Analog Spiking Neuron Model for Unsupervised STDP-based learning in Neuromorphic Circuits

Yannaël Bossard, Zalfa Jouni, Siqi Wang, Pietro Maris Ferreira

Analog Spiking Neuron Model for Unsupervised STDP-based learning in Neuromorphic Circuits

Yannaël Bossard, Zalfa Jouni †, Siqi Wang, Pietro M. Ferreira
Université Paris-Saclay, Ecole Normale Supérieure Paris-Saclay, DER N. Tesla, 91190, Gif-sur-Yvette, France
Université Paris-Saclay, CentraleSupélec, CNRS, Lab. de Génie Électrique et Électronique de Paris,91192,Gif-sur-Yvette,France
Sorbonne Université, CNRS, Lab. de Génie Électrique et Électronique de Paris, 75252, Paris, France
Univ. Savoie Mont Blanc, Univ. Grenoble Alpes, Grenoble INP, CNRS, CROMA, Grenoble, France
† zalfa@ieee.org

***Abstract*— Neuromorphic computing has emerged as a promising solution to meet the computational demands of artificial intelligence and enhance the energy efficiency through spiking neural networks (SNNs). This paper models a previously published biomimetic electronic neuron (eNeuron), focusing on the effects of its random noise within an analog SNN framework. The analog SNN is trained via Spike-Timing-Dependent Plasticity (STDP) using unsupervised learning method. Its computational efficiency is evaluated against two benchmarks. Preliminary results have demonstrated that the analog SNN with either simplified or omitted random noise modeling perform better than those with full noise modeling. This highlights the significant impact that random noise of transistors in the eNeuron has on STDP learning processes. However, if random noise is neglected during the training phase, the reintroduction of noise modeling in the testing phase causes an average accuracy drop of only 0.2% for the simplified model of random noise and 4.5% for the full model.**

***Index Terms*— Neuromorphic circuit, Analog Spiking Neuron, Spiking Neural Network, Unsupervised STDP, Random noise**

I Introduction

Artificial intelligence (AI), notably Artificial Neural Networks (ANNs), has demonstrated remarkable success in solving various complex problems. However, their increasing complexity often runs on traditional Von Neumann architectures with limited CPU and memory performance. As a result, a substantial cost in terms of energy consumption is required, which limits their widespread application [1].

Recently, neuromorphic systems have gained significant attention as potential biomimetic solutions. In fact, the human brain still stands out for its exceptional capabilities, with approximately 86 billion neurons [2], of which less than 1% are active simultaneously [3], leading to a low power consumption of a few tens of Watts [4]. Neuromorphic systems consist of neuron-equivalent processing elements interconnected by synapses. By replicating the differential equations governing neurons and synapses behavior through analog circuits, these systems can achieve power consumption reduced by several orders of magnitude, compared to the numerical solution of these equations using digital processors [5].

Although Spiking Neural Networks (SNNs) offer a variety of learning algorithms, the more efficient and well-established gradient-based learning algorithms used in ANNs are not directly applicable to SNNs [6], [7]. In addition, those learning frameworks do not consider physical design limitations from analog SNNs, as noise [8]. In fact, one of the primary challenges in implementing SNNs in analog hardware is the availability of efficient and accurate learning techniques [9].

Spike-Timing-Dependant Plasticity (STDP) is a biological plausible learning rule through the temporal correlation of events [10]. Previous studies have demonstrated that STDP is a robust learning rule for SNNs, enabling on-chip unsupervised learning [11]. The implementations in literature have been proposed using various theoretical neuron models [12]-[14]. However, the analog electronic neurons (eNeurons) exhibit a random noise contributions, due to transistor noise sources, (ie. thermal and shot noise). In [15], tools were propsed to analyze random noise contributions in ring oscillators. It demonstrated that such noise can cause variations in switching timing. Referring to this work, and given that spiking eNeuron circuits operate on similar oscillation principles, random noise affects spike timing in these circuits as proved in previous work [16]. This may influence the accuracy of analog SNN using temporal learning methods such as STDP-based methods.

To achieve an unsupervised STDP-based learning within a analog SNN using eNeurons and conductance-based synapses, physical eNeuron models should be considered. This paper proposes

- Physical-informed neuron model from Morris-Lecar (ML) eNeuron inplementation from the post-layout simulation (PLS) results of its analog circuit from [17], [18].
- A synthesis framework for analog SNN training considering such eNeuron model, through unsupervised STDP.
- Two benchmark problems XOR and MNIST to assert common neural networks figure-of-merit for analog SNN implementation.

Section II presents literature overview of ML eNeurons and STDP learning rule. The synthesis of the analog SNN is detailed in Sec. III. Detailed code for the synthesis framework is available [19]. Section IV presents a detailed analysis of the results obtained from the eNeuron modeling and the two benchmarks. Conclusions are drawn in Sec. V.

Digital Object Identifier 10.29292/jics.v19i3.889

II Background

II.A Analog Spiking eNeurons

The Morris-Lecar model is a biomimetic neuron model that achieves a balance between complexity and biological accuracy [20]. It describes the conductance-based dynamics of neuron behavior, with membrane potential changes driven by calcium during depolarization and potassium during repolarization, as follows:

$$C_m\frac{dV_m}{dt} = I - g_{\mathrm{Ca}}M_{\mathrm{ss}}(V_m)(V_m - V_{\mathrm{Ca}}) - g_{\mathrm{L}}(V_m - V_{\mathrm{L}}) - g_{\mathrm{K}}N(V_m - V_{\mathrm{K}}) \quad (1)$$

$$\frac{dN}{dt} = \frac{N_{\mathrm{ss}}(V_m) - N}{\tau_N(V_m)} \quad (2)$$

Figure 1 illustrates the ML eNeuron circuit and its corresponding layout, redesigned from [17], and studied in this paper. It is chosen for its biomimetic properties and broad dynamic range, which enhance learning accuracy [8]. Briefly, upon receiving a synaptic current I_{syn} from presynapses, the membrane capacitance C_m charges via MP_{Na} (pull-up) and discharges through MN_K (pull-down), causing rapid fluctuations in membrane potential V_m and spike generation.

In this eNeuron circuit, the transistors are biased in the deep-subthreshold region to optimize power usage and achieve the required non-linear functions [21]. As a low current I_{syn} excites an eNeuron, it flows through these transistors, causing carrier diffusion to dominate the channel current. This results in a significant variation of current that flows across V_m. This variation, amplified in the subthreshold regime, allows shot noise to become a dominant factor due to stochastic electron and hole recombination at V_m [22]. The spectral density of the shot noise current in the subthreshold region is given by:

$$S_I = 2 \cdot q \cdot I_D \cdot |Z_{\mathrm{out}}|^2 \quad (3)$$

where q is the electronic charge; I_D is the drain current; and Z_{out} is the output impedance (which is here the parallel combination of C_m and the total output resistances of transistors).

Previous work [16] demonstrates that this noise strongly affects the spike timing of the low-power analog eNeurons with high spike frequencies, such as the ML eNeuron. Post-layout simulation results reveal that the spike timing variability, measured as the ratio of standard deviation to mean (σ/μ), exceeds 76%.

II.B Spike-Time-Dependant Plasticity

STDP provides a biological plausible mechanism of learning that adjusts synaptic strength based on the relative timing of pre-synaptic and post-synaptic spikes [23]. If a presynaptic neuron fires before a postsynaptic neuron within a short time window, the synaptic weight is strengthened. If the postsynaptic neuron fires before, the synapse is weakened. This temporal relationship is defined by the STDP function or learning window, which varies across excitatory and inhibitory synapses connectivity [10]. A common STDP window function, defined by A^+ and A^- (maximum synaptic changes) and τ^+ and τ^- (time constants), is shown in (4).

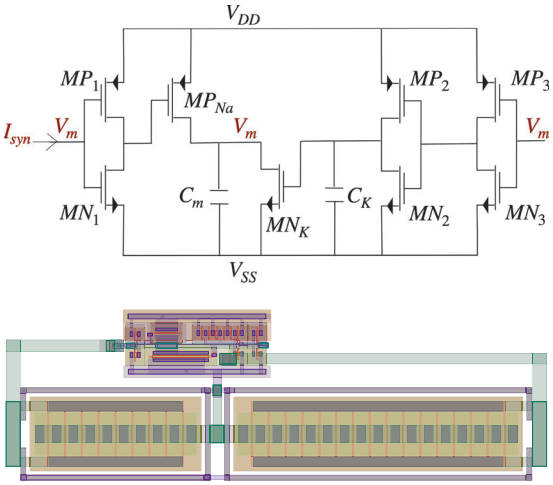

Fig. 1: The Morris-Lecar eNeuron at (a) circuit and (b) layout levels, designed in BiCMOS 55 nm technology (V_{DD} = 100 mV, V_{SS} = -100 mV), with a layout size of 5.7 × 17.3 μm^2.

The change in synaptic weight Δw is computed by aggregating the values of the STDP function applied to the differences between all corresponding presynaptic t_{pre} and postsynaptic spike times t_{post}.

$$W(\Delta t) = \begin{cases} A^+ \exp\left(-\frac{\Delta t}{\tau^+}\right) & \text{if } \Delta t > 0 \\ -A^- \exp\left(\frac{\Delta t}{\tau^-}\right) & \text{if } \Delta t < 0 \end{cases}, \Delta t = t_{\mathrm{post}} - t_{\mathrm{pre}} \quad (4)$$

Its straightforward implementation via spike traces makes it a favorable option for analog SNNs hardware. In addition, STDP is an unsupervised local learning rule. On unsupervised learning, the algorithm receives inputs but, neither supervised target outputs nor rewards. Thus labelled data are not required. It is traditionally used for finding patterns into data which limits the applications [24], [25]. STDP-based learning often requires additional techniques, such as winner-take-all strategies to impose competition or weight normalization to prevent intrinsic instability [26]. It can also be used in supervised learning or reinforcement learning scenarios with several different methods [27].

II.C SNN using Morris-Lecar Neurons

Su et al. have exploited (1) and (2) to implement a Morris-Lecar neuron model within SNNs and proceed machine learning in [28]. A STDP-based unsupervised learning of memristive SNN using ML neuron model is proposed in [29]. The ML neuron model is based on simplified version of (1) and (2). A ML eNeuron inspired from a 65 nm CMOS technology is proposed in [30]. It is employed for extraction of electrocardiography features with low power consumption. However, none of these implementations have demonstrated the SNN robustness over circuit constraints such as transistor noise.

III Analog SNN model and learning framework

To synthesize the analog SNN effectively, the initial step involves modeling the ML eNeuron using data from the

PLS results of the eNeuron circuit (detailed in Sec. III.A). Next, a detailed framework for analog SNN synthesis is presented, applying unsupervised STDP learning (detailed in Sec. III.B). This framework is finally showcased to demonstrate the capabilities of the analog SNN through solving the XOR and MNIST (detailed in Sec. III.C)

III.A eNeuron and Synapses

The ML model presented in (1) and (2) is complex and do not take in consideration the physical constraints of analog circuits. To accurately model the ML eNeuron behavior from analog circuit design shown in Fig. 1, it is essential to incorporate three key properties from PLS results

1. The dynamics of potential membrane behavior over time, including aspects such as resting potential and spike timing.
2. The firing rate's response to different input excitation levels through the eNeuron's activation function.
3. The random noise from transistors, which is crucial in STDP learning, where noise affects spike timing and synaptic adjustments.

This model stands out for its flexibility, allowing for easy adaptation to various eNeuron types while maintaining low computational demands and high precision in fitting eNeuron properties.

1. Dynamics of the spike behavior: The dyanmics of the spike is presented as

$$\frac{dV_m}{dt} = \frac{(V_{rest} - V_m)}{\tau_{leak}} + I_{syn} \cdot \frac{R_m}{\tau_m}, \tag{5}$$

which describes the membrane potential V_m of the ML eNeuron, influenced by the leakage current and the synaptic input current (I_{syn}). The first term ($(V_{rest} - V_m)/\tau_{leak}$) represents the leaky behavior, driving V_m back towards the resting potential V_{rest} over time, ensuring stability and preventing runaway excitation. The time constant τ_{leak} indicates the system dynamics during C_m discharges. PLS results reveal that speed values vary with different input currents. A constant value of τ_{leak}, equal to the mean over the entire range of current, is selected.

The second term ($I_{syn} \cdot R_m/\tau_m$) represents the effect of the external current on the eNeuron's membrane, driving V_m away from V_{rest}, potentially towards the threshold of firing an action potential. The membrane time constant τ_m (typically $R_m \cdot C_m$) represents how quick the membrane responds to input currents. Here, the eNeuron model in (5) closely resembles the equation of the LIF model in [31].

To accurately model the spike shape of the ML eNeuron, precise control over the dynamics of V_m is necessary.This can be achieved through a variable membrane resistance R_m, which changes in a non-linear piecewise manner based on the membrane potential as described by

$$R_m = \sum_{i=1}^{n} R_i \cdot [V_m \leq V_i] \cdot (1 - [V_m \leq V_{i-1}]). \tag{6}$$

[P] is the Iverson bracket, returning 1 if P is true and 0 if P is false. This approach segments R_m into different segments R_i depending on V_m relative to a series of voltage thresholds V_i. This extention into the model provided by (6) enhances the fidelity of the model over the simple Leaky Integrate-and-Fire (LIF) model by reflecting the ion channels dynamics occurring as the ML eNeuron depolarizes or hyperpolarizes.

The proposed model is a physical-informed version of the typical ML model (1)-(2), while maintaining similar behavior and avoiding the need for numerous constants. Finer segmentation of R_m allows a more precise adjustments to small changes in V_m, better reflecting the nonlinear properties of the ML eNeuron. However, increased segmentation enhances precision and realism of the model with complexity increased.

2. Firing rate response property: To consider the activation function property of the eNeuron model, its firing rate over input synaptic current I_{syn} in (5) is replaced by

$$I'_{syn} = \delta(I_{syn}) \cdot f_{\text{FT\_eNeuron}}(I_{syn}), \tag{7}$$

where $f_{\text{FT\_eNeuron}}(I_{syn})$ is the interpolation function derived from the PLS results of the nonlinear ML eNeuron activation function. The coefficient $\delta(I_{syn})$ adjusts the interpolated function affected by the leakage term. Additionally, the current I'_{syn} is limited to 15 nA to prevent transistors operating out of deep-subthreshold region.

3. Noise model in spike timing property: To improve the model's precision in analog circuits, incorporating noise is crucial. The eNeuron typically encounters two types of noise: external noise from input signals and intrinsic noise from transistors. The eNeuron already mitigates external noise by averaging input currents. Therefore, this study focuses on addressing the random noise caused by transistors. This random noise in the eNeuron refers to the shot noise, as detailed in Sec. II.A.

Figure 2 compares the spike timing of an ideal eNeuron, which ignores the random noise, with a real eNeuron that considers it. It highlights t_{rise}, the moment where the membrane potential surpasses the threshold, and T, the interval between consecutive spikes.

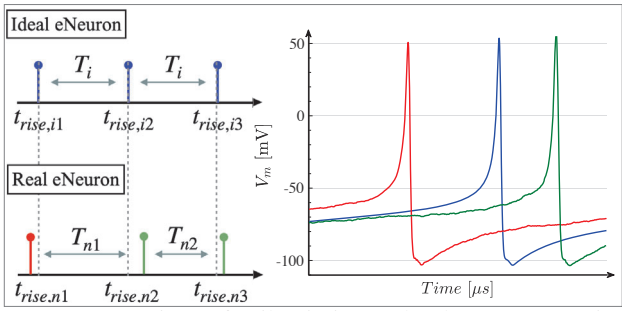

Fig. 2: Comparison of spike timing and voltage response in ideal and real eNeurons. Left panel: spike intervals and rise times. Right panel: Post-layout simulations results of the V_m with a noise-free transient simulation (in blue) and two trans-noise simulations (in red and green).

In the ideal model, spikes occur at regular intervals, with each k^{th} spike firing at $t_{rise,ik} = t_{rise,i1} + k \cdot T_i$. However, in a real eNeuron, spike timing varies due to shot noise, causing deviations from expected rise timings (e.g., $t_{rise,n1} < t_{rise,i1}$) and irregular periods (T_{n1}, T_{n2}, etc., not equal to the ideal

period T_i). The graph in Fig. 2 illustrates these variations through PLS results, showing spikes that occur either earlier or later compared to the ideal regular timings (in blue).

To accurately model the impact of noise on spike timing in the eNeuron, timing deviations caused by noise can be defined in two types: noise-driven period deviation and noise-driven rise deviation. Noise-driven period deviation ($\Delta T_n = T_n - T_i$) refers to variations in the intervals between consecutive spikes. Noise-driven rise deviation ($\Delta t_{rise,n} = t_{rise,n} - t_{rise,i}$) pertains to deviations in the time it takes for a spike to reach its threshold. This study concentrates on noise-driven rise deviation, as it adopts the STDP learning that relies on precise spike timing. The focus on noise-driven rise deviation provides a more precise account of deviations in the timing of spike occurrences, rather than the intervals between spikes.

At each spike index, the noise follows a Gaussian distribution characterized by a mean and standard deviation that accumulate and scale with each spike occurrence, as presented later in Sec. IV. Therefore, this noise follows a Gaussian random walk distribution over spike occurrences. The mean of the noise-driven rise deviation for spike k increases linearly, described by

$$\mu(\Delta T_{rise,n}) = k \times \mu_1(\Delta T_{rise,n}), \tag{8}$$

where k is the spike index, and $\mu_1(\Delta T_{rise,n})$ represents the mean at the first spike. This pattern results from the linear accumulation of noise due to membrane capacitance dynamics, allowing for a straightforward adjustment to achieve a zero-mean distribution.

The standard deviation of the noise-driven rise deviation shows a nonlinear increase with spike index, modeled as

$$\sigma(\Delta T_{rise,n}) = \sqrt{k} \times \sigma_1(\Delta T_{rise,n}), \tag{9}$$

where $\sigma_1(\Delta T_{rise,n})$ is the initial standard deviation. This nonlinear trend likely stems from the accumulation of noise in the feedback loops of the eNeuron circuit over time. Thus, the noise can be described as following a Gaussian random walk distribution with mean zero and standard deviation $\sqrt{k} \cdot \sigma_1(\Delta T_{rise,n})$, where $\sigma_1(\Delta T_{rise,n})$ is calculated from the noise analysis in the eNeuron circuit, detailed in [16]

$$\sigma_1^2(\Delta T_{rise,n}) = \frac{q}{I_{syn}} \left(\frac{1}{f_{spike}} + \frac{r_{ds} \cdot C_m}{\pi} \right), \tag{10}$$

where r_{ds} is the output resistance of the transistors, and f_{spike} is the spiking frequency over a certain period of time. Therefore, the noise model of the eNeuron, as defined by (9) and (10), depends on circuit variables C_m, f_{spike}, r_{ds}, and the external input current I_{syn}. The model scales with the index of spike occurrence. This noise model is added to the eNeuron model through the threshold level.

4. Synapses: Finally, a conductance-based synapse model is employed to link neurons to each other. When a spike occurs, the synaptic conductance is increased by an amount corresponding to the synaptic weight, conductance which exponentially decays towards zero (11). g_l and τ_l are respectively conductance and time constant for either excitatory or inhibitory synapses.

$$\frac{dg_l}{dt} = -\frac{g_l}{\tau_l}, \quad l \in (e,i) \tag{11}$$

Synapses are classified into two types: excitatory and inhibitory. An instantaneous change in conductance leads to the generation of a post-synaptic current. Depending on the manipulation of the equilibrium potential V_{eq} in (12), a synapse is considered excitatory if $V_{eq} = V_{thresh}$ leading to a positive increase in the current that decays exponentially towards zero. The synapse is considered inhibitory if $V_{eq} = V_{reset}$ leading to a negative increase in the current, tending exponentially towards zero. A setting value α_l, modulating the postsynaptic current, is used to differentiate the impact of excitatory and inhibitory synapses.

$$I_{syn,l} = \alpha_l \cdot g_l \cdot (V_{eq} - V_m), \quad l \in (e,i) \tag{12}$$

Algorithm 1 Analog SNN Synthesis Framework for unsupervised STDP learning

```
1: 1. PLS activation function of eNeuron
2: activation_function =
   load_PLSresults(ML_eNeuron_Ferreira2021)
3: 2. Definition of eNeuron and synapses
4: MLeNeuron = [activation_function,
5:   Rm:  segmentation(nb=3),
6:   dynamic_range_restriction:  15nA,
7:   noise:  scenario 1 or 2 or 3 ];
8: synapses = [conductance_based,
9:   type:  exci or inhi];
10: 3. Network Structure for the MNIST problem
11: SNN = [ #Network can be adjusted for any problem
12:   Input_layer(nb=784,type=MLeNeurons)
13:   Output_layer(nb=1225,type=MLeNeurons)
14:   fully_connect(Input_layer,Output_layer,
    synapses = exci & inhi)
15:   Lateral_connect(Output_layer,Output_layer,
    synapses = inhi)];
16: 4. Encoding input data    #from pixel to current value
17: train,test_data = load_MNIST()*3nA/255
18: 5. Network Model for learning
19: training = (SNN,train_data,time=120μs,
    learning=STDP,weight_normalization)
20: labeling = (SNN,train_data)
21: testing = (SNN,test_data)
22: 6. Network Model Training on MNIST dataset
23: for epoch = 0 to max_epoch do
24:    accuracy = learning(SNN, epoch)
25:    if accuracy ≥ accuracy_min then
26:       break;
27:    end if
28: end for
29: final_accuracy,weights = testing()
30: if noise == 3 then
31:    accuracy_noise =
    testing(noise=1,2)
32: end if
```

III.B Analog SNN Synthesis with unsupervised STDP Learning

The synthesis framework Algo. 1 outlines the entire process to build and train modeled analog SNN. First, it loads

`activation_function` from PSL results of the ML eNeuron. This function is employed to define `MLeNeuron`, its `random noise`, and `synapses` models. Once the eNeuron and synapse models are integrated into the Brian2 simulator [32], the architecture of the analog SNN can be designed with varying numbers of layers, eNeurons, and excitatory/inhibitory synapses, configured to address the specific complexity and requirements of the given problem. It corresponds to the class `SNN`, built in Step 3 of Algo. 1.

Figure 3 illustrates the analog SNN architecture designed to address either the XOR or MNIST problem presented in Sec. III.C.

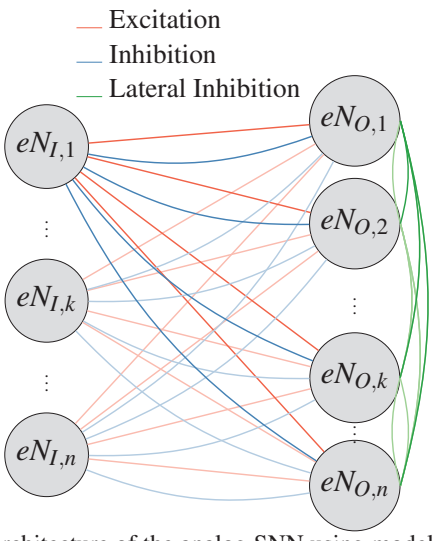

Fig. 3: Architecture of the analog SNN using modeled eNeuron.

The architecture selection was optimized through multiple simulations on the XOR problem (e.g with hidden layers or not, etc.). It consists of an input layer and an output layer, fully connected with both excitatory and inhibitory synapses. The first layer of eNeurons receives input data and feeds it into the analog SNN using a rate-code approach. Jouni et al. have demonstrated the impact of a random noise, in the used eNeuron, on the precise-timing of a spike (temporal-code) which leads up to 76% of variation where the impact are at 6% on the firing rate (rate-code) [16]. Moreover, the analog SNN, which employs only one type of eNeuron, facilitates the conversion to rate-code. In this method, the intensity of the input is translated into excitation currents, causing each input eNeuron to spike at a rate that corresponds to the encoded information. This corresponds to Step 4 of Algo. 1. These specific spike rates are calculated based on (7) from the eNeuron model, which maps the firing rate response to the input excitation. Additionally, a lateral inhibition is implemented in the final layer through inhibitory synapses, connecting each eNeuron to all other eNeurons in the same layer. It plays a crucial 'soft-winner-take-all' role for fostering competitive learning that effectively differentiates the inputs, as explained in [33].

The unsupervised learning strategy adopted is influenced by Diehl and Cook's 2015 framework [34], which uses unsupervised STDP for training SNNs. However, their implementation focuses on a simple LIF model and does not account for the circuit properties or the effects of random noise of transistors, which are crucial for a more comprehensive neuromorphic system. The learning process is structured into three distinct phases: `training`, `labeling`, and `testing` initialized in Step 5 of Algo. 1.

During the training process, the network uses STDP, depicted in Sec. II.B, to adjust synaptic weights based on the timing of spikes between eNeurons. In this phase, the network self-organizes without supervision or reinforcement. The weights are initialised randomly. Only the weights between the input layer and the output layer are adjusted by STDP, while those in the lateral inhibition remain constants. A normalization of weights conducted by STDP according to the divisive enforcement rule, as described in [35], is performed. The normalization factor employed is made twice as small for inhibitory synapses as for excitatory synapses to ensure an active SNN. To address the lack of weight boundary function in STDP function, a clipping function is used. Each input data is processed during 120 μs including 20 μs of resting time allowing the variables to return to their resting values, preventing the influence of neighboring data simulations.

Following training, a labeling process assigns meaningful labels to the eNeurons in the output layer. This step is crucial for evaluating the network's performance in tasks requiring classification. During labeling, the activity of each eNeuron in the output layer is monitored as the network is exposed to a portion of the last inputs used during training. Each eNeuron is then labeled, based on the input type that most frequently causes it to fire. Finally, the network's performance is evaluated during the testing phase. In this phase, new inputs are presented to the network, and the response of the output neurons is observed. The response of the neurons is checked against the labels assigned during the labeling phase. These three functions are repeatedly executed for a specified number of iterations, as detailed in Step 6 of Algo. 1.

In the framework, the dataset includes both features and labels, even though the training algorithm is unsupervised. This is essential for the testing phase, which requires correctly labeled data to compare against the network's output and determine accuracy. Additionally, the framework incorporates a boolean setting named *noise* to enable or disable the random noise of transistors within the eNeuron model. This feature allows for performance comparisons under various scenarios depicted in Sec. III.C and helps quantify the impact of intrinsic random noise on STDP training.

III.C Analog SNN assesses through two benchmarks

To evaluate the effectiveness of the analog SNN using unsupervised STDP, two benchmarks are undertaken. The first benchmark is the exclusive OR (XOR) logical operator problem. The XOR problem is historically a fundamental issue in the field of neural networks and symbolizes a simple yet significant challenge for any computational model [36]. It involves the classification of inputs into two classes that are not linearly separable. Thus, it requires the neural network to effectively learn and represent non-linear boundaries between classes. Multiple methodologies to solve the problem

with SNNs trained by tuned-STDP have already bean proposed [37]-[39].

The second benchmark is the MNIST problem, a standard benchmark in the field of machine learning for recognizing and classifying handwritten digits [40], [41]. It involves the MNIST dataset that comprises 70,000 images of digits ranging from 0 to 9, with each image having a resolution of 28x28 pixels.

The number of eNeurons in both layers varies with the complexity of the problem. The XOR problem employs 2 input neurons and 13 output neurons are used, making a total of 208 synapses. The MNIST problem employs 784 input neurons and 1225 output neurons, making a total of 3420200 synapses. In the framework, input encoding uses a rate-code approach. For the XOR problem, the two logical values 0 and 1 respectively correspond to a low power input constant current of 0.3 nA and a higher input current of 2 nA. It respectively triggers spike rate of 215 kHz and 540 kHz in the input eNeuron layer. For the MNIST problem, a grayscale image is presented to the network. The intensity of each pixel is translated into a current that varies from 0 to 3 nA, driving eNeuron spike rates between 0 and 610 kHz.

This analysis compares three scenarios, each accounting for different the random noise models provided in Sec. III.A. Scenario 1 incorporates the eNeuron model with a constant Standard Deviation (SD) for its random noise. This scenario does not include the Gaussian walk distribution of the noise over spike occurrences, which leads to a non-linear accumulation of its standard deviation over time. Here, the standard deviation considered is the average calculated across the entire range of input current. Scenario 2 involves simulations with the complete eNeuron model, where the full noise model is considered. This includes the Gaussian walk distribution of the noise and its variable standard deviation over spike occurrences. In scenario 3, simulations are conducted using the eNeuron model without considering its random noise. Thus, it allows to assess if the training framework is effective when eNeurons do no present random noise. In addition, the reintroduction of noise model from scenarios 1 and 2 in the testing phase enables to assess the impact of the random noise on post-trained analog SNN, respectively named scenario 3.1 and 3.2.

IV Results and Discussion

IV.A eNeuron Model

This section presents the validation results of the ML eNeuron model. Figure 4 illustrates the potential membrane fitting for the ML eNeuron, using different levels of resistance R_m segmentation: one (orange plot), two (yellow plot), and three (green plot), compared with the actual post-layout potential membrane over time (blue plot). This figure demonstrates that increasing the segmentation fineness enhances the model's accuracy, reducing the error from 40.3% to 4.6%. For ongoing work on solving XOR and MNIST problems, the model with three segmentations was selected due to its acceptable error margin in approximating the post-layout potential membrane. Further refinement in segmentation could reduce the error margin to negligible levels, but at the cost of an increased computational time when implementing this model in learning analog SNN.

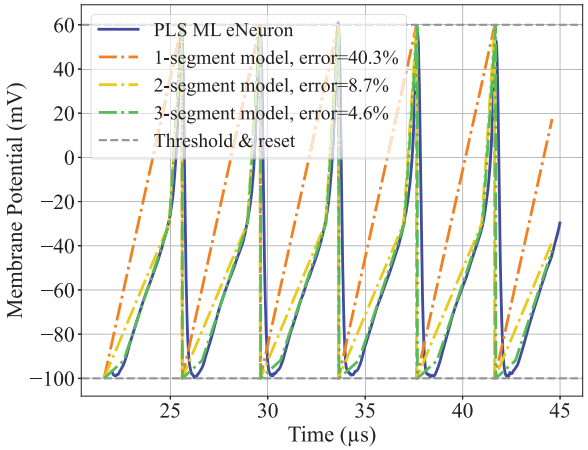

Fig. 4: Membrane potential of the ML eNeuron from PLS results (in blue)) and membrane potential of the model for different segmentation of the resistance R_m. The error corresponds to the relative error between the area under the curve of the rise of the membrane potential of the ML eNeuron and the area under the various presented models.

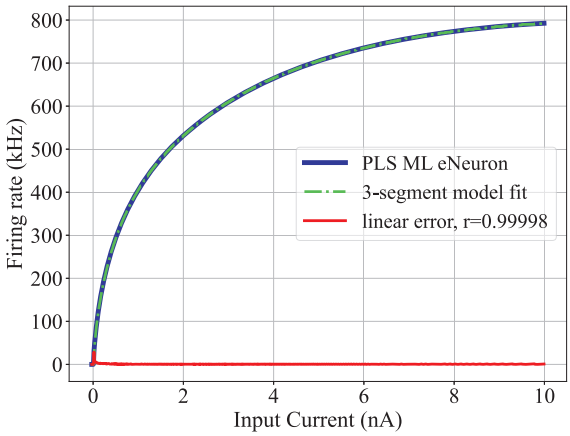

Fig. 5: Firing rate activation function of the ML eNeuron from PLS results (in blue) and the activation function of the model (in green). The red curve corresponds to the subtraction of the blue and green curves.

Figure 5 shows the activation function of the ML eNeuron from post-layout simulations (in blue) and from the previously depicted model (in green). The correction coefficient $\delta(I_{syn})$ from (7) enables the correlation coefficient r to tend to 1, resulting in an accurate modeling of the ML eNeuron PLS activation function. Figure 6 illustrates the post-layout distribution of noise-driven rise deviation in spike timings of ML eNeuron. The multiple plots in the figure represent the distributions of noise-driven rise deviation at various spike occurrences over time. As shown in this figure, the noise-driven rise deviation exhibits a Gaussian random walk distribution over all spike occurrences, having different means and standard deviations that exhibit a cumulative effect scaling with the spike index. The average noise-driven rise deviation, defined in (8), demonstrates a linear increase corresponding to spike occurrences. Figure 7(a) confirms

this trend, displaying the measured post-layout mean of the noise-driven rise deviation in blue and the estimated mean in red.

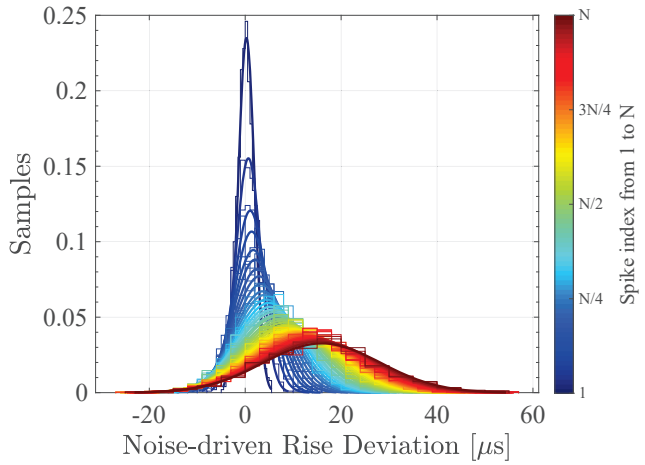

Fig. 6: Distribution of noise-driven rise deviation in ML eNeuron, illustrating noise characteristics across spike occurrences.

Figure 7(b) illustrates the standard deviation of noise-driven rise deviation through three different considerations. The blue line represents the post-layout standard deviation, the red dashed line the estimated standard deviation, and the yellow line the analytical standard deviation. As shown in this figure, the post-layout standard deviation, shown in blue, demonstrates a non-linear accumulation with each spike occurrence. The estimated equation for this deviation, defined in (9), closely matches the post-layout results, as depicted by the red dashed line. The standard deviation of the noise accumulates incrementally, beginning with the initial spike's standard deviation from (10). The analytical expression for the standard deviation, shown in yellow, incorporates this initial value and matches the observed data well. These findings affirm the accuracy of the noise model for the eNeuron, supporting its integration into analog SNN synthesis with unsupervised STDP learning.

IV.B The XOR benchmark

The analog SNN has been trained and tested using the algorithm outlined in Sec. III.B. Figure 8 illustrates the training and testing accuracy faced to the XOR problem through different scenarios of noise consideration depicted in Sec. III.C. The analog SNN is trained over 100 epochs, averaged through multiple simulations, where one epoch consists of 10x a randomly arranged list of the four binary inputs ([0,0], [0,1], [1,0], [1,1]). During labeling phase, the network is exposed to 30% of the last inputs used during training. One may notices that when the training is proceeded with eNeurons presenting random noise, corresponding to scenarios 1 and 2, the network usually fails to converge and do not go up to the lucky threshold within 100 epochs. However, the network presents a high accuracy when using noiseless eNeurons, corresponding to scenario 3. By replacing noiseless

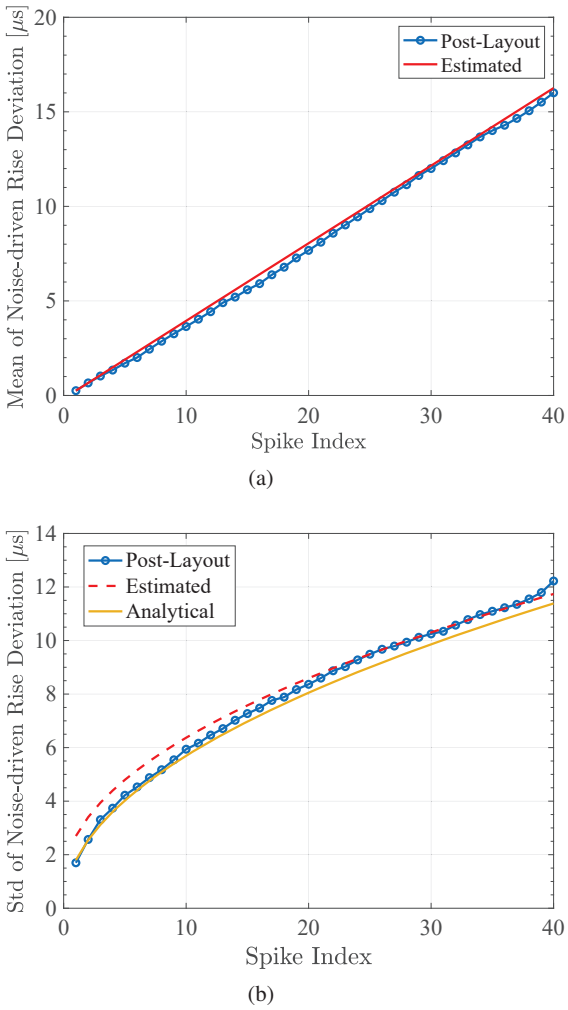

Fig. 7: (a) Mean noise-driven rise deviation of ML eNeuron plotted against spike index, measured in blue and estimated in red, (b) Standard deviation of noise-driven rise deviation with measured values in blue, estimated in red dashed, and analytical in yellow.

eNeurons with noisy eNeurons during only the testing period, the drop of accuracy is higher with the complete model of random noise (corresponding to scenario 3.2, pink boxplot) than with the simplified random noise (corresponding to scenario 3.1, yellow boxplot). These findings indicate that random noise characterized by a dynamic standard deviation exerts a greater disruptive effect on the accuracy of post-STDP-learning predictions compared to random noise with a static standard deviation.

Here, a positive aspect of unsupervised training is that clustering is performed. Thus, analog SNN is trained until at least a clear separation of ([0,1] and [1,0]) from ([0,0] and [1,1]) is achieved. By extending the training period even after the XOR problem has been resolved, the SNN is enabled to cluster all four distinct inputs. Consequently, it becomes capable of solving any two-input logic gate without additional training, requiring only a re-labeling of the output neurons. For example, a post-training neuron that responds

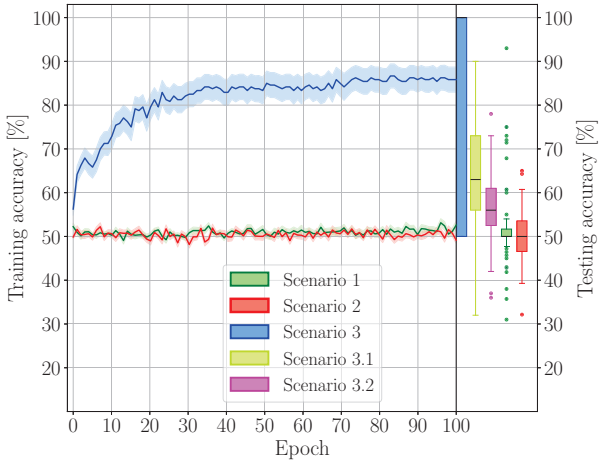

Fig. 8: Training and testing accuracy over 100 epochs, for XOR problem and for different scenarios of noise incorporation in the eNeuron model.

more strongly to the input [0,0] will be labeled '0' for the XOR problem and '1' for the NAND problem.

Figure 9 provides a 3D visualization of the XOR problem as resolved by a trained analog SNN. Each of the plots, labeled (a) through (e), demonstrates how the network interprets varying combinations of binary inputs, which are finely segmented into 10 slices ranging from 0 to 1. The X-input and Y-input axes correspond to the binary inputs [x, y] that feed the two input neurons of the network. The Z-input represents the averaged predictions of the network across numerous simulations. The color gradient on this axis, from blue to yellow, visually encodes the prediction values, where blue indicates a prediction closer to 0 and yellow closer to 1.

Figure 9(c) illustrates the network behavior under Scenario 3, where the network has successfully learned the XOR problem without noise, showing a clear separation between the classes. Figures 9(d) and 9(e) correspond to Scenarios 3.1 and 3.2, respectively. These plots show the effects of introducing simplified and more complex random noise models during the testing phase, affecting the clarity and definition of the predictive surface. Figures 9(a) and 9(b) reflect Scenarios 1 and 2, where the network was exposed to noise during the training phase, leading to less effective learning outcomes as evidenced by the less defined predictive surfaces, shown by a prominent valley.

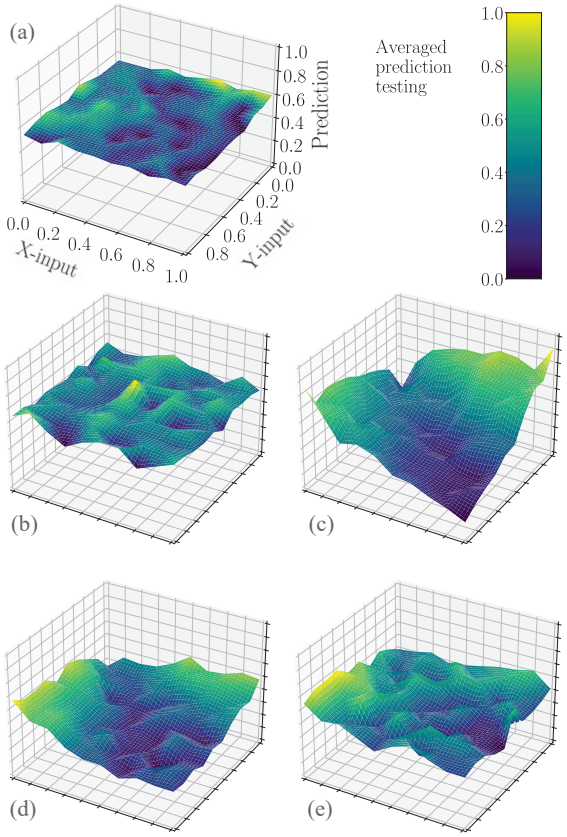

Fig. 9: 3D visualization of XOR problem resolution by a trained A-SNN, with inputs [x, y] mapped on the X and Y axes and predictions color-coded from blue (0) to yellow (1) on the Z-axis, for (a) Scenario 1, (b) Scenario 2, (c) Scenario 3, (d) Scenario 3.1, and (e) Scenario 3.2.

IV.C The MNIST benchmark

Regarding to the MNIST problem, the analog SNN has been trained over 10000 input images. During labeling phase, the network is exposed to 15% of the last inputs used during training. Figure 10 illustrates the training and testing accuracy of analog SNN across different scenarios depicted in Sec. III.C. As shown, Scenario 3 demonstrates the most rapid learning, achieving a peak accuracy of 50.6%. However, this scenario learning plateaus after processing a few thousand samples, indicating a limitation in further improving its accuracy with additional training data. Scenario 1, using a simplified noise model, shows a slightly slower but steady increase in learning, achieving an average accuracy

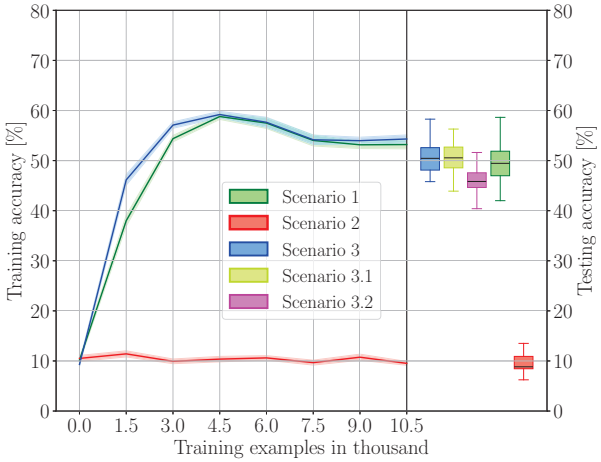

Fig. 10: Training and testing accuracy of the analog SNN for MNIST problem for different scenarios of noise incorporation in the eNeuron model.

nearly equivalent to Scenario 3 at 49.9%. In contrast, Scenario 2, which applies a complete noise model, demonstrates a poorer performance. This suggests that the full noise model may be disruptive, avoiding the network effective training. The boxplots on the right provide further insights into test-

ing accuracy variations. Notably, there is a more significant accuracy drop in Scenario 3.2, with an average decrease of 4.5% (indicated by the pink boxplot), compared to a minor 0.2% drop in Scenario 3.1 (shown by the yellow boxplot). This highlights the impact of different noise considerations on the network robustness in testing phases.

Figure 11 shows learning results for each scenario for a network, as a tendency. Figure 11(a) illustrates the labeling distribution of the 1225 neurons. Scenario 2 (in red) shows 1225 neurons all with the same label, which explains why its accuracy is close to 10%, the neurons keep repeating the same prediction. Scenarios 1 and 3 each have multiple neurons labeled by digits, though distributed unevenly. This means that the network has multiple neurons presenting a maximum firing rate for each digit. Figure 11(b) represents the receptive field of three labeled neurons per digit per scenario. The excitatory synaptic weights are restructured (from 784 to 28x28), at post-training phase. Consistent with Fig. 11(a), the analog SNN in scenarios 1 and 3 has neurons labeled for each digit, whereas the analog SNN in scenario 2 has only neurons labeled as 1. As the neurons labeled for the other digits do no exist, they are set as equal to 0 (dark blue). The receptive fields in scenarios 1 and 3 tend to be shaped as the corresponding labeled digit. However, one may notice that some receptive fields are incorrect, such as the presence of shapes resembling a 9 in neurons labeled as 4. This explains why the analog SNNs achieve accuracy well above the chance level (10%), yet remain far from perfect.

IV.D Discussion

Previous results seemed to indicate that random noise in spike timing significantly disrupted unsupervised training by STDP-based learning. Analog SNNs with random noise, as described by (10), failed to learn when faced with both benchmarks. However, results on the MNIST benchmark showed that when the random noise had a constant standard deviation, the learning was slightly slower but almost as efficient as when the eNeurons were noiseless. This suggests that it is not just any intrinsic random noise, but specifically noise with a dynamic standard deviation dependent on internal eNeuron variables, that impacts and inhibits learning through STDP. Further investigations could delve deeper into the relationship between the reduction in the learning rate and the standard deviation of noise to assess an acceptable noise magnitude. Additionally, studying the impact of each term in the dynamic standard deviation (9) on STDP-based learning could help redesign an eNeuron model with intrinsic noise that is less inhibitory to learning.

The drop in accuracy caused by random noise in both scenario 3.1 and 3.2 was below 5%. This result encourages the development of an efficient learning method that enables high accuracy with minimal impact, affected by only a few percentage points.

Finally, the simulation of this type of SNNs was very time-consuming and required substantial computational resources. Indeed, Brian2 is well-suited for modeling biological neurons but not for machine learning with precise models of electronic neurons operating at high frequencies. The use of interpolation functions significantly increases computation time. Libraries based on Torch, like BindsNet, including a differential equation solver, could enable faster simulations for further investigations [42].

V Conclusion

Recent advancements in neuromorphic hardware had highlighted SNNs as a viable energy-efficient solution for AI applications. This paper proposed unsupervised STDP-based learning within an analog SNN using a complete model of ML eNeuron and conductance-based synapses. The analog SNN was tested on XOR and MNIST benchmarks to assess the impact of noise on the temporal learning STDP method employed. Preliminary results demonstrated that the analog SNN did not learn as effectively as analog SNNs with simplified or even neglected random noise modeling, indicating that the eNeuron random noise significantly impacted STDP learning. However, when random noise was neglected during the training phase, the reintroduction of noise modeling in the testing phase caused an average accuracy drop of only 0.2% for the simplified model of random noise and 4.5% for the full model.

References

[1] C. Schuman, et al., "Opportunities for neuromorphic computing algorithms and applications", *Nature Computational Science*, vol. 2, n. 1, 2022. doi: 10.1038/s43588-021-00184-y

[2] F. Azevedo et al., "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain", *Journal of Comparative Neurology*, vol. 513, n. 5, pp. 532-541, 2009. doi: 10.1002/cne.21974

[3] P. Lennie, "The cost of cortical computation", *Current biology*, vol. 13, pp. 493-497,2003. doi: 10.1016/S0960-9822(03)00135-0

[4] F. Javed et al., "Brain and high metabolic rate organ mass: contributions to resting energy expenditure beyond fat-free mass", *The American journal of clinical nutrition*, vol. 91, no. 4, pp. 907-912, 2010. doi: 10.3945/ajcn.2009.28512

[5] A. Shrestha, et al., "A survey on neuromorphic computing: Models and hardware", *IEEE Circuits and Systems Magazine*, vol. 22, n. 2, 2022. doi: 10.1109/MCAS.2022.3166331

[6] A. Tavanaei and A. Maida, "BP-STDP: Approximating backpropagation using spike timing dependent plasticity. Neurocomputing", *Neurocomputing*, vol. 330, pp. 39-47, 2019. doi: 10.1016/j.neucom.2018.11.014

[7] K. Yamazaki, et al., "Spiking neural networks and their applications: A review" *Brain Sciences*, vol. 12, no. 7, pp. 863, 2022. doi: 10.3390/brainsci12070863

[8] T. Soupizet, et al., "Analog Spiking Neural Network Synthesis for the MNIST", *Journal of Integrated Circuits and Systems*, vol. 18, n. 1, 2023. doi: 10.29292/jics.v18i1.663

[9] C. Schuman, et al., "A survey of neuromorphic computing and neural networks in hardware", *arXiv preprint*, 2017. doi: arXiv:1705.06963

[10] N. Caporale and Y. Dan, "Spike timing-dependent plasticity: A hebbian learning rule", *Annu. Rev. Neurosci.*, vol. 31, n. 1, pp. 25-46, 2008. doi: 10.1146/annurev.neuro.31.060407.125639

[11] C. Sun et al., "An energy efficient STDP-based SNN architecture with on-chip learning", *IEEE Trans. Circuits Syst. I Reg. Papers.*, vol. 69, n. 12, pp. 5147-5158, 2022. doi: 10.1109/TCSI.2022.3204645

[12] B. Joo et al., "Energy-and area-efficient CMOS synapse and neuron for spiking neural networks with STDP learning", *IEEE Trans. Circuits Syst. I Reg. Papers.*, vol. 69, n. 9, pp. 3632-3642, 2022. doi: 10.1109/TCSI.2022.3178989

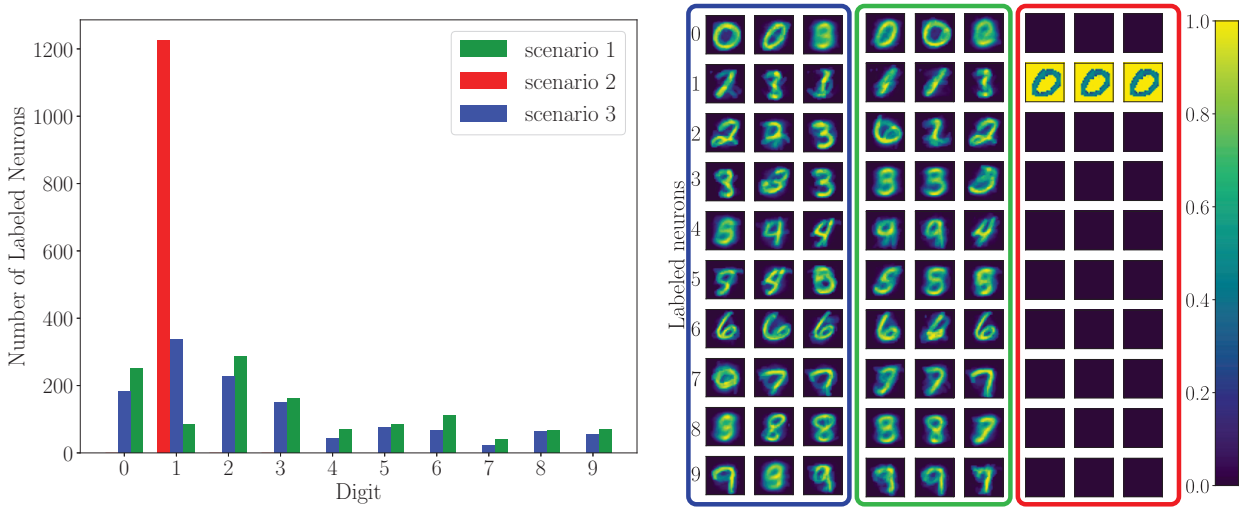

Fig. 11: (a) Number of labeled neurons per digit for a network. (b) Restructured weights (from 784 to 28 x 28) of excitatory connections from input to output neurons for a network. 3 output neurons labeled per digit are randomly picked for each 3 scenarios (making three 10 by 3 grids).

[13] S. k. Vohra et al., "Energy-and area-efficient CMOS synapse and neuron for spiking neural networks with STDP learning", *arXiv preprint*, 2022. arXiv: 2204.04430

[14] F. Danneville et al., "Sub-0.3 V CMOS neuromorphic technology and its potential application", *2021 International Conference on Content-Based Multimedia Indexing (CBMI)*, Lille, France, pp. 1-6, 2021. doi: 10.1109/CBMI50038.2021.9461899

[15] A. A. Abidi, "Phase noise and jitter in CMOS ring oscillators" *IEEE J. Solid-State Circuits*, vol. 41, no. 8, pp. 1803-1816, 2006. doi: 10.1109/JSSC.2006.876206

[16] Z. Jouni, et al., "Jitter Noise Impact on Analog Spiking Neural Networks: STDP Limitations", *36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, Aug. 2023. doi: 10.1109/SBCCI60457.2023.10261661

[17] P. M. Ferreira, et al., "Neuromorphic analog spiking-modulator for audio signal processing", *Analog Integrated Circuits and Signal Processing*,vol. 106, n. 1, 2021. doi: 10.1007/s10470-020-01729-3

[18] T. P. Rioufol, et al., "Revisiting the ultra-low power electronic neuron towards a faithful biomimetic behavior", *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, Rio de Janeiro, Brazil, 2023, pp. 1-6, 2023. doi: 10.1109/SBCCI60457.2023.10261961

[19] Y. Bossard and al., "Analog SNN unsupervised STDP learning, open softw.", *preprint*, 2024. doi: 10.5281/zenodo.12839336

[20] C. Morris and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber", *Biophysical Journal*, vol. 35, pp. 193-213, 1981. doi: 10.1016/S0006-3495(81)84782-0

[21] I. Sourikopoulos, et al., "A 4-fJ/spike artificial neuron in 65 nm CMOS technology" *Frontiers in Neuroscience*, vol. 11, pp. 123, 2017. doi: 10.3389/fnins.2017.00123

[22] T. Grasser, "Noise in Nanoscale Semiconductor Devices", *Springer Nature*, 2020.

[23] G. Bi and M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type", *Journal of neuroscience*, vol. 18, no. 24, pp. 10464-10472, 1998. doi: 10.1523/JNEUROSCI.18-24-10464.1998

[24] Z. Ghahramani, "Unsupervised learning", *Summer school on machine learning. Berlin, Heidelberg : Springer Berlin Heidelberg*, 2003, pp. 72-112. doi: 10.1007/978-3-540-28650-9_5

[25] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity", *PLoS computational biology*, vol. 3, n. 2, pp. e31, 2015. doi: 10.1371/journal.pcbi.0030031

[26] P. Yger and M. Gilson, "Models of Metaplasticity: A Review of Concepts", *Frontiers in computational neuroscience*, vol. 9, pp. 138, 2007. doi: 10.3389/fncom.2015.00138

[27] A. Vigneron and J. Martinet, "A critical survey of STDP in spiking neural networks for pattern recognition", *2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, pp. 1-9. 2020. doi: 10.1109/IJCNN48605.2020.9207239

[28] W. H. Su, et al., "Deep learning of biological models from data: applications to ode models", *Bulletin of mathematical biology*,vol. 83, pp. 1-19, 2021. doi: 10.1007/s11538-020-00851-7

[29] A. Amirsoleimani, et al., "STDP-based unsupervised learning of memristive spiking neural network by Morris-Lecar model", *2017 International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, USA, pp. 3409-3414, 2017. doi: 10.1109/IJCNN.2017.7966284

[30] Z. Li and L. E. Calvet, "Extraction of ECG features with spiking neurons for decreased power consumption in embedded devices", *19th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, Funchal, Portugal, pp. 1-4, 2023. doi: 10.1109/SMACD58065.2023.10192147.

[31] A. Taherkhani, et al., "A review of learning in biologically plausible spiking neural networks", *Neural Networks*, vol. 122, pp. 253-272, 2020. doi: 10.1016/j.neunet.2019.09.036

[32] D. Goodman and R. Brette, "Brian: A simulator for spiking neural networks in python", *Frontiers in Neuroinformatics*, vol. 2, pp. 350, 2008. doi: 10.3389/neuro.11.005.2008

[33] W. Maass, "On the computational power of winner-take-all", *Neural Computation*, vol. 12, n. 11, p. 2519-2535, 2000. doi: 10.1162/089976600300014827

[34] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity", *Frontiers in Computational Neuroscience*, vol. 9, p. 99, 2015. doi: 10.3389/fncom.2015.00099

[35] G. J. Goodhill and H. G. Barrow, "The role of weight normalization in competitive learning", *Neural Computation*, vol. 6, no. 2, pp. 255-269, 1994. doi: 10.1162/neco.1994.6.2.255

[36] M. Minsky and S. A. Papert, "Perceptrons, reissue of the 1988 expanded edition with a new foreword by Léon Bottou: an introduction to computational geometry", *MIT press*, 2017.

[37] R. V. Florian, "Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity", *Neural computation*, vol. 19, no. 6, pp. 1468-1502, 2007. doi: 10.1162/neco.2007.19.6.1468

[38] A. Cyr, et al., "Revisiting the xor problem: a neurorobotic implementation", *Neural Computing and Applications*, vol. 32, no. 14, pp. 9965-9973, 2020. doi: 10.1007/s00521-019-04522-0

[39] L. Mo and M. Wang, "LogicSNN: A unified spiking neural networks logical operation paradigm", *Electronics*, vol. 10, no. 17, p. 2123, 2021. doi: 10.3390/electronics10172123

[40] C. Lee, et al., "Training deep spiking convolutional neural networks with STDP-based unsupervised pre-training followed by supervised fine-tuning", *Frontiers in Neuroscience*, vol. 12, pp. 435, 2018. doi: 10.3389/fnins.2018.00435

[41] D. A. Nguyen, et al., "A review of algorithms and hardware implementations for spiking neural networks", *Journal of Low Power Electronics and Applications*, vol. 11, no. 2, p. 23, 2021. doi: 10.3390/jlpea11020023

[42] H. Hazan, et al., "Bindsnet: A machine learning-oriented spiking neural networks library in python", *Frontiers in Neuroinformatics*, vol. 12, pp. 89, 2018. doi: 10.3389/fninf.2018.00089