Effect of Line Resistance of Passive Memristive Crossbars on Spiking Neural Network Performance

Pierre Lewden, Adrien Vincent, Jean Tomas, Chip-Hong Chang, Sylvain Saïghi

Effect of Line Resistance of Passive Memristive Crossbars on Spiking Neural Network Performance

Pierre Lewden*, Adrien F. Vincent†, Jean Tomas†, Chip-Hong Chang‡*, and Sylvain Saïghi†*
Email: pierre.lewden@cnrsatcreate.sg, {adrien.vincent, jean.tomas, sylvain.saighi}@u-bordeaux.fr, echchang@ntu.edu.sg
* CNRS@CREATE, 1 Create Way, 08-01 Create Tower, Singapore 138602
† Univ. Bordeaux, CNRS, Bordeaux INP, IMS, UMR 5218, F-33400 Talence, France
‡ School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

***Abstract*—Hardware spiking neural networks using memristors as synapses are promising candidates for highly integrated and low-power event-based computations. However, when using passive synaptic crossbars, a number of limitations linked to the high parallelism of the memristive synapses appear. In this paper, we focus on the detrimental effect of the synaptic line resistance of passive crossbars made of ferroelectric tunnel junction memristive devices on learning capabilities. Using a model of such memristive device that considers the actual voltage applied on the synapses when updating their weights, we show how this line resistance impacts the performance of a single layer all-to-all spiking neural network using unsupervised and reward-modulated learning, and how these detrimental effects can be mitigated.**

***Index Terms*—Memristors, passive crossbar, reward-modulated learning, spiking neural networks, unsupervised learning.**

I. Introduction

Neuromorphic systems built with Neural Networks (NNs) is an active research field [1]–[3]. It has emerged as a very promising solution to overcome the von Neumann bottleneck in conventional computing architectures, especially for cognitive memory-intensive tasks where the conventional computing architecture shows limited performances [4]. In particular, Spiking Neural Networks (SNNs), the third generation of neural networks [5], use Leaky-Integrate-and-Fire (LIF) neurons as computational units. With the notion of time incorporated into the computing model, they are very efficient when combined with event-based sensors compared with feedforward multi-layer NNs, thanks to their asynchronous (cycle-free) propagation of information.

For compact and low-power integrated systems, the hardware implementation of SNNs can be built using analog neurons and passive synaptic arrays that leverage Kirchhoff's current law for memory storage and transmission of signals. Such designs require specific devices to implement the synapses which is one of the most important element in a NN as they store the synaptic weights (memory points) and they have to be intrinsically plastic for learning capable systems. As in other types of NNs, synapses significantly outnumber neurons in SNNs.

This work is supported by a public grant overseen by the French National Research Agency (ANR) as part of the 'Chaires IA' programme (GrAI project ANR-19-CHIA-0003) and as part of the 'PEPR IA France 2030' programme (Emergences project ANR-23-PEIA-0002). This research is part of the programme DesCartes and is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

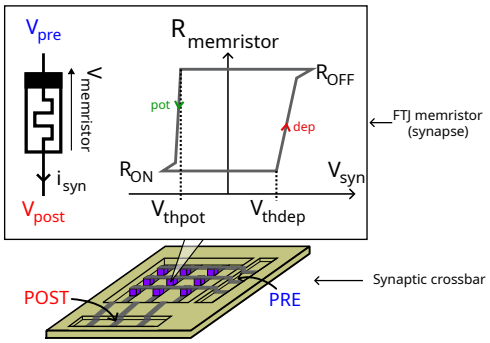

Fig. 1. Passive synaptic crossbar composed of FTJ memristive devices. The presynaptic (PRE lines) and postsynaptic (POST columns) metallic lines connect the corresponding memristive devices terminals in parallel.

Ferroelectric Tunnel Junction (FTJ) memristors are good candidates to make fast, low-power and highly integrable synaptic arrays for hardware implementation of SNNs. FTJ devices can be around a hundred nanometers wide [6]. They have a high conductance state around a few microsiemens [7], which reduces the power consumption during inference, while allowing a weight update by repetitively applying short square voltages (a few hundreds of nanoseconds [7] or shorter [8]).

This work focuses on the impact of postsynaptic line resistance in a crossbar as this is where the number of connected synapses is highest in the system under study (single layer all-to-all SNN). Section II gives the details of the FTJ memristor model used in the simulation as a device and in an array. Section III gives the details of the SNN architecture, the main parameters and how multiple crossbars can be used in a mixed-mode SNN, including digital control, analog synapses and analog neurons. Section IV shows the impact of the postsynaptic line resistance on the performance of the system when using different numbers of crossbar arrays to build the SNN. The results are obtained by simulating a 1156×100 single layer SNN using an in-house Python simulator.

II. Ferroelectric Tunnel Junction Memristors

A. Single device behavior

To be able to study the impact of the voltage drop on a SNN using FTJ devices, one needs a synapse model that takes into

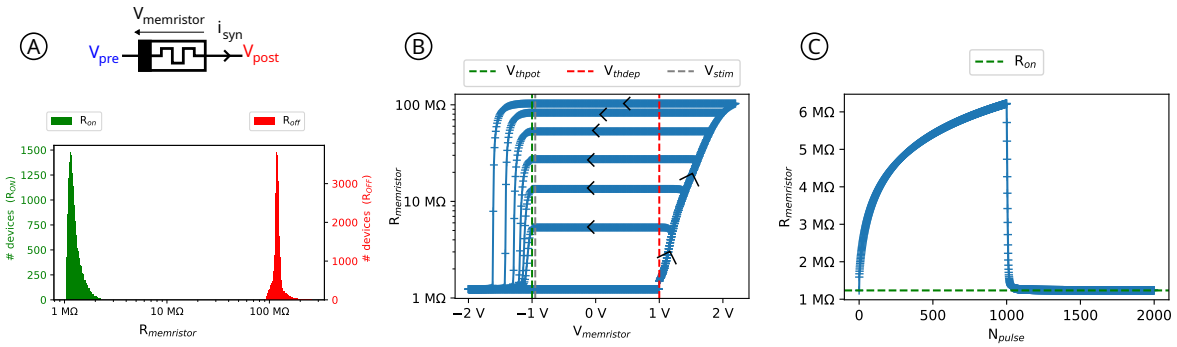

Fig. 2. Model behavior of the FTJ memristor used in simulations. A. An example of the distributions of the R_{ON} and R_{OFF} values of the devices. B. Hysteresis of the device obtained in the simulation by applying voltage pulses of 100 ns. C. Resistance evolution of a simulated device when applying 1000 successive pulses of 1.05 V followed by 1000 pulses of -1.05 V.

account the actual voltage applied on the devices to compute the weight change. For this work, we adapted in Python the Verilog-A behavioral model from Meyer et al. [9] based on BFO memristors with some slight modifications to match our application. The variability of the resistance state changes was removed and the possible R_{ON} and R_{OFF} resistance states were shifted to higher values to have the low resistance state of the devices higher than 1 MΩ and the high resistance state around 100 MΩ, which is among the possible values of such technology [6]. The purpose of scaling up the resistance is to reduce power consumption while limiting the voltage drop along the synaptic lines. Fig. 2.A shows the shifted distributions used to select the R_{ON} and R_{OFF} states for each memristive device in the simulated architecture.

Fig. 2.B shows the kind of behavior (hysteresis in blue) that can be obtained with the model using voltage pulses of 100 ns for different maximum amplitudes, showing the ability of the model to reach different non-volatile states depending on the voltage used. The two writing thresholds V_{thdep} (dashed red) and V_{thpot} (dashed green) that need to be surpassed to cause a weight change are also shown. The inference pulse amplitude V_{stim} (dashed grey) used to charge the postsynaptic neurons in an inference phase must be contained between the two writing thresholds.

Fig 2.C. represents the resistance evolution of a simulated device when successive fixed amplitude pulses greater than the writing thresholds are applied. We can see that the application of successive pulses enables the progressive change of the synapse weight, a phenomenon observed with real FTJ devices [7], [10]. This progressive programming through repetitive pulses is used to enable the SNN to learn progressively with each training sample.

B. Driving a crossbar

When building a single all-to-all layer of a SNN with FTJ devices, crossbar arrays are the standard structure chosen for their integrability. In the case of passive crossbars (see Fig. 1), the connection between each synaptic terminal connected to the same neuron is made by a single metallic line. Thus, a small resistance r_{x} exists on a synaptic line between each pair of connected devices and it causes the apparition of a voltage drop between successively connected devices. Fig. 3.A

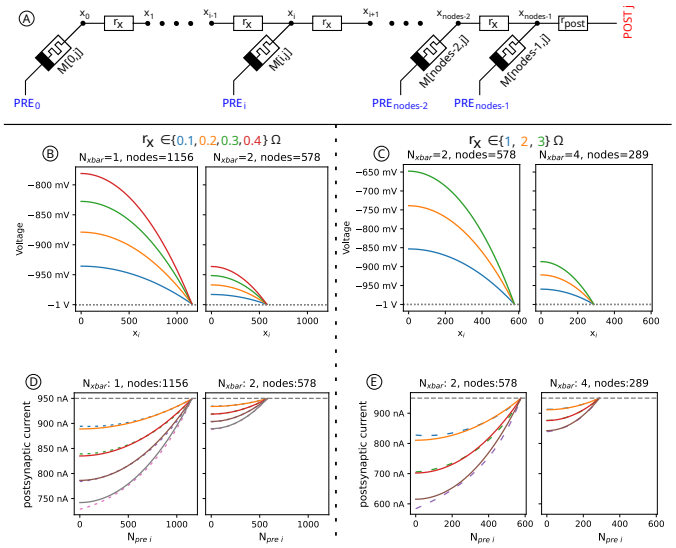

Fig. 3. Examples of results obtained by the voltage solver and by the fitted inference current approximation in the worst scenario (all synaptic memristive devices at $G_{\mathrm{max}} = 1$ µS) for different numbers of connected devices (nodes) and different resistance values r_{x}. A. Equivalent circuit of a postsynaptic line of "nodes" elements. B, C. Computed voltage on the postsynaptic memristive device nodes x_i for different small r_{x} values and a postsynaptic line connecting 1156 inputs ($n_{\mathrm{xbar}} = 1$ crossbar), 578 inputs or 289 inputs. D, E. Postsynaptic current computed with the solver (solid lines) and the adjusted estimator (dashed lines) for different [r_{x} , n_{xbar}] configurations when an inference voltage $V_{\mathrm{stim}}= 0.95$ V is applied at $N_{\mathrm{pre\ i}}$. The grey dashed line represents the ideal case when r_{x} and r_{post} are negligible. For simplicity, $r_{\mathrm{post}}= 1\,\Omega$.

shows the electrical model of a postsynaptic line including this r_{x} resistance. As the number of presynaptic inputs is higher than the number of postsynaptic neurons in our simulations (1156 inputs versus 100 outputs), we assume that the voltage drop along the presynaptic lines is negligible as they are connected to only 100 postsynaptics. This allows us to simplify the mathematical equations and reduce the computation time of SNN simulations. Figs. 3.B and 3.C show the voltage applied on the postsynaptic terminal along the line, assuming that the presynaptic inputs are at 0 V, the postsynaptic voltage of this line is at 1 V, and all the synapses are at 1 MΩ (the worst-case scenario). The x_i voltages at each device terminal are obtained by solving a linear algebraic equation. This method is used to determine the actual voltage on each device for the training phase.

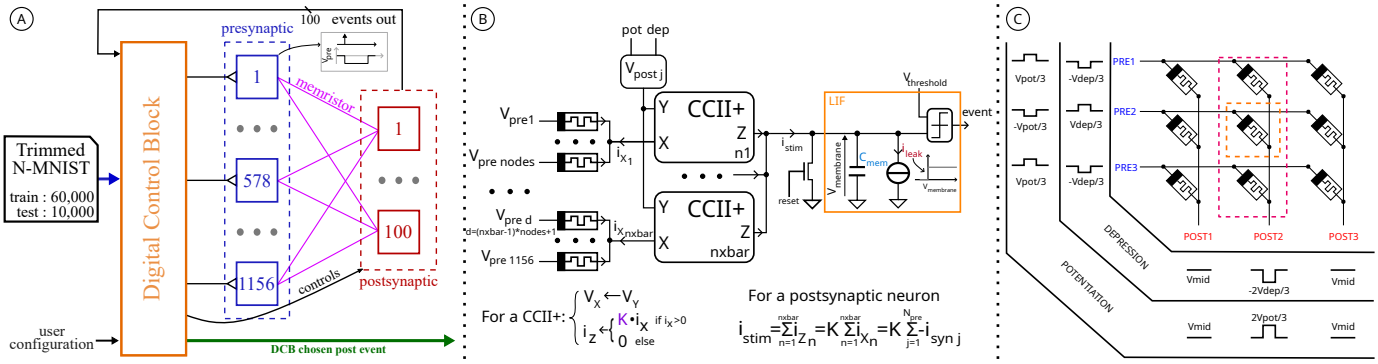

Fig. 4. Architecture of the system and weight modification protocol. A. General architecture of the system. All-to-all single layer analog SNN controlled by a Digital Controlled Block (DCB) that triggers the presynaptic and postsynaptic waveforms according to the dataset and learning rules. B. An overview of a postsynaptic neuron made of a LIF neuron and multiple CCII+ to combine the synaptic currents issued by the synapses of different crossbars if there is more than a single crossbar. C. Example of the weight modification protocol of one memristor (orange dashed box) in a small 3×3 crossbar of the device if the second postsynaptic line (red dashed box) needs to be updated.

Using this method, the postsynaptic current (going through r_{post}) can also be determined by knowing the value of $x_{\text{nodes}-1}$. However, as the computation of the postsynaptic current of each postsynaptic neuron is one of the most called operations in the simulation, the computation time increases drastically. To alleviate this problem, we define an estimator using a simpler generic equation that is empirically fitted to each postsynaptic line length used. Figs 3.D and 3.E show the computed synaptic current obtained when a presynaptic $N_{\text{pre i}}$ is active while all other terminals are at $0\,\text{V}$. We can see that the best fitted curve (dashed lines) produced by the estimator follows the trajectory of the synaptic current obtained with the solver based on Kirchhoff's and Ohm's laws (solid lines).

III. Spiking Neural Network Architecture

A. General architecture

To study the effect of this voltage drop, we consider an all-to-all single layer SNN with 1156 inputs and 100 outputs (architecture and most parameters taken from Lewden et al. [11], [12]) as shown in Fig. 4.A. The SNN is composed of analog neurons with presynaptic neurons applying a $10\,\mu\text{s}$ wide negative voltage ($V_{stim} = 0.95\,\text{V}$) each time a presynaptic event happens. The postsynaptic neurons illustrated in Fig. 4.B collect the postsynaptic current coming from a number of postsynaptic lines through a positive second generation current conveyor (CCII+). The latter copies the synaptic current (only the positive current in our system) with a factor $K = 0.01$ while maintaining the postsynaptic voltage V_{post} on the connected postsynaptic line. If the synaptic array is made using only one crossbar, only one CCII+ is used. If more crossbars are used, by splitting the crossbar along the postsynatpic lines into smaller ones, one CCII+ per crossbar is used to combine the currents.

To test the SNN, we use a subset of the N-MNIST dataset from G. Orchard et al. [13] (handwritten digits from 0 to 9 filmed by a DVS), by using only the first $100\,\text{ms}$ (first saccadic movement) and only the ON events similar to the setting of Lewden et al. [12]. When a postsynaptic neuron fires upon reaching the membrane threshold ($V_{\text{threshold}} = 1\,\text{V}$), the sample presentation is stopped. If the network is in testing mode, the neurons are reset and the next sample is presented. If the SNN is in training mode, a weight update protocol is executed in two steps before presenting the next sample. This weight update protocol is illustrated in Fig. 4.C. It potentiates (increase, step 1) or depreciates (decrease, step 2) the weights (conductances) of interest depending on the learning rule for a postsynaptic line. During those weight updates, the devices that are not updated will see a residual voltage ($\pm V_{\text{pot}}/3$ or $\pm V_{\text{dep}}/3$) that should be lower than the writing thresholds, while the devices being updated should see the writing voltage $V_{\text{pot}} = -1.05\,\text{V}$ or $V_{\text{dep}} = 1.05\,\text{V}$.

B. Learning rules, labeling and performances

We consider two learning rules taken from Lewden et al. [12] in this work. The first one is the 1P1D unsupervised learning rule. It potentiates synapses connected to inputs that have fired at least once and depreciates the others when a postsynaptic neuron is updating its weights. The second rule is called R$_\varnothing$1P1D. It is a weak supervision rule or reward-modulated rule that applies the 1P1D rule if a postsynaptic neuron fires on a correct class in the training phase or do not update any weights at all if the class of the sample does not match the desired label for the neuron. The supervision labels are organized by the index of the neurons. The first 10 postsynaptic neurons are dedicated to the digit 0, the next 10 to 1, etc. To attribute the labels after the learning phase for both the unsupervised and weakly supervised learning rules, we use a heuristic method based on the most recent samples upon which a postsynaptic neuron fired.

Lastly, in our results, the performance we consider is the recognition rate of the neural network defined by how many times the output given by the network (label of the firing neuron) matches the sample class, divided by the number of test samples.

IV. Results

To see the impact of the resistance r_{x} on the performance of the system, we simulated the SNN for multiple configurations

using 5 different initial random weight distributions (uniform law) and synapses parameters (random R_{ON} and R_{OFF} picked according to the distributions given in Fig. 2.A). The different simulated configurations studied were related to the crossbar structure: different values of r_{x} while r_{post} fixed at $1\,\Omega$ for every simulations (see figure Fig 3.A) when using only 1 crossbar of 1156×100, 2 crossbars of 578×100 or 4 crossbars of 289×100.

Fig. 5 shows the average, maximum and minimum results obtained for the 5 different initial random weight distributions. According to our results, when using a single crossbar (blue line), a sharp drop in the average performance is observed for both rules when r_x only increases from $0.1\,\Omega$ to $0.2\,\Omega$ despite synaptic weights that are all higher than $1\,\mathrm{M}\Omega$. The variability in performance is also high depending on the synaptic parameters (R_{ON}, R_{OFF} and starting weights) before plateauing at a lower recognition rate.

Splitting the crossbar can mitigate the loss in performance. Indeed, if the resistance between devices is higher, the performances can be conserved by dividing again the crossbar as shown by the results obtained for a system with 2 or 4 crossbars when $r_{\mathrm{x}}\geq 1\,\Omega$. Thus, from our results, we can conclude that the higher r_{x} is, the more the crossbar should be cut into smaller ones to mitigate the voltage drop. To corroborate this, we can observe from our results Fig. 5 that the performance starts to drop when $r_{\mathrm{x}} = 3\,\Omega$ even with 4 crossbars (green line). This indicates that the crossbar needs to be split again if r_{x} is even higher.

If we compare the performance of the unsupervised learning rule 1P1D (top panel of Fig. 5) with the performance of the weakly supervised rule R$_{\varnothing}$1P1D (bottom panel of Fig. 5), we can see that as expected the recognition rate is higher when using the weakly supervised rule while having apparently better performance with the same increase in r_{x}.

To complement those results, Fig. 6 shows an example of the conductance (sensitivity) maps of the same postsynaptic neuron for different r_{x} and n_{xbar} values. When r_{x} is too high for the size of the crossbar, we can clearly observe from our results the delimitation in the sensitivity maps from where the writing voltages are not enough to modify the weights. This delimitation is indicated on Fig. 6 by a red arrow on the right side of the sensitivity maps when it is relevant. If we consider the case of a single crossbar, when r_{x} is too high ($r_{\mathrm{x}}\geq 0.3\,\Omega$), we can see that the top of the sensitivity map that corresponds to the farthest devices on the postsynaptic line are still in their random initial value while the bottom has been modified mostly towards lower weights as almost no part of the digit is located there. This is symptomatic of the voltage drop as the farther a device is from the postsysnaptic voltage terminal, the less voltage it will see during a writing phase. Thus, if the voltage drop is too high, the farthest devices will not see a voltage important enough (above the writing thresholds) to be modified. In case of the use of two crossbars, for example, when $r_{\mathrm{x}}= 2\,\Omega$, $n_{\mathrm{xbar}}= 2$, we can see two demarcations (for two crossbars) where only the last third of the synapses close to the postsynaptic of a crossbar connection have been modified (decreased).

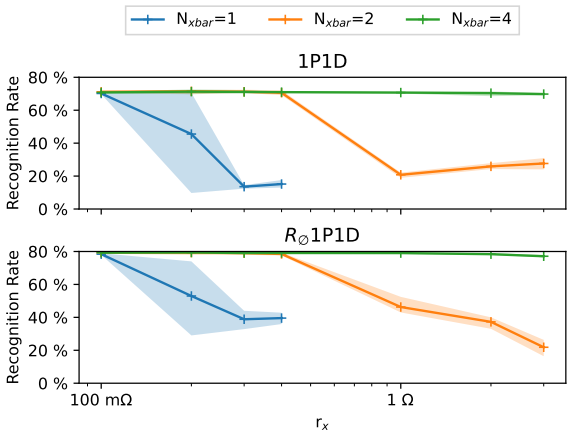

Fig. 5. Performance of the system for different numbers of crossbars and values of the resistance between connected memrisors r_{x}, and the two learning rules of interest obtained after one epoch of training for 5 simulations. Each point (+) represents the average recognition rate. The tinted area around each point represents the minimum and maximum recognition rates at that resistance.

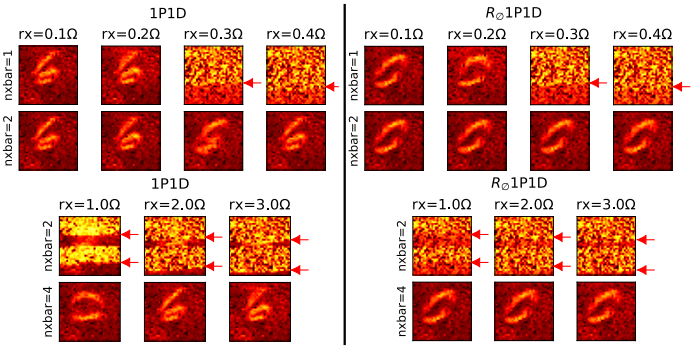

Fig. 6. Example of conductance (sensitivity) maps of the first postsynaptic neuron obtained with different numbers of crossbars and values of r_x for the two learning rules of interest. Red arrows show the observable delimitation where the voltage is not enough to modify the weights.

V. Conclusion

The results presented here showed how detrimental the resistance line of the postsynaptic neurons can be. The results were obtained by assuming negligible voltage drop along the presynaptic line as there is often in practice more inputs than outputs. Thus, the demonstrated detrimental effects can only be more pronounced in an actual implementation, leading to the necessity to use smaller passive crossbars when there are a lot of parallel synaptic connections. However, splitting a single crossbar into smaller ones can also lead to more surface used as more electronic circuits would be needed (especially the CCII+). Using a passive crossbar shows potential in terms of simplification of the design as long as their conception takes into account the limitations on the synaptic lines connections. Lastly, while both unsupervised and weakly supervised learning rules show similar behavior, R$_{\varnothing}$1P1D (the weakly supervised) shows better overall results confirming results from Lewden et al. [12]. R$_{\varnothing}$1P1D also demonstrates slightly less adverse voltage drop effect while using potentially lower energy during training as it skips the weights modification if an output neuron fires on a wrong class.

REFERENCES

[1] G. Indiveri and S. Liu, "Memory and information processing in neuromorphic systems," *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, 2015.

[2] C. S. Thakur, J. L. Molin, G. Cauwenberghs, G. Indiveri, K. Kumar, N. Qiao, J. Schemmel, R. Wang, E. Chicca, J. Olson Hasler, J.-s. Seo, S. Yu, Y. Cao, A. van Schaik, and R. Etienne-Cummings, "Large-scale neuromorphic spiking array processors: A quest to mimic the brain," *Frontiers in Neuroscience*, vol. 12, p. 891, 2018.

[3] D. V. Christensen, R. Dittmann, B. Linares-Barranco, A. Sebastian, M. L. Gallo, A. Redaelli, S. Slesazeck, T. Mikolajick, S. Spiga, S. Menzel, I. Valov, G. Milano, C. Ricciardi, S.-J. Liang, F. Miao, M. Lanza, T. J. Quill, S. T. Keene, A. Salleo, J. Grollier, D. Marković, A. Mizrahi, P. Yao, J. J. Yang, G. Indiveri, J. P. Strachan, S. Datta, E. Vianello, A. Valentian, J. Feldmann, X. Li, W. H. P. Pernice, H. Bhaskaran, S. Furber, E. Neftci, F. Scherr, W. Maass, S. Ramaswamy, J. Tapson, P. Panda, Y. Kim, G. Tanaka, S. Thorpe, C. Bartolozzi, T. A. Cleland, C. Posch, S. Liu, G. Panuccio, M. Mahmud, A. N. Mazumder, M. Hosseini, T. Mohsenin, E. Donati, S. Tolu, R. Galeazzi, M. E. Christensen, S. Holm, D. Ielmini, and N. Pryds, "2022 roadmap on neuromorphic computing and engineering," *Neuromorphic Computing and Engineering*, vol. 2, no. 2, p. 022501, may 2022. [Online]. Available: https://dx.doi.org/10.1088/2634-4386/ac4a83

[4] Editorial, "Big data needs a hardware revolution," *Nature*, vol. 554, no. 7691, p. 145, 2018.

[5] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[6] S. Boyn, S. Girod, V. Garcia, S. Fusil, S. Xavier, C. Deranlot, H. Yamada, C. Carrétéro, E. Jacquet, M. Bibes *et al.*, "High-performance ferroelectric memory based on fully patterned tunnel junctions," *Applied Physics Letters*, vol. 104, no. 5, 2014.

[7] S. Boyn, J. Grollier, G. Lecerf, B. Xu, N. Locatelli, S. Fusil, S. Girod, C. Carrétéro, K. Garcia, S. Xavier *et al.*, "Learning through ferroelectric domain dynamics in solid-state synapses," *Nature communications*, vol. 8, no. 1, p. 14736, 2017.

[8] C. Ma, Z. Luo, W. Huang, L. Zhao, Q. Chen, Y. Lin, X. Liu, Z. Chen, C. Liu, H. Sun *et al.*, "Sub-nanosecond memristor based on ferroelectric tunnel junction," *Nature communications*, vol. 11, no. 1, p. 1439, 2020.

[9] C. Meyer, A. Chanthbouala, S. Boyn, J. Tomas, V. Garcia, M. Bibes, S. Fusil, J. Grollier, and S. Saïghi, "Verilog-a model of ferroelectric memristors dedicated to neuromorphic design," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2018, pp. 557–560.

[10] A. Chanthbouala, V. Garcia, R. O. Cherifi, K. Bouzehouane, S. Fusil, X. Moya, S. Xavier, H. Yamada, C. Deranlot, N. D. Mathur *et al.*, "A ferroelectric memristor," *Nature materials*, vol. 11, no. 10, pp. 860–864, 2012.

[11] P. Lewden, A. F. Vincent, C. Meyer, J. Tomas, S. Siami, and S. Saïghi, "Hardware spiking neural networks: Slow tasks resilient learning with longer term-memory bits," in *2019 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2019, pp. 1–4.

[12] P. Lewden, A. F. Vincent, C. Meyer, J. Tomas, and S. Saïghi, "Toward hardware spiking neural networks with mixed-signal event-based learning rules," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[13] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in neuroscience*, vol. 9, p. 437, 2015.