

In Situ Training of CMOL CrossNets

Jung Hoon Lee and Konstantin K. Likharev

Abstract— Hybrid semiconductor/nanodevice (“CMOL”) technology may allow the implementation of digital and mixed-signal integrated circuits, including artificial neural networks (“CrossNets”), with unparalleled density and speed. However, previously suggested methods of CrossNet training may be impracticable for large-scale applications of these networks. In this work, we are describing two new methods of “in situ” training of CrossNets, based on either genuinely stochastic or pseudo-stochastic multiplication of analog signals, which may be readily implemented in CMOL circuits. The methods have been tested by numerical simulation of CrossNet-based perceptrons by error backpropagation on three problems of the Proben1 benchmark dataset. The testing gave very encouraging results: CMOL CrossNets with their binary elementary synapses may provide, after the in situ training, classification performance at least on a par with the best results reported for software-based networks with continuous synaptic weights.

I. INTRODUCTION

The recent advances of nanotechnology, and in particular the molecular electronics (see, e.g., Ref. 1 for a recent collection of reviews), give every hope for the practical introduction, perhaps within the next 10 years, of hybrid CMOS/nanodevice integrated circuits of unparalleled density – up to 10^{12} functions per cm^2 [2-4]. In particular, our group is working on the development of hybrid circuits of specific topology, dubbed CMOL [3, 4]. As in some earlier proposals (for their review, see [2]), nanodevices in CMOL circuits are formed at each crosspoint of a crossbar array, consisting of two levels of nanowires. However, in order to overcome the CMOS/nanodevice interface problems pertinent to the architectures proposed earlier [2], in CMOL circuits the interface is provided by pins that are distributed all over the circuit area, on the top of the CMOS stack. (The technology necessary for fabrication of pins with nanometer-scale tips has already been developed in the context of field-emission arrays.) Such structure, with a certain angle between the nanowire crossbar and the square lattice of interface pins, gives the CMOS subsystem an individual access to any nanodevice, even if the wire half-pitch F_{nano} is much less than that (F_{CMOS}) of the semiconductor subsystem [3, 4].

The main idea of CMOL circuits is combine the

advantages of their components: the high functionality (in particular, high voltage gain) and reliability of silicon field-effect transistors of the CMOS subsystem with the extremely high possible density of nanowires and molecular nanodevices. This is why the devices may be rather simple; for example, the functionality of two-terminal latching switches, which may be based on single-electron tunneling [3, 4] is sufficient for most prospective applications of CMOL circuits [5-7]. This simplicity may allow us to apply inexpensive “bottom-up” methods of nanodevice formation, for example the self-assembled monolayer-based technique which has already been demonstrated to provide high device yield (so far, for rather simple molecules [8]).

Analyses have already shown that CMOL circuits may be used for very efficient digital memory [5] and logic [6, 7] architectures enabling high performance (at acceptable power) at relatively high defect tolerance. (Such tolerance is deemed necessary for any nanodevice-based electronics.) Moreover, the very structure of these circuits makes them uniquely suitable for the hardware implementation of artificial neural networks (“CrossNets” [9]). In these networks, relatively large and sparse neural cell bodies (“somas”) are implemented in the CMOS subsystem, crossbar nanowires serve as dendrites and axons, while the latching switches serve as elementary (binary-weight) synapses. The specific spatial distribution of somatic cells determines CrossNet connectivity; in particular, it is straightforward to implement feedforward multilayer perceptrons (MLP) [9, 10].

In our earlier work we have shown that despite restrictions imposed by CMOL hardware, CrossNets can perform not only simple associative memory functions [9], but more complex tasks such as pattern classification [10]. (For the latter tasks, CrossNet MLPs should use composite synapses providing L -level synaptic weights.) The significance of this result is in the CrossNet’s potential unparalleled density and speed: for realistic component parameters, the elementary synapse density may be as high as 10^{12} cm^{-2} , while the average cell-to-cell communication delay may be as low as ~ 10 ns (i.e. orders of magnitude lower than that in mammal’s cerebral cortex), at acceptable power. As a result, CMOL CrossNet chips of even modest size (below 1 cm^2), with unoptimized architecture, could provide unparalleled classification performance, e.g., recognition of a person’s face in a crowd may be achieved as fast as in $100 \mu\text{s}$, the number to be compared with ~ 3 hrs

This work was supported in part by AFOSR, NSF and MARCO via FENA Center.

The authors are with Department of Physics and Astronomy, Stony Brook University, Stony Brook, NY 11794-3800 (phone: 631-632-9786, fax: 631-632-4977, e-mail: jlee@grad.physics.sunysb.edu).

for the same task running the same algorithm (and hence providing similar fidelity) on a high-performance digital microprocessor [10]. Simultaneously, CrossNets may be very defect-tolerant, sometimes operating well with a fraction of bad synapses well above 30% [9, 11].

One of the major issues of CMOL CrossNet development is training of these networks. If a CrossNet is not too large, its synaptic weights may be set up with a simple import procedure [9]. Unfortunately, this method may not be practicable for very large CrossNets, because it requires the preliminary calculation of the weights on a precursor network implemented using either an ordinary computer or a digital CMOL chip. This is why the CrossNet training “in situ” (i.e., without a precursor network) is an important problem.

Virtually every supervised learning algorithm (e.g., the generic Hebb rule, error backpropagation, etc.) requires the synaptic weight change Δw_{ij} to be proportional to the product of two analog signals. Earlier we suggested [9] a method of such multiplication based on the internal stochastic dynamics of single-electron latching switches working as elementary synapses. Unfortunately, this method would not work for the most realistic case when the single-electron island of the switch has sub-nm size – the necessary condition of room temperature operation of single-electron devices.

The goal of this work has been to develop methods of approximate multiplication which use the natural randomness of nanodevice synapses. The suggested methods, based on the comparison of the analog signals to be multiplied with either random or periodic clock signals, have been successfully tested on such a well-known benchmark set of classification problems as Proben1 [12].

II. SINGLE ELECTRON LATCHING SWITCH

Fig.1 shows a latching switch [13] which is a two-terminal combination of two single-electron devices, a “transistor” and a “trap” [3]. If the applied voltage $V = V_1 - V_2$ is low, the trap island in equilibrium has no extra electrons ($n = 0$), and its net electric charge $Q = -ne$ is zero. As a result, the transistor is in the closed (“Coulomb-blockade”) state [3], and input and output wires are essentially disconnected (Fig. 1b). If V is increased, the electrochemical potential of the trap island increases above V_2 due to the effect of capacitance C_1 . If V exceeds a certain threshold value V_+ , this increase forces one more electron to tunnel into the box: $n \rightarrow 1$. This change of box charge affects, through the coupling capacitance C_c , the potential of the transistor island, and lifts the Coulomb blockade; as a result, the transistor connects the nanowires with a finite resistance R_0 (Fig. 1b). If V stays above V_+ , this connected (ON) state is stable, but if it is decreased below a lower value, V_- , the additional electron is ejected from the trap. As

a result, the transistor closes, disconnecting the wires.

Actually, ON and OFF switching processes (as all dynamics of single-electron devices) are random, and only their rates (switching probabilities as functions of time) Γ_{\pm} are definite functions of the applied voltage V [3]. (The simplified picture presented on Fig. 1b is valid only if either the rates, or the characteristic time t of experiment, or both are large enough: $\Gamma_{\pm}t \gg 1$.) The “orthodox” theory of single-electron tunneling predicts the rates to be exponentially-linear functions of V . (In this case, the desired proportionality of the average synaptic weight adjustment speed to the product of two analog signals may be implemented using a simple time multiplication procedure [9].) However, for room temperature operation, the single-electron island size should be so small (below ~ 1 nm) that electron motion in it is substantially quantum-confined, making the energy spectrum discrete [3]. Theory [14] shows that in this case, for a substantially large distance between the switching threshold values, $V_+ - V_- \equiv 2V_{th} \gg k_B T/e \approx 26$ meV, the dependences $\Gamma_{\pm}(V)$ may be well approximated by step functions (Fig. 1c). We will use this approximation. For the sake of notation simplicity, we will also select the voltage origin in the middle of the interval, i.e. accept $V_+ = +V_{th}$, $V_- = -V_{th}$.

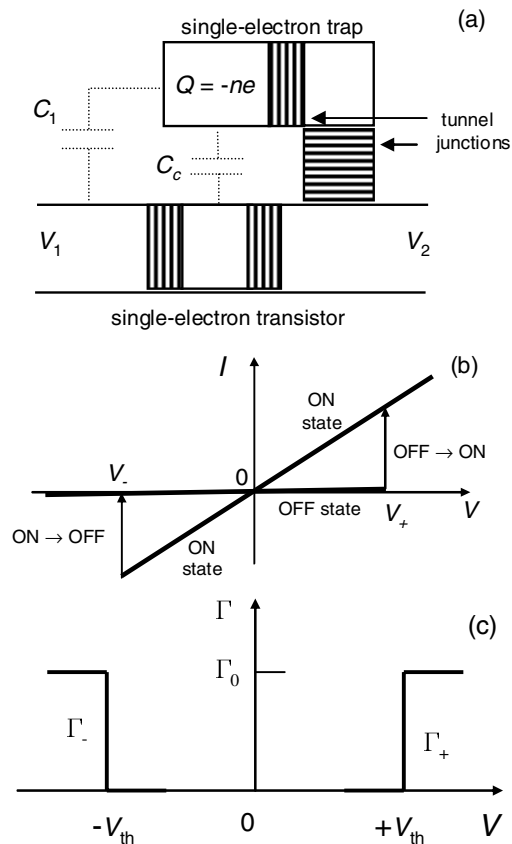


Fig. 1. Single-electron latching switch: (a) schematics, (b) dc I - V curve, and (c) the model for switching rates accepted in this work.

III. METHOD 1: STOCHASTIC MULTIPLICATION

Fig.2 shows our idea of our first method, the stochastic multiplication of analog signals.¹ It uses the dual-rail presentation for both pre-synaptic (V_1) and post-synaptic (V_2) signals. Each synapse consists of four groups (arrays) of $n \times n$ similar elementary latching switches. In the operation mode, the open switches supply currents, proportional to axonic voltages, to the inputs of differential dendritic amplifiers, with the same polarities as shown in Fig. 2 [9]. As a result, the net synaptic weight is

$$w = \alpha N = \alpha [N_{++} + N_{--} - N_{+-} - N_{-+}] \quad (1)$$

where each component of N is the number of ON-state switches in each group ($0 \leq N \leq n^2$), so that w may have $L = 4n^2 + 1$ different values. (Our prior work [9] has shown that, for example, $n = 5$, i.e., $L = 101$, is quite sufficient for high-fidelity pattern classification.)

In the training (weight adjustment) mode, voltages $V_1(t)$ and $V_2(t)$ are developed by comparators $C_{1,2}$ with binary outputs. The comparators are fed by:

(i) the analog signals $x_{1,2}$ to be multiplied, with the magnitude limited to a certain range $[0, x_{max}]$, and the signs changed in time to perform 4-stage time division multiplexing (see Table I), and

(ii) random, analog reference signals $REF_{1,2}$ from two independent random signal generators, with the uniform probability within the same range $[0, x_{max}]$.

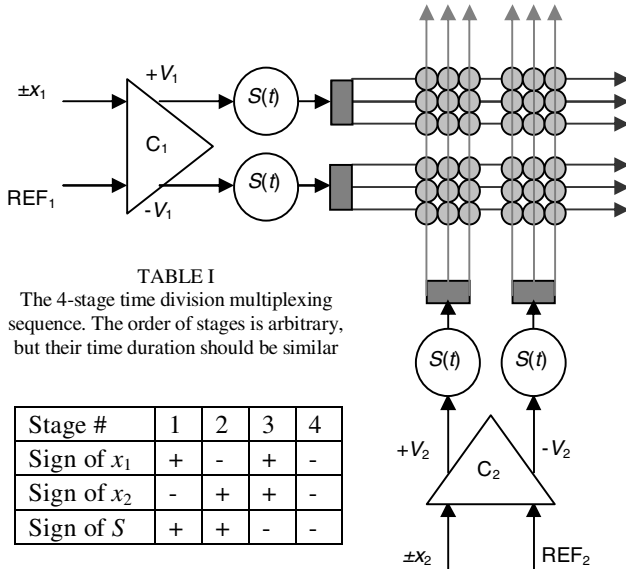


TABLE I

The 4-stage time division multiplexing sequence. The order of stages is arbitrary, but their time duration should be similar

Stage #	1	2	3	4
Sign of x_1	+	-	+	-
Sign of x_2	-	+	+	-
Sign of S	+	+	-	-

Fig.2. The scheme of our in-situ training methods. In the first (stochastic multiplication) method, references $REF_{1,2}$ are random analog signals, while in the second method they are deterministic sawtooth waveforms with different frequencies. Each small circle is the latching switch (Fig. 1); the composite synapse consists of four groups of $n \times n$ similar switches. (The figure is for the

¹ A somewhat similar idea, but in application to digital rather than analog signals, has been discussed in [15].

case $n = 3$). The lines connected by switches are, respectively, the “axonic” and “dendritic” nanowires [9], while $C_{1,2}$ are the signal comparators with binary output signals. The alternating global shift signal $S(t) = \pm S_0$, together with flipping the signs of signals $x_{1,2}$, performs time division multiplexing.

Each comparator performs the following function: if the input signal $\pm x_i$ ($i = 1,2$) is larger than random reference REF_i , the comparator generates voltage $V_i \equiv (V_{th}/2)\text{sgn}(\pm x_i)$, and applies it, in the dual-rail format, to its outputs; otherwise no output is produced. A global shift signal $S(t) = \pm S_0$, with an amplitude S_0 somewhat below $V_{th}/2$, and the sign alternating in accordance with Table I, is added to the output voltages. It is easy to see that the simultaneous switching of the signs of x_i and $S(t)$ ensures that the net voltage may exceed V_{th} , and hence cause switching with a finite rate Γ_0 (see Fig. 1c), only in one group of synapses.

Let us consider, for example, stage 1 for the case when both x_1 and x_2 are positive. In this case only the top left array of switches may be activated, and indeed is if each x_i is larger than REF_i . Due to the uniform probability distribution of each REF_i , the probability that each comparators generate the finite output signal is $P_i = x_i/x_{max}$. Since the signals REF_i are independent, the probability that both comparators produce the finite output, i.e. that the switching rate is finite (Γ_0 , see Fig. 1c) is

$$P_{\Gamma} = P_1 P_2 = x_1 x_2 / x_{max}^2 \quad (2)$$

This is the essence of our idea of signal multiplication. However, the real law of the average synaptic weight change is slightly different, because the change of probability p of having any latching switch in ON state depends not only of Γ_{\pm} , but also on p itself:

$$dP/dt = \Gamma_+(1-p) + \Gamma_-p \quad (3)$$

Combining (1)-(3), and taking into account all 4 stages of the time division multiplexing period, we get the following final formula for the synaptic weight change during a relatively short time interval $\Delta t \ll 1/\Gamma_0$:²

$$\langle \Delta w \rangle = \eta x_1 x_2 \times \begin{cases} (w_{max} - w), & x_1 x_2 > 0, \\ (w_{max} + w), & x_1 x_2 < 0, \end{cases} \quad (4)$$

with $\eta \propto \Gamma_0 \Delta t$. Here $w_{max} = 2\alpha N = 2\alpha n^2$ is the maximum synaptic weight, corresponding to all binary switches in ON state. Note the relaxation term in the right-hand part of (4),

² This condition means that we are actively exploiting the random character of nanodevice switching. In our simulations described below, the product $\Gamma_0 \Delta t$ was 4×10^{-3} . Still, the interval Δt should include at least one full time division multiplexing period. In our simulations, the period was equal to Δt .

giving a contribution proportional to $-w$ into $\langle \Delta w \rangle$; it plays the “finite memory time” role to some extent similar to that in the renowned Oja Rule [16].

Having a generator of random analog signals REF_i in each somatic cell might significantly increase CrossNet hardware complexity. This is why we have also explored a simpler version of the method, in which one reference signal is used by many cells, e.g., all cells of one MLP layer. This simplification does not affect the network performance – see Sec. V below.

IV. METHOD 2: PSEUDO-STOCHASTIC MULTIPLICATION

The hardware complexity may be reduced even further using two periodic sawtooth waveforms with different periods τ_i , instead of analog random signals, as the reference signals REF_i , within the same general scheme (Fig. 2). When a sawtooth waveform is fed to a comparator, the device generates nonzero output signal with the corresponding period and the duty cycle linearly depending on the input signal x (Fig. 3a).

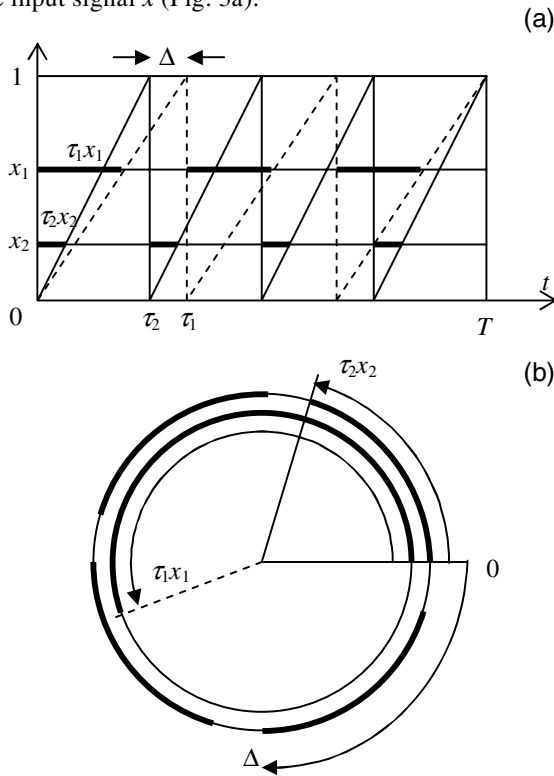


Fig. 3. Time diagrams for comparators fed by sawtooth reference signals, for the case $n = 3$: (a) linear presentation and (b) presentation on a cylinder. In both cases, the periods of nonzero signal generation by each of comparator are shown with bold lines. (Latching switches are updated during the time segments when these periods overlap.)

In order to analyze statistical properties of the resulting “beats” (considering input signals x_1 and x_2 as random numbers), let us consider the case of a simple and practically convenient relation of reference signal periods: $\tau_1 = n\Delta$, where $\Delta \equiv \tau_1 - \tau_2$, and n is an integer. In this case

the beat process has a time period $T = n(n-1)\Delta = (n-1)\tau_1 = n\tau_2$ commensurate with both τ_1 and τ_2 , and the analysis may be restricted to one beat period. Let us wrap the time axis on a cylinder with circumference τ_1 – see Fig. 3b drawn for the particular value $n = 4$, and $\tau_1 x_1 > \tau_2 x_2$. (For convenience, starting from after this point, signals x_i are normalized to their maximum value, so that $0 < x_i < 1$.) In this presentation, the repeating periods when the first comparator gives finite output are overlapped, while those for the second comparator move clockwise by Δ each period τ_2 , until they come to the initial position after the full beat period T . The duration τ of time segments when the both comparators give nonzero output signal (and hence the synapse is being updated) may be presented by different formulas in three possible cases:

Case 1 ($-m\Delta + x_2\tau_2 > 0$):

$$\tau = \sum_{m=0}^{m_1} (\tau_2 x_2 - m\Delta) = \frac{\tau_2 x_2}{2} \left(\frac{\tau_2 x_2}{\Delta} + 1 \right) + \frac{1}{2} \Delta a (1-a),$$

where

$$m_1 \equiv \left\lfloor \frac{\tau_2 x_2}{\Delta} \right\rfloor = \frac{\tau_2 x_2}{\Delta} - a, \quad 0 \leq a < 1.$$

Case 2 ($-\tau_1(1-x_1) + m\Delta > 0$):

$$\begin{aligned} \tau &= \sum_{m=m_2}^{m_3} m\Delta - \tau_1(1-x_1) \\ &= \frac{\tau_2 x_2}{2} \left(\frac{\tau_2 x_2}{\Delta} + 1 \right) + \frac{1}{2} \Delta (b-c)(1-b-c) - c\tau_2 x_2, \end{aligned}$$

where

$$\begin{aligned} m_2 &\equiv \left\lceil \frac{\tau_1(1-x_1)}{\Delta} \right\rceil = \frac{\tau_1(1-x_1)}{\Delta} + b, \\ m_3 &\equiv \left\lfloor \frac{\tau_2 x_2 + \tau_1(1-x_1)}{\Delta} \right\rfloor = \frac{\tau_2 x_2 + \tau_1(1-x_1)}{\Delta} - c, \\ c &= \begin{cases} 1+a-b, & a < b, \\ a-b, & b \leq a. \end{cases} \quad 0 \leq b, c < 1. \end{aligned}$$

Case 3 ($-\tau_1(1-x_1) + m\Delta - x_2\tau_2 > 0$):

$$\begin{aligned} \tau &= \tau_2 x_2 \left(\frac{\tau_1}{\Delta} - m_3 - 1 \right) \\ &= \tau_1 \tau_2 \frac{x_1 x_2}{\Delta} - \tau_2 x_2 \left(\frac{\tau_2 x_2}{\Delta} + 1 \right) + c\tau_2 x_2. \end{aligned}$$

Summing these three periods and dividing the sum by the total beat period T , we obtain that both comparators are open the following fraction of the total time:

$$\begin{aligned} f &= x_1 x_2 + \frac{\Delta^2}{2\tau_1 \tau_2} [a(1-a) + b(1-b) - c(1-c)] \\ &= x_1 x_2 + \frac{\Delta^2}{2\tau_1 \tau_2} \varepsilon = x_1 x_2 + \frac{1}{2n(n-1)} \varepsilon, \end{aligned} \quad (5a)$$

$$\varepsilon = 2 \min[a, b] [1 - \max[a, b]], \quad \varepsilon_{\min} = 0, \quad \varepsilon_{\max} = 1/2.$$

If we consider the normalized signals x_i as random variables uniformly distributed between 0 and 1, we can

readily calculate statistical properties of ε :

$$\begin{aligned}\langle \varepsilon \rangle &= \int_0^1 dz \int_0^z dy 2y(1-z) = \frac{1}{12}, \\ \langle \varepsilon^2 \rangle &= \int_0^1 dz \int_0^z dy 4y^2(1-z)^2 = \frac{1}{9}.\end{aligned}\quad (5b)$$

Equations (5) show that this method also gives a result which at $n \gg 1$ approaches the desired law given by Eq. (2).

V. CROSSNET TRAINING SIMULATION

A. Task, network, and training

Our both methods of approximate multiplication may be used for both supervised and unsupervised learning. In this work we have only explored the former option: in order to evaluate their quality, we have used them for training a CrossNet-based feedforward perceptron used for classification of patterns of 3 datasets (cancer1, card1, and diabetes1) from the popular benchmark set Proben1 [12]. The input layer had the number of cells equal to the input vector size (9 for cancer1, 51 for card1, and 8 for diabetes1). The network also had one hidden layer of 10 cells, and 2 output cells. Each output represented one class and WTA (Winner Take All) rule has been used. (The details of suitable CrossNet geometry may be found in [10], however, they have no bearing on the results presented below.) Two different activation functions (the hyperbolic tangent and piecewise-linear dependence) have been used. Although the latter activation function is not differentiable, it presents practical interest, because it can be readily implemented in CMOS circuits.

The network was trained using the error backpropagation technique in which the usual exact signal-by-error multiplication is replaced by the stochastic multiplication described in the previous section, leading to Eq. (4). Before training, all switches were initialized to be randomly ON or OFF with 50% probability. During training, synapses were updated after presenting each pattern of the training set.

Our methods require that the range x_{\max} of each input signal (x_i) and the corresponding reference signal (REF_{*i*}) should be equal. At error backpropagation, one of x_i is feedforward signal, while the other one is feedback error. The feedforward signal is limited by the activation function, so there is no problem with limiting its range to that of REF_{*i*}. However, error backpropagation is usually linear, so that the range of this signal (ε) arriving at the synapses between the input and hidden layer, should be fixed by different means.³ In our calculations, we have limited (“clipped”) both ε and the corresponding RND at $\pm\sigma_\varepsilon$, where σ_ε is the expected r.m.s. value of the error signal. In

³ To the synapses connecting the hidden and output layers, the error signals are strictly limited by certain ε_{\max} , because they are just the differences

order to find σ_ε , we may assume that throughout the training procedure the fraction of latching switches in ON state remains close to the initial 50%. (We have checked that this assumption is indeed confirmed in our numerical experiments.) According to the binomial distribution, the standard deviation of weights from the average in one synapse ($4n^2$ switches) is $\sigma_w = \alpha n$. Then, according to the central limit theorem, $\sigma_\varepsilon = \sqrt{2\alpha n \varepsilon_{\max}}$, because error signals arrive from 2 cells of the output layer.

It is generally accepted that reaching the global minimum of the error function at supervised training does not guarantee the best classification performance on the test set. For example, Prechelt recommended [17] and himself used [12] an earlier training termination, using validation datasets for to optimize the stopping condition. We have used relatively simpler condition. Namely, for the first 300 training epochs,⁴ the validation set error is measured (after every 5th training epoch), and the minimum validation set error is stored. After 300 epochs, training is stopped as soon as the validation error is smaller than that stored value. Otherwise, training is stopped after 1000 epochs.

The second (pseudo-stochastic) method discussed in Sec. IV has been also characterized on the same tasks. For the cancer task, we took $\tau_1 = 50\Delta t$ and $\tau_2 = 40\Delta t$ (i.e. $n = 5$), while to card1 and diabetes1, slightly different parameters $\tau_1 = 40\Delta t$, $\tau_2 = 30\Delta t$ have been used. A lower tunneling rate has been assumed ($\Gamma\Delta t = 10^{-5}$) in order to make the resulting learning rate of the same order as for the stochastic multiplication training. Just as in the second version of the “genuine-stochastic” method, sawtooth waveforms have been shared by all cells of the same layer.

B. Results

Generalization properties of the network depend on the degree of activation function nonlinearity. If the activation functions $y(h)$ per se are fixed (as it was in our study), the effective nonlinearity may be regulated by the preamplifier gain, i.e. by the coefficient α in (1):

$$h_i = \frac{1}{\sqrt{M}} \sum_{j=1}^M w_{ij} x_j \equiv \frac{\alpha}{\sqrt{M}} \sum_{j=1}^M N_{ij} x_j, \quad (6)$$

where M is the number of cells in the previous layer.

In our simulation, CMOL based networks were trained with a few α 's, for 4 different values of n^2 (the number of elementary nanodevices in each synaptic group). The performance is more sensitive to the weight scale α , i.e. cell nonlinearity, rather than to the number $L = 4n^2 + 1$ of discrete levels of synaptic weights.⁵

between the output feedforward signal (limited by the activation function of the output layer) and the binary target values.

⁴ This number has been inferred from the results of [12].

⁵ Surprisingly, the performance of networks with 400 switches per synapse was worse than that for smaller n^2 . This may be due to the fact that quasi-continuous synapses with big α saturates neurons very easily, making the error signal propagation harder. Our plans are to explore this effect further.

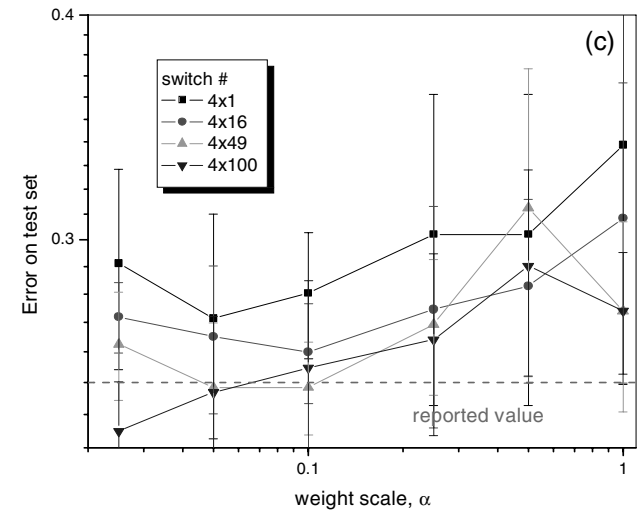
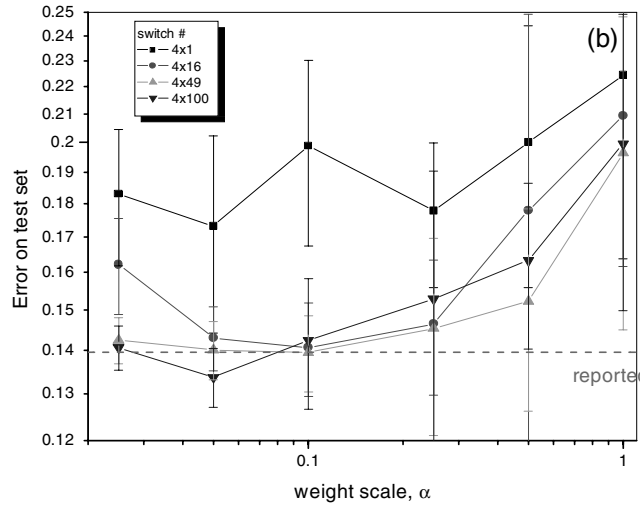
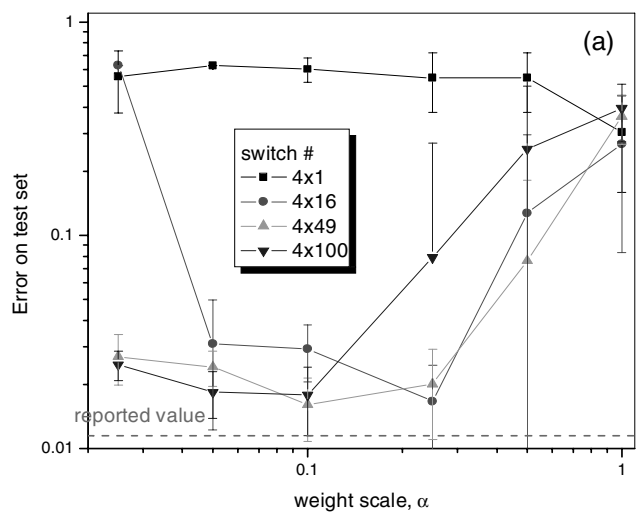
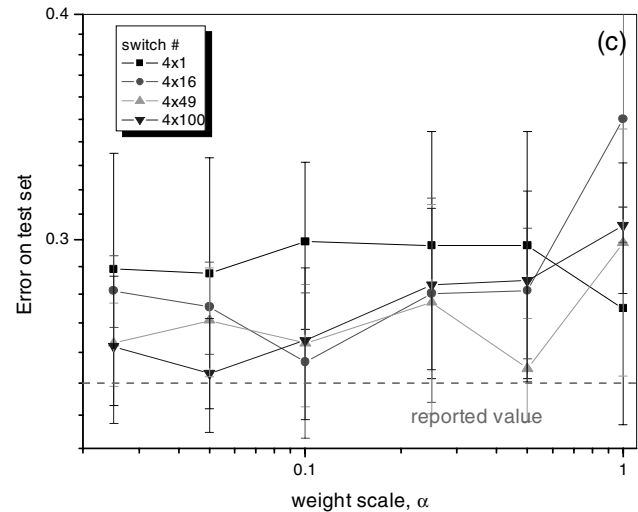
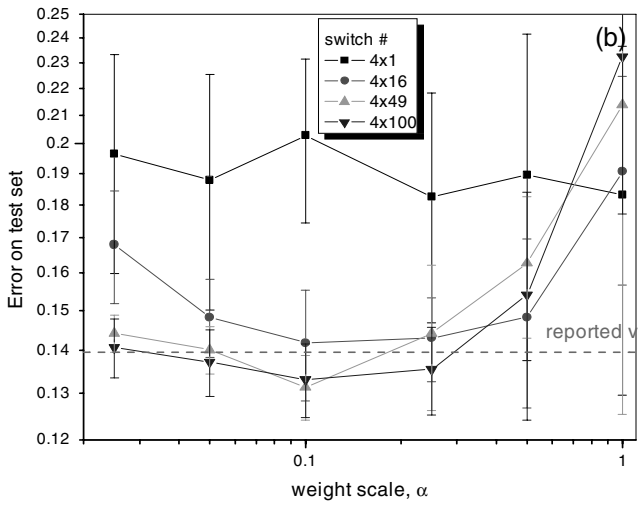
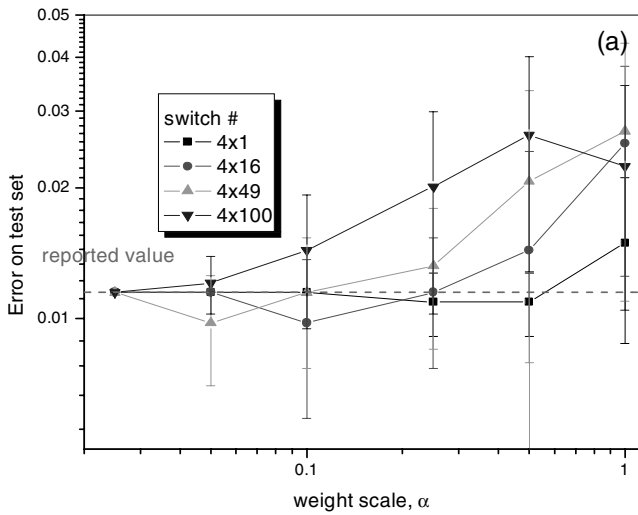


Fig. 4. Classification fidelity of in situ trained CrossNet perceptron with the tanh activation function, for 3 Proben1 datasets: (a) cancer1, (b) card1, and (c) diabetes1, as function of the synaptic weight scale α , for four values of the total number $4n^2$ of latching switches per synapse. For each point, training was carried out 10 times, and statistics calculated. Dashed lines show the best results reported in [12] for multilayered perceptrons with continuous synapses.

Fig.5. Same as Fig. 4, but for the piecewise-linear activation function.

We have used an elementary signal preprocessing – pattern centering:

$$\left(x_i^\mu\right)^{\text{centered}} = x_i^\mu - \frac{1}{P} \sum_{\mu=1}^P x_i^\mu, \quad (7)$$

where P is the number of data vectors in the training set. (For centering test patterns, the averaging is still carried out over the training set.) This procedure has improved results for diabetes1 rather significantly, from ~37% error to ~27%. For other datasets, the initial vectors are already virtually centered, so that the preprocessing has improved the results only marginally.

Figure 4 shows the results with hyperbolic activation function whereas Fig. 5 shows the performance for the piecewise-linear function. Table II summarizes our best results (with 4x16 level synapses). One can see that for all three problems they are at least on a par (and actually better) than those reported in literature for software-implemented networks with deterministic, continuous synaptic weights [12].

TABLE II.

Method performance (test set error) comparison. The results for the continuous backprop training are from Ref. 12 which gives only the best run results (no statistics data).

Problem	Stochastic independent	Stochastic shared	Pseudo-stochastic	Continuous backprop
Cancer1	0.010±0.004	0.010±0.002	0.011±0.002	0.011
Card1	0.14±0.01	0.14±0.01	0.14±0.01	0.14
Diabetes1	0.26±0.02	0.26±0.02	0.26±0.02	0.25

VI. DISCUSSION

We have shown that stochastic and pseudo-stochastic multiplication of analog signals can be implemented by nanoelectronic “CMOL” circuits. This allows CMOL neuromorphic networks (“CrossNets”) to be trained “in situ”, for example by error backpropagation. The initial simulation of such training on several problems of Proben1 benchmark dataset gave a very encouraging result: despite the fact that CMOL CrossNets have binary elementary synapses with finite range, their classification performance after in situ training is quite comparable with that reported for software-based networks with continuous synaptic weights.

We are of course very much aware of the finite value of error backpropagation training. For some classification problems this procedure gives results inferior in comparison with, e.g., the Support Vector Machine algorithm. For us the backprop was rather the simplest testbench for the CMOL CrossNet implementation of the stochastic multiplication for in situ synaptic adaptation. Our results show that this procedure, described by Eq. (4), works very

much similar to the Hebb Rule which will probably remain the key ingredient of all advanced algorithms of neural network learning. The significance of this fact is in the potentially enormous density and speed of CMOL CrossNets.

A natural alternative method of the signal multiplication, e.g., the implementation of the basic rule $\Delta w_{ij} \propto x_1 x_2$, is using spiking neurons [18]. Indeed, assuming that the spike trains, whose rates represent x_1 and x_2 , are statistically independent on time intervals of the order of the average spiking period, the probability of the spikes to coincide in time is proportional to the product of the two rates. With the proper choice of the synaptic switching threshold V_b , this again results in the weight adjustment law expressed by (4), which provide a very natural means for the signal multiplication, given the statistical nature of nanoscale latching switches. One of our future goals is to compare the hardware complexity (say, the minimum area of CMOS circuit per one somatic cell) of this option with the methods analyzed in this report. If spiking neurons may provide a smaller area, using such somas may be the natural choice for the further CMOL CrossNet development.

ACKNOWLEDGMENTS

Important contributions by Xiaolong Ma, as well as valuable comments and suggestions by P. Adams, J. Barhen and D. Hammerstrom, are gratefully acknowledged.

REFERENCES

- [1] G. Cuniberti, G. Fagas, and K. Richter (eds.), *Introducing Molecular Electronics*, Berlin: Springer, 2005.
- [2] M. R. Stan, P. D. Franzon, S. C. Goldstein, J. C. Lach, and M. M. Ziegler, “Molecular electronics: From devices and interconnect to circuits and architecture,” *Proc. IEEE*, vol. 91, pp. 1940-1957, Nov. 2003.
- [3] K. K. Likharev, “Electronics below 10 nm”, in: J. Greer *et al.* (eds.), *Nano and Giga Challenges in Microelectronics*, Amsterdam: Elsevier, 2003, pp. 27-68.
- [4] K. K. Likharev and D. B. Strukov, “CMOL: Devices, circuits, and architectures,” in *Introducing Molecular Electronics*, G. Cuniberti, G. Fagas, and K. Richter (eds.) Berlin: Springer, 2005, pp. 447-477.
- [5] D. B. Strukov and K. K. Likharev, “Prospects for terabit-scale nanoelectronic memories,” *Nanotechnology*, vol. 16, pp. 137-148, Jan. 2005.
- [6] —, “CMOL FPGA: A cell-based, reconfigurable architecture for hybrid digital circuits using two-terminal nanodevices,” *Nanotechnology*, vol. 16, pp. 888-900, Apr. 2005.
- [7] —, “A Reconfigurable Architecture for hybrid CMOS/nanodevice circuits”, in *Proc. FPGA '2006*, pp. 131-140.
- [8] N. Zhitenev, W. Zhang, A. Erbe, Z. Bao, E. Garfunkel, D. M. Tennant, and R. A. Cirelli, “Control of topography, stress and diffusion at molecule–metal interfaces”, *Nanotechnology*, vol. 17, pp. 1272-1277, Mar. 2006.
- [9] Ö. Türel, J. H. Lee, X. L. Ma, and K. K. Likharev, “Neuromorphic architectures for nanoelectronic circuits,” *Int. J. Circ. Theory App.*, vol. 32, pp. 277-302, Sep/Oct. 2004.
- [10] J. H. Lee and K. K. Likharev, “CMOL CrossNets as pattern classifiers,” *Lecture Notes in Computer Science*, vol. 3512, pp. 446-454, June 2005.
- [11] —, “Defect tolerance of CrossNet MLP classifiers”, paper in preparation.

- [12] L. Prechelt, "Proben1 - A set of neural network benchmark problems and benchmarking rules", *Technical Report 21/94*, Fakultät für Informatik, U. Karlsruhe, 1994.
- [13] S. Fölling, Ö Türel, and K. K. Likharev, "Single-electron latching switches as nanoscale synapses", in: *Proc. IJCNN'2001*, pp. 216-221.
- [14] D.V. Averin and A.N. Korotkov, "Correlated single-electron tunneling via mesoscopic metal particles: effects of energy quantization", *J. Low Temp. Phys.* vol. 80, pp. 173-185, Aug. 1990.
- [15] Y. Kondo and Y. Sawada, "Functional abilities of a stochastic logic neural network", *IEEE Trans. on Neural Networks*, vol. 3, pp. 434-443, May 1992.
- [16] E. Oja, "A simplified neuron model as a principal component analyzer", *J. Math. Biology*, vol. 15, pp. 267-273, 1982.
- [17] L. Prechelt, "Automatic early stopping using cross validation: quantifying the criteria", *Neural Networks*, vol. 11, pp. 761-767, June 1998.
- [18] C. Mead, *Analog VLSI and Neural Systems*, Reading, MA: Addison-Wesley, 1989, p. 198.