Check for updates

Initial state-dependent implementation of logic gates with memristive neurons

Franciska Rajki,[1] András Horváth,[1] Alon Ascoli,[2,✉] and Ronald Tetzlaff[3]

[1]*Faculty of Information Technology and Bionics, Peter Pazmany Catholic University, Budapest, Hungary*

[2]*Department of Electronics and Telecommunications, Politecnico di Torino, Turin, Italy*

[3]*Institute of Circuits and Systems, TUD | Dresden University of Technology, Dresden, Germany*

✉ E-mail: alon.ascoli@polito.it

This study introduces a simple memristor cellular neural network structure, a minimalist configuration with only two cells, designed to concurrently address two logic problems. The unique attribute of this system lies in its adaptability, where the nature of the implemented logic gate, be it AND, OR, and XOR, is determined exclusively by the initial states of the memristors. The memristors' state, alterable through current flow, allows for dynamic manipulation, enabling the setting of initial conditions and consequently, a change in the circuit's functionality. To optimize the parameters of this dynamic system, contemporary machine learning techniques are employed, specifically gradient descent optimization. Through a case study, the potential of leveraging intricate circuit dynamics is exemplified to expand the spectrum of problems solvable with a defined number of neurons. This work not only underscores the significance of adaptability in logical circuits but also demonstrates the efficacy of memristive elements in enhancing problem-solving capabilities.

Introduction: Logical functions assume a pivotal role in contemporary problem-solving, necessitating the integration of versatile and reusable circuit elements within modern and emerging circuits.

Neural networks facilitate the resolution of diverse and intricate problems by constructing elaborate structures from fundamental building blocks. While neural networks demonstrate proficiency in addressing specific problems for which they are trained, contemporary artificial intelligence tends to exhibit a narrow and singular purpose.

Over the preceding decade, an escalating introduction of increasingly complex neural networks has yielded enhanced performance across widely explored benchmark datasets. This discernible trend highlights that augmenting network complexity commonly arises from the augmentation of neurons and layers within their architectural framework. However, elevated complexity can also be attained through the enhancement of cellular dynamics.

The evolutionary trajectory of neural networks in recent years showcases the practical efficacy of complex architectures featuring millions of processing elements and parameters. Notably, these parameters can be finely tuned through the application of modern machine learning algorithms, underscoring the adaptability of neural networks in addressing practical problems.

A discernible trend in the past decade has been the escalating complexity of neural networks. In 2012, computer vision predominantly utilized neural networks with 8 layers, exemplified by Alexnet [1]. Subsequent advancements saw an increase to 19 layers in VGG in 2014 [2] and a remarkable elevation to 152 layers in residual networks in 2016 [3], indicating an ongoing trajectory of network complexity augmentation.

A parallel augmentation of complexity is observed in natural language processing models. GPT-2, a transformer network, employed 48 layers and 1.5 billion parameters in 2019 [4]. In a remarkable stride, GPT-3 in 2020 increased this complexity to 96 layers and a staggering 175 billion parameters [5] within a mere year.

An alternative avenue for enhancing neural network capabilities lies in augmenting the complexity of individual building blocks. The incorporation of cells with memory and higher-dimension state-space representations, embedding higher-order dynamics, holds promise for solving challenging non-linearly separable problems. Such problems often necessitate the execution of numerous sequential operations through a singular operation. Ideally, these extended dynamics should not be implemented using multiple simple building blocks to maintain efficiency. Instead, they should be realized through basic circuit elements.

The memristor emerges as a promising element for efficiently introducing both memory and non-linearity into circuits. Demonstrated in previous works [6, 7], memristive dynamics offer energy- and time-efficient solutions in circuit design. This study delves into a fundamental problem from a similar perspective, exploring the potential of leveraging memristive elements for efficient and effective circuit design.

The XOR problem, featuring linearly inseparable input-output pairs, stands out as a renowned elementary problem. Logic gates, including AND, OR, and XOR, serve as essential tools for describing complex and arbitrary algorithmic problems. Enhancing the efficiency of logic gates could prove advantageous in addressing a diverse array of challenging problems.

Numerous neural network-based solutions for the XOR gate have been proposed in the literature, encompassing single neural solutions with complex values [8], spiking neural networks [9], and efficient solutions with memristive systems [10, 11]. This problem gains particular significance as it eludes resolution with two neurons in a simple fully-connected network, employing either Hebbian or gradient-based optimization [12], or within a standard CellNN consisting of two cells [13].

The human nervous system, with its inherently general architecture, adeptly tackles various tasks relying on the input and the initial state to determine a solution for a given problem. An ideal scenario in biomorphic chip design envisions a similar case, where memristive circuits exhibiting programmability, enable the implementation of diverse logic gates without necessitating changes to the system's wiring, solely by manipulating the initial state.

In this study, we demonstrate resolution of AND, OR, and XOR logic gates by a basic CellNN comprising two memristive cells. The circuit's functionality is uniquely dictated by the initial state of the memristors. To achieve this, we employ a gradient-based optimization technique to train both the network weights and the memristor parameters, encompassing their initial states.

Our exploration demonstrates how the incorporation of complex circuit dynamics expands the range of problems that can be efficiently solved, even with a limited number of neurons. By examining a well-known problem, we aim to underscore the vast potential inherent in such structures. The study emphasizes the adaptability and efficacy of leveraging complex circuitry to extend the problem-solving capabilities of neural networks.

Cell dynamics of a memristive cellular neural network: The subject of our investigation is a memristive cellular neural network (M-CellNN) [14] comprising two cells.

Prior studies have demonstrated that such networks exhibit emergent behaviour [15, 16], as well as complex and chaotic dynamics [17, 18], even with just two conventional cell structures [19, 20], and [21]. However, these dynamics alone could not offer a solution to the XOR problem.

The solution we propose is based on continuous-time dynamics that emerge in an M-CellNN, featuring two analogue neurons, each with two degrees of freedom. Specifically, the states of cell $j \in \{1, 2\}$ encompass its capacitor voltage x_j and its memristor state m_j. The dynamics of the capacitor C_{x_j} in cell $C(j)$ can be succinctly expressed by the following equation:

$$\dot{x_j} = -x_j + a_0 y_j + b_0 u_j - i_{m,j} + a_{\zeta_i} y_i + b_{\zeta_i} u_i + z \tag{1}$$

with $(j, i) \in \{(1, 2), (2, 1)\}$, and $\zeta_i \in -(+)1$ for $i = 1(2)$. Here y_j is the output of the cell $C(j)$ which is defined by the standard non-linearity $f(x_j)$ via:

$$y_j = f(x_j) = \frac{1}{2}\left|x_j + 1\right| - \frac{1}{2}\left|x_j - 1\right| \tag{2}$$

In (1) z denotes the bias parameter, whereas a_{-1}, a_0 and a_{+1} (b_{-1}, b_0 and b_{+1}), known as feedback (feedforward) synaptic weights, are represented in compact form via the feedback (feedforward) template, defined as $\underline{\underline{\mathbf{A}}} = [a_{-1}, a_0, a_{+1}]$ ($\underline{\underline{\mathbf{B}}} = [b_{-1}, b_0, b_{+1}]$). In particular, a_{-1} and

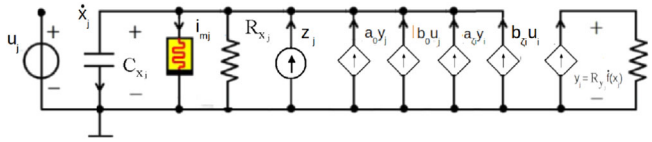


Fig. 1 *Schematic of the circuit of cell $C(j)$, including the dependent sources controlled by the cell $C(i)$. Here $(j, i) \in \{(1, 2), (2, 1)\}$ and $\zeta_i = -(+)1$ for $i = 1(2)$. Here $C_{x_j} = 1F$ and $R_{x_j} = R_{y_j} = 1\ \Omega$*

b_{-1} (a_{+1} and b_{+1}) respectively are the feedback and feedforward synaptic weights by which cell C(1) (C(2)) acts on the dynamics of cell C(2) (C(1)). Finally a_0 and b_0 respectively are self feedback and self feedforward weights.

$\underline{\underline{\mathbf{A}}}$, $\underline{\underline{\mathbf{B}}}$ and z determine the dynamic behaviour, i.e. the mapping of inputs and initial states of the network onto the respective outputs, and are usually referred to as programming templates.

CellNNs have proven efficiency across diverse applications [22]. It has been established, that a two-cell array can exhibit chaotic dynamics. However, the utilization of a straightforward, M-CellNN for implementing any logic gate from a number of options, including especially the XOR function has not been explored previously.

The current i_{m_j} through the memristor M_j in cell $C(j)$ is expressed according to Ohm's law as:

$$i_{m_j} = G(m_j, x_j) \cdot x_j, \tag{3}$$

where the evolution of the state m_j of this voltage-controlled first-order extended memristor is governed by the following dynamics:

$$\dot{m}_j = g(m_j, x_j) \tag{4}$$

Here x_j represents the voltage across the memristor and m_j denotes the state of the memristor. The initial state and initial voltage of the memristor are denoted by $m_j(0)$ and $x_j(0)$, respectively.

In accordance with Leon Chua's latest memristor classification, g and G characterize an extended memristor [23, 24]. By employing Chua's Unfolding Principle [25], these functions are approximated through third-order polynomials:

$$\begin{aligned}\dot{m}_j = g(m_j, x_j) =& \alpha_0 + \alpha_1 \cdot x_j + \alpha_2 \cdot x_j^2 + \alpha_3 \cdot x_j^3 + \\ &+ \beta_1 \cdot m_j + \beta_2 \cdot m_j^2 + \beta_3 \cdot m_j^3 + \\ &+ \gamma_1 \cdot x_j \cdot m_j + \gamma_2 \cdot x_j \cdot m_j^2 + \gamma_3 \cdot x_j^2 \cdot m_j\end{aligned} \tag{5}$$

$$\begin{aligned}G(m_j, x_j) =& \delta_0 + \delta_1 \cdot x_j + \delta_2 \cdot x_j^2 + \delta_3 \cdot x_j^3 + \\ &+ \epsilon_1 \cdot m_j + \epsilon_2 \cdot m_j^2 + \epsilon_3 \cdot m_j^3 + \\ &+ \phi_1 \cdot x_j \cdot m_j + \phi_2 \cdot x_j \cdot m_j^2 + \phi_3 \cdot x_j^2 \cdot m_j\end{aligned} \tag{6}$$

The circuit schematic of cell $C(j)$, incorporating the influence of the neighbouring cell C_i on its dynamics, is depicted in Figure 1. The memristive cell introduces only one additional circuit element compared to a standard CellNN cell—a memristor in parallel to the capacitor. This singular addition does not significantly amplify power consumption or manufacturing complexity. Nevertheless, it substantially broadens the spectrum of implementable functions, as elucidated in [26] and further demonstrated in this study.

Within this two-cell array, both the feedback synaptic template $\underline{\underline{\mathbf{A}}}$ and the feedforward synaptic template $\underline{\underline{\mathbf{B}}}$ incorporate three non-zero coefficients each. Coupled with the bias parameter z, the total number of tunable parameters reaches seven. Simultaneously, the properties of each memristor are delineated by an additional set of 20 parameters. While presuming perfectly matched memristive devices in the cells, we introduce variability solely in their initial memristive states, representing the only parameter distinct in the implementation of different logic gates. All other parameters remain consistent across all scenarios. Consequently, this introduces two additional parameters for each logic gate, and as we investigate three distinct logic gates (AND, OR, XOR), this results in an additional six parameters. Thus, the overall system is characterized by 33 parameters, encompassing three distinct problems and functionalities. Each functionality involves the application of 29 parameters, with 27 of these shared among the three problems.

Table 1. This table describes the representation of inputs and outputs for the investigated logic gates (AND, OR, and XOR). The input voltage u_j to cell $C(j)$, $j \in \{1, 2\}$ is either 0 (false) or 1 (true) and denotes one of the input operands. The sum of the cell's output voltages is either -2 (false) or 2 (true), and represents the result. The output of the M-CellNN, i.e. $y_1 + y_2$, where y_j is the output voltage of cell $C(j)$, matches the expected output of the operations, if we consider output cell values above zero as true and those below zero as false

Input 1	Input 2	AND	OR	XOR	M-AND	M-OR	M-XOR
0	0	−2	−2	−2	−2	−2	−2
0	1	−2	2	2	−2	2	2
1	0	−2	2	2	−2	2	2
1	1	2	2	−2	2	2	−2

The extensive nature of this parameter set renders it impractical for direct exploration using exhaustive or grid search methodologies. Therefore, we have chosen to employ a machine learning algorithm for parameter optimization. This strategy proves particularly beneficial when dealing with more intricate and practical problems that necessitate larger CellNNs, resulting in a linear augmentation of the parameter count with the number of cells.

Training of the network parameters: The implemented network utilized PyTorch [27], where both the programming templates and the parameters of the memristor model underwent the training process.

Training involved simulating the continuous-time differential equations on digital hardware, employing the Euler formula with the TorchDiffEq module [28]. The PyTorch code for training the network parameters and a straightforward Python script for testing the optimized network can be accessed in [29].

For the XOR gate, there are four possible input-output pairs, as detailed in Table 1. In instances of a false input on cell i, u_i was set to zero; conversely, for true inputs, it was set to 1. The overall output of the computation was determined as the sum of the cells' outputs, which could range from -2 (false) to 2 (true) due to the standard CellNN non-linearity constraints. System dynamics were allowed to evolve for 100 iterations, with a timestep of 0.01 s, ensuring an accurate integration of the continuous-time differential equation, evaluating the global output and comparing it to the expected XOR operation output.

The training spanned 3000 iterations, utilizing the Adam optimizer [30] with an initial step size of 0.01. Initial parameter values were randomly generated from a standard normal distribution. The ℓ_2 loss function guided the network training, assessing the disparity between actual and expected outputs at the conclusion of each training iteration.

We conducted training on the network using all four possible input pairs for the three problems. Theoretically, no stimulus pair other than the twelve specified in Table 1 is permissible.

To enhance the robustness of the network outputs, we implemented a data augmentation strategy by introducing uniform noise $\xi \in [0, 0.1]$ into each input value. Consequently, the inputs were permitted to vary within $[-0.1, 0.1]$ for the false case and within $[0.9, 1.1]$ for the true case.

After the execution of the training, the programming templates were found to assume the following form:

$$\begin{aligned}\mathbf{A} &= \begin{bmatrix}-0.7379 & 1.1183 & -0.8916\end{bmatrix}, \\ \mathbf{B} &= \begin{bmatrix}-1.5312 & 0.6234 & -1.2246\end{bmatrix}, \\ z &= -1.9875\end{aligned} \tag{7}$$

1350911x, 2024, 11, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ell2.13172 by Politecnico Di Torino Sist. Bibl Del Polit Di Torino, Wiley Online Library on [03/10/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

The differential equations governing the dynamics of the capacitor voltages were found to be expressed by

$$\begin{aligned}\dot{x}_1 = &- x_1 + 1.1183y_1 + -0.8916y_2 \\ &0.6234u_1 + -1.2246]0u_2 - 1.9875 - i_{m1},\end{aligned} \tag{8}$$

and

$$\begin{aligned}\dot{x}_2 = &- x_2 + -0.7379y_1 + 1.1183y_2 \\ &+ 0.6098u_1 + -1.5312u_2 - 1.9875 - i_{m2},\end{aligned} \tag{9}$$

for cells $C(1)$ and $C(2)$ respectively.

The values of the optimized memristor model parameters turned out to be as follows: $\alpha_0 = -0.9602$, $\alpha_1 = -0.6280$, $\alpha_2 = -2.1831$, $\alpha_3 = 1.5359$, $\beta_1 = 1.6686$, $\beta_2 = 1.2363$, $\beta_3 = -1.1509$, $\gamma_1 = 0.5227$, $\gamma_2 = -1.8543$, $\gamma_3 = -0.0450$, and $\delta_0 = -1.2040$, $\delta_1 = -1.2903$, $\delta_2 = -1.5158$, $\delta_3 = -0.2210$, $\epsilon_1 = 0.6489$, $\epsilon_2 = 0.8607$, $\epsilon_3 = -0.8444$, $\phi_1 = 0.2612$, $\phi_2 = -0.4397$, $\phi_3 = 1.0842$.

The states of the memristors of the two neurons for the different logic gates were initialized as follows: AND gate: $m_1(0) = 0.4841$, $m_2(0) = -0.0494$; OR gate: $m_1(0) = -0.6606$, $m_2(0) = 0.3078$, XOR gate: $m_1(0) = -0.4568$, $m_2(0) = -0.2965$. The initial values of the capacitor voltages $x_1(0)$ and $x_2(0)$ were set as the values of the inputs u_1 and u_2 respectively.

We have repeated parameter training multiple times and different optimized parameters were obtained in each training, thereby indicating the existence of multiple potential solutions to this problem.

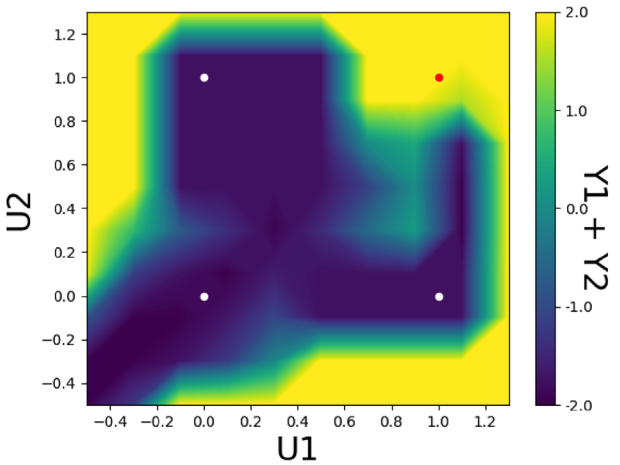
Results: We conducted training for both the programming templates and the parameters of the memristor model. The average ℓ_2 distance decreased to 0.0093 after the final training iteration. While this value indicates close proximity to the ideal solution, the fact that it is not zero suggests that perfection has not been achieved. In practical terms, this accuracy can be easily enhanced by applying thresholding to the actual outputs of the cells and converting them to a digital format. This adjustment would yield correct XOR outputs in all four cases.

As a point of reference, we implemented a classical CellNN featuring two non-memristive cells. This reference model struggled to solve the problem and could only handle either the OR or the AND problem individually, as they are linearly separable. The lowest ℓ_2 distance, averaged over the four possible input-output pairs attainable during training, was 0.724. This comparison demonstrates that the integration of two memristors into such a simple circuit is adequate to expand its functionalities. Furthermore, we aim to illustrate how memristors facilitate the modulation of the decision boundary in the traditional CellNN, allowing for the seamless execution of the XOR Boolean logic operation between two operands and switching between logical operations by altering only the initial state of the system.

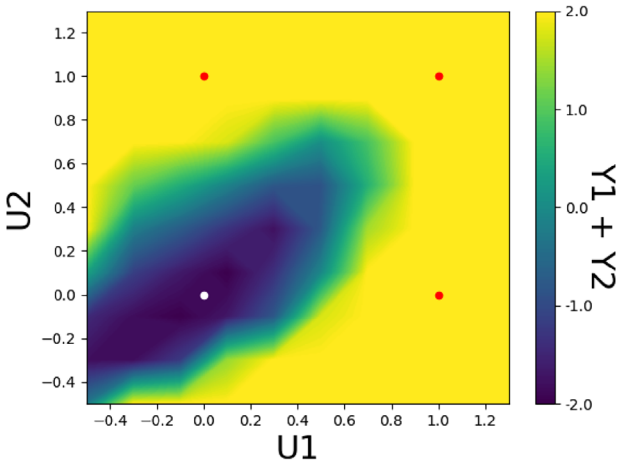
Beyond merely extending the circuit's functionality to solve the XOR problem, which is unfeasible for the linear implementation, memristors enable the creation of a dynamic network. This network maintains constant connection weights, but the initial condition of the network can be set to achieve any of the three desired functionalities.

Decision boundaries hold significance from two distinct perspectives. In practical applications, merely deriving a theoretical solution may prove inadequate. Real circuits are invariably affected by noise and parasitic elements. Therefore, ensuring a solution for each theoretically admissible input is not sufficient for a robust design. The system must yield correct solutions even in scenarios where inputs deviate slightly from their nominal values. Examination of decision boundaries unveils the extent to which parameters can change without affecting the digitized, thresholded outputs.

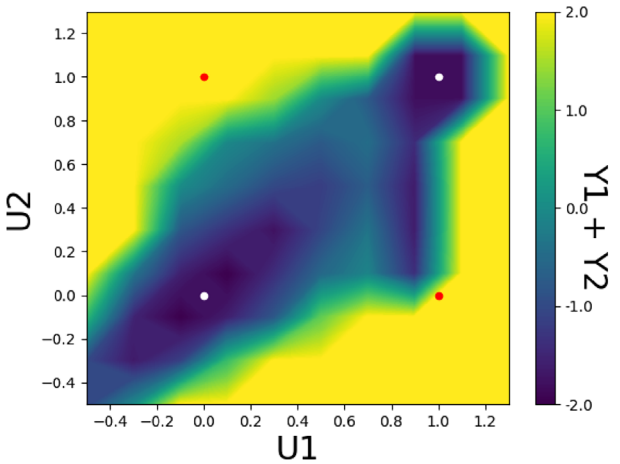
From another viewpoint, decision boundaries can unveil the complexity of the system. Linear systems exhibit simple linear decision boundaries, limiting their utility for most practical applications. The incorporation of higher nonlinearities into a network results in a more intricate and complex decision boundary, which is a crucial element for solving complex problems. One can also notice how altering the initial conditions has influenced the decision boundaries of the system.



(a) Decision boundary for AND operation



(b) Decision boundary for OR operation



(c) Decision boundary for XOR operation

Fig. 2 *This diagram shows the output ($y_1 + y_2$) of a M-CellNN for all possible input combinations where $u_j \in [-0.4, 1.2]$, $\Delta u_j = 0.05$, $j \in \{1, 2\}$. The colors visualize the output according to the legend. At the positions of white (red) markers the system should emit low (high) output values*

Figure 2(a)–(c) illustrates the AND, OR and XOR outputs accordingly $y_1 + y_2$ for every possible input pair (u_1, u_2) with $u_j \in [-0.4, 1.2]$, $\Delta u_j = 0.05$, $j \in \{1, 2\}$, for our M-CellNN (for the memristive two-cell array).

In the case of the traditional CellNN, a linear boundary separates the regions containing the red and white markers, hindering the two-cell array from solving the XOR logical Boolean problem.

As these illustrations demonstrate, our network adeptly computes the correct output for each of the three investigated problems and

1350911x, 2024, 11, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ell2.13172 by Politecnico Di Torino Sist. Bibl Del Polit Di Torino, Wiley Online Library on [03/10/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

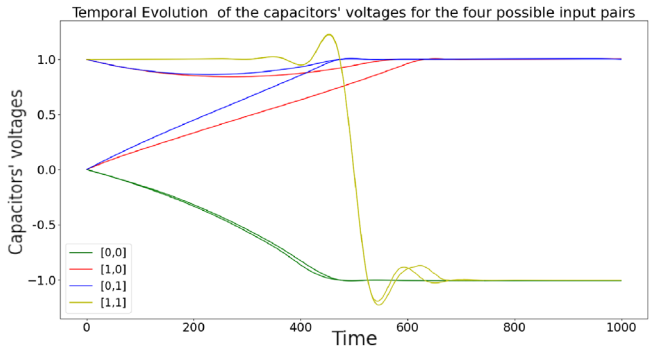


Fig. 3 *Evolution of the states x_1 and x_2 of the M-CellNN for the four possible input pairs for the XOR operation. The state evolution for the two other logic gates are not reported for the sake of space and simplicity but their simulations resulted in similar figures*

all four possible input pairs. Notably, the memristive system exhibits an intricate decision boundary, enabling the two-cell array to accurately compute the XOR solution for each of the four possible input combinations—a feat unattainable with a linear system. Moreover, the design showcases robustness as all of the outputs fall considerably far from the decision boundaries except the input of $u_1 = 1$ and $u_2 = 0$ for the XOR gate, which falls close to the decision boundary, but even this input case provides the expected output for the desired operation.

The intricate decision boundary map is shaped by the rich nonlinear dynamics of the two memristors, playing a pivotal role in enabling the two-cell array to solve the AND, OR, and XOR problems without modifying the connection weights in the network, showcasing the significance of altering only the initial conditions.

Given that our network is a dynamic system, it is crucial to scrutinize how its internal state evolves over time during computation. We have explored the progression of the network states (x_1, x_2) for all possible input pairs across all three logical operations. The temporal evolution of the cells' outputs y_1 and y_2 can be directly inferred from the time course of the capacitors' voltages x_1 and x_2, respectively. The dynamics of the capacitors' voltages are illustrated in Figure 3. As it is depicted, the output states align with the anticipated outputs of the logical operations. The state evolution for the other two problems is not provided due to space constraints, given that those problems are comparatively simpler than the implementation of the XOR problem.

Conclusions: In this paper, we have successfully showcased the capability of a memristive cellular neural network with two ideal memristive cells to execute the AND, OR, and XOR operations between two inputs. The resulting output solutions for the respective problems solely depend on the initial states of the memristors, which can ideally be modified by the current flowing through them.

It is noteworthy that our implementation marks the first instance where the functionalities of memristors are altered based on their initial conditions, enabling their application for various logical operations. Importantly, the XOR task, unattainable for a traditional 2×1 CellNN, underscores the potential of enriching the dynamical properties of neurons to expand the range of functions implementable by a given neural network. This expansion can lead to more efficient hardware solutions for complex problems.

It is essential to clarify that our simulations involved the training of parameters in an idealized M-CellNN setup, free from noise, with all circuit elements and connections assumed to be ideal. Additionally, our memristor is defined according to Chua's unfolding principle, implementing ideal and optimized memristor characteristics for this task. Because of this the investigated scenarios might not completely be feasible with existing memristive devices. The translation of similar functionalities to real devices necessitates further investigation and exploration.

Author contributions: All the authors equally contributed to the scientific work presented in this manuscript.

Acknowledgments: The support of the Alfréd Rényi Institute of Mathematics and the following grants: 2018-1.2.1-NKP00008: Exploring the Mathematical Foundations of Artificial Intelligence and TKP2021_02-NVA-27 – Thematic Excellence Program are gratefully acknowledged.

Conflict of interest statement: The authors declare no conflicts of interest.

Data availability statement: No data is available in the manuscript.

© 2024 The Authors. *Electronics Letters* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
Received: *1 December 2023* Accepted: *22 March 2024*
doi: 10.1049/ell2.13172

References

1 Krizhevsky, A., Nair, V., Hinton, G.: The cifar-10 dataset. https://www.cs.toronto.edu/~kriz/cifar.html (2014). Accessed 8 Apr 2024
2 Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (2014)
3 He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern recognition, pp. 770–778 (2016)
4 Radford, A., et al.: Language models are unsupervised multitask learners. *OpenAI blog* **1**(8), 9 (2019)
5 Brown, T., et al.: Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **33**, 1877–1901 (2020)
6 Bearden, S.R., Pei, Y.R., Di Ventra, M.: Efficient solution of Boolean satisfiability problems with digital memcomputing. *Sci. Rep.* **10**(1), 1–8 (2020)
7 Ascoli, A., et al.: Graph coloring via locally-active memristor oscillatory networks. *J. Low Power Electron. Appl.* **12**(2), 22 (2022)
8 Nitta, T.: Solving the XOR problem and the detection of symmetry using a single complex-valued neuron. *Neural Networks* **16**(8), 1101–1105 (2003)
9 Reljan-Delaney, M., Wall, J.: Solving the linearly inseparable xor problem with spiking neural networks. In: 2017 Computing Conference, pp. 701–705. IEEE, Piscataway, NJ (2017)
10 Vourkas, I., et al.: Memristor-based logic circuits. In: *Memristor-Based Nanoelectronic Computing Circuits and Architectures*, pp. 61–100. Springer, Cham (2016)
11 Horváth, A., Ascoli, A., Tetzlaff, R.: Implementation of the XOR gate with two memristive neurons. In: 2023 12th International Conference on Modern Circuits and Systems Technologies (MOCAST), pp. 1–5. IEEE, Piscataway, NJ (2023)
12 Hamey, L.G.: XOR has no local minima: a case study in neural network error surface analysis. *Neural Networks* **11**(4), 669–681 (1998)
13 Chua, L.O., Roska, T.: The CNN paradigm. *IEEE Trans. Circuits Syst. I: Fundam. Theory Appl.* **40**(3), 147–156 (1993)
14 Ascoli, A., et al.: System-theoretic methods for designing bio-inspired mem-computing memristor cellular nonlinear networks. *Front. Nanotechnol.* **3**, 633026 (2021)
15 Ascoli, A., et al.: Edge of Chaos Is Sine Qua Non for Turing Instability. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **69**(11), 4596–4609 (2022)
16 Ascoli, A., et al.: Edge of chaos theory resolves Smale paradox. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **69**(3), 1252–1265 (2022)
17 Tetzlaff, R., et al.: Theoretical foundations of memristor cellular nonlinear networks: memcomputing with bistable-like memristors. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **67**(2), 502–515 (2019)
18 Ascoli, A., et al.: Theoretical foundations of memristor cellular nonlinear networks: stability analysis with dynamic memristors. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **67**(4), 1389–1401 (2019)
19 Dellnitz, M., et al.: Cycling chaos. *IEEE Trans. Circuits Syst. I: Fundam. Theory Appl.* **42**(10), 821–823 (1995)
20 Solving the XOR problem and the detection of symmetry using two memristive neurons. IEEE Trans. Circuits Syst. I: Regul. Pap. (2022)
21 Ascoli, A., et al.: Edge of chaos explains Prigogine's instability of the homogeneous. *IEEE J. Emerging Sel. Top. Circuits Syst.* **12**(4), 804–820 (2022). https://doi.org/10.1109/JETCAS.2022.3221156

1350911x, 2024, 11, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ell2.13172 by Politecnico Di Torino Sist. Bibl Del Polit Di Torino, Wiley Online Library on [03/10/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

22 Horváth, A., et al.: Cellular neural network friendly convolutional neural networks–CNNs with CNNs. In: Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 145–150. IEEE, Piscataway, NJ (2017)
23 Corinto, F., Civalleri, P.P., Chua, L.O.: A theoretical approach to memristor devices. *IEEE J. Emerging Sel. Top. Circuits Syst.* **5**(2), 123–132 (2015)
24 Ascoli, A., Corinto, F., Tetzlaff, R.: Generalized boundary condition memristor model. *Int. J. Circuit Theory Appl.* **44**(1), 60–84 (2016)
25 Ascoli, A., et al.: Unfolding the local activity of a memristor. In: 2014 14th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA), pp. 1–2. IEEE, Piscataway, NJ (2014)
26 Ascoli, A., et al.: Theoretical foundations of memristor cellular nonlinear networks: a dRM 2-based method to design memcomputers with dynamic memristors. *IEEE Trans. Circuits Syst. I: Regul. Pap*. **67**(8), 2753–2766 (2020)
27 Paszke, A., et al.: Automatic differentiation in pytorch (2017).
28 Chen, R.T., et al.: Neural ordinary differential equations. arXiv:1806.07366 (2018)
29 Horváth, A.: Memristive cellular neural networks in pytorch. https://github.com/horan85/memristive_cnn (2023). Accessed 21 Feb 2023
30 Zhang, Z.: Improved Adam optimizer for deep neural networks. In: *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1–2. IEEE, Piscataway, NJ (2018)

1350911x, 2024, 11, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ell2.13172 by Politecnico Di Torino Sist. Bibl Del Polit Di Torino, Wiley Online Library on [03/10/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License