Supplementary Material

Simulation details

In this work, the same parameters employed in [1] were used for the Monte Carlo simulations of the Stochastic Resistor Network.

The network is structured as an undirected weighted graph, resembling a cubic lattice with $L_x = 27 \times L_y = 42 \times L_z = 3$ nodes. Each link in this model, representing a coarse-grained abstract of a small portion of the gold film, can exhibit one of four conductance values: a vanishing insulating level $\sigma_\alpha = 10^{-10}\Omega^{-1}$ and three discrete conducting levels, $\sigma_\beta = 0.01\,\Omega^{-1}$, $\sigma_\gamma = 0.02\,\Omega^{-1}$, and $\sigma_\delta = 0.04\,\Omega^{-1}$. The initial coverage is random and beyond percolation threshold. Similar to the experiments, the simulated network is subjected to a constant overall applied voltage ΔV_{tot}. At each MC step, we utilize the Spectral Theory of Laplacian Matrix, as described in [2, 3], to solve the network's circuit and determine the effective resistance R_{tot} and voltage ΔV_{ij} across node pairs $\{i, j\}$. This method resembles traditional Circuit Theory nodal analysis but extends the basic Kirchhoff equation calculations, offering an efficient approach to manage current redistribution in large systems. Following this approach, at each step, we compute the Laplacian matrix L associated with the network. Subsequently, we derive R_{tot} by computing the Moore-Penrose pseudo-inverse L^+. We then use Ohm's Law to compute the total current I_{tot}. Individual link voltages ΔV_{ij} are then determined via nodal potential methods, which rely on the total computed current and the spectral decomposition of L. After this computation, we proceed to probabilistically update each link conductance $\sigma_{\{ij\}}$. Each link conductance $\sigma_{\{ij\}}$is probabilistically updated depending on 1) its dissipated power $W_{ij}{}^{(d)} = \frac{\Delta V_{ij}{}^2}{R_{ij}}$(this may decrease $\sigma_{\{ij\}}$) and 2) the power dissipated by its neighbors $W_{ij}{}^{(a)} \sim \frac{\Sigma_{N_{neigh}(kl)} \Delta V_{kl}{}^2}{R_{kl}}$ (this may increase $\sigma_{\{ij\}}$). Moreover, each ΔV_{ij} is compared with a threshold voltage, which determines whether a downgrade/upgrade of σ_{ij} has to be attempted.

The model was implemented in C++ language (http://www.cplusplus.com/reference/). High-Performance Computing (HPC) resources were needed, due to the matrix size (approximately $N \times N \approx 10^7$ elements) and simulations spanning thousands of MC steps. We utilized the Armadillo library in C++ (https://arma.sourceforge.net/) for matrix calculations, leveraging its built-in parallelization with OpenMP to efficiently perform Spectral Theory computations.

To analyze the results of the simulations, we used tools from the NetworkX package [8] and custom Python libraries.

Receptron Setup

The network shown in Fig. 1b has three specialized nodes called electrode-nodes (ENs) whose connections to the source and the network are designed to mimic the experimental multi-electrode setup (see 1, 2, and 3 in Fig. S1).

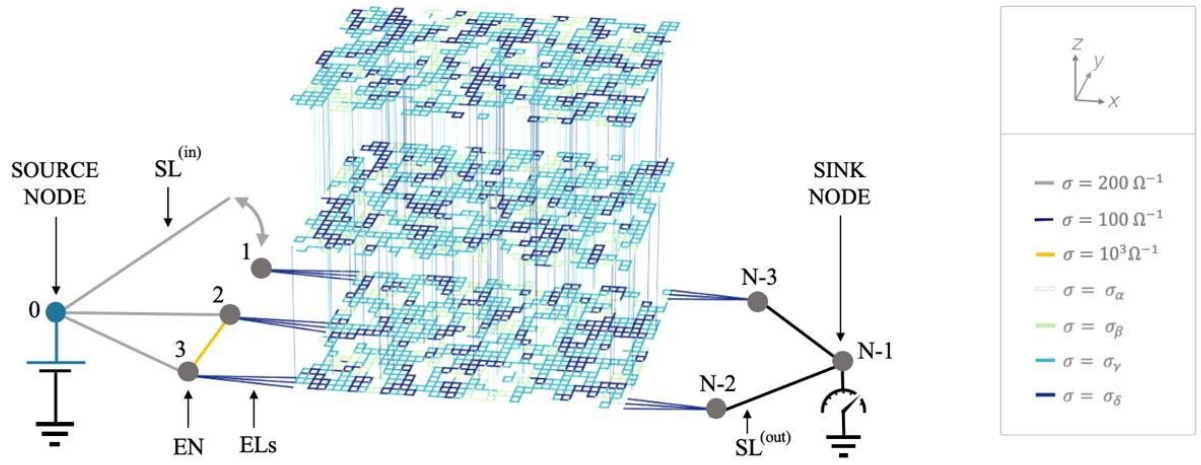

Figure S1: Network depiction. Specialized nodes and links are illustrated here.

These electrodes are connected to the left side of the lower layer by three separate groups of a few permanent-conductance links called electrode-links (ELs). The conductance of the input-ELs is set to $\sigma_{EL}^{(in)} = \sigma_\delta$ / $\sigma_{EL}^{(in)} = \sigma_\alpha$ when the input-SL is closed/open. The groups of ELs are spaced apart from each other and are intended to mimic three electrodes. Each of them can be connected to the source node through its own input-EN via a permanent switch-link (SL), which acts as a closed (or open) switch that connects (or disconnects) the device to the current generator. The conductance of the input-SLs depends on the desired electrode configuration: $\sigma_{SL}^{(in)} = 200\Omega^{-1}$ and $\sigma_{SL}^{(in)} = \sigma_\alpha$, representing respectively a closed or an open switch. The input electrodes can be connected or disconnected from the source according to the desired configuration, which is represented using a binary notation: a value of 0/1 indicates an open/closed switch. In configurations involving the passage of current through more than one electrode, two connections - whose conductance is set to $\sigma_{1,2} = \sigma_{2,3} = 10^3\Omega^{-1}$ to minimize the voltage difference at their ends - are added between two input-ENs to ensure the same amount of current can flow through all input channels. A similar setup is used to represent two output electrodes using two separate groups of permanent ELs that connect the right side of the lower layer to the corresponding output-EN (nodes A and C in Fig. S1). The output-ENs are linked to node N−1 (the sink through which current exits the system) via two SLs with fixed $\sigma = 100\Omega^{-1}$.

Reprogramming/computing procedures

During the reprogramming, a voltage above a certain threshold is applied, triggering local updates of the link weights [4] and producing new values of the output V_{out} during the subsequent reading phase. By exploiting the topological reorganization induced by the reprogramming, each of which lasts 90 MC steps, different output values can be explored (as shown in Fig.1b). The range of applied voltages spans $\Delta V \in$ [3 V, 45 V]. During the reading phase, the output of the device is measured sequentially for all input configurations and is averaged under the influence of a voltage of 1 V for 3 Monte Carlo steps.

The reprogramming voltage threshold is operationally defined as the electrical potential required to trigger resistive switching [5-7]. At the same threshold ($V_{th} = 3V$), a dramatic change in slope is also observed in the *I(V)* curve (see [1] for details).

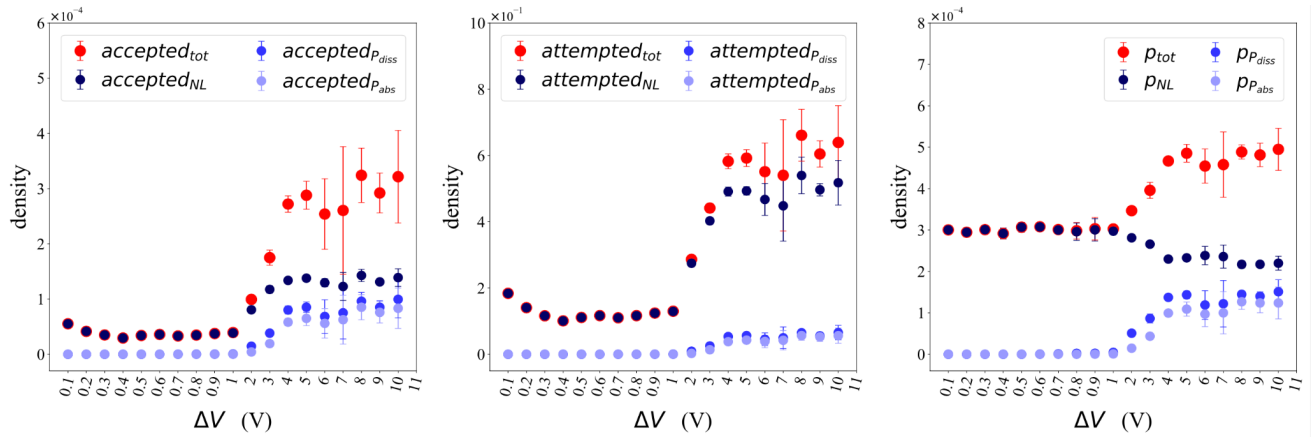

*Figure S2 Protocol of sequential simulations, each lasting 1000 MC steps and starting from the same networks configuration. The simulation of the sequence differ for the magnitude of costant ΔV delivered to the system. The MC moves of the algorithm of simulation are here considered separately (labelled with P_{abs}, P_{diss} and NL) and cumulatively (labelled with $\mathtt{tot}$). Simulations are labelled with the magnitude of costant voltage delivered (x-axis). **(Left panel)** curves of accepted moves, **(Central panel)** curves of attempted moves, **(Right panel)** acceptance probabilities of the MC moves as function of the costant voltage applied during the simulation.*

Similar behavior is also observed in the curves of MC move acceptance probabilities as the potential difference applied to the network changes. Fig. S2 shows the trends in the number of link updates (MC moves accepted) (left panel) over the number of MC moves attempted (central panel) for simulations of 1000 MC steps under a fixed potential difference. The four curves in each plot show respectively the count of acceptances/attempts due to the total number of moves, the absorbed power (P_{abs}) and the dissipated power (P_{diss}) moves, and nonlinear (NL) moves. As shown, below 1V the moves related to heat dissipation cannot be effective, while the nonlinear move is. At 1V all moves are active; for the network size utilized in this study, comprising approximately 9000 links, it's worth noting that a single link undergoes modification every two Monte Carlo steps within this regime (see total acceptance rate curve in Fig. S2-right panel). For potentials above 1V the acceptance rate of the total moves (Fig. S2-right panel) rises dramatically leading to frequent rearrangement of conductive links within the network. It is therefore reasonable to assume that at 1V there are insufficient changes to reconfigure the network: this is akin to the experimental computing regime. We instead assume that the number of accepted moves from 3V onward is sufficient to induce structural changes in the network, as observed for the reprogramming in the experimental system.

Markovianity Assumption

The receptron weights can be altered due to reprogramming, which physically consist of high voltage pulses that redistribute current pathways in the resistive medium. We restrict our investigation to the magnitude, polarity, and localization of the voltage stimulus applied:

$$r = (\Delta V, \pm, \vec{l})$$

as control parameters that shape the evolution of the receptron weighting of inputs. Additional parameters that may be considered are the length of the reprogramming pulse or a more complicated pulse train stimulation, which goes beyond the scope of the present work.

As formalized in section 2.3, depending on the actual internal resistive state *c(t)* of the system and the features of the reprogramming (summarized by *r*), a series of reprogramming phases with specific properties will induce a series of transitions between internal configurations.

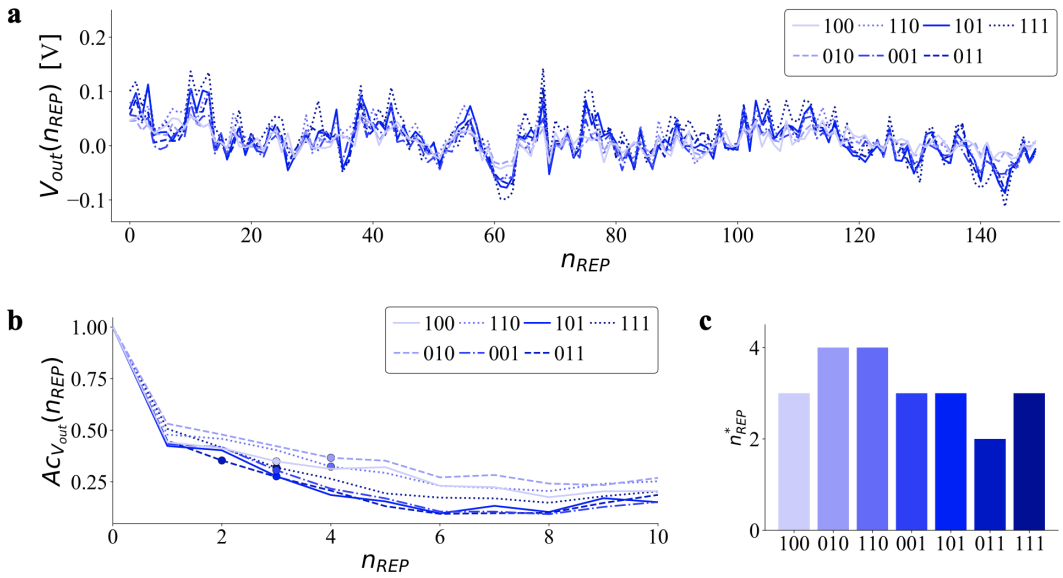

Figure S3 Experimental protocol. ***(a)*** *$V_{out}(t)$ curves, for $\vec{l}$ fixed. Protocol of 150 alternated reprogramming and computation phases.* ***(b)*** *Autocorrelation function of $V_{out}(t)$ for $\vec{l}$ fixed.* ***(c)*** *Characteristic autocorrelation time for each $\vec{l}$.*

Consequently, in the most general case, the output functions will follow a certain probability distribution P_{out} which depends on **all** previous internal configurations:

$$f_n \in P_{out}(R, \{f_1, f_2, \cdots, f_{n-1}\})$$

where $R = \{r_1, r_2, \ldots, r_n\}$ collects all the characteristics of the sequence of reprogrammings performed on the receptron implementation and $\{f_1, f_2, \cdots, f_{n-1}\}$ represent all the past output functions computed.

We have verified the extent to which the system memorizes the history of applied stimuli with the usual protocol involving alternating cycles of reprogramming and output computation and computing the autocorrelation of each analog output, being $V_{out} = \{V_{out}(t=1), V_{out}(t=2), \ldots, V_{out}(t=n)\}$ the analog outputs of the computation sequence.

We apply this protocol to both the experimental and simulated receptrons. For the experimental realization of the protocol, $\Delta V \in [-35\,V, -15V\,]\;U\;[15\,V, 35V]$, $\pm \in \{+,-\}$ and $\vec{l} = \{0,1\} \times \{0,1\} \times \{0,1\}$, with reprogramming lasting 1 sec, while for simulated protocol $\Delta V \in [-35\,V, -25V\,]\;U\;[25\,V, 35V]$, $\pm \in \{+\}$ and $\vec{l} = \{0,1\} \times \{0,1\} \times \{0,1\}$, with reprogramming lasting 90 MC steps. Figure S3-a and Figure S2-a show the sequence V_{out} for a protocol 150 reprogrammings, respectively for experimental and simulated realizations. The V_{out} curves are plotted for each parameter $\vec{l}$ composition.

To examine the system's memory of previous stimuli we analyze the autocorrelation function of each output for a given stimulus (Figure S3-b and Figure S4-b) and we compute the characteristic time n^*_{REP} (Figure S3-c and Figure S4-c).

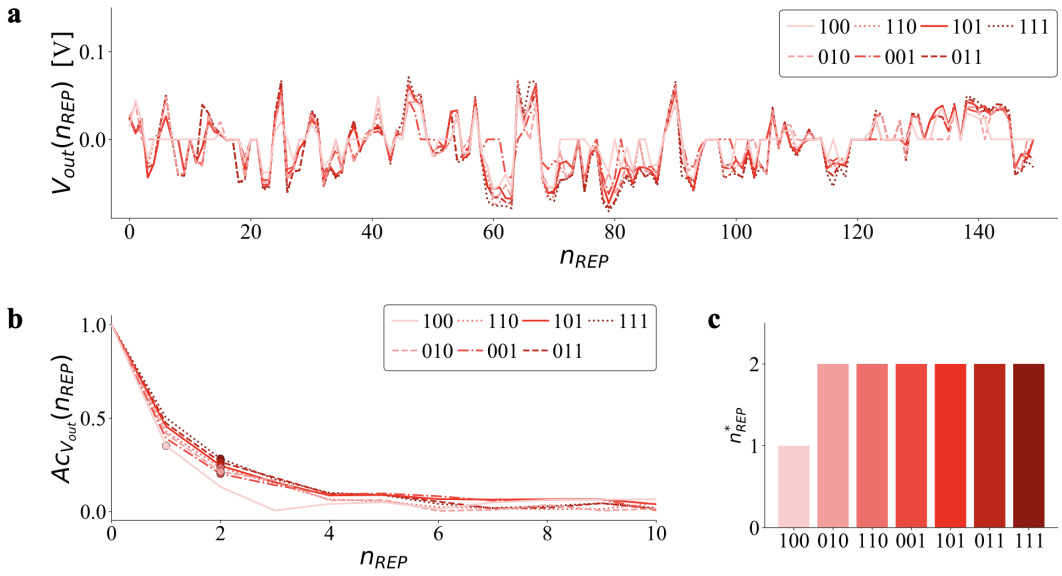

Figure S4 Simulated protocol. **(a)** $V_{out}(t)$ curves, for $\vec{l}$ fixed. Protocol of 150 alternated reprogramming and computation phases. **(b)** Autocorrelation function of $V_{out}(t)$ for $\vec{l}$ fixed. **(c)** Characteristic autocorrelation time for each $\vec{l}$.

Results show short memory retention of the last reprogrammings both for experimental (n^*_{REP} between 2 and 4 reprogramming steps) and simulated (n^*_{REP} between 1 and 2 reprogramming steps) cases. n^*_{REP} is calculated as the time at which the autocorrelation reaches a value smaller than 1/e for the first time.

It is therefore reasonable to limit the P_{out} dependency to r and f_{n-1}, discarding the terms referred to inputs that were further back. In other words, we assume that the output functions follow a certain probability distribution P_{out}:

$$f_n \in P_{out}(r_n, f_{n-1})$$

with the assumption of Markovianity in first approximation.

MI and II Calculation in the SRN Model

The data in Fig. 3b have been obtained with a simulation which followed a different protocol. To compute Mutual and Integrated Information, which are entropy-based measurements, one needs a probability distribution, and thus a definition of what is a 'state'. Therefore, we simulated at 1V a virgin network for 10^4 steps, taking the first 6×10^3 as equilibration and then using the last 4×10^3, split into 2 windows (made of 3000 MC steps each), with a stride of 1000 steps. After a reprogramming phase (long as well 10^4 steps, with r = (15 V, +, 100)), another reading at 1V was performed for 10^4 MC steps. To monitor the immediate effects of reprogramming, only the initial 4×10^3 steps have been included in the analysis, with the same window size and stride of the pre-reprogramming phase.

For the calculation of MI and II, we use a coarse-graining procedure by dividing the network into seven parallelepipeds. For each Monte Carlo step, the conductance value of each coarse-grained area is given by averaging the links belonging to the edges of the relevant parallelepiped. The choice to average only the edges is because only the conductivity of the edges is responsible for the passage of current between one zone and another. Formally, the links that constitute these boundaries are those that share at least one node with the links of neighboring areas. Since we are interested in understanding how the sub-regions of the SRN communicate with each other by exchanging information, we only included in the computation of the average σ only the boundary of each sub-region. Formally, the links that constitute these boundaries are those that share at least one node with the links of neighboring areas. This implies discarding all the internal links of each zone, if they only marginally contribute to the information exchange process. We expect that within each 3D zone there is always at least a conductive path which connects a region with the rest of the system through its interfaces.

In Fig. 3b, we have chosen the (100) configuration since we expect it to induce more marked changes with respect, e.g., to (010), since the current tends to flow close to one of the borders of the layer, thus making its effects more detectable. For each spatial region, the time series of the average conductance in that zone was used to build a histogram, with 1000 bins. The two histograms (of 6000 entries each) were then discretized, using deciles to obtain 10 distinct conductive states. The probability of such states was then computed, used to build the entropy H and from it the MI and II were obtained for each region.

References

1. Mambretti F, Mirigliano M, Tentori E, Pedrani N, Martini G, Milani P, et al. Dynamical stochastic simulation of complex electrical behavior in neuromorphic networks of metallic nanojunctions. *Sci Rep* (2022) 12(1):12234. doi:10.1038/s41598-022-15996-9
2. Rubido N, Grebogi C, Baptista MS. General analytical solutions for DC/AC circuit-network analysis. *Eur Phys J ST, Spec Top* (2017) 9. doi:10.1140/epjst/e2017-70074-2
3. Kagan M. On equivalent resistance of electrical circuits. *Am J Phys* (2015) 83(1):53–63. doi:10.1119/1.4900918
4. Teuscher Christof. Unconventional computing catechism. *Front Robotics AI* (2014) 1:10. doi:10.3389/frobt.2014.00010
5. Mirigliano M, Borghi F, Podestà A, Antidormi A, Colombo L, Milani P. Nonohmic behavior and resistive switching of Au cluster-assembled films beyond the percolation threshold. *Nanoscale Adv* (2019) 1(8):3119–30. doi:10.1039/c9na00256a
6. Mirigliano M, Decastri D, Pullia A, Dellasega D, Casu A, Falqui A, et al. Complex electrical spiking activity in resistive switching nanostructured Au two- terminal devices. *Nanotechnology* (2020) 31(23):234001. doi:10.1088/1361-6528/ab76ec
7. Mirigliano M, Paroli B, Martini G, Fedrizzi M, Falqui A, Casu A, et al. A binary classifier based on a reconfigurable dense network of metallic nanojunctions. *Neuromorphic Comput Eng* (2021) 1(2):024007. doi:10.1088/2634-4386/ac29c9
8. NetworkX, https://github.com/networkx/networkx