pattern as shown in Fig. 5, consists of one input layer with 700 neurons, one hidden layer with 400 neurons, and 250 output neurons. We provide an input of 700 Poisson spike trains over 250 time steps with varying inter-spike intervals. The target was generated by taking a clean pattern and multiplying it with Bernoulli noise ($p = .95$). We use the van Rossum (VR) distance [25] as a loss function; the VR distance calculations are performed at floating point precision and the energy for these calculations is ignored in the experiments. The parameters defining neuron and synaptic dynamics are set values amenable to compact hardware, *e.g.* single-tap FIR filters for membrane and synaptic dynamics. The VR distance is then recorded over 10000 epochs of training.

We determine the energy required to encode the two layers of the SNN through the different encoding schemes. Different bit-precisions affect the accuracy (VR distance) achieved by the network while simultaneously affecting the energy associated with accessing and writing to the weight memories. We capture these trade-offs in Fig. 6. The highest accuracy is achieved with 6 bit weight resolution, where the lowest energy consumption corresponds to the CB structure. However, the PB-BMP scheme, when used with 5-bit quantized weights is the most energetically-efficient scheme across all experiments. This is in part due to the network having a sparsity of 27% over the course of the training.

V. Conclusion

To summarize, we proposed a new functional method to encode connectivity and weight storage for convolutional layers. We tested both fully-connected and convolutional layers and analyzed the impact of the storage cost as well as weight access cost on the net energy. This provides a more holistic view of the design of digital systems implementing SNNs, unlike previous work [16], [17] which focused mostly on storage cost. Using the proposed function to encode structured connectivity approximately doubles the energy efficiency of implementing the backward pass of a convolutional layer composed of 8-bit synapses when compared to the PB-CSR. In the forward pass, this function incurs an overhead of 5% in energy over the PB-CSR. Leading to a net energy saving when weight updates are frequent, such as in the context of continuous learning. Additionally, we compared the energy required to train a quantized SNN stored using various data encoding schemes. An SNN stored using PB-BMP required 19.66% more energy than CB for a VR distance of 3015 using 5-bit weights. However, when trained with 2-bit weight precision, then for the same VR distance, PB-BMP can exploit the increased sparsity (27%) in connectivity to consume 14.59% lower energy than CB.

REFERENCES

[1] T. Delbruck and M. Lang, "Robotic goalie with 3 ms reaction time at 4% cpu load using event-based dynamic vision sensor," *Frontiers in neuroscience*, vol. 7, p. 223, 2013.

[2] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "Hats: Histograms of averaged time surfaces for robust event-based object classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1731–1740.

[3] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.

[4] J. Kaiser, A. Friedrich, J. Tieck, D. Reichard, A. Roennau, E. Neftci, and R. Dillmann, "Embodied event-driven random backpropagation," *arXiv preprint arXiv:1904.04805*, 2019.

[5] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses," *Frontiers in neuroscience*, vol. 9, 2015.

[6] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[7] J. Schemmel, D. Brüderle, A. Grübl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *International Symposium on Circuits and Systems, ISCAS 2010*. IEEE, 2010, pp. 1947–1950.

[8] M. Davies, N. Srinivasa, T. H. Lin, G. Chinya, P. Joshi, A. Lines, A. Wild, and H. Wang, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. PP, no. 99, pp. 1–1, 2018.

[9] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He *et al.*, "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nature*, vol. 572, no. 7767, p. 106, 2019.

[10] S. Friedmann, J. Schemmel, A. Grübl, A. Hartel, M. Hock, and K. Meier, "Demonstrating hybrid learning in a flexible neuromorphic hardware system," *IEEE transactions on biomedical circuits and systems*, vol. 11, no. 1, pp. 128–142, 2017.

[11] E. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks," *Signal Processing Magazine, IEEE*, Dec 2019, (accepted).

[12] J. Kim, J. Koo, T. Kim, and J.-J. Kim, "Efficient synapse memory structure for reconfigurable digital neuromorphic hardware," *Frontiers in neuroscience*, vol. 12, p. 829, 2018.

[13] A. Aimar, H. Mostafa, E. Calabrese, A. Rios-Navarro, R. Tapiador-Morales, I.-A. Lungu, M. B. Milde, F. Corradi, A. Linares-Barranco, S.-C. Liu *et al.*, "Nullhop: A flexible convolutional neural network accelerator based on sparse representations of feature maps," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 3, pp. 644–656, 2018.

[14] B. U. Pedroni, S. Sheik, S. Joshi, G. Detorakis, S. Paul, C. Augustine, E. Neftci, and G. Cauwenberghs, "Forward table-based presynaptic event-triggered spike-timing-dependent plasticity," Oct 2016.

[15] M. Payvand, M. Fouda, A. Etawil, F. Kurdahi, and E. Neftci, "Error-triggered three-factor learning dynamics for crossbar arrays," *arXiv preprint arXiv:1910.06152*, Dec 2019.

[16] S. Joshi, B. U. Pedroni, and G. Cauwenberghs, "Neuromorphic event-driven multi-scale synaptic connectivity and plasticity," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2017, pp. 1–5.

[17] B. U. Pedroni, S. Joshi, S. Deiss, S. Sheik, G. Detorakis, S. Paul, C. Augustine, E. O. Neftci, and G. Cauwenberghs, "Memory-efficient synaptic connectivity for spike-timing-dependent plasticity," *Frontiers in neuroscience*, vol. 13, p. 357, 2019.

[18] R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee, and V. Srinivas, "Cacti 7: New tools for interconnect exploration in innovative off-chip memories," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 14, no. 2, p. 14, 2017.

[19] W. Gerstner and W. Kistler, *Spiking Neuron Models. Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[20] F. Zenke and S. Ganguli, "Superspike: Supervised learning in multi-layer spiking neural networks," *arXiv preprint arXiv:1705.11146*, 2017.

[21] S. Wu, G. Li, F. Chen, and L. Shi, "Training and inference with integers in deep neural networks," *arXiv preprint arXiv:1802.04680*, 2018.

[22] A. Yousefzadeh, E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco, "On practical issues for stochastic stdp hardware with 1-bit synaptic weights," *Frontiers in neuroscience*, vol. 12, 2018.

[23] X. Jin, A. Rast, F. Galluppi, S. Davies, and S. Furber, "Implementing spike-timing-dependent plasticity on spinnaker neuromorphic hardware," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2010, pp. 1–8.

[24] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *Custom Integrated Circuits Conference (CICC), 2011 IEEE*, Sept. 2011, pp. 1–4.

[25] M. v. Rossum, "A novel spike distance," *Neural computation*, vol. 13, no. 4, pp. 751–763, 2001.