Immunity to Device Variations in a Spiking Neural Network With Memristive Nanodevices
Damien Querlioz, Olivier Bichler, Philippe Dollfus, Christian Gamrat

To cite this version:
10.1109/TNANO.2013.2250995. hal-01826840

HAL Id: hal-01826840
https://hal.archives-ouvertes.fr/hal-01826840
Submitted on 29 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Immunity to Device Variations in a Spiking Neural Network with Memristive Nanodevices

Damien Querlioz, Member, IEEE, Olivier Bichler, Philippe Dollfus, Member, IEEE, and Christian Gamrat

Abstract—Memristive nanodevices can feature a compact multi-level non-volatile memory function, but are prone to device variability. We propose a novel neural network-based computing paradigm, which exploits their specific physics, and which has virtual immunity to their variability. Memristive devices are used as synapses in a spiking neural network performing unsupervised learning. They learn using a simplified and customized “spike timing dependent plasticity” rule. In the network, neurons’ threshold is adjusted following a homeostasis-type rule. We perform system level simulations with an experimentally verified-model of the memristive devices’ behavior. They show, on the textbook case of character recognition, that performance can compare with traditional supervised networks of similar complexity. They also show that the system can retain functionality with extreme variations of various memristive devices’ parameters (a relative standard dispersion of more than 50% is tolerated on all device parameters), thanks to the robustness of the scheme, its unsupervised nature, and the capability of homeostasis. Additionally the network can adjust to stimuli presented with different coding schemes, is particularly robust to read disturb effects and does not require unrealistic control on the devices’ conductance. These results open the way for a novel design approach for ultra-adaptive electronic systems.

Index Terms—spiking neural networks, memristors, memristive devices, spike timing dependent plasticity, unsupervised learning, neuromorphic

I. INTRODUCTION

MEMRISTIVE nanodevices provide fantastic opportunities for microelectronics. They can indeed provide a compact multi-level non-volatile memory function [1]. However, they are often subject to strong variability [2], [3], [4], so that fully exploiting their potential would be easier with architectures offering a strong immunity to device variations. Spiking neural networks could provide a serious lead since the brain itself relies on variable neurons and synapses [5] and manages computational efficiency that outperforms man-made systems. This idea takes particular meaning in that many groups (constituting the “neuromorphic” community) already imitate the brain with electronics, using Complementary Metal Oxide Semi-conductor (CMOS) circuits to model its spiking neurons and synapses. These works, however, are often limited by the number of implementable synapses: implementing plastic synapses requires many transistors [6], [7]. Memristive nanodevices could provide the compact synapses required to advance neuromorphic circuits. In recent years several classes of them have indeed emerged, as e.g. resistive RAMs and memristors [1], [2], or adaptive transistors [8], [3]. It has been suggested [9], [10], [11], [12], and shown experimentally [13], [14], [15], [16], [17], [18], that such devices could reproduce a learning rule of biological synapses – spike timing dependent plasticity (STDP) [19], [20] – that is believed to be a foundation of learning in the brain [21]. A system consisting of nanoscale synapses and CMOS neurons could be a major breakthrough in computing, allowing cognitive-type tasks with high efficiency.

This idea is currently receiving considerable interest [22], [23], [24], [25]. However, its sustainability is still to be demonstrated, especially with regards to the variability issue that is common to all memristive technologies [26], [27], [4]. It is also not clear if biological STDP is the best approach for electronics, the constraints of which differ strongly from Biology’s.

In this paper, system simulations introduce quantitative results in terms of computing performance and robustness to variability. We exploit a simplified and customized STDP scheme for memristive devices, which is key to achieve effective learning with extreme robustness to memristive devices’ variability. It is associated with the use of unsupervised learning, and of a homeostasis-type mechanism. We describe the required technology and architecture (section II) and perform system-level simulations on a standard database of machine learning [28] that show the potential of the approach and its robustness. The system consists of an unsupervised layer that extracts features of the inputs using a simplified spike timing dependent plasticity (section III). The network performance compares favorably with traditional – but supervised – networks with similar numbers of adjustable parameters and achieves excellent tolerance to various memristive devices’ parameters variability. Finally, the robustness of the network to other device nonidealities (read disturb and limited resolution effects) is established (section IV).

Partial and preliminary results have appeared in [12]. This papers adds new discussions and results, especially the plausibility of the device model with regards to measurements of real devices, adaptations of the programming scheme, and the impact of diverse device nonidealities. Other proposals have been made to exploit variable adaptive devices in the context of nanotechnological implementations. Most proposed architectures rely on reconfigurable logic [29] or on state-based supervised neural networks [30], [31]. In the first approach...
variability is addressed through error mapping and redundancy, in the second case through a standard neural network approach using supervised learning based on error gradient descent.

Our approach of using unsupervised learning with asynchronous spiking neural networks to tackle the variability issue of nanodevices is original and takes inspiration from recent ideas in computational neuroscience and neural networks [32], [33], [34]. Different works have been published that go into that direction. As mentioned above, several proposals exist to use memristive devices for STDP [10], [9], [11], [35]. In this paper we use a simplified STDP scheme that is easier to implement. Additionally, we propose to give to the neurons a homeostasis property and that is shown to be essential for the robustness of the scheme to variations. One work had already shown that memristive devices with STDP could allow the emergence of receptive fields in a variability-compatible unsupervised approach and synchronous neurons [36]. Our work uses asynchronous designs, like the ones used in the neuromorphic community [6], [7], and performs full learning on a standard dataset. Finally, an alternative way to allow learning with memristive devices in a variability-compatible way can be to use all digital designs [22]. This requires more devices per synapses [37]. In this paper we show that variation-tolerance can be retained by using nanodevices with continuous variation of the conductance.

II. ARCHITECTURE AND IMPLEMENTATION OF THE NETWORK

We first introduce the architecture that we propose for our classifier system. CMOS input and output “neurons” are connected by the nanodevices that act as synapses. It is natural to lay out the nanodevices in the widely studied crossbar as illustrated on Fig. 1, where CMOS silicon neurons and their associated synaptic driving circuitry are the dots, the squares being the nanodevices. The synapses indeed act as adaptive resistors. With the crossbar layout, if several synapses are active at the same time (i.e. receiving voltage spikes), the output receives directly the sum of the currents flowing through the synapses. In a more futuristic design, the system could also be laid out in a CMOL architecture where nanodevices crossbar is fabricated on top of the CMOS neurons and driving circuits [30].

This kind of connectivity corresponds to a feed-forward architecture in a neural network. Of particular interest, in the case of our network, it limits the sneak path issue because both programming and reading are performed in parallel. This issue usually limits the competitiveness of crossbars [38] and requires complex counter-measures like complementary resistive switches [39], or the use of nonlinear devices [40].

As a replacement of memristive devices, the architecture may also exploit phase change memories associated in “2-PCM” circuits as evidenced experimentally in [14], [18], [41], which has the advantage of technological maturity.

The input neurons present the stimuli as asynchronous voltage spikes using several possible coding schemes described in section III-C. Spiking rate is proportional to stimulus intensity. These stimuli may originate for example directly from a spiking retina [42] or cochlea [43] designed in the neuromorphic community that present data as asynchronous spikes, similarly to their biological counterparts.

As a result of learning, the output neurons should become selective to the different stimuli classes that are presented in a fully unsupervised manner: the output neurons should develop selectivity to specific features contained in the input patterns. The learning rule of the nanodevices needs to be fully local to the nanodevices to which it is connected. When an output neuron spikes it applies an Output pulse (b) to the nanodevices to which it is connected. When the voltage applied on the device (difference between the voltages applied at the two ends (c) or (d)) reaches $V_T^+$ or $V_T^-$, its conductance is increased or decreased, respectively.

Figure 2. Pulses for simplified STDP (voltage pulses as a function of time). When an input neuron spikes, it applies an Input pulse (a) to the nanodevices to which it is connected. When an output neuron spikes it applies an Output pulse (b) to the nanodevices to which it is connected. When the voltage applied on the device (difference between the voltages applied at the two ends (c) or (d)) reaches $V_T^+$ or $V_T^-$, its conductance is increased or decreased, respectively.
When an output neuron spikes, it applies a pulse that is dependent on the activity of the neurons to which they are connected, which provides the foundation of learning by the system.

The memristive nanodevices are programmed as follows. When being applied a positive voltage pulse higher than a threshold $V_{T+}$, they increase their conductance. When applied a voltage pulse smaller than a negative threshold $V_{T-}$, they decrease their conductance [13], [10]. Previous works have shown that memristive devices can implement spike timing dependent plasticity (STDP), a learning rule used by brain synapses [13], [14], [15], [9], [11], [10], [16], [17], [35]. These works focused on faithful imitation of Biology. In this work, we focus on proposing simpler scheme, targeted toward pattern extraction, as illustrated in Figure 2.

- When an input neuron spikes, it applies a long voltage pulse to its synapses (Input pulse, Figure 2(a)). This voltage is high enough to drive some current into a memristive device, but not enough to reprogram it. This current is integrated by the output neurons (resistor role of the synapse). If several synapses connected to the same output neuron are active at the same time their currents are summed.

- When an output neuron spikes, it applies a pulse that is a succession of a negative bias and of a positive bias (Output pulse, Figure 2(b)). If no Input pulse is being applied to the device, only the second part reaches a threshold and the conductance of the synapse is increased by $\delta G_p$ (Figure 2(d)). However, if the input neuron had spiked recently, and the Input pulse is still being applied on the other end of the device, the voltage applied on the device actually increases its conductance by $\delta G_p$ (Figure 2(c)).

This simple learning rule, easily implemented with nanodevices, is the ground for learning. Compared with the purely bioinspired and more complex scheme introduced in [9], no delay matching is necessary between the Input and Output synaptic waveforms, which should make the driving circuitry much easier to design.

The way this simple learning rule works is straightforward. When an output neuron declares a spike (at time $t_{\text{spike}}$), it increases by $\delta G_p$ the conductance of the synapses connected to input neurons that spiked recently (from $t_{\text{spike}} - t_{\text{PRE}}$ to $t_{\text{spike}}$, if $t_{\text{PRE}}$ is the duration of the Input pulse), and decreases by $\delta G_m$ the conductance of the synapses that did not. This increases the sensitivity of the neuron to the specific pattern that activated it, making it more likely to spike for a similar (correlated) pattern in the future. This process – that works surprisingly well in practice, as we show in this paper – has been partially theorized in [34]. A comparison with the traditional biological STDP scheme is presented in Figure 4.

A disadvantage of this rule is the long Input pulse, which drives current for a long time, and thus increases power consumption. If it becomes significant, a low power version of the learning rule is possible and illustrated in Figure 3, for the cost of limited added complexity. The input neuron does not apply a voltage pulse during the whole time it is active, but only at the beginning to drive current into the output neuron. Additionally, as soon as one of the output neuron becomes active, a signal is sent back to the input neurons, which apply a short Input pulse again if they are still active. This lower power version of the learning scheme is more realistic with most current technologies like [13], [14].

2) Memristive devices modeling: To model the conductance increments and decrements in our system simulations, we use the model introduced in [44]. It takes inspiration from experimental memristive devices measurements [13], [15]. An increase in the conductance is modeled by the equation:

$$
\delta G_p = \alpha_p e^{-\beta_p \frac{G - G_{\text{min}}}{G_{\text{max}} - G_{\text{min}}}}. 
$$

Similarly, a decrease is modeled by:

$$
\delta G_m = \alpha_m e^{-\beta_m \frac{G_{\text{max}} - G}{G_{\text{max}} - G_{\text{min}}}}. 
$$

The exponential factor expresses the fact, observed in most memristive technologies, that a given voltage pulse has a reduced effect on the device conductance if applied several times [13], [15], [14]. Agreement with the experimental data of [13] is presented in Figure 5. The parameters $\alpha_p$, $\alpha_m$, $\beta_p$, $\beta_m$ depend heavily on the Input and Output pulse voltages that are chosen. These parameters, as well as minimum and maximum conductances $G_{\text{min}}$ and $G_{\text{max}}$ are subject to device variability in real devices.

B. Output neurons

1) Output neurons’ dynamics: Exploiting the devices requires connecting them to processing units – silicon neurons able to process and generate spikes in a bioinspired manner...
by integration of their input. We call $X$ the state variable (a current or a voltage, equivalent of the biological “membrane potential”) of the neuron (expressed in normalized unit where the maximum value of the state variable $X$ is 1). Neurons are leaky integrate-and-fire type, which is meant to solve the simple following equation.:

$$\tau \frac{dX}{dt} + gX = \gamma I_{input}$$

(3)

where $\tau$ is a leak time constant, and $g$ and $\gamma$ are constants. $I_{input}$ represents the current flowing through the line of the crossbar connected to the neuron:

$$I_{input} = \sum_j I_j$$

(4)

where $I_j$ are the currents flowing through each memristive device $j$ connected to output neuron.

The neuron declares a spike if $X$ reaches a given threshold $X_{th}$, in which case $X$ is reset to zero.

An approach widely studied in the neuromorphic community is to use analog circuits (generally with transistors operating in the sub-threshold regime) able to receive and generate asynchronous spikes [6], [7] to design such neurons. This kind of CMOS design is particularly low power, because most transistors operate in the subthreshold regime, and thanks to the use of asynchronous computation. Though smaller than in nanodevices, variability is also a problem for such transistor designs. It is a challenge for any neuromorphic design [43] and will become even more crucial when scaling to modern technology. Digital designs may also be used that do not suffer from variability directly but may have higher area and power requirements [45].

2) Output neurons’ lateral inhibition: When an output neuron spikes, it sends inhibitory signals to the other output neurons of the layer that prevent them from spiking during the inhibition time and resets their potential to zero. With this inhibition, the network is reminiscent of a Winner-Takes-All topology [34].

More precisely, when an output neuron spikes, the state variable $X$ of the other output neurons is reset to zero during a time $t_{inhbit}$.

$$X = 0 \text{ if } t_{spike} < t < t_{inhbit}.$$  

(5)

In hardware, this inhibition between the neurons can be implemented in a compact way through diffuser networks as in [6], which require a minimum number of transistors.

3) Homeostasis: A final issue for the architecture is the adjustment of the neurons’ threshold. Simple algorithms exist for traditional (non spiking) artificial neuron networks, but they do not work directly for spiking neurons. A bioinspired original route is homeostasis [5]. A target activity is defined for the neurons (i.e. a number of times an output neuron should spike over an extended period of time, like 100 digits presentation). Regularly the threshold of the neuron is increased if the average activity of the neuron is above the target, and decreased if it is below.

$$\frac{dX_{th}}{dt} = \gamma (A - T),$$

(6)

where $A$ is the mean activity (or firing rate) of a neuron, $T$ is the target activity, and $\gamma$ is a multiplicative positive constant.

This ensures that all the output neurons are used and adjust the neurons’ thresholds to the stimuli for which they become specialized. In neuromorphic hardware, homeostasis has been implemented with analog memories like in [46] or could be implemented digitally. The advantage inherent in this technology is evidenced in this paper in section III-B.

C. Simulations of the System

In this paper, all simulations are system-level and are based on a C++ special purpose code (“Xnet”) [47], [12]. The code is event-based for simulation performance and runs on traditional central processing units (CPUs). Simulation parameters as introduced above are $\tau = 100 \text{ ms}$, $g = 1$, $X_{th} = 0.5$ (normalized unit), inhibition time $t_{inhbit} = 10 \text{ ms}$, $\alpha_p = 10^{-2}$, $\alpha_m = 5 \cdot 10^{-3}$, $G_{min} = 10^{-4}$, $G_{max} = 1$ (normalized units where the mean of the maximum conductance value is 1), $\beta_p = 3.0$, $\beta_m = 3.0$. The width of the Input pulses is 25 ms. Parameter variations are introduced around all the parameters using Gaussian random numbers (the value of their standard deviation is given in Section III). The initial conductances are selected randomly around mid-range (0.5). The stimuli are applied using the coding schemes described in section III-C.

The two variations of voltages pulses (Figures 2 and 3) gave identical performance in terms of recognition rate. All the results presented in the paper use the pulses of Figures 2. For demonstration of the concept, in this paper we use the widely studied case of handwritten number recognition.
The MNIST database is used, which consists in handwritten number of 28 × 28 pixels by 250 writers [28].

In order to achieve learning, we present the full MNIST training database (60,000 digits) three times to the system. Each input neuron is connected with one pixel of the image. It emits spikes with a jittered rate that is proportional to the pixel intensity (maximum rate is 22 Hz) as illustrated in Figure 11(b). The initial phase is random. Input neurons present spikes corresponding to a given digit during a period of 350 ms, after which they present spikes corresponding to another digit. No kind of preprocessing on the digits is used and the set is not augmented with distortions. The network is then tested on the MNIST test database, which consists in 10,000 digits that have not been presented during training. Simulation time was about 8 hours per run on an AMD Opteron 2216 CPU.

Figure 6 plots the synaptic conductances (or weights) learned by the system in a configuration with only 10 output neurons. It is remarkable that without any supervision and using only our local custom STDP rule, the system has identified 9 (out of 10) different numbers, the real features of the input. Moreover it has learned the distinctive features of the digits (and not just the most likely handwriting): it has learnt the loop of the digit two, the bottom of the six, or the horizontal parts of three and eight.

To evaluate the capability of the system, we can define a recognition rate. For that purpose, we associate output neurons with the digit for which they spike the most frequently a posteriori, using a subset of 1000 well identified numbers. In hardware this association could be performed with complex digital circuitry. An alternative can be to associate the unsupervised network with a supervised one [48]. All the simulations were repeated ten times, and the recognition rate given is averaged on the ten runs.

In order to evaluate quantitatively the network’s recognition rate, Figure 7 plots the final recognition rate on the test database. With ten output neurons the recognition rate reaches 60%. To improve the recognition rate, we can introduce additional output neurons, in which case some output neurons respond to different handwritings of the same digit. All the output neurons respond to some kind of digits, as explained in section II-B3. With 50 output neurons, the recognition rate reaches 81%, and with 300 output neurons 93.5%. A traditional artificial neural network with back-propagation and 300 hidden neurons (obtained with the same number of adjustable parameters) reaches 95% [28], which compares to our rate of 93.5%. In the literature, the best algorithm has a largely superior 99.7% recognition rate, but using 12 million adjustable parameters (vs. 235, 200 here) and a largely augmented training set [49]. Though our numbers are clearly more modest, the interest here is that the network is fully unsupervised, with simple local learning rules, and variability immunity as is seen in the next section.

Thanks to the unsupervised nature of learning, the complicated “labeling” step does not need to be performed right away. Lots of unknown data may be presented for training, and the labeling can be performed from a limited subset of well identified data. This is a strong advantage for many problems involving natural data. In many cases, a lot of data is available, but that was not classified, and the system can here perform this analysis by itself.

III. VARIABILITY IMMUNITY

A. Synaptic variability

We now exploit our system-level simulations to analyze the robustness of our approach. We first study the impact of nanodevices (synaptic) variability on the recognition rate of the network. For this study, we simulated the network with 50 output neurons, introducing different kinds of variability. Results are reported in Figure 8.

In the top curve of Figure 8, we evaluate the impact of variability of the initial conductance (i.e. the initial conductance of the nanodevices, before learning starts). We can see it has no
impact on the final recognition rate, even when the variability is extreme. This means that we do not need to control the initial conductance precisely. This is fortunate since this would be complex: controlling the exact state of memristive devices organized in a crossbar is a difficult issue, due to the device variations and the sneak paths.

In the middle curve of Figure 8, we evaluate the impact of the variations on the learning increments and decrements (the parameters \( \alpha_p, \alpha_m \) of equations 1 and 2). These variations can be caused by dispersion of the device thresholds and of their programming characteristics. We can see that the network is immune to variations up to 25% of the mean value of these parameters (i.e. \( \sigma/\mu = 25\% \)). This is already an extremely high level of variation for an electron device, but typical for research nanodevices. With 50% of variation, there is a small decrease of recognition rate (79% instead of 82% with no variability). With an extreme variability of 100% on the synaptic parameters, the recognition rate decreases significantly, but interestingly the functionality of the network is not challenged. We should note that with a variability of 50%, as many as 4% of the nanodevices cannot be programmed in at least one direction (i.e. they have an \( \alpha \) value of 0). The latter figure becomes 30% when parameter variability is 100%. This is an indication of the overall tolerance to defects. To give a feel of the magnitude of these dispersions, we plotted on Figure 9, the conductance as a function of pulse number (plotted similarly to Figure 5), for 100 devices with different relative standard dispersion \( \sigma/\mu \) on the parameters \( \alpha_p, \alpha_m \).

The bottom curve of Figure 8 adds variations of \( G_{\text{min}}, G_{\text{max}} \). In that case some nanodevices overpower the others due to increased maximum conductance. Results are however similar to the previous case, and extreme robustness is preserved: with 50% variability, the recognition rate is lowered from 81.9% to a reasonable 77.2%. The most sensitive parameter is the maximum conductance. It should be noted that memristive currently devices developed in academic groups may have relative standard variations in \( G_{\text{min}}, G_{\text{max}} \) in the range between 50% or 100% ([4], [26] suppl. info.). The situation is expected to improve dramatically when the technology becomes mature. However, some significant degree of variability might be intrinsic to their physics [27].

This degree of robustness to device variation is exceptional in electronic systems and constitutes one of the strongest points of the approach. It takes root in the unsupervised nature of learning: output neurons learn features for which they are naturally suited. Synaptic variability determines how a neuron reacts initially, and thus what it is going to learn. This means that, in a way, variability is not a real problem, but a feature to initiate learning.

The simplicity of the voltages pulses is also fundamental to ensure this robustness.

B. CMOS neuron variability

We now study the impact of the variability of the CMOS neurons. The impact of the variability of their threshold is seen in Figure 10. We can see that without homeostasis the impact of the variability is dramatic. The more excitable neurons (the ones with lower thresholds) spike predominantly and the other neurons do not specialize efficiently. In a typical simulation with a threshold variation of 25%, the most excitable neuron spikes 51% of the time, and in a simulation with a threshold variation of 50%, it spikes 91% of the time. Homeostasis however fully compensates this issue (all the neurons spike between 1.5 and 3% of the time, most neurons spiking around 2%). And the same recognition rate as without threshold variation is achieved as evidenced in Figure 10. Homeostasis appears as particularly valuable in this application. This also stresses the importance to evaluate the robustness of the system not only to nanodevices’ variations, but also to the CMOS’.

C. Stimulus encoding

Another remarkable point is the insensitivity to stimulus encoding scheme. Different schemes are possible to encode the stimulus into spikes as illustrated in Figure 11. In the first two schemes (Figures 11(a) and 11(b)), the input neurons spike periodically, the firing rate being proportional to the input quantity. The input neurons can spike either in phase (Figure 11(a)) or out of phase (Figure 11(b)). This is a natural scheme based on how spiking retinas currently work [42]. In the last case (Figure 11(c)), input neurons spike in a Poissonian way, the time constant being inversely proportional to the input. All three coding styles lead to a recognition rate between 81% and 82%. The only condition for this is that the window for
potentiation in the STDP scheme is long enough with regards to the typical interspike interval. In this work, the average interspike interval is 45 ms for a black pixel, and insensitivity to stimulus encoding scheme is achieved if the window for potentiation is longer than 25 ms (while 15 ms is sufficient in the jittered periodic case). This shows again the robustness of the approach.

Additionally, noise can be inserted in the input with no impact. Adding 10% of perfectly random spikes distributed on all the input neurons has no impact on the system’s recognition rate. This kind of robustness to spikes’ precise timing is consistent with computational neuroscience works on STDP [32]. This result is not surprising since the precise timing of the spikes does not play a major role neither for the neurons nor for the simplified STDP learning rule.

IV. DEVICE NON-IDEALITIES

Finally, we check that our approach is not sensitive to two crucial non-idealities of real memristive devices: the limited resolution of the conductance and the read disturb effect.

A. Limited Resolution of the Memristive Devices

The nanodevices that we are considering have an “analog memory” property. It has been evidenced in many real-life devices [13], [14], [15], [16], [17]. In these real devices, however, the conductance cannot be programmed with an arbitrary precision [13]. It is thus important to clarify which precision is required for our application. To that purpose, we performed simulations with different increments \( \alpha_p \) of equation (1) (the decrement \( \alpha_m \) of equation (2) is kept proportional to \( \alpha_p \)). The results in terms of recognition rate are plotted in Figure 12 (\( \alpha_p \) is expressed in normalized units where the maximum conductance is 1). We see that recognition rate is maximum for \( \alpha_p \) until 0.05, which is achievable by several experimental devices. For example the best measurements in [15] achieve \( \alpha_p \sim 0.01 \), or \( \alpha_p \sim 0.02 \) in [14] and \( \alpha_p \sim 0.07 \) in [13]. For higher values of \( \alpha_p \), the recognition rate decreases and the devices’ resolution is not sufficient.

B. Impact of Read Disturb

Another significant device non-ideality can materialize if the Input pulses (which are subthreshold) have an impact on the devices’ conductance. This corresponds to the Read Disturb phenomenon for memory devices. To assess the impact of this phenomenon, we performed simulations where the Input pulses increase the device conductance by a fraction \( \epsilon \) of the increase of programming threshold:

\[
\delta G_{\text{subthreshold}} = \epsilon \delta G_p.
\]  

The results are presented in Figure 12. It appears that read-disturb parameter \( \epsilon \) as high as 0.1 (meaning that a read pulse has 10% of the impact of a write pulse) may be tolerated, since it is fully compensated by learning. Such a value is a lot higher than what is seen in real devices [26], [13]. Read disturb should thus not be an issue for our approach.

REFERENCES


![Figure 12. Impact of the conductance increment on the recognition rate for a system with 50 outputs. Inset: Impact of the read disturb parameter \( \epsilon \) on the recognition rate for a system with 50 outputs (as defined by equation (7)). Every simulation was repeated five times.](image-url)


