nature electronics

Review article

https://doi.org/10.1038/s41928-024-01133-z

Physical reservoir computing with emerging electronics

Received: 25 April 2023

Accepted: 9 February 2024

Published online: 12 March 2024

Check for updates

Xiangpeng Liang[1], Jianshi Tang[1,2] ✉, Yanan Zhong[3], Bin Gao[1,2], He Qian[1,2] & Huaqiang Wu[1,2]

Physical reservoir computing is a form of neuromorphic computing that harvests the dynamic properties of materials for high-efficiency computing. A wide range of physical systems can be used to implement this approach, including electronic, optical and mechanical devices. Electronics can, in particular, provide mixed-signal and fully analogue systems, and could be used to deliver large-scale implementations. Here we examine the development of physical reservoir computing with emerging electronics. We discuss the different architectures, physical nodes, and input and output layers of electrical reservoir computing. We also explore performance benchmarks and the competitiveness of different implementations. Finally, we consider the future development of the technology and highlight challenges that need to be addressed for it to deliver practical applications.

Over the past 50 years, the downscaling of transistor sizes—and the resulting exponential growth in the number of transistors in an integrated circuit—has driven improvements in computer performance and led to applications such as artificial intelligence (AI)[1]. However, in recent years, such scaling has begun to approach its physical limits, facing challenges related to quantum effects and reliability issues[2,3]. At the same time, the physical separation of memory and computing units in conventional hardware limits energy efficiency—an issue known as the von Neumann bottleneck—creating particular problems for data-intensive applications such as AI[4].

Neuromorphic computing aims to create energy-efficient computing systems by emulating the working mechanisms of the brain[5–7]. Unlike conventional computers, the brain uses an analogue computing approach with complex recurrent dynamics. Although the nervous system is extremely complex and yet to be fully understood, simplified structures can be recreated in artificial systems.

Recently, it has been shown that recorded neuronal activities in the brain can be decoded into behaviours via a linear readout[8]. For example, electrode arrays were implanted into a mouse's brain to trace its electrophysiology during a foraging task. By calculating the weighted sum of multi-channel neural signals from an electroencephalogram measured in the frontal cortex, the decision variables that affect the mouse's behaviours could be calculated (Fig. 1a). This biological working principle is mirrored in a type of a recurrent neural network (RNN) algorithm known as reservoir computing (RC)[8].

It is the case that RC (together with other RNNs) has frequently been used to study brain functionalities because of the similarities in their complex recurrent dynamics[9–12]. The origins of RC can be traced back to 2001, when Herbert Jaeger invented the echo state network (ESN) as an efficient implementation of an RNN[13]. That work used the echo states in a complex RNN layer, followed by a linear readout layer trained by linear regression. In 2002, Wolfgang Maass proposed the liquid-state machine (LSM), which operates in a continuous and spiking fashion[14,15]. Liquid state is an important concept that can be visualized by considering what happens when continuously throwing stones into a lake. The multiple ripples created are interactive and result in a complex ripple pattern—the liquid states. Information related to the stone-throwing activities over the past few seconds can be retrieved by applying a linear layer to the liquid states. Both works were subsequently experimentally unified as RC based on their similar principles (Fig. 1b)[16]: an input layer, high-dimensional mapping in a fixed reservoir layer with complex time-dependent dynamics, and a linear readout layer.

In RC with N neurons, a d-dimensional input and a k-dimensional output, the network topology comprises three layers: an input layer W_{in} ($N \times d$), a reservoir layer W_{res} ($N \times N$) and an output layer W_{out} ($k \times N$).

[1]School of Integrated Circuits, Beijing Advanced Innovation Center for Integrated Circuits, Tsinghua University, Beijing, China. [2]Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing, China. [3]Institute of Functional Nano & Soft Materials (FUNSOM), Jiangsu Key Laboratory for Carbon-Based Functional Materials & Devices, Soochow University, Suzhou, Jiangsu, P. R. China. ✉e-mail: jtang@tsinghua.edu.cn

https://doi.org/10.1038/s41928-024-01133-z

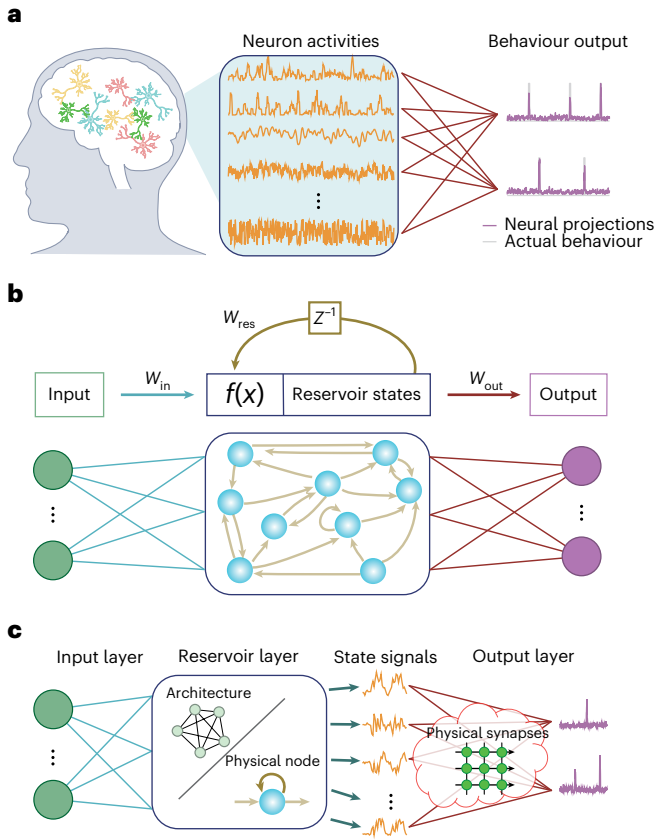

Fig. 1 | Biological, digital and physical RC. a, Biological RC. The weighted sum of electrical signals emitted by neuron activities in a mouse's brain can predict actual behaviours, providing a biological foundation for RC algorithms. **b**, Digital RC. A conventional RC algorithm implemented in the digital domain. W_{in}, W_{res} and W_{out} represent input weights, reservoir weights and output weights, respectively. Z^{-1} denotes a delay operation over a discrete time step, and $f(x)$ is a nonlinear function. **c**, PRC. Dynamic behaviours of physical systems can be used as computing resources that mimic the functionalities of RC in the physical domain. The reservoir layer is implemented using interconnected physical nodes, and the output layer is normally realized using physical synapses that can perform vector–matrix multiplication.

Typically, W_{res} and W_{in} are randomly generated, and W_{out} requires training using simple methods like linear regression. Based on the theory of ESN, the W_{res} should be appropriately scaled according to its spectral radius to create a desirable echo state property[17]. At each time step n, a standard software-based reservoir follows the following process:

$$\mathbf{s}(\mathbf{n}) = f[aW_{in}\mathbf{u}(\mathbf{n}) + bW_{res}\mathbf{s}(\mathbf{n}-1)] \quad (1)$$

$$\mathbf{y}(\mathbf{n}) = W_{out} \cdot \mathbf{s}(\mathbf{n}) \quad (2)$$

where $\mathbf{u}(\mathbf{n})$, $\mathbf{s}(\mathbf{n})$ and $\mathbf{y}(\mathbf{n})$ are the input vector, state vector and output vector, respectively at the nth step, a and b denote the input and feedback scaling factors, respectively, and f represents a nonlinear function.

RC has distinct features in how it analyses temporal input data due to its recurrent connections, which establish a dependency between the current and past neuron dynamics, referred to as short-term memory or fading memory. Short-term memory allows historical information to be retained within the reservoir state over the time steps and is quantified by the measurement tech nique memory capacity (MC)[18]. The purpose of the reservoir layer is to nonlinearly map the low-dimensional input into a high-dimensional feature space, that is, represented by the states of a neuron. Therefore, the state channel $\mathbf{s}(\mathbf{n})$ has to provide linearly uncoupled dynamics to enhance the separation of the different input information in the high-dimensional space. The state richness of the RC can be quantified by the number of linearly uncoupled dynamics[19].

The advantage of RC is the simplicity of its output-layer training, which avoids the gradient exploding and vanishing problems in conventional RNNs[20,21]. It could also enable fast learning with few samples[17]. The output layer can be easily trained by linear regression. Beyond linear regression, other training methods such as reinforcement learning[22], gradient descent[23–25] and augmented direct feedback alignment[26] have also been proposed. In contrast to the widely used feed-forward deep neural networks (DNNs) trained by gradient descent, RC is less expressive in approximating target function, but has advantages in terms of training, dynamical systems approximation and temporal signal processing[20,21,27].

Physical systems can also be used to approximate the functionalities of RC, where the behaviour of specifically designed dynamical systems can be analogous to the reservoir layer[28,29]. In 2003 it was shown that a LSM in the physical domain could be used to solve XOR and speech recognition problems[30], bridging the gap between RC algorithms and physical systems. Then, in 2011, a delay-coupled RC[31] that reduces the number of physical neurons was reported.

The aim of physical reservoir computing (PRC) is to harvest physical dynamics as computational resources under the framework of RC to achieve resource-efficient information processing. It is an increasingly important branch of neuromorphic computing[28]. Compared with the other neuromorphic computing approaches—such as spiking neural networks (SNNs) and feed-forward DNNs—PRC can perform the majority of computing in the physical domain in a more straightforward and elegant manner[28], and can even achieve all-analogue computing[32,33].

Its hardware implementation is compatible with a range of physical systems, including electronics[23,31,33], optical components[34–43], mechanical structures[44–52] and quantum devices[53–61]. Various PRC systems have already been used for low-power computing, edge computing and in-/near-sensor computing[28,29,62]. However, in this interdisciplinary field there is a lack of consensus regarding RC architectures, implementations and benchmarks, which has hindered the development of large-scale PRC for practical applications. Material scientists will, for example, typically focus on the computational capabilities of various materials and devices using the concept of RC, whereas computer scientists will focus on the novelty and validity of RC algorithms.

In this Review we explore the development of PRC with emerging electronics[27–29,63]. Such electrical RC (eRC) has the potential for large-scale implementation and is more compatible with conventional circuits than implementations such as optical and mechanical RC. We first consider how architectures of the reservoir layer play a critical role in the hardware design of eRC, and propose a taxonomy of eRC architectures. This taxonomy provides guidelines for designing eRC, characterizing the main features and comparing different implementations. We then examine the design of the physical nodes, as well as the design of the input (pre-processing) and output layers. We also consider optimization strategies for the development of eRC, and performance benchmarks for the technology. Finally, we discuss the competitiveness of different eRC implementations and consider the challenges that need to be addressed for eRC to deliver practical applications. (Brief definitions of commonly used terminologies of eRC are provided in Supplementary Table 1.)

eRC architectures

Delay-coupled RC

During the early stages of RC, the concept of delay was incorporated to describe the state update[13,64]. In 2011, the delay-coupled RC was demonstrated[31], in which a time-multiplexing operation oscillates the input signal by multiplying a mask matrix within every time step. The masked signal then injects into a nonlinear physical node where a complex dynamic is generated by fine-tuning the intervals of time-multiplexing and the time constant of the physical node. The state vector—the virtual

https://doi.org/10.1038/s41928-024-01133-z

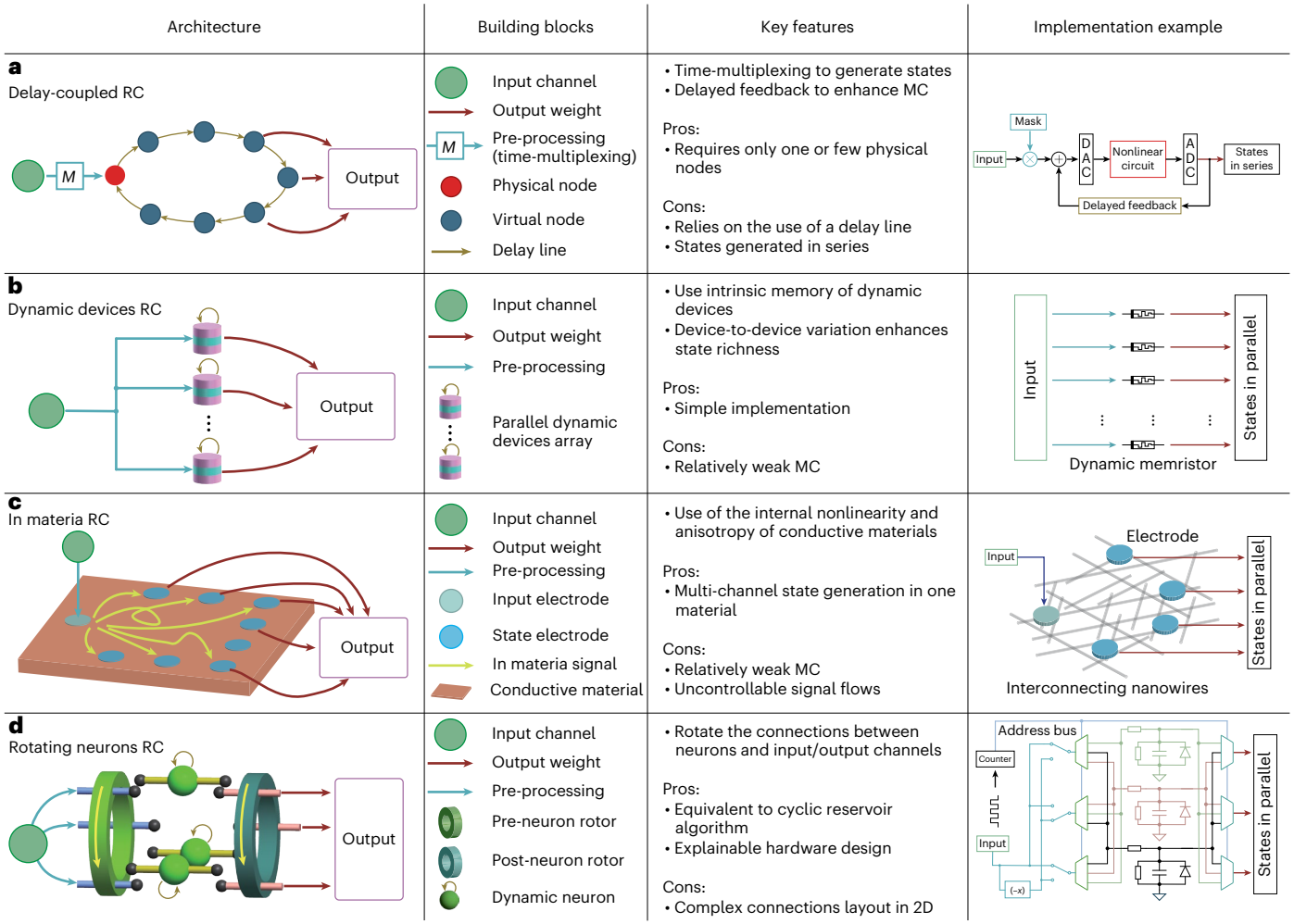

Fig. 2 | eRC architecture taxonomy. a–d, Four architectures: delay-coupled RC (**a**), dynamic devices RC (**b**), in materia RC (**c**) and rotating neurons RC (**d**). Also shown are their building blocks, key features and implementation examples.

node—is obtained by sampling the output of the physical node in series. The output of the physical node is delayed and added to the masked input signal via a feedback line[31,65,66]. By linking the current input with the previous node states, the delay line preserves the historical information of the network and forms an RNN[31,67]. Multiple delay lines and a longer delay time can enhance the MC to retain a longer historical signal[68–70]. A non-resonant delay cycle can also boost the MC and the corresponding performance[71]. In typical eRC implementations (Fig. 2a), feedback and time-multiplexing are realized in the digital domain, where analogue and digital conversions are necessary. Here, the physical node can be a circuit or an electronic device, because the nonlinear region exists in most electronic behaviours[28].

An advantage of delay-coupled RC is that only one physical node is needed to construct a reservoir layer, which dramatically reduces the implementation complexity compared with interconnected neurons in conventional RC[28]. On the other hand, of importance when designing delay-coupled RC is the implementation of the delay line. In early works, the delay is realized digitally (Fig. 2a), and the conversions of analogue and digital signals can be challenging[28]. To minimize cost, an analogue-to-digital converter (ADC) and a digital-to-analogue converter (DAC) with low precisions (usually 4–10 bits) can be employed, but the noise implication needs to be studied carefully[67]. Recent implementations of delay-coupled RC are integrated with mature Si complementary metal–oxide–semiconductor (CMOS) circuits, such as 180-nm (a spike-based delay-coupled RC)[72] and 65-nm technology[73]. In fact, the delay-coupled RC architecture is of particular interest for optical RC systems, in which a long optical fibre is employed to delay the signal[37,74], thus enabling ultrahigh-speed signal processing[39]. One disadvantage of delay-coupled RC is its serial operations due to the time-multiplexing of the physical node. Parallel computing is preferred in analogue neuromorphic computing[75]. Nevertheless, delay-coupled RC provides useful insights for PRC based on dynamical hardware and has inspired the development of PRC with different physical systems, as well as other eRC architectures.

Dynamic devices RC

In recent years, an eRC architecture based on dynamic devices has emerged. In this architecture, the physical node is a single dynamic device, and the entire reservoir layer consists of multiple dynamic devices that receive input signals in parallel (Fig. 2b). This technique is also known as spatial multiplexing[76] (analogous to time-multiplexing in delay-coupled RC), where the effective number of physical nodes is increased to enhance the system performance. The key idea is to harness the intrinsic memory property and nonlinearity of dynamic devices to encode temporal input signals into internal device states, such as the conductance of a volatile memristor, which are then read as the reservoir state vector. The dynamic device under this architecture thus usually possesses short-term memory and nonlinearity, for example, a dynamic memristor[23,24,33,77–81], spintronic oscillator[82], ferroelectric device[83–86], ionic transistor[87,88] or quantum system[76]. In 2017, a memristive RC based on multiple independent dynamic memristors was proposed[23] in which the different spike-based input sequences

were encoded into the memristor conductance states via short-term memory. The state richness can be improved to a certain degree through device-to-device variation of the temporal dynamics[23,24,86]. A group of devices with low device-to-device variation might cause issues like overfitting in the readout layer. Time-multiplexing to a spintronic oscillator has also been introduced to substantially increase the length of the state vector[82]. However, the absence of a delayed feedback line requires that a single device has to support the MC of hundreds of virtual nodes. The effective MC declines as the mask length increases. Inspired by the above studies, different mask vectors with relatively short lengths (around five) were applied to parallel dynamic memristors to avoid the vanishing of MC[77]. Subsequently, efforts have been made to generate high-quality state vectors using various dynamic devices. For example, the number of states was increased by reading the outputs of ferroelectric tunnelling junctions (FTJs) at multiple time steps[84]; applying different gate voltages to an α-In_2Se_3 transistor array resulted in different relaxation times for each device and thus increased the reservoir's state richness[89]; ionic transistors were configured with different gate lengths to induce different relaxation times and nonlinearities to enhance state richness[88]; and integrating a configurable memristor, capacitor and resistor in a single cell—the temporal kernel[90]—provided more flexibility for the reservoir's dynamics, allowing the system to adapt to a wide range of signal-processing tasks, from ultrasound (time constant of 10^{-7} s) to electrocardiogram (time constant of 1 s).

Compared with other eRC architectures, dynamic devices RC fully explores the internal device behaviour, and several techniques can be used to improve the quantity and quality of the reservoir states. This architecture simplifies connectivity and facilitates hardware implementation, minimizes system complexity and hardware cost, while ensuring sufficient state richness. It is commonly used in in-sensor computing[25,91,92] and self-powered computing[93], because the devices can directly receive environmental stimulation such as light sensing when using optoelectrical devices[25], and they can perform in situ computing simultaneously, opening a novel edge-computing paradigm. However, the challenge of dynamic devices RC is that the internal states are less likely to encode and distinguish information with a long timescale. This is because the MC depends only on the capacity of a single device, whereas the MCs of multiple parallel devices are mostly overlapped rather than accumulating to a higher value. This limits the MC to a relatively low level, but could be mitigated by storing intermediate states in a buffer, at the cost of external memory[23–25,89].

In materia RC

The in materia RC architecture is a fast-growing approach in recent years to implement designless PRC[94]. It was first implemented using a neuromorphic atomic switch network[95]. It operates under the premise that the internal dynamics of an aeolotropic material—a material whose properties depend on the direction of measure—are sufficiently intricate to allow high-dimensional mapping[94,96,97]. A conductive material suitable for the architecture is analogous to the complex spatially dependent ripple patterns observable on a lake surface after stone throws (as in the case of the LSM example discussed earlier). Injecting signals at a particular location of the conductive material results in distinct responses at different state-vector collection locations. This requires the material to exhibit heterogeneous conductivity between every in–out channel, allowing the in materia signal to undergo different nonlinear transformations. Here, the physical node is the material itself, which determines the MC. To enhance the MC, memristive materials are preferable (Fig. 2c). For example, an in materia RC was developed by drop-casting interconnected Ag nanowires[94]. The memristive property resulted from Ag^+ ion migration in the polyvinylpyrrolidone (PVP) at the cross-point junction (acting like neurons) between nanowires, forming a volatile conductive bridge. The connectivity and junction density of the Ag–PVP–Ag nanowires are relatively controllable in this system, which is an advantage over other materials[98–101]. The signal input and state output can be accessed via electrodes such as Au pads for the pre- and post-processing. A number of materials and substrates have been investigated, including Ag/AgI nanowires[102], carbon nanotubes[97,103], organic electrochemical network devices[104,105], magnetic skyrmion memristors[106] and ferroelectric devices[107–109]. Other examples include wave propagation and transmission through materials[110–113], such as spin waves travelling through a magnetic material[114,115] and metamaterial[116].

In materia RC fully explores the internal dynamics of a conductive material itself. Therefore, the hardware design outside the material can be minimized or is even needless, highlighting its unique advantage of extremely low cost and simple hardware implementation. Moreover, a single material can produce multiple state channels in parallel. However, its main disadvantage is its shortage in MC, as this mainly depends on the material properties. An exception was found in a spin wave-based RC simulation that yielded an MC of 38 (ref. 115), but further studies are needed to reveal the underlying mechanism. Furthermore, in-material signal flows are hardly accessible and controllable, as they are usually sealed inside the material body, unlike other architectures that have independent circuits or devices. Although the physical reservoir layer, in principle, could be designed in a highly random manner, it is still desirable to monitor the signal flows of each neuron to understand how the inner structure affects the state vector and the corresponding dynamics, especially during the development and optimization stages. In existing studies, the signal flows and inner dynamics are mainly analysed through materials modelling[94,99,100,117], which may not fully reveal the actual picture.

Rotating neurons RC

In a recently proposed architecture[32], rotating neurons RC, an equivalent pair of physically rotating objects and a cyclic RC algorithm were discovered. Cyclic RC is a simplified version of the classical RC[64], where the reservoir layer has a ring topology and can be designed in a deterministic manner with fewer parameters in comparison to randomly generated connections. It has been mathematically proven that if we periodically shift the input and output connections of a first-order dynamical node array, its physical behaviours are equivalent to the in/out of a cyclic reservoir, which we name 'rotating neurons RC'. So far, this is the only network-level equivalent pair of an RC hardware architecture and algorithm that enables the mapping of cyclic RC to rotational hardware and makes the hardware designs interpretable by algorithm. Pre- and post-neuron rotors can physically implement the weight matrix of the reservoir layer (that is, a shifted identity matrix). This explainable hardware design brings unique benefits. It enables straightforward and elegant implementation of the reservoir layer so that the assistive peripheral circuits and the interface between different modules, such as the ADC, buffer and memory, are not required, which substantially reduces the system complexity and power. Furthermore, rotating neurons RC could offer a much higher MC than none-rotation systems (similar to the parallel dynamic devices RC)[32], because the MC can be enhanced as the number of physical nodes increases, as has been analytically studied with the cyclic RC algorithm[64]. A follow-up study has shown an over threefold MC improvement by adding rotation to the PRC system[118].

An example implementation based on electrical circuits is shown in Fig. 2d, where a multiplexer array subject to a counter is used to realize the pre- and post-neuron rotors, and the first-order dynamic neuron can be properly approximated by an integration–ReLU–leakage circuit. When applying a driving pulse to the counter, every multiplexer in the pre-neuron rotor periodically and sequentially links its input channel and each neuron circuit, thus implementing the rotation. The main challenge associated with rotating neurons RC is the layout of the great number of connections on a two-dimensional (2D) substrate, as a rotating connection is naturally a 3D motion. As the number of

https://doi.org/10.1038/s41928-024-01133-z

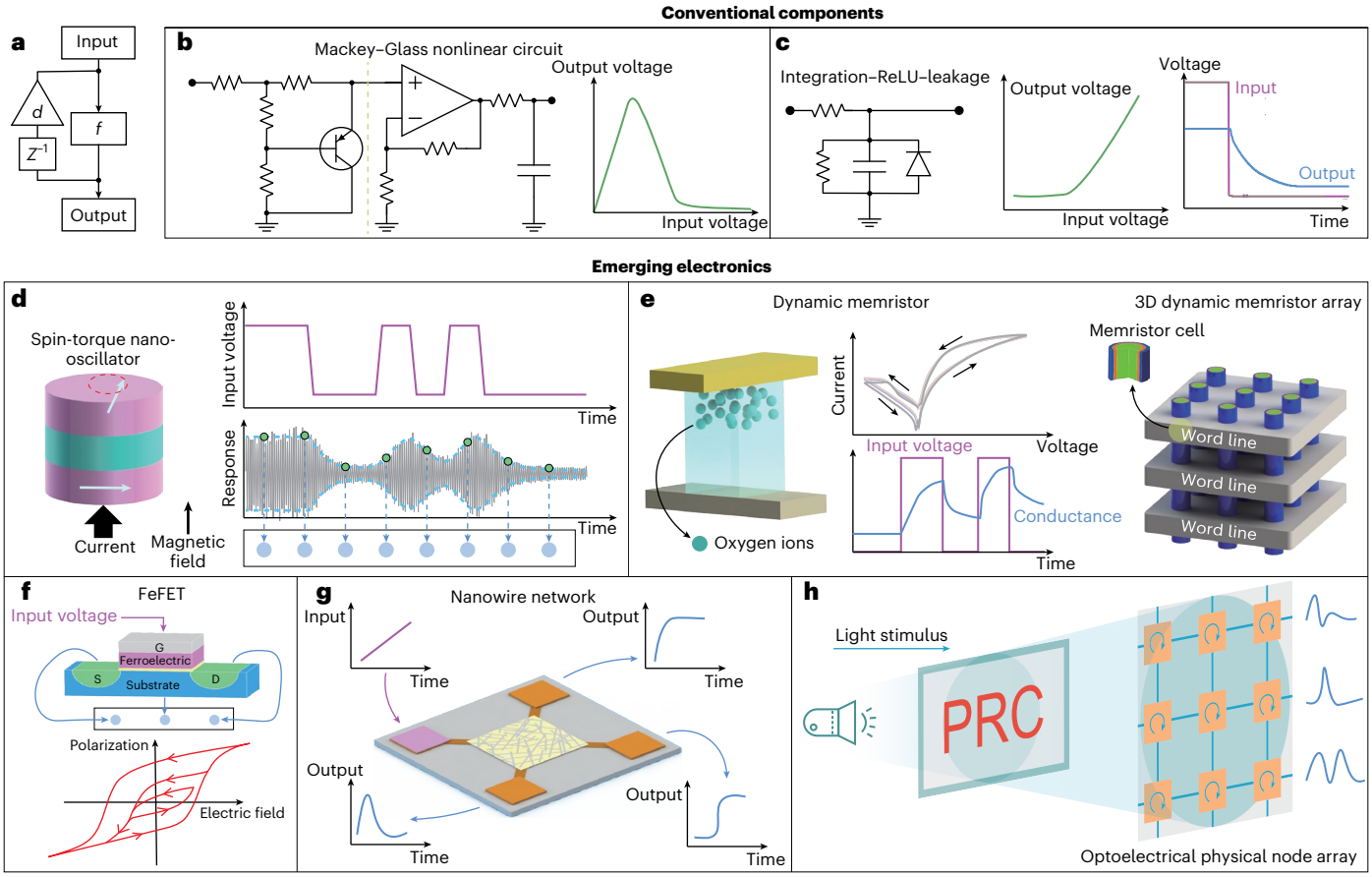

Fig. 3 | Physical nodes. a, A general model of the physical node. The two basic characteristics of physical nodes are nonlinearity and dynamical property. f and d represent a nonlinear function and a decay factor, respectively, and Z^{-1} denotes a delay operation over discrete time steps. **b**, A Mackey–Glass nonlinear circuit with its characteristic input voltage versus output voltage curve. It exhibits a Mackey–Glass-type nonlinear transformation and dynamical property provided by the resistor–capacitor network. **c**, An integration–ReLU–leakage circuit. The diode offers a nonlinearity similar to the ReLU function, and the resistor–capacitor circuit offers integration and dynamical property. **d**, A spin–torque nano-oscillator. A magnetic tunnel junction subjected to current input and an external magnetic field can output an oscillating signal to generate states. **e**, A dynamic or volatile memristor, whose conductance can be dynamically modulated by external stimulation and gradually relaxes to its initial value in the absence of stimulation, thus providing short-term memory and nonlinearity. A 3D memristor array can provide high-density memristors as physical nodes for generating node states. The input signal can be injected at corresponding word lines. **f**, A ferroelectric field-effect transistor (FeFET), whose gate (G) receives an input signal, induces distinct outputs at the drain (D), source (S) and substrate (silicon), resulting in three state channels in a single device. **g**, A nanowire network as a physical node can produce different nonlinearities and dynamical behaviours at the output electrodes in response to a common input signal at the input electrode. **h**, Optoelectronic physical nodes are naturally compatible with in- and near-sensor computing, where optical stimulation from the environment (for example, light bouncing off a PRC sign) can be directly presented to the node array to induce reservoir states generation.

neurons increases, the wiring between rotors and neurons will grow exponentially on the 2D substrate. This issue could be mitigated by exploring 3D integration.

Physical node design

Physical node is another key element for PRC (Fig. 1c). It usually refers to the interconnected dynamical components acting like neurons in RC algorithms. The connectivity and interaction between physical nodes is governed by the chosen PRC architecture. In delay-coupled RC, a single physical node receives the time-multiplexing signal and generates virtual nodes in series. In dynamic devices RC, multiple physical nodes independently receive stimulation and generate output in parallel, without interaction. In in materia RC, physical nodes could be the junctions of signal flow between each in/out channel, whose interactions are defined by the random connections inside the material. In rotating neurons RC, physical nodes interact with each other via the pre- and post-neuron rotors.

Generally, physical nodes possess two main characteristics (Fig. 3a): nonlinearity and dynamics. The physical node, acting as a kernel function, should offer a nonlinear function between the input–output signals. Otherwise, the PRC can only solve linearly separable problems, because the rest of the network is usually linear[27,28]. Unlike feed-forward DNNs, PRC is a dynamical system exhibiting short-term memory, which mainly stems from the physical node. Other implementation-specific properties of physical nodes in eRCs include oscillation[82], integration[32], leaky integrate-and-fire[72] and device-to-device variation[24]. Figure 3 illustrates several examples of physical nodes and their main characteristics. Conventional electronics can provide dynamics with specific nonlinearity and time constants to implement a physical node, and typical examples to realize the Mackey–Glass nonlinear and ReLU functions are shown in Fig. 3b,c refs. 31,32,67. The magnetic tunnel junction in dynamic devices RC can serve as a nonlinear oscillator for state generation (Fig. 3d)[82,119]. Memristive devices have also been widely studied to provide dynamics for eRC in a different way: historical stimulation changes the conductance in a volatile manner, thus connecting the current node state and historical node states[23–25,33,63,77,93,120]. Memristive devices and materials are usually preferred in dynamic devices RC and in materia RC to

compensate for the weak MC of these. Recently, a 3D memristor array was proposed to generate more states[80,120] (Fig. 3e). A ferroelectric field-effect transistor can produce three channels of node states in parallel (Fig. 3f). Its ferroelectricity offers a history-dependent polarization state and nonlinear polarization dynamics[107,108]. In materia RC is commonly based on junctions created by interconnected nanowires (Fig. 3g), where multiple output pads respond nonlinearly and differently to the input[100,102,104]. Recent developments in physical nodes have focused on incorporating more and more versatility. For example, optoelectronic physical nodes can simultaneously receive optical stimulation and produce a state output, thus enabling in-sensor computing[25,92] or multimode computing[87,89] (Fig. 3h).

Pre-processing techniques

Beyond the extensively studied reservoir layer, the pre-processing (input layer) and output layers must also be taken into consideration to form a complete PRC hardware. The pre-processing methods vary for different RC architectures, physical nodes and tasks.

A commonly used technique to increase state richness is time-multiplexing[31,33]. Figure 4a illustrates a time-multiplexing operation with a mask length of 10. The duration of each piece of discrete data (black line) is equally divided into ten units, followed by allocating the data value randomly multiplied by (or added to) −1 or 1 to each unit. The generated mask vector remains constant for the duration of every data point. Bias and scaling may also be applied after time-multiplexing to find an optimal input range[68]. To induce complex dynamics (blue line), the empirical duration of each unit (θ in Fig. 4a) is approximately a quarter of the time constant of the physical node[31,82]. The resulting node response of each time-multiplexing unit is the state vector that can be collected in series, namely the virtual node. Here, the unit duration θ and mask length need to be carefully chosen. A duration that is too long would cause saturation before the next value comes and lead to an inadequate state richness due to bistable state values in the node response[31,33]. At the same time, if the duration is too short, the physical node may work in a small linear range with a low amplitude. Therefore, the time constant needs to be chosen carefully to adapt to the intervals of the source signal and the desired number of virtual nodes. The mask vector is usually randomly generated. Studies have also revealed that the mask generated by the chaotic system[121] and maximum length sequences[122] could perform better. This technique is essential for delay-coupled RC to produce sufficient virtual node states, as only one physical node exists[31]. Meanwhile, it is optional for other architectures to increase the number of states[33,77,83,85,120].

Another widely used pre-processing technique is spike encoding, which is particularly useful for 2D inputs such as images. Here we use the volatile memristor-based dynamic devices RC as an example, although similar methods work for other architectures. For a simple 2D character (such as 'E' in Fig. 4b), the map can be directly converted to a multi-channel spike sequence as the input to the reservoir layer[23,25,80,85,94,101,120]. The spike sequence adds a temporal dynamic to the physical nodes. The state vector can be measured once the last spike is injected. In principle, the last value of the physical node output contains the information of the whole spike sequence due to short-term memory. The intermediate values within a spike sequence can also be collected to enhance the state vector and MC[24,25,84,108,109], but collecting and storing those intermediate values requires additional serial operations and memory units. Furthermore, for a more informative image input, such as the handwritten-digit dataset[123], chopping, merging and rotating the image before spike encoding (Fig. 4c) can increase the number of parallel spike sequences at the input channels and shorten the length of every sequence to avoid MC shortage[33,92,94,124].

Output layer

The RC output layer multiplies the state vector by the output weights, which is a typical vector–matrix multiplication (VMM) operation. During the early stage of PRC development, VMMs were mostly implemented using a computer. Recently, hardware-based VMM has been intensively investigated in the field of neuromorphic computing because it consumes the majority of computational resources in artificial neural network applications[5,125]. The solutions to the output layer are similar across eRC research—a memristor crossbar array is used that has been proven extremely energy-efficient for VMM operation compared with digital computers[125]. Different from the reservoir layer, the memristor[126] used in the output layer must be non-volatile to store the trained output weights (Fig. 4d). Ideally, the voltage signals flow through each memristor, resulting in a current according to Ohm's law, which is then integrated at the output to obtain the weighted sum results of state vectors according to Kirchhoff's law. However, the conductance variation in the memristors is a big challenge, and the output weights should be noise-tolerant to avoid substantial accuracy loss. In the memristor array, each unit cell commonly consists of a one-transistor–one-resistor (1T1R) structure to avoid the sneak path problem. Memristor array-based output layers have been implemented for rotating neurons RC[32], parallel dynamic devices RC[33,83,84,92,93] and in materia RC[94]. In 2022, a real-time fully analogue RC system was demonstrated by integrating volatile memristors in the reservoir layer and non-volatile memristors in the output layer, forming a dynamic devices RC architecture[33]. That work overcame the difficulties in interfacing different modules and handling the real-time end-to-end signal flows in an all-analogue fashion, while yielding orders-of-magnitude lower power consumption than digital RC.

Optimization strategies

The development of eRC always involves hyperparameter optimization, where the tunable parameters in the hardware are more sophisticated than software algorithms[28]. This section discusses the key factors that can be optimized. First, the reservoir size is the prime feature that determines the degree of freedom of the dynamics in the reservoir layer that affects the MC and approximation capacity[16,27,64,127]. The reservoir size (also the length of the state vector) in conventional RC indicates the number of neurons. However, the concept of virtual nodes in eRC suggests that the reservoir size could be increased without adding physical nodes[31]. In general, the minimum reservoir size depends on the task complexity and target accuracy and error. A larger reservoir size requires more hardware resources to implement. The computational capacity cannot be endlessly improved by expanding the reservoir size[27] and hence should be optimized for a specific application.

The RC performance is highly sensitive to its input range. Unlike gradient-based neural networks, the RC training process cannot automatically adapt to achieve the optimal range of each neuron for feature extraction by tuning every pre-neuron weight. Therefore, the input range, including the scaling and bias for the input signal, has to be fine-tuned to maximize performance. A straightforward approach is to scan and find the best parameters. For example, the input scaling of delay-coupled RC[31] and rotating neurons RC[32], together with the feedback strength or the time constant of the physical node, need to be scanned to find the optimal setting for nonlinear system approximation tasks.

Another crucial factor is the time constant of the physical node (τ), which determines the operating timescale for the overall system. Basically, the value mainly depends on the desired computing speed and whether time-multiplexing is used. Before optimization, the purpose of the RC system needs to be known. To accelerate signal processing, the physical node could be designed with a short τ, such as picosecond level in delay-coupled RC[38], as long as the architecture and other components allow this. On the other hand, the physical nodes used in neuromorphic computing such as in-sensor or near-sensor applications could exhibit biologically realistic timescales (larger than 1 ms) to interact with the environment and natural signals[7]. A short τ would be unable to couple with the real-time signal and fail to generate effective

https://doi.org/10.1038/s41928-024-01133-z

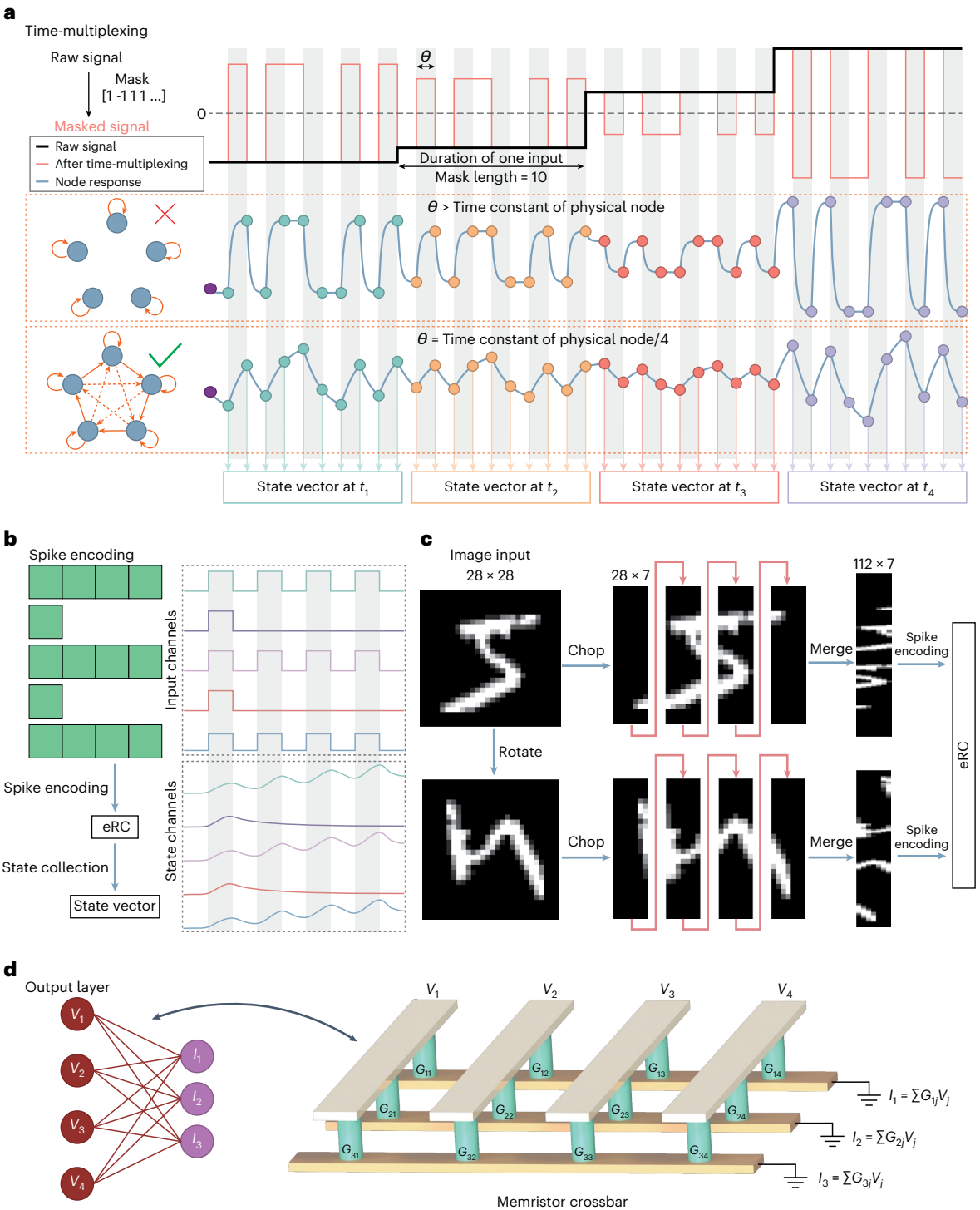

Fig. 4 | Input layer and output layer. a, Time-multiplexing operation for the input signal with a mask length of 10. During every input interval, the data point is multiplied by a randomly generated mask vector (top). By coupling the time constant of the physical node (τ) and the mask duration (θ), the time-multiplexing signal can induce a complex context-dependent response in the physical nodes, which can be collected in series as the state vector (bottom) at different time steps t_1, t_2, ... In the case of mismatch between θ and τ, for example when θ is larger than τ (middle), the node will fail to generate an effective node state because the node rapidly saturates before the next input comes. **b**, Two-dimensional data can be encoded in spike sequences as the input of eRC. For example, the pattern representing the letter 'E' is encoded into five spike sequences. The bottom right panel is an illustration of the output of physical nodes (such as a volatile memristor in dynamic devices RC) in response to such spike sequences. Applying a state collection scheme on the state channels can obtain the state vector. **c**, For informative image inputs like handwritten digits from the MNIST dataset, the pre-processing usually involves chopping the image to shorter length sequences before converting them into spike sequences. To increase the length of the state vector (reservoir size), the image can also be rotated by an angle before chopping and merging, which creates a different input spike map for a common data source. **d**, The output layer of RC is usually a fully connected linear network, which can calculate the weighted sum of the state vector. This can be physically implemented by a memristor crossbar array, in which VMM operations can be realized in parallel, in situ, by taking advantage of computing-in-memory. The output currents indicate the VMM results. V_1 to V_4 are the voltage values corresponding to the state vector, G_{11} to G_{34} are the conductance values of the memristor crossbar array, and I_1 to I_3 denote the current values representing the VMM results.

https://doi.org/10.1038/s41928-024-01133-z

reservoir states. Rotating neurons eRC, with τ around 1 s, can process handwriting traces in real time, and the state vector collected at the end of the handwriting trace contains the information of the past roughly 0.8 s thanks to the effect of MC. The storage of intermediate states is unnecessary in this case[32]. In contrast, a shorter τ with a faster rotating speed leads to rapid forgetting of the past handwriting trace, thus degrading the eRC system performance.

In addition, there are various configurations provided by the physical node and architecture that can be optimized in eRC development, such as the mask length of time-multiplexing[77], the delay line (number and duration) in delay-coupled RC[70,128] and the rotation speed of rotating neurons RC[32]. Most eRC optimization currently relies on trial and error assisted by simulation models (Supplementary Note 1), and in-depth analysis is still lacking. Advanced strategies in future development should be more physics-aware and heuristic, such as hardware-in-the-loop optimization subjected to genetic algorithms.

Performance benchmark

From the viewpoint of trainable dynamic systems, eRCs exhibit several task-independent network characteristics that are important for performance evaluation and comparison[32,69]. A widely recognized characteristic is MC, indicating the network's capacity to retain historical information[13,18,64]. MC can be quantified by a random binary sequence-recalling experiment, where the input is a randomly generated i.i.d. sequence, and the expected output (after training) at any time step is the value at the i step behind the current point in the sequence. The sum of the square of correlations between the expected and actual sequences for $i = 1, 2, \ldots \infty$ is the value of MC[18]. This task tests the volume of historical information preserved in the network, which is a standard measurement for both hardware- and software-based RC. Previous work has also revealed that an RC with linear nodes normally yields much higher MC than one with nonlinear nodes, as the historical information is distorted by nonlinearity[129]. MC is particularly important in physical implementations, because maximizing MC can free the eRC system from using assistive memory and even allow all-analogue computing[32], thus reducing the cost of system implementations and operations. To achieve this, implementing delayed feedback and pre- and post-neuron rotation are effective methods. In addition, coupling the timescales between the physical nodes and input signals can yield a higher MC. Supplementary Table 2 summarizes recent eRC implementations whose MCs were clearly analysed, and an extended discussion on MC is provided in Supplementary Note 2.

As a nonlinear extension of MC, information processing capacity (IPC) has been proposed to quantify the computational capacity of dynamical systems with a fading memory condition[130]. Given the random i.i.d. inputs, the approximation target in IPC is a product of the multivariate orthogonal functions of inputs. In most cases, Legendre polynomials have been used[130–133]. The total IPC is the sum of the approximation results with different degrees of nonlinearity. IPC can simultaneously measure the capabilities of eRCs in terms of short-term memory and nonlinear approximation, which is more scientifically meaningful and hence recommended for cross-comparison.

Kernel quality (KQ) is also used to assess an eRC's ability in terms of high-dimensional nonlinear mapping. Similar to the kernel function, an ideal eRC would produce node states that are linearly separable in a higher-dimensional space. KQ can be measured by calculating the rank of a matrix composed of the state vectors produced at the end of distinct random sequence inputs[134]. For similar purposes, the richness of the state vector can be measured by the number of linearly uncoupled dynamics, for example, the number of principal components that can explain more than 90% of the state variables[19].

Furthermore, generalization rank (GR) indicates how fast the reservoir responds to the recent input sequence while minimizing the impact of initial states[32,69,134]. Note that a noise-tolerant scheme should be applied when calculating the ranks of KQ and GR by considering the noise of reservoir states. Recently, a CHARC framework was proposed to evaluate and predict eRC performance by combining MC, KQ, GR and a high-level approximation algorithm/network[135].

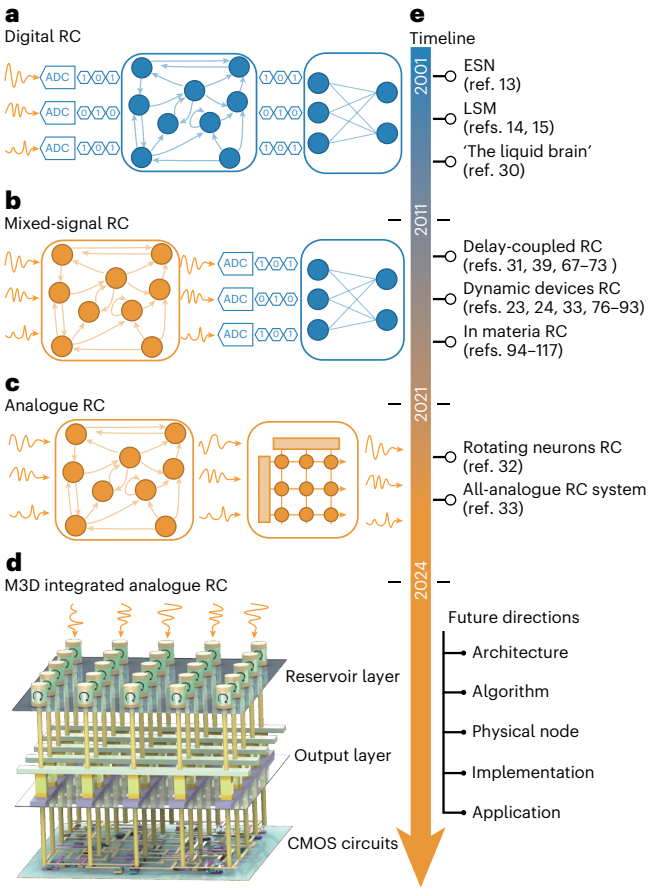

Fig. 5 | Evolution of RC implementations. a, RC was initially proposed as a machine-learning algorithm, in which the implementation was in the digital domain and its interactions with the environment were interfaced through ADC modules. **b**, Since 2011, the reservoir layer has been approximated by physical systems, and the output layer has remained in the digital domain, forming mixed-signal RC systems. **c**, Recently, eRC has been integrated with a memristor array-based output layer to create all-analogue RC, whose sensing, computing and interaction with the environment can all occur in the analogue domain. **d**, Future monolithic three-dimensional (M3D) integration of RC. The reservoir layer based on emerging devices is situated at the top of the system to interact with the environment, and the memristor-based output layer in the middle serves as a VMM accelerator. The bottom layer, composed of silicon CMOS circuits, provides the control units for the overall system. Data can be shuffled across these functional layers through high-density interlayer vias with ultrahigh bandwidth, which helps to further boost the system performance. **e**, Timeline of RC development, summarizing the milestones, the corresponding references, and the future perspectives.

Apart from these task-independent properties, eRCs have been comprehensively evaluated using benchmark tasks. To demonstrate temporal signal processing, eRCs have been used to forecast or approximate a number of classical chaotic signals and time series, including Mackey–Glass chaotic systems[24,31,32,94,116], Hénon Map time series[77] and Santa Fe laser intensity[69,122,136]. Also, nonlinear autoregressive moving average (NARMA) systems[137] are a widely accepted benchmark to test the capabilities of both nonlinear system approximation and memory[32,88,106,138]. For example, an interesting simulation[69] attempted to correlate a delay-coupled RC's KQ and GR with its performance in approximating a tenth-order NARMA system.

A wide range of machine-learning tasks have been used to evaluate the performance of eRCs in practical scenarios, including simple

waveform classification/generation[33,104], handwritten-digit recognition[23,78,85,88,90,94,106,110,128,139], spoken-digit recognition[31,77,82,139,140], biosignal processing[68,81,141], soft actuators[133,142], wearables[143], gesture classification[33,120,139] and computer vision[93]. Typical tasks implemented with eRC are summarized in Supplementary Fig. 1.

Comparing eRC implementations

The development of eRC implementations can be loosely classified into three stages (Fig. 5). In the first decade after the invention of ESN and LSM in 2001, the development of RC was focused on algorithms for the efficient training and implementation of RNNs representing a fully digital RC paradigm. Interactions with the environment were interfaced by ADCs and DACs (Fig. 5a). Later, discussions around RC and dynamical systems started the second development wave, with a focus on a physical reservoir layer. In this stage, the concept of PRC attracted researchers from different backgrounds to empower their hardware with computing capability. However, eRC remained a mainstream choice because of its compatibility with conventional circuits and large-scale implementations. The physical node of the reservoir layer was the main focus of research, and system-level considerations and evaluation were lacking. The eRC systems proposed at this stage were mainly mixed-signal systems, where the output layer was still implemented digitally (Fig. 5b). Since 2020, more and more works have attempted to implement end-to-end solutions from the system viewpoint, going beyond only engineering the reservoir layer. The designs of the input and output layers were also taken into consideration to form all-analogue RC systems (Fig. 5c), which is in line with the design primitives of neuromorphic computing[75]. The techniques commonly found in the three stages of developing eRC are summarized in Table 1. The important milestones and representative works during the development of eRC are listed in Fig. 5d.

Representative eRC implementations from recent years are surveyed and compared in Supplementary Table 3 according to the principles outlined in the following. It is important to examine to what extent eRC implementations rely on peripherals and assistive memory as compared to ideal eRC, which can be naturally embedded into the environment and perform all computing in the analogue domain. More specifically, an ideal eRC can directly receive analogue signals from sensors and output the computing results by fully using the eRC's intrinsic capacities without auxiliary processing or memory. A golden standard that can compare different eRC systems is difficult to find, but criteria important for considerations towards practical development are parallel computing, low power and system complexity.

The first factor for comparison is the architecture, considering the key features of the overall eRC system. In practice, the four fundamental RC architectures are not exclusive, and one can combine two or more architectures to form a hybrid eRC system. For instance, a delayed feedback line can be added to architectures other than delay-coupled RC to enhance the MC, and pre- and post-neuron rotation can also be added to dynamic devices or in materia RC. However, it should be noted that such a hybrid system combines not only those architectures' strengths, but also their weaknesses. Adding delay lines to an eRC would also lead to additional costs in hardware implementation.

The second factor for comparison is the physical node, which determines the nonlinearity, timescale and input/output modality. The number of physical nodes, the number of virtual or intermediate nodes and the total length of the state vector in each eRC system are also important. Virtual or intermediate nodes with numbers greater than one occur when intermediate node states are stored or time-multiplexing is used, indicating that every physical node is reused over time to increase the state richness. The length of the state vector is equal to the number of physical nodes times the number of virtual or intermediate nodes, which is actually the reservoir size in software-based RC. Meanwhile, the configuration of the output layer should also be listed for comparisons—the memristor array-based output layer should be highlighted in particular. These values can be used to estimate and compare the cost of system implementations. Finally, their demonstration tasks and results need to be compared.

A comparison of earlier works reveals four important facts: (1) eRC systems with various physical nodes are capable of performing machine-learning tasks and temporal signal processing in the physical domain; (2) the results, such as the accuracy or error rate, are acceptable, but not better than state-of-the-art gradient-based DNN algorithms due to the nature of RC algorithms; (3) because of the noise in hardware, software-based RC normally outperforms eRC under similar configurations, but exceptions can be found by exploring the rich dynamics offered by hardware behaviour[32]; (4) instead of accuracy and error rate, the merits of eRCs involve system complexity, power consumption, training cost, computing speed, and analogue and neuromorphic

Table 1 | A design toolbox for eRC

Layer	Purpose	Technique
Input layer (pre-processing)	One-dimensional input	• Spiking encoding • Filtering
	Two-dimensional input	• Spiking encoding • Image rotating/ chopping/merging
	State richness enhancement	• Time-multiplexing
	Optimization	• Input range scanning
Reservoir layer	Architecture	• Delay-coupled RC • Dynamic devices RC • In materia RC • Rotating neurons RC
	Physical node optimization	• Time-constant coupling • Nonlinear range tuning
	State richness enhancement	• Parallel eRCs with different configurations • Increase network size (number of physical nodes or length of time-multiplexing) • Increase device-to-device variation
	Memory capacity enhancement	• Delayed feedback • Pre- and post-neuron rotors • Time-constant coupling
Output layer	Training	• L_1, L_2 and logistic regression • Reinforcement learning • Gradient descent • Augmented direct feedback alignment
	Implementation	• Memristor crossbar array
Overall network	Task-independent characteristics	• Memory capacity • Kernel quality • Generalization rank • CHARC • Information processing capacity
	Performance benchmark and demonstrations	• Mackey–Glass chaotic series • Hénon map time series • Santa Fe laser time series • Simple waveform classification/generation • Handwritten-digit recognition • Spoken-digit recognition • Biosignal processing • Gesture classification • Control • ...

computing, which are particularly appealing for resource-limited applications such as edge computing and the Internet of Things (IoTs).

Outlook

Despite the recent success of eRC, various challenges still need to be addressed for the technology to deliver practical applications. Here we discuss potential future developments in five areas: architecture and connectivity, algorithm, physical node, hardware implementation and application.

Architecture and connectivity

Architecture determines the connectivity between physical nodes and how physical dynamics are used to create state vectors. A key consideration in architecture design is the enhancement of the richness of reservoir states without needlessly increasing the reservoir size. From the viewpoint of inter-node connectivity, existing architectures provide limited flexibility in defining the inter-node connections in the reservoir layer: the delay-coupled RC and rotating neurons RC use a ring structure and have sparse inter-node connections (cyclic reservoir), while dynamic devices RC lacks interaction between nodes. In in materia RC, state changes in the junction are only allowed when electrodes to which the signal is applied are adjacent. Therefore, the connectivity mainly depends on the signal transmission path, which is complicated and not readily accessible.

To improve the theoretical foundations, a systematic study of the correlation between performance metrics and different architectures is needed. Proposing a new architecture could start from the basic RC principle, keeping the feasibility of hardware implementation in mind—a PRC that costs more resources than digital RC is meaningless. Furthermore, eRC architectures can take inspiration from RC algorithms and also biological systems (Fig. 1a), exploring more bio-plausible neural activities, synaptic plasticity and even structural plasticity.

Algorithm

RC algorithms determine the computational potential of eRC. As the reservoir layer becomes deeper and wider, the in–out range of each neuron is usually not optimized, as their connections are randomly generated and fixed, unlike the gradient descent-based DNNs where the working condition of every neuron can be fine-tuned by the pre-neuron weights and bias. This leads to saturation of the RC performance at a relatively smaller network scale (usually fewer than 2,000 neurons). It is thus challenging for RC to obtain state-of-the-art results in machine-learning benchmarks, indicating its relatively weaker computational capabilities when compared to gradient descent-based DNNs[20].

Novel RC algorithms—such as the recently proposed hierarchical RC[144], deep RC[145,146], echo state graph neural networks[147] and next-generation RC[148]—could inspire innovation in eRC architectures and physical nodes. New RC algorithms could propose different requirements for physical implementations. For example, investigations in cyclic RC have enabled the development of delay-coupled RC[31] and rotating neurons RC[32].

Theoretical insight into RC algorithms could facilitate hardware implementations. For example, investigations into the input/output layer, MC and benchmarks can be directly used to optimize eRC. In addition, the algorithm work is not limited to RC itself. As a physical implementation of RNN, eRC can act as a trainable arithmetic unit in a larger physical network or system to handle more complicated scenarios, which is an open question for the future application of eRC.

Physical node

Discovering physical nodes for eRC is not demanding, as most electronics possess nonlinearity and controllable time constants. Therefore, simply replacing the physical node with another type of electronics may offer limited innovation. Instead, physical nodes that are specifically designed for eRC are preferred. In particular, the underlying mechanism determining their characteristics and how their device-to-device variations affect the eRC system performance and implementation cost is of interest. A well-designed physical node array should be able to efficiently extract uncorrelated or complementary information from the temporal input, without needlessly expanding the reservoir size and while minimizing the power consumption and implementation costs.

Hardware implementation

Demonstrating eRC performance by measuring the standalone physical nodes—while implementing the entire system through software simulation—can only validate the principle of an approach, and it is a long way from having engineered a technically sound eRC system. Therefore, future works should systematically consider all three layers, the end-to-end signal flows, power consumption, real-time demonstration and interfaces between modules.

Currently, there are three levels of eRC systems integration: measurement of discrete devices[77], circuit-level integration on printed circuit boards (PCBs)[32,33] and chip-level integration. Recently, the chip design for CMOS-implemented eRC has been reported for delay-coupled RC[72,73], and more efforts regarding chip-level integration are expected to emerge in the near future. Furthermore, monolithic 3D integration of analogue RC (Fig. 5d) could maximize the node density and interlayer connectivity[149,150], which could be an appealing perspective for the hardware implementation of RC.

Application

Generally, we should maximize the advantages of eRC in lightweight analogue computing, low-power computing and temporal signal processing, while avoiding its shortcomings in computing capability. The applications of eRC can also be considered from a continuous dynamical system viewpoint, rather than a machine-learning classifier viewpoint.

In particular, eRC is naturally well-suited to compute at the edge, in which analogue sensors are integrated for signal pre-processing or information extraction (that is, near- and in-sensor computing). eRC is also free from the memory wall limitations of digital computers. Thus, a complete all-analogue eRC system could provide ultrahigh-speed nonlinear computing, which has already been demonstrated with optical RC[38] (though this was in an implementation that was bulky and expensive). Finally, eRC could provide a solution for trainable nonlinear controllers, in which combination with feedback sensors is also possible.

References

1. Moore, G. E. Cramming more components onto integrated circuits (reprinted from Electronics, 114–117, 19 April 1965). *Proc. IEEE* **86**, 82–85 (1998).
2. Frank, D. J. et al. Device scaling limits of Si MOSFETs and their application dependencies. *Proc. IEEE* **89**, 259–288 (2001).
3. Schaller, R. R. Moore's Law: past, present and future. *IEEE Spectr.* **34**, 52–59 (1997).
4. Backus, J. Can programming be liberated from the von Neumann style? *Commun. ACM* **21**, 613–641 (1978).
5. Zhang, W. Q. et al. Neuro-inspired computing chips. *Nat. Electron.* **3**, 371–382 (2020).
6. Ielmini, D. & Wong, H. S. P. In-memory computing with resistive switching devices. *Nat. Electron.* **1**, 333–343 (2018).
7. Indiveri, G. & Liu, S. C. Memory and information processing in neuromorphic systems. *Proc. IEEE* **103**, 1379–1397 (2015).
8. Cazettes, F. et al. A reservoir of foraging decision variables in the mouse brain. *Nat. Neurosci.* **26**, 840–849 (2023).
9. Mante, V., Sussillo, D., Shenoy, K. V. & Newsome, W. T. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**, 78–84 (2013).

10. Sussillo, D. & Abbott, L. F. Generating coherent patterns of activity from chaotic neural networks. *Neuron* **63**, 544–557 (2009).
11. Enel, P., Procyk, E., Quilodran, R. & Dominey, P. F. Reservoir computing properties of neural dynamics in prefrontal cortex. *PLoS Comput. Biol.* **12**, e1004967 (2016).
12. Suárez, L. E., Richards, B. A., Lajoie, G. & Misic, B. Learning function from structure in neuromorphic networks. *Nat. Mach. Intell.* **3**, 771–786 (2021).
13. Jaeger, H. *The 'Echo State' Approach to Analysing and Training Recurrent Neural Networks—with an Erratum Note* Technical Report 148, 13 (German National Research Center for Information Technology, 2001).
This paper proposed the concept of the ESN.
14. Natschläger, T., Maass, W. W. & Markram, H. The 'liquid computer': a novel strategy for real-time computing on time series. *Spec. Issue Found. Inf. Process. TELEMATIK* **8**, 39–43 (2002).
15. Maass, W., Natschläger, T. & Markram, H. Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput.* **14**, 2531–2560 (2002).
This paper proposed the LSM.
16. Verstraeten, D., Schrauwen, B., D'Haene, M. & Stroobandt, D. An experimental unification of reservoir computing methods. *Neural Netw.* **20**, 391–403 (2007).
This paper unified ESN and LSM as reservoir computing.
17. Lukoševičius, M. A practical guide to applying echo state networks. *Lect. Notes Comput. Sci.* **7700 LECTU**, 659–686 (2012).
18. Jaeger, H. *Short Term Memory in Echo State Networks* (GMD Forschungszentrum Informationstechnik, 2002).
19. Gallicchio, C. & Micheli, A. Richness of deep echo state network dynamics. In 2019 *Advances in Computational Intelligence: 15th International Work-Conference on Artificial Neural Networks (IWANN) Part I 15* 480–491 (Springer, 2019).
20. Sun, C. et al. A systematic review of echo state networks from design to application. *IEEE Trans. Artif. Intell.* **5**, 23–37 (2022).
21. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149 (2009).
22. Chang, H. T. & Futagami, K. Reinforcement learning with convolutional reservoir computing. *Appl. Intell.* **50**, 2400–2410 (2020).
23. Du, C. et al. Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **8**, 2204 (2017).
This paper proposed dynamic devices RC.
24. Moon, J. et al. Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nat. Electron.* **2**, 480–487 (2019).
25. Sun, L. et al. In-sensor reservoir computing for language learning via two-dimensional memristors. *Sci. Adv.* **7**, eabg1455 (2021).
26. Nakajima, M. et al. Physical deep learning with biologically inspired training method: gradient-free approach for physical hardware. *Nat. Commun.* **13**, 7847 (2022).
27. Cucchi, M., Abreu, S., Ciccone, G., Brunner, D. & Kleemann, H. Hands-on reservoir computing: a tutorial for practical implementation. *Neuromorphic Comput. Eng.* **2**, 032002 (2022).
28. Tanaka, G. et al. Recent advances in physical reservoir computing: a review. *Neural Netw.* **115**, 100–123 (2019).
This paper reviews more general PRC.
29. Nakajima, K. Physical reservoir computing-an introductory perspective. *Jpn J. Appl. Phys.* **59**, 060501 (2020).
30. Fernando, C. & Sojakka, S. in *Advances in Artificial Life* (eds Banzhaf, W. et al.) 588–597 (Springer, 2003).
This paper fulfilled the idea of LSM in physical domain.
31. Appeltant, L. et al. Information processing using a single dynamical node as complex system. *Nat. Commun.* **2**, 468 (2011).
This paper proposed delay-coupled RC.
32. Liang, X. et al. Rotating neurons for all-analog implementation of cyclic reservoir computing. *Nat. Commun.* **13**, 1549 (2022).
This paper proposed rotating neurons RC.
33. Zhong, Y. N. et al. A memristor-based analogue reservoir computing system for real-time and power-efficient signal processing. *Nat. Electron.* **5**, 672–681 (2022).
This paper reports the design an all-analogue dynamic devices RC system.
34. Vandoorne, K. et al. Toward optical signal processing using photonic reservoir computing. *Opt. Express* **16**, 11182–11192 (2008).
35. Duport, F., Schneider, B., Smerieri, A., Haelterman, M. & Massar, S. All-optical reservoir computing. *Opt. Express* **20**, 22783–22795 (2012).
36. Larger, L. et al. Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing. *Opt. Express* **20**, 3241–3249 (2012).
37. Paquot, Y. et al. Optoelectronic reservoir computing. *Sci. Rep.* **2**, 287 (2012).
38. Brunner, D., Soriano, M. C., Mirasso, C. R. & Fischer, I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat. Commun.* **4**, 1364 (2013).
39. Larger, L. et al. High-speed photonic reservoir computing using a time-delay-based architecture: million words per second classification. *Phys. Rev. X* **7**, 011015 (2017).
40. Brunner, D., Soriano, M. C. & Van der Sande, G. *Photonic Reservoir Computing: Optical Recurrent Neural Networks* (Walter de Gruyter, 2019).
41. Rafayelyan, M., Dong, J., Tan, Y. Q., Krzakala, F. & Gigane, S. Large-scale optical reservoir computing for spatiotemporal chaotic systems prediction. *Phys. Rev. X* **10**, 041037 (2020).
42. Dambre, J. et al. in *Reservoir Computing: Theory, Physical Implementations and Applications* (eds Nakajima, K. & Fischer, I.) 397–419 (Springer, 2021).
43. Nakajima, M., Tanaka, K. & Hashimoto, T. Scalable reservoir computing on coherent linear photonic processor. *Commun. Phys.* **4**, 20 (2021).
44. Nakajima, K. et al. A soft body as a reservoir: case studies in a dynamic model of octopus-inspired soft robotic arm. *Front. Comput. Neurosci.* **7**, 91 (2013).
45. Caluwaerts, K. et al. Design and control of compliant tensegrity robots through simulation and hardware validation. *J. R. Soc. Interface* **11**, 20140520 (2014).
46. Nakajima, K., Li, T., Hauser, H. & Pfeifer, R. Exploiting short-term memory in soft body dynamics as a computational resource. *J. R. Soc. Interface* **11**, 20140437 (2014).
47. Nakajima, K., Hauser, H., Li, T. & Pfeifer, R. Information processing via physical soft body. *Sci. Rep.* **5**, 10487 (2015).
48. Bhovad, P. & Li, S. Physical reservoir computing with origami and its application to robotic crawling. *Sci. Rep.* **11**, 13002 (2021).
49. Tanaka, K. et al. Flapping-wing dynamics as a natural detector of wind direction. *Adv. Intell. Syst.* **3**, 2000174 (2021).
50. Sakurai, R., Nishida, M., Jo, T., Wakao, Y. & Nakajima, K. Durable pneumatic artificial muscles with electric conductivity for reliable physical reservoir computing. *J. Robot. Mechatron.* **34**, 240–248 (2022).
51. Tanaka, K. et al. Self-organization of remote reservoirs: transferring computation to spatially distant locations. *Adv. Intell. Syst.* **4**, 2100166 (2021).
52. Hauser, H. in *Reservoir Computing Natural Computing Series* (eds Nakajima, K. & Fischer, I.) Ch. 8 (Springer, 2021).
53. Fujii, K. & Nakajima, K. Harnessing disordered-ensemble quantum dynamics for machine learning. *Phys. Rev. Appl.* **8**, 024030 (2017).
54. Ghosh, S., Paterek, T. & Liew, T. C. H. Quantum neuromorphic platform for quantum state preparation. *Phys. Rev. Lett.* **123**, 260404 (2019).

55. Chen, J. Y., Nurdin, H. I. & Yamamoto, N. Temporal information processing on noisy quantum computers. *Phys. Rev. Appl.* **14**, 024065 (2020).
56. Martinez-Pena, R., Giorgi, G. L., Nokkala, J., Soriano, M. C. & Zambrini, R. Dynamical phase transitions in quantum reservoir computing. *Phys. Rev. Lett.* **127**, 100502 (2021).
57. Kubota, T. et al. Temporal information processing induced by quantum noise. *Phys. Rev. Res.* **5**, 023057 (2023).
58. Tran, Q. H. & Nakajima, K. Learning temporal quantum tomography. *Phys. Rev. Lett.* **127**, 260401 (2021).
59. Mujal, P. et al. Opportunities in quantum reservoir computing and extreme learning machines. *Adv. Quantum Technol.* **4**, 2100027 (2021).
60. Fujii, K. & Nakajima, K. in *Reservoir Computing Natural Computing Series* (eds Nakajima, K. & Fischer, I.) Ch. 18 (Springer, 2021).
61. Ghosh, S., Nakajima, K., Krisnanda, T., Fujii, K. & Liew, T. C. H. Quantum neuromorphic computing with reservoir computing networks. *Adv. Quantum Technol.* **4**, 2100053 (2021).
62. Nakajima, K. & Fischer, I. *Reservoir Computing* (Springer, 2021).
63. Cao, J. et al. Emerging dynamic memristors for neuromorphic reservoir computing. *Nanoscale* **14**, 289–298 (2022).
64. Rodan, A. & Tino, P. Minimum complexity echo state network. *IEEE Trans. Neural Netw.* **22**, 131–144 (2011).
65. Ortin, S. et al. A unified framework for reservoir computing and extreme learning machines based on a single time-delayed neuron. *Sci. Rep.* **5**, 14945 (2015).
66. Soriano, M. C., Brunner, D., Escalona-Moran, M., Mirasso, C. R. & Fischer, I. Minimal approach to neuro-inspired information processing. *Front. Comput. Neurosci.* **9**, 68 (2015).
67. Soriano, M. C. et al. Delay-based reservoir computing: noise effects in a combined analog and digital implementation. *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 388–393 (2015).
68. Liang, X., Li, H., Vuckovic, A., Mercer, J. & Heidari, H. A neuromorphic model with delay-based reservoir for continuous ventricular heartbeat detection. *IEEE Trans. Biomed. Eng.* **69**, 1837–1849 (2022).
69. Appeltant, L. *Reservoir Computing based on Delay-Dynamical Systems*. PhD thesis, Univ. Illes Balears (2012).
70. Ortín, S. & Pesquera, L. Tackling the trade-off between information processing capacity and rate in delay-based reservoir computers. *Front. Phys.* **7**, 210 (2019).
71. Stelzer, F., Rohm, A., Ludge, K. & Yanchuk, S. Performance boost of time-delay reservoir computing by non-resonant clock cycle. *Neural Netw.* **124**, 158–169 (2020).
72. Bai, K. J., Liu, L. J. & Yi, Y. Spatial-temporal hybrid neural network with computing-in-memory architecture. *IEEE Trans. Circuits Syst. I Regul. Pap.* **68**, 2850–2862 (2021).
73. Chandrasekaran, S. T., Bhanushali, S. P., Banerjee, I. & Sanyal, A. Toward real-time, at-home patient health monitoring using reservoir computing CMOS IC. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **11**, 829–839 (2021).
74. Van der Sande, G., Brunner, D. & Soriano, M. C. Advances in photonic reservoir computing. *Nanophotonics* **6**, 561–576 (2017).
75. Kendall, J. D. & Kumar, S. The building blocks of a brain-inspired computer. *Appl. Phys. Rev.* **7**, 011305 (2020).
76. Nakajima, K., Fujii, K., Negoro, M., Mitarai, K. & Kitagawa, M. Boosting computational power through spatial multiplexing in quantum reservoir computing. *Phys. Rev. Appl.* **11**, 034021 (2019).
77. Zhong, Y. et al. Dynamic memristor-based reservoir computing for high-efficiency temporal signal processing. *Nat. Commun.* **12**, 408 (2021).
78. Yang, J. et al. Tunable synaptic characteristics of a Ti/TiO_2/Si memory device for reservoir computing. *ACS Appl. Mater. Interfaces* **13**, 33244–33252 (2021).
79. Wang, T., Huang, H. M., Wang, X. X. & Guo, X. An artificial olfactory inference system based on memristive devices. *Infomat* **3**, 804–813 (2021).
80. Jaafar, A. H. et al. 3D-structured mesoporous silica memristors for neuromorphic switching and reservoir computing. *Nanoscale* **14**, 17170–17181 (2022).
81. Zhu, X., Wang, Q. & Lu, W. D. Memristor networks for real-time neural activity analysis. *Nat. Commun.* **11**, 2439 (2020).
82. Torrejon, J. et al. Neuromorphic computing with nanoscale spintronic oscillators. *Nature* **547**, 428–431 (2017).
83. Tang, M. F. et al. A compact fully ferroelectric-FETs reservoir computing network with sub-100-ns operating speed. *IEEE Electron Device Lett.* **43**, 1555–1558 (2022).
84. Yu, J. et al. Energy efficient and robust reservoir computing system using ultrathin (3.5 nm) ferroelectric tunneling junctions for temporal data learning. In *2021 IEEE Symposium on VLSI Technology* 1–2 (IEEE, 2021).
85. Duong, N. T. et al. Dynamic ferroelectric transistor-based reservoir computing for spatiotemporal information processing. *Adv. Intell. Syst.* **5**, 2300009 (2023).
86. Chen, Z. et al. All-ferroelectric implementation of reservoir computing. *Nat. Commun.* **14**, 3585 (2023).
87. Liang, X. C., Luo, Y. Y., Pei, Y. L., Wang, M. Y. & Liu, C. Multimode transistors and neural networks based on ion-dynamic capacitance. *Nat. Electron.* **5**, 859–869 (2022).
88. Nishioka, D. et al. Edge-of-chaos learning achieved by ion-electron-coupled dynamics in an ion-gating reservoir. *Sci. Adv.* **8**, eade1156 (2022).
89. Liu, K. et al. An optoelectronic synapse based on α-In_2Se_3 with controllable temporal dynamics for multimode and multiscale reservoir computing. *Nat. Electron.* **5**, 761–773 (2022).
90. Jang, Y. H. et al. Time-varying data processing with nonvolatile memristor-based temporal kernel. *Nat. Commun.* **12**, 5727 (2021).
91. Du, W. et al. An optoelectronic reservoir computing for temporal information processing. *IEEE Electron Device Lett.* **43**, 406–409 (2022).
92. Zhang, Z. et al. In-sensor reservoir computing system for latent fingerprint recognition with deep ultraviolet photo-synapses and memristor array. *Nat. Commun.* **13**, 6590 (2022).
93. Lao, J. et al. Ultralow-power machine vision with self-powered sensor reservoir. *Adv. Sci.* **9**, e2106092 (2022).
94. Milano, G. et al. In materia reservoir computing with a fully memristive architecture based on self-organizing nanowire networks. *Nat. Mater.* **21**, 195–202 (2022).
95. Sillin, H. O. et al. A theoretical and experimental study of neuromorphic atomic switch networks for reservoir computing. *Nanotechnology* **24**, 384004 (2013).
This paper proposed in materia RC.
96. Kan, S. H. et al. Simple reservoir computing capitalizing on the nonlinear response of materials: theory and physical implementations. *Phys. Rev. Appl.* **15**, 024030 (2021).
97. Tanaka, H. et al. In-materio computing in random networks of carbon nanotubes complexed with chemically dynamic molecules: a review. *Neuromorphic Comput. Eng.* **2**, 022002 (2022).
98. Diaz-Alvarez, A. et al. Emergent dynamics of neuromorphic nanowire networks. *Sci. Rep.* **9**, 14920 (2019).
99. Daniels, R. K. et al. Reservoir computing with 3D nanowire networks. *Neural Netw.* **154**, 122–130 (2022).
100. Hochstetter, J. et al. Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nat. Commun.* **12**, 4008 (2021).
101. Milano, G., Montano, K. & Ricciardi, C. In materia implementation strategies of physical reservoir computing with memristive nanonetworks. *J. Phys. D* **56**, 084005 (2023).

102. Lilak, S. et al. Spoken digit classification by in-materio reservoir computing with neuromorphic atomic switch networks. *Front. Nanotechnol*. https://doi.org/10.3389/fnano.2021.675792 (2021).
103. Tanaka, H. et al. A molecular neuromorphic network device consisting of single-walled carbon nanotubes complexed with polyoxometalate. *Nat. Commun.* **9**, 2693 (2018).
104. Usami, Y. et al. In-materio reservoir computing in a sulfonated polyaniline network. *Adv. Mater.* **33**, e2102688 (2021).
105. Cucchi, M. et al. Reservoir computing with biocompatible organic electrochemical networks for brain-inspired biosignal classification. *Sci. Adv.* **7**, eabh0693 (2021).
106. Jiang, W. C. et al. Physical reservoir computing using magnetic skyrmion memristor and spin torque nano-oscillator. *Appl. Phys. Lett.* **115**, 192403 (2019).
107. Nako, E., Toprasertpong, K., Nakane, R., Takenaka, M. & Takagi, S. Experimental demonstration of novel scheme of HZO/Si FeFET reservoir computing with parallel data processing for speech recognition. In *2022 IEEE Symposium on VLSI Technology and Circuits* (*VLSI Technology and Circuits*) 220–221 (IEEE, 2022).
108. Toprasertpong, K. et al. Reservoir computing on a silicon platform with a ferroelectric field-effect transistor. *Commun. Eng.* **1**, 21 (2022).
109. Liu, K. et al. Multilayer reservoir computing based on ferroelectric α-In_2Se_3 for hierarchical information processing. *Adv. Mater.* **34**, e2108826 (2022).
110. Momeni, A. & Fleury, R. Electromagnetic wave-based extreme deep learning with nonlinear time-Floquet entanglement. *Nat. Commun.* **13**, 2651 (2022).
111. Marcucci, G., Pierangeli, D. & Conti, C. Theory of neuromorphic computing by waves: machine learning by rogue waves, dispersive shocks and solitons. *Phys. Rev. Lett.* **125**, 093901 (2020).
112. Silva, N. A., Ferreira, T. D. & Guerreiro, A. Reservoir computing with solitons. *New J. Phys.* **23**, 023013 (2021).
113. Maksymov, I. S. & Pototsky, A. Reservoir computing based on solitary-like waves dynamics of liquid film flows: a proof of concept. *Europhys. Lett.* **142**, 43001 (2023).
114. Nakane, R., Hirose, A. & Tanaka, G. Spin waves propagating through a stripe magnetic domain structure and their applications to reservoir computing. *Phys. Rev. Res.* **3**, 1–15 (2021).
115. Nakane, R., Hirose, A. & Tanaka, G. Performance enhancement of a spin-wave-based reservoir computing system utilizing different physical conditions. *Phys. Rev. Appl.* **19**, 034047 (2023).
116. Gartside, J. C. et al. Reconfigurable training and reservoir computing in an artificial spin–vortex ice via spin–wave fingerprinting. *Nat. Nanotechnol.* **17**, 460–469 (2022).
117. Zhu, R. et al. Information dynamics in neuromorphic nanowire networks. *Sci. Rep.* **11**, 13047 (2021).
118. Vidamour, I. T. et al. Reconfigurable reservoir computing in a magnetic metamaterial. *Commun. Phys.* **6**, 230 (2023).
119. Marković, D. et al. Reservoir computing with the frequency, phase and amplitude of spin–torque nano-oscillators. *Appl. Phys. Lett.* **114**, 12409 (2019).
120. Sun, W. et al. 3D reservoir computing with high area efficiency (5.12 TOPS/mm^2) implemented by 3D dynamic memristor array for temporal signal processing. In *2022 IEEE Symposium on VLSI Technology and Circuits* (*VLSI Technology and Circuits*) 222–223 (IEEE, 2022).
121. Kuriki, Y., Nakayama, J., Takano, K. & Uchida, A. Impact of input mask signals on delay-based photonic reservoir computing with semiconductor lasers. *Opt. Express* **26**, 5777–5788 (2018).
122. Appeltant, L., Van der Sande, G., Danckaert, J. & Fischer, I. Constructing optimized binary masks for reservoir computing with delay systems. *Sci. Rep.* **4**, 3629 (2014).
123. LeCun, Y. *The MNIST Database of Handwritten Digits* (1998); http://yann.lecun.com/exdb/mnist/
124. Yu, J. et al. Energy efficient and robust reservoir computing system using ultrathin (3.5 nm) ferroelectric tunneling junctions for temporal data learning. *Proc. 2021 Symp. VLSI Technol.* **2**, 16–14 (2021).
125. Yao, P. et al. Fully hardware-implemented memristor convolutional neural network. *Nature* **577**, 641–646 (2020).
126. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).
127. Jaeger, H. & Haas, H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science* **304**, 78–80 (2004).
128. Stelzer, F., Rohm, A., Vicente, R., Fischer, I. & Yanchuk, S. Deep neural networks using a single neuron: folded-in-time architecture using feedback-modulated delay loops. *Nat. Commun.* **12**, 5164 (2021).
129. Inubushi, M. & Yoshimura, K. Reservoir computing beyond memory-nonlinearity trade-off. *Sci. Rep.* **7**, 10199 (2017).
130. Dambre, J., Verstraeten, D., Schrauwen, B. & Massar, S. Information processing capacity of dynamical systems. *Sci. Rep.* **2**, 514 (2012).
131. Kubota, T., Takahashi, H. & Nakajima, K. Unifying framework for information processing in stochastically driven dynamical systems. *Phys. Rev. Res.* **3**, 043135 (2021).
132. Vettelschoss, B., Rohm, A. & Soriano, M. C. Information processing capacity of a single-node reservoir computer: an experimental evaluation. *IEEE Trans. Neural Netw. Learn. Syst.* **33**, 2714–2725 (2022).
133. Akashi, N. et al. Input-driven bifurcations and information processing capacity in spintronics reservoirs. *Phys. Rev. Res.* **2**, 043303 (2020).
134. Legenstein, R. & Maass, W. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Netw.* **20**, 323–334 (2007).
135. Dale, M., Miller, J. F., Stepney, S. & Trefzer, M. A. A substrate-independent framework to characterize reservoir computers. *Proc. Math. Phys. Eng. Sci.* **475**, 20180723 (2019).
136. Soriano, M. C. et al. Optoelectronic reservoir computing: tackling noise-induced performance degradation. *Opt. Express* **21**, 12–20 (2013).
137. Atiya, A. F. & Parlos, A. G. New results on recurrent network training: unifying the algorithms and accelerating convergence. *IEEE Trans. Neural Netw.* **11**, 697–709 (2000).
138. Bai, K. J. & Yi, Y. DFR: An energy-efficient analog delay feedback reservoir computing system for brain-inspired computing. *ACM J. Emerg. Technol. Comput. Syst.* **14**, 1–22 (2018).
139. Sun, J. et al. Novel nondelay-based reservoir computing with a single micromechanical nonlinear resonator for high-efficiency information processing. *Microsyst. Nanoeng.* **7**, 83 (2021).
140. Dion, G., Mejaouri, S. & Sylvestre, J. Reservoir computing with a single delay-coupled non-linear mechanical oscillator. *J. Appl. Phys.* **124**, 152132 (2018).
141. Donati, E. et al. Processing EMG signals using reservoir computing on an event-based neuromorphic system. In *2018 IEEE Biomedical Circuits and Systems Conference* (*BioCAS*) 1–4 (IEEE, 2018).
142. Kan, S., Nakajima, K., Asai, T. & Akai-Kasaya, M. Physical implementation of reservoir computing through electrochemical reaction. *Adv. Sci.* **9**, e2104076 (2022).
143. Akashi, N. et al. A coupled spintronics neuromorphic approach for high-performance reservoir computing. *Adv. Intell. Syst.* **4**, 2200123 (2022).
144. Moon, J., Wu, Y. & Lu, W. D. Hierarchical architectures in reservoir computing systems. *Neuromorphic Comput. Eng.* **1**, 014006 (2021).
145. Gallicchio, C. & Micheli, A. in *Reservoir Computing Natural Computing Series* (eds Nakajima, K. & Fischer, I.) Ch. 4 (Springer, 2021).
146. Gallicchio, C., Micheli, A. & Pedrelli, L. Deep reservoir computing: a critical experimental analysis. *Neurocomputing* **268**, 87–99 (2017).

147. Wang, S. et al. Echo state graph neural networks with analogue random resistive memory arrays. *Nat. Mach. Intell.* **5**, 104–113 (2023).
148. Gauthier, D. J., Bollt, E., Griffith, A. & Barbosa, W. A. S. Next generation reservoir computing. *Nat. Commun.* **12**, 5564 (2021).
149. Li, Y. et al. Monolithic 3D integration of logic, memory and computing-in-memory for one-shot learning. In *2021 IEEE International Electron Devices Meeting* (*IEDM*) 21.25.21–21.25.24 (IEEE, 2021).
150. An, R. et al. A hybrid computing-in-memory architecture by monolithic 3D integration of BEOL CNT/IGZO-based CFET logic and analog RRAM. In *2022 International Electron Devices Meeting* (*IEDM*) 18.11.11–18.11.14 (IEEE, 2022).

Acknowledgements

This work was in part supported by STI 2030-Major Projects 2022ZD0210200, the National Natural Science Foundation of China (92264201, 62025111 and 62104126) and the XPLORER Prize. X.L. is supported by the Shuimu Tsinghua Scholar Program of Tsinghua University.

Author contributions

X.L. and J.T. conceived the idea and wrote the paper. All authors discussed and commented on the paper.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41928-024-01133-z.

Correspondence and requests for materials should be addressed to Jianshi Tang.

Peer review information *Nature Electronics* thanks Cheol Seong Hwang, Hans Kleemann, Wei Lu and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.