Harnessing Disordered-Ensemble Quantum Dynamics for Machine Learning

Keisuke Fujii[1,2,3,4] and Kohei Nakajima[2,4,5]

[1]*Photon Science Center, Graduate School of Engineering, The University of Tokyo,*
2-11-16 Yayoi, Bunkyo-ku, Tokyo 113-8656, Japan
[2]*The Hakubi Center for Advanced Research, Kyoto University,*
Yoshida-Ushinomiya-cho, Sakyo-ku, Kyoto 606-8302, Japan
[3]*Department of Physics, Graduate School of Science, Kyoto University,*
Kitashirakawa Oiwake-cho, Sakyo-ku, Kyoto 606-8502, Japan
[4]*JST, PRESTO, 4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan*
[5]*Graduate School of Information Science and Technology,*
The University of Tokyo, Bunkyo-ku, 113-8656 Tokyo, Japan

The quantum computer has an amazing potential of fast information processing. However, the realization of a digital quantum computer is still a challenging problem requiring highly accurate controls and key application strategies. Here we propose a platform, quantum reservoir computing, to solve these issues successfully by exploiting the natural quantum dynamics of ensemble systems, which are ubiquitous in laboratories nowadays, for machine learning. This framework enables ensemble quantum systems to universally emulate nonlinear dynamical systems including classical chaos. A number of numerical experiments show that quantum systems consisting of 5–7 qubits possess computational capabilities comparable to conventional recurrent neural networks of 100–500 nodes. This discovery opens up a paradigm for information processing with artificial intelligence powered by quantum physics.

I. INTRODUCTION

Quantum physics, which is one of the most fundamental frameworks of physics, exhibits rich dynamics, sufficient to explain natural phenomena in microscopic worlds. As Feynman pointed out [1], the simulation of quantum systems on classical computers is extremely challenging because of the high complexity of these systems. Instead, they should be simulated by using a machine of which the operation is based on the laws of quantum physics.

Motivated by the recent rapid experimental progress in controlling complex quantum systems, nonconventional information processing utilizing quantum physics has been explored in the field of quantum information science [2,3]. For example, certain mathematical problems, such as integer factorization, which are believed to be intractable on a classical computer, are known to be efficiently solvable by a sophisticatedly synthesized quantum algorithm [4]. Therefore, considerable experimental effort has been devoted to realizing full-fledged universal quantum computers [5,6]. On the other hand, quantum simulators are thought to be much easier to implement than a full-fledged universal quantum computer. In this regard, existing quantum simulators have already shed new light on the physics of complex many-body quantum systems [7–9], and a restricted class of quantum dynamics, known as adiabatic dynamics, has also been applied to combinatorial optimization problems [10–13]. However, complex real-time quantum dynamics, which is one of the most difficult

tasks for classical computers to simulate [14–16] and has great potential to perform nontrivial information processing, is now waiting to be harnessed as a resource for more general-purpose information processing. Specifically, the recent rapid progress in sensing and Internet technologies has resulted in an increasing demand for fast, intelligent big-data analysis with low energy consumption. This has motivated us to develop brain-inspired information-processing devices of a non–von Neumann type, on which machine-learning tasks are able to run natively [17].

Here we propose a framework to exploit the complexity of real-time quantum dynamics in ensemble quantum systems for nonlinear and temporal learning problems. These problems include a variety of real-world tasks such as time-dependent signal processing, speech recognition, natural language processing, sequential motor control of robots, and stock-market predictions. Our approach is based on a machine-learning technique inspired by the way the brain processes information, so-called *reservoir computing* [18–20]. In particular, this framework focuses on real-time computing with time-varying input that requires the use of memory, unlike feedforward neural networks. In this framework, the low-dimensional input is projected to a high-dimensional dynamical system, which is typically referred to as a *reservoir*, generating transient dynamics that facilitates the separation of input states [21]. If the dynamics of the reservoir involves both adequate memory and nonlinearity [22], emulating nonlinear dynamical systems requires only adding a linear and static

readout from the high-dimensional state space of the reservoir.

A number of different implementations of reservoirs have been proposed, such as abstract dynamical systems for echo-state networks (ESNs) [18] or models of neurons for liquid-state machines [19]. The implementations are not limited to programs running on the PC but also include physical systems, such as the surface of water in a laminar state [23], analog circuits and optoelectronic systems [24–29], and neuromorphic chips [30]. Recently, it has been reported that the mechanical bodies of soft and compliant robots have also been successfully used as a reservoir [31–36]. In contrast to the refinements required by learning algorithms, such as in deep learning [37], the approach followed by reservoir computing, especially when applied to real systems, is to find an appropriate form of physics that exhibits rich dynamics, thereby allowing us to outsource a part of the computation. Nevertheless, no quantum physical system has been employed yet as a physical reservoir.

Here we formulate quantum reservoir computing (QRC) and show, through a number of numerical experiments, that disordered quantum dynamics can be used as a powerful reservoir. Although there have been several prominent proposals on utilizing quantum physics in the context of machine learning [38–43], they are based on sophisticatedly synthesized quantum circuits on a full-fledged universal quantum computer. Contrary to these software approaches, the approach followed by QRC is to exploit the complexity of natural (disordered) quantum dynamics for information processing, as it is. Instead of using quantum gates, which contrasts our approach to the digital one, we employ analog quantum dynamics under a time-independent Hamiltonian. The parameters in the Hamiltonian are randomly chosen, and hence no fine-tuning of the parameters of the Hamiltonian is required. Any quantum-chaotic (nonintegrable) system can be harnessed, in general, and its computational capabilities are specified. This is a great advantage, because we can utilize existing quantum simulators or complex quantum systems as resources to boost information processing. Among existing works on quantum machine learning [38–41,43], our approach attempts to exploit quantum systems for temporal machine-learning tasks, which essentially require a memory effect to the system. As we see below, our benchmark results show that quantum systems consisting of 5–7 qubits exhibit a powerful performance, with the help of time multiplexing, comparable to the ESNs of 100–500 nodes [44]. Not only its computational power, QRC also provides us an operational means to approach complex real-time quantum dynamics. While there had been a missing operational link between classical chaos and quantum-chaotic systems manifested by a Wigner-Dyson-type statistics of the energy-level spacing [45,46], it is quite natural to connect them via the QRC framework naturally as an emulation of classical chaos by quantum-chaotic systems. Moreover, since complex quantum dynamics is ubiquitous,

this framework provides us an operational understanding of quantum physics, such as quantum simulation, thermodynamics in a closed quantum system, and fast scrambling in a black hole.

The rest of this paper is organized as follows. In Sec. II, we provide a detailed description of QRC, introducing quantum mechanics and how it is related to nonlinear temporal machine learning. In Sec. III, we provide demonstrations of QRC for prototypical nonlinear machine-learning tasks. These demonstrations are aimed to explore potential applications of QRC in practical problems. In Sec. IV, the performances of QRC are investigated in detail with respect to their memory and nonlinear capacities and compared to a classical reservoir computing (ESNs) as a reference. The effect of imperfections in experimental settings is also argued there. Section V is devoted to discussion. The detailed settings and methods of the numerical experiments are provided in the Appendix.

II. QUANTUM-RESERVOIR COMPUTING

A. Description of quantum system and dynamics

In this subsection, we explain how to describe the quantum system and dynamics for readers who are not familiar with quantum information. The minimum unit of information in quantum physics is a quantum bit (qubit), which consists of a two-level quantum system, namely, a vector in a two-dimensional complex vector space spanned by $\{|0\rangle, |1\rangle\}$. Let us consider a quantum system consisting of N qubits, which is described as a tensor product space of a complex vector space of two dimensions. (Below, we use the number of qubits, N, to measure the size of a quantum system, since it reflects a physical size of the system, like the number of atoms or particles.) A pure quantum state is represented by a state vector $|\psi\rangle$ in a 2^N-dimensional complex vector space. We may also consider a statistical (classical) mixture of the states of the pure states, which can be described by a $2^N \times 2^N$ Hermitian matrix ρ known as a density matrix. For a closed quantum system, the time evolution for a time interval τ is given by a unitary operator $e^{-iH\tau}$ generated by a Hermitian operator H called the Hamiltonian. Specifically, for the density matrix the time evolution is given by

$$\rho(t + \tau) = e^{-iH\tau}\rho(t)e^{iH\tau}, \tag{1}$$

where the Hamiltonian H is a $2^N \times 2^N$ Hermitian matrix and defines the dynamics of the quantum system.

B. Measurements in ensemble quantum systems

Measurements in a quantum system are described by a set of projective operators $\{P_i\}$, which satisfies $\sum_i P_i = I$ and $P_i P_j = \delta_{ij} P_i$. Then the probability to obtain the measurement outcome i for the state ρ is given by $p_i = \mathrm{Tr}[P_i \rho]$. The state after the measurement gets a

backaction and is given by $P_i\rho P_i/\text{Tr}[P_i\rho]$, that is, a single quantum system inevitably disturbed by the projective measurement. By repeating the projective measurements, we can calculate average values $\langle O \rangle := \text{Tr}[O\rho]$ of an observable $O = \sum_i a_i P_i$.

Here we consider an ensemble quantum system, where the system consists of a huge number of the copies of ρ, i.e., $\rho^{\otimes m}$. Cold atomic ensembles and liquid- or solid-state molecules are natural candidates of such an ensemble quantum system. For example, in a NMR (nuclear-magnetic-resonance) spin-ensemble system, we have typically 10^{18-20} copies of the same molecules [47,48]. Their nuclear spin degrees of freedom can be employed as the quantum system, like NMR spin-ensemble quantum computers or synthetic dimensions of ultracold atoms for quantum simulations. We here assume that we can obtain the signals as a macroscopic observable from the ensemble quantum system directly, where the ensemble quantum system and the probe system are coupled by an extremely weak interaction. Actually, the NMR bulk-ensemble average measurement is done in this way. There is almost no backaction, or backaction is much smaller than other imperfections like the T_1 relaxation [47,48]. In QRC, we make active use of such a property of the ensemble quantum systems to exploit the complex quantum dynamics on the large degrees of freedom.

C. Definition of quantum-reservoir dynamics

As nodes of the network of the QR, we use an orthogonal basis of quantum states. The idea is similar to the quantum walks [39,49,50], where each individual node is defined not by qubits (subsystems) but by basis states like $\{|000\rangle, |001\rangle, ..., |111\rangle\}$. Therefore, for N qubits, we have 2^N basis states for a pure quantum state. Moreover, here we employ the density matrix, in general; we define the nodes of the network by an orthogonal basis of the operator space of the density matrices. By using the Hilbert-Schmidt inner product, the density matrix can be represented as a vector $\mathbf{x}$ on a 4^N-dimensional operator space. Here the ith coefficient x_i of $\mathbf{x}$ is defined by $x_i = \text{Tr}[B_i\rho]$ by using the set of N-qubit products of the Pauli operators $\{B_i\}_{i=1}^{4^N} = \{I, X, Y, Z\}^{\otimes N}$ (where $B_i B_j = \delta_{ij}I$). Specifically, we choose the first N elements such that $B_i = Z_i$ for convenience in the definition of the observables later.

In this operator space, the time evolution is reformulated as a linear map for the vector $\mathbf{x}$:

$$\mathbf{x}(t + \tau) = U_\tau \mathbf{x}(t). \tag{2}$$

Here U_τ is a $4^N \times 4^N$ matrix whose element is defined by

$$(U_\tau)_{ji} := \text{Tr}[B_j e^{-iH\tau} B_i e^{iH\tau}]. \tag{3}$$

Owing to the unitarity of the dynamics $e^{-iH\tau}(e^{-iH\tau})^\dagger = I$, we have $U_\tau U_\tau^T = I$. If the system is coupled to an external system for a measurement and/or a feedback operation, the time evolution (for the density matrix) is not given by the conjugation of the unitary operator $e^{-iH\tau}$; instead, it is generally given by a complete positive trace-preserving (CPTP) map $\mathcal{D}$ for the density matrix ρ. Even in such a case, the dynamics is linear, and hence the time evolution for $\mathbf{x}(t)$ is given in a linear form:

$$\mathbf{x} \rightarrow W\mathbf{x}, \tag{4}$$

where the matrix element is defined

$$W_{ji} := \text{Tr}[B_j \mathcal{D}(B_i)]. \tag{5}$$

In order to exploit quantum dynamics for information processing, we have to introduce an input and the signals of the quantum system [see Fig. 1(a)]. Suppose $\{s_k\}_{k=1}^M$ is an input sequence, where s_k can be a binary ($s_k \in \{0, 1\}$) or a continuous variable ($s_k \in [0, 1]$). A temporal learning task here is to find, using the quantum system, a nonlinear function $y_k = f(\{s_l\}_{l=1}^k)$ such that the mean-square error between y_k and a target (teacher) output $\bar{y}_k$ for a given task becomes minimum. To do so, at each time $t = k\tau$, the input signal s_k is injected into a qubit, say, the first qubit,
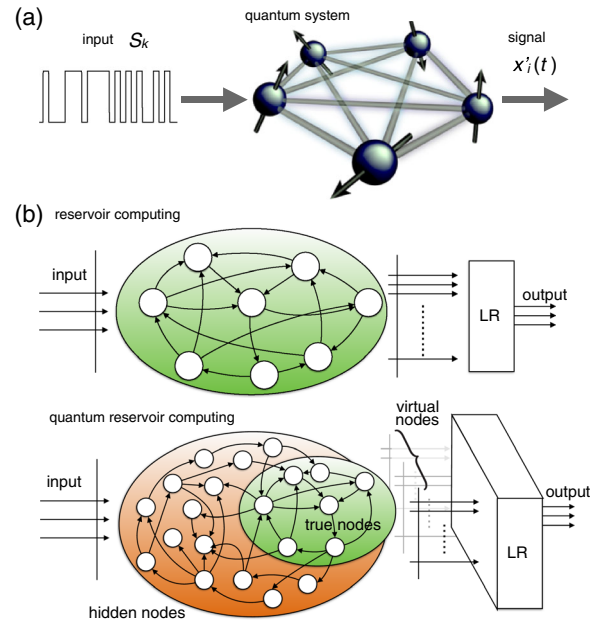

FIG. 1. Information-processing scheme in QRC. (a) The input sequence $\{s_k\}$ is injected into the quantum system. The signal $x_i'(t)$ is obtained from each qubit. (b) Comparison between conventional (upper) and quantum (lower) reservoir-computing approaches. Note that the circles in the QRC do not represent qubits but the basis of the Hilbert space like the nodes in a quantum walk [39,49,50]. The true nodes correspond to a subset of basis of the operator space that are directly monitored by the ensemble measurements. The hidden nodes correspond to the remaining degrees of freedom.

by replacing (or by using measurement and feedback) the first qubit with the state $\rho_{s_k} = |\psi_{s_k}\rangle\langle\psi_{s_k}|$, where

$$|\psi_{s_k}\rangle := \sqrt{1 - s_k}|0\rangle + \sqrt{s_k}|1\rangle. \qquad (6)$$

The density matrix ρ of the system is transformed by the following CPTP map:

$$\rho \rightarrow \rho_{s_k} \otimes \mathrm{Tr}_1[\rho], \qquad (7)$$

where Tr_1 indicates the partial trace with respect to the first qubit. The above action of the kth input on the state $\mathbf{x}(t)$ is again rewritten by a matrix S_k by using Eq. (5). After the injection, the system evolves under the Hamiltonian H for a time interval τ. Thus, the time evolution of the state for a unit time step is given by

$$\mathbf{x}(k\tau) = U_\tau S_k \mathbf{x}[(k-1)\tau]. \qquad (8)$$

After injecting the kth input, the system evolves under the Hamiltonian for τ time. The time interval τ should be chosen within a physically allowed time scale for the input injections, which is determined by both the time scale for the initialization of the qubit and the operation for the input. Then, within such a time scale, τ can be chosen to optimize the performance of the QR. As we see in Sec. IV, the trade-off between memory and nonlinear degrees of performance can be controlled by τ.

The signal, which is exploited for the learning process, is defined as an average value of a local observable on each qubit. We here employ, as observables, the Pauli operator Z_i acting on each ith qubit. Recall that, for an appropriately ordered basis $\{B_i\}$ in the operator space, the observed signals are related with the first N elements of the state $\mathbf{x}(t)$ via $x_i(t) = \mathrm{Tr}[Z_i \rho(t)]$ $(i = 1, ..., N)$. As we mentioned before, we do not consider the backaction of the measurements to obtain the average values $\{x_i(t)\}$ by considering an ensemble quantum system. We call the directly observed signals $\{x_i(t)\}_{i=1}^{N}$ as the *true nodes*. Then, the remaining $(4^N - N)$ nodes of $\mathbf{x}(t)$ are *hidden nodes*, as they are not employed as the signals for learning. For the learning, we employ $x_i'(t)$ defined by

$$x_i'(t) := \mathrm{Tr}[(I + Z_i)/2\rho(t)] = [x_i(t) + 1]/2 \qquad (9)$$

by adding a constant bias and rescaling with $1/2$ just for convenience for the presentation.

The unique feature of QRC in the reservoir computing context is that the exponentially many hidden nodes which originated from the exponentially large dimensions of the Hilbert space are monitored from a polynomial number of the signals defined as the true nodes as shown in Fig. 1(b). For this purpose, in the next section, we introduce *time multiplexing* by dividing a unit time interval τ into V subintervals to construct V virtual nodes. This effectively increases the total number of computational nodes

employed in the learning process from N true nodes to NV computational nodes.

In contrast to a single quantum system, the ensemble quantum system allows us to get real-time signals directly from exponentially large degrees of freedom. It should be noted that classical nonlinear delay systems mathematically contain infinite degrees of freedom. However, in an actual physical system, the time resolution is physically limited, and hence the effectively available degrees of freedom are also limited. This allows us to simulate them efficiently on a classical computer, as long as their time evolution is given appropriately. However, in the case of the quantum systems, we need exponentially many degrees of freedom in general to execute their brute-force simulation (prediction of the time evolution) on a classical computer, while the descriptions of their time evolution are explicitly given. This is the case even if the time resolution and precision of the readouts are finite as long as they are sufficiently small. Another physical implementation, photonic reservoir computing with a laser [24–29,51], would have a similar motivation, since photonics and lasers rely heavily on quantum physics. In contrast to these photonic approaches, we here should emphasize that our implementation crucially exploits the quantum computational supremacy region, where the classical simulation of the system is computationally hard for a classical computer. Note that at the injected one a clean qubit at each time step and the single-qubit averaged outputs after a unitary time evolution are hard enough for a classical computer to simulate efficiently, in general [14,15].

D. Emerging nonlinearity from a linear system

We here provide a physical insight why quantum-disordered dynamics can be employed for a nonlinear learning task. One might think that the quantum system is totally linear and, hence, that we cannot employ it for learning tasks, which essentially require nonlinearity. However, this is not the case. The definitions of the nonlinearity defined for the learning task and the linearity of the dynamics on the quantum system are quite different. Let us, for example, consider a quantum circuit shown in Fig. 2. For two input states $|\psi_{s_1}\rangle = \sqrt{1 - s_1}|0\rangle + \sqrt{s_1}|1\rangle$ and $|\psi_{s_2}\rangle = \sqrt{1 - s_2}|0\rangle + \sqrt{s_2}|1\rangle$, we obtain $\langle Z_{\mathrm{out}}\rangle = (1 - 2s_1)(1 - 2s_2)$, which has second-order nonlinearity with respect to s_1 and s_2. Or equivalently, in the Heisenberg picture, the observable Z_{out} corresponds to the nonlinear observable $Z_1 Z_2$. Whereas dynamics is described as a linear map, information with respect to any kind of correlation exists in exponentially many degrees of freedom. In the QRC, such higher-order correlations or nonlinear terms are mixed by the linear but quantum-chaotic (nonintegrable) dynamics U_τ. There exists a state corresponding to an observable $B_l = Z_i Z_j$, i.e., $x_l(t) = \mathrm{Tr}[Z_i Z_j \rho(t)]$ storing the correlation between $x_i(t) = \mathrm{Tr}[Z_i \rho(t)]$ and $x_j(t) = \mathrm{Tr}[Z_j \rho(t)]$, which can be monitored
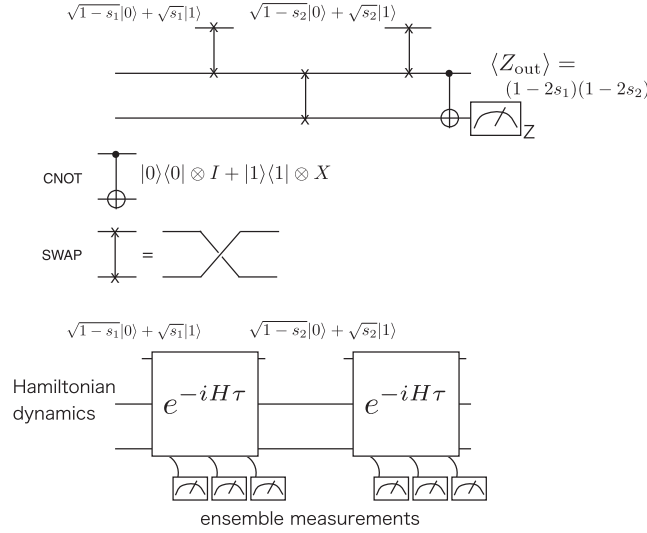
FIG. 2. Physical insight of QRC. (Top) A quantum circuit whose output has second-order nonlinearity with respect to the input variables s_1 and s_2. (Bottom) The quantum circuit is replaced by a unitary time evolution under a Hamiltonian H. The observables are monitored by the ensemble average measurements.
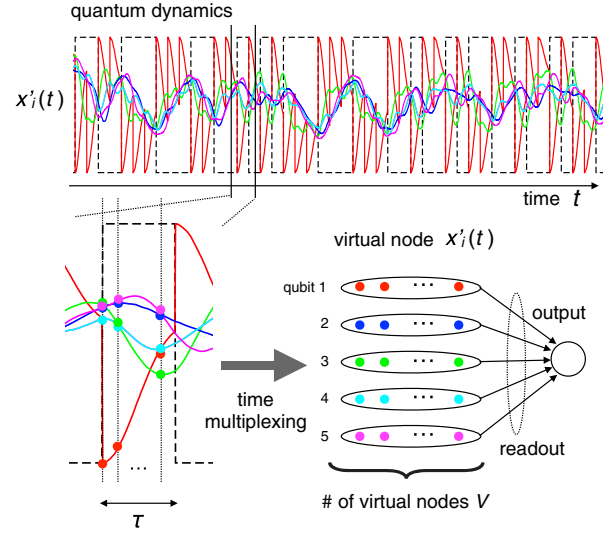

FIG. 3. Quantum-reservoir dynamics and virtual nodes. The time interval τ is divided into V subdivided time steps. At each subdivided time step, the signals are sampled. Using the NV signals as the computational nodes for each time step k in the learning phase, the linear readout weights $\{w_i^{\text{LR}}\}$ are trained for a task.

from another true node via U_τ. This mechanism allows us to find a nonlinear dynamics with respect to the input sequence $\{s_k\}$ from the dynamics of the true nodes $\{x_i(t)\}_{i=1}^{N}$. The emergent nonlinearity is not as special, because classical (nonlinear) dynamics appears as (coarse-grained) dynamics of averaged values of the observables in the quantum system. However, our main contribution goes beyond it to use such nonlinear dynamics on the disordered (chaotic) quantum systems for the simulation of general nonlinear dynamics by training an optimal observable. Such a simulation of general nonlinear dynamics with analog quantum dynamics provides an alternative paradigm to digital universal quantum computing.

E. Training readout weights

Here we explain how to train the QR from the observed signals. We harness complex quantum dynamics in a physically natural system by utilizing the reservoir computing approach. Here the signals are sampled from the QR not only at the time $k\tau$ but also at each of the subdivided V time steps during the unitary evolution U_τ as shown in Fig. 3. That is, at each time $t + v(\tau/V)$ with an integer $1 \leq v \leq V$, the signals $x_i'[t + v(\tau/V)] = \text{Tr}\{Z_i\rho[t + v(\tau/V)]\}$ are sampled. Thus, at each time step k, we have NV computational nodes in total. These time-multiplexed signals are denoted by x_{ki}' with $i = n + vN$ with integers $1 \leq n \leq N$ and $0 \leq v \leq V$, x_{ki}', which means the signal of the nth qubit at time $t = k\tau + v(\tau/V)$, i.e., $x_{ki}' := x_i'[k\tau + v(\tau/V)]$. A similar technique of time multiplexing is also used in, e.g., Ref. [24]. As explained below, these NV computational nodes are employed in the learning

phase. This allows us to make full use of the richness of quantum dynamics, because unitary real-time evolution is essential for nonlinearity.

Suppose learning is performed by using L time steps. Let $\{x_{ki}'\}$ ($1 \leq i \leq NV$ and $1 \leq k \leq L$) be the states of the computational nodes in the learning phase. We also introduce $x_{k0}' = 1.0$ as a constant bias term. Let $\{\bar{y}_k\}_{k=1}^{L}$ be the target sequence for the learning. In the reservoir computing approach, learning of a nonlinear function $y_m = f(\{s_k\}_{k=1}^{m})$, which emulates the target sequence $\{\bar{y}_k\}$, is executed by training the linear readout weights of the reservoir states such that the mean-square error

$$\frac{1}{L}\sum_{k=1}^{L}(y_k - \bar{y}_k)^2 \tag{10}$$

is minimized. That is, what we have to do is to find linear readout weights $\{w_i\}_{i=0}^{NV}$ to obtain the output sequence

$$y_k = \sum_{i=0}^{NV} x_{ki}' w_i \tag{11}$$

with the minimum mean-square error. This problem corresponds to solving the following equations:

$$\bar{\mathbf{y}} = \mathbf{X}\mathbf{w}, \tag{12}$$

where $\{x_{ki}'\}$, $\{\bar{y}_k\}_{k=1}^{L}$, and $\{w_i\}_{i=0}^{NV}$ are denoted by an $L \times (NV + 1)$ matrix $\mathbf{X}$ and column vectors $\bar{\mathbf{y}}$ and $\mathbf{w}$, respectively. Here we assume that the length of the training

sequence L is much larger than the total number of the nodes $NV + 1$ including the bias term. Thus, the above equations are overdetermined, and hence the weights that minimize the mean-square error are determined by the Moore-Penrose pseudoinverse $\mathbf{X}^+ := (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$ [$(NV + 1) \times L$ matrix] of $\mathbf{X}$ as follows:

$$\mathbf{w}^{\text{LR}} := \mathbf{X}^+ \bar{\mathbf{y}}. \tag{13}$$

Using $\mathbf{w}^{\text{LR}}$, we obtain the output from the QR

$$y_k = \sum_{i=0}^{NV} w_i^{\text{LR}} x'_{ki}. \tag{14}$$

Or, equivalently, an optimal observable

$$O_{\text{trained}} \equiv \sum_{i=1}^{N} w_i^{\text{LR}} (I + Z_i)/2 + w_{N+1}^{\text{LR}} I \tag{15}$$

is trained, and the output is obtained as $\langle O_{\text{trained}} \rangle$.

Specifically, as is the case in the conventional reservoir computing approach, none of the parameters of the system (Hamiltonian) requires fine-tuning except for the linear readout weights. Thus, we can employ any quantum system (Hamiltonian) as long as it exhibits dynamics with appropriate properties for our purpose, such as fading memory and nonlinearity. That is, as long as the QR is sufficiently rich, we can find an optimal observable O_{trained} capable of exploiting the preferred behavior via the training (learning) process. In the following numerical experiments, we employ, as an example, the simplest quantum system, a fully connected transverse-field Ising model, which exhibits a Wigner-Dyson statistics of the energy-level spacing [45,46,52]:

$$H = \sum_{ij} J_{ij} X_i X_j + h Z_i, \tag{16}$$

where the coupling strengths are randomly chosen such that J_{ij} is distributed randomly from $-J/2$ to $J/2$. We introduce a scale factor Δ so as to make $\tau \Delta$ and J/Δ dimensionless. When we mention the size of QRs, we refer to the number of qubits, N, simply because it reflects the physical size of the quantum systems as is also the case in the quantum-information literature. However, we should recall that the total number of computational nodes in the learning phase is NV because of time multiplexing. In any numerical experiments, we show the number of virtual nodes V explicitly. Note that we do not employ any approximation, but the quantum dynamics of the above Hamiltonian is exactly calculated to evaluate the potential performance of the QRs. The imperfections including decoherence and noise on the observed signals, which might occur in actual experiments, are further taken into account in Sec. IV B.

III. DEMONSTRATIONS OF QRC FOR TEMPORAL LEARNING TASKS

We start by providing several demonstrations to obtain a sense of QRC using a number of benchmark tasks in the context of machine learning. Our demonstrations consist of a timing control (the timer task), learning of input-driven dynamical systems or nonautonomous dynamical systems [the *nonlinear autoregressive moving average* (NARMA) task], and learning of autonomous dynamical systems including chaotic attractors (the Mackey-Glass prediction task), which cover typical cases of the target applications in reservoir computing schemes [18–20,53]. Detailed analyses of the performance for each task are given in the Appendix.

A. Timer task

Our first experiment is to construct a timer. One important property of QRC is having memory to be exploited. Whether the system contains memory or not can be straightforwardly evaluated by performing this timer task (see, e.g., Ref. [53]). The input is flipped from 0 to 1 at certain time step (k') as a cue, and the system should output 1 if τ_{timer} time steps have passed from the cue; otherwise, it should output 0 [see Fig. 4(a), left diagram]. To perform this task, the system has to be able to "recognize" the duration of time that has passed since the cue was launched. This clearly requires memory. Here we used six-qubit QRs with $\tau \Delta = 1$ to perform this task by incrementally varying V.

Figure 4(a) shows the task performance with trained readouts. We can clearly observe that by increasing V the performance improves, which means that the amount of memory, which can be exploited, also increases. In particular, when $V = 5$ and 10, the system outputs overlap with the target outputs within the certain delay, which clearly demonstrates that our QR system is capable of embedding a timer. By increasing the delay time steps τ_{timer}, we can gradually see that the performance declines, which expresses the limitation of the amount of memory that can be exploited within the QR dynamics. It is interesting to note that, while the systems are highly disordered, we can find an observable O_{trained} or a mode, at which the wave function of the system is focused after a desired delay time τ_{timer}. This is very useful as a control scheme for engineering quantum many-body dynamics. For further information, see detailed settings, experimental and learning procedures, and analyses for the timer task in Appendix A 1.

B. NARMA task

The second task is the emulation of nonlinear dynamical systems, called NARMA systems, which is a standard benchmark task in the context of recurrent neural-network learning. This task presents a challenging problem for any computational system because of its nonlinearity and dependence on long time lags [54]. The first NARMA
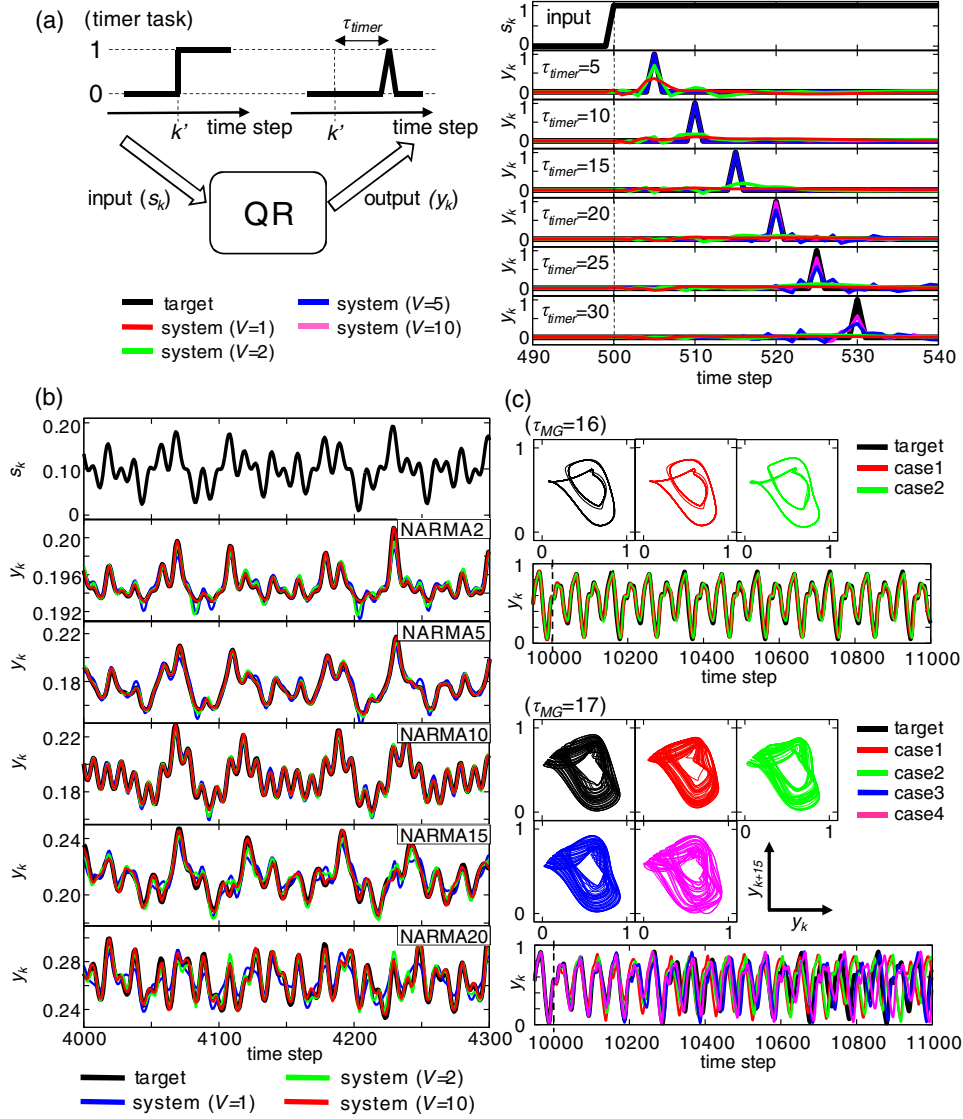
FIG. 4. Typical performances of QR for temporal machine-learning tasks. (a) The timer task. A six-qubit QR system is prepared, and, starting from different initial conditions, ten trials of numerical experiments were run for each τ_{timer} setting. k' is set to 500 throughout the numerical experiments. The plots overlay the averaged system performance over ten trials for $V = 1, 2, 5,$ and 10 with the target outputs. (b) The NARMA emulation task. This task requires five different NARMA systems driven by a common input stream to be emulated. The upper plot shows the input stream, and the corresponding task performances of a six-qubit QR system for five NARMA tasks are plotted, overlaying the case for each V with the target outputs. (c) The Mackey-Glass prediction task. The performances for $\tau_{\text{MG}} = 16$ (nonchaotic) and 17 (chaotic) are shown. The trained system outputs are switched to the autonomous phase at time step 10000. Two-dimensional plots (y_k, y_{k+15}) are depicted for the autonomous phase in each case. For each setting of τ_{MG}, case 1 represents the case for six qubits and, otherwise, represent the cases for seven qubits. In this task, the number of virtual nodes is fixed to be $V = 10$. For all tasks, the detailed settings and analyses are provided in the Appendix.

system is the following second-order nonlinear dynamical system:

$$y_{k+1} = 0.4y_k + 0.4y_k y_{k-1} + 0.6s_k^3 + 0.1. \quad (17)$$

This system was introduced in Ref. [55] and used, for example, in Refs. [33,35]. For descriptive purposes, we call this system NARMA2. The second NARMA system is the following nonlinear dynamical system that has an order of n:

$$y_{k+1} = \alpha y_k + \beta y_k \left(\sum_{j=0}^{n-1} y_{k-j} \right) + \gamma s_{k-n+1} s_k + \delta, \quad (18)$$

where $(\alpha, \beta, \gamma, \delta)$ are set to $(0.3, 0.05, 1.5, 0.1)$, respectively. Here, n is varied using the values of 5, 10, 15, and 20, and the corresponding systems are called NARMA5, NARMA10, NARMA15, and NARMA20, respectively. In particular, NARMA10 with this parameter setting is introduced in Ref. [55] and broadly used (see, e.g., Refs. [20,33,35]). As

a demonstration, the input s_k is expressed as a product of three sinusoidal functions with different frequencies. This input setting is just for illustrative purposes. The results of the different types of input stream, such as randomly drawn real values, are also provided in Appendix A 2, along with the detailed analyses. (Note that, when the input is projected to the first qubit, the value is linearly scaled to [0, 1]; see Appendix A 2 for details). Here, according to an input stream expressed as a product of three sinusoidal functions with different frequencies, the system should simultaneously emulate five NARMA systems (NARMA2, NARMA5, NARMA10, NARMA15, and NARMA20), which we call multitasking.

Figure 4(b) plots the input sequence and the corresponding task performance of our six-qubit QR system with $\tau \Delta = 1$ with a trained readout by varying V. We can clearly observe that, by increasing V, the performance improves, so that when $V = 10$ the system outputs almost overlap with the target outputs. Further information and extended

analyses on the tasks with random input streams can be found in Appendix A 2.

C. Mackey-Glass prediction task

The third experiment is a Mackey-Glass (MG) time series prediction task, including a chaotic time series. This is also a popular benchmark task in machine learning (e.g., Ref. [18]). Here, unlike the previous two cases, the system output is fed back as the input for the next time step, which means that, when the system with a trained readout generates outputs, it receives its own output signals through the feedback connections instead of through external inputs. To train the readout weights, the system is forced by the correct teacher output during the presentation of the training data, without closing the loop. A slight amount of white noise is added to the reservoir states in the training phase to make the trained system robust, and the weights are trained through the usual procedure (see Appendix A 3 for further information). The MG system has a delay term τ_{MG}, and when $\tau_{\mathrm{MG}} > 16.8$ it exhibits a chaotic attractor. We first test a nonchaotic case ($\tau_{\mathrm{MG}} = 16$) for comparisons and then test the chaotic case, where $\tau_{\mathrm{MG}} = 17$, which is the standard value employed in most of the MG system prediction literature.

Figure 4(c) depicts the typical task performances of six- and seven-qubit QR systems. When $\tau_{\mathrm{MG}} = 16$, the system outputs overlap the target outputs, which implies successful emulations. When $\tau_{\mathrm{MG}} = 17$, our systems tend to remain relatively stable in the desired trajectory for about 200 steps but, after switched from the teacher-forced condition, start to deviate perceptibly large. Although it is difficult to compare the performance of the closed-loop task directly with that of different types of reservoirs due to the differences in experimental conditions and reservoir settings, we would like to mention that, in the MG prediction task (for $\tau_{\mathrm{MG}} = 17$ case) using an ESN equipped with 400 leaky integrator neurons, it is observed that the predictions started to deviate perceptibly from the target not earlier than after about 1200 steps [56]. Recent results from an analog hardware implementation of RC using 600 computational nodes reported that the mean-squared error (MSE) of the predictions calculated by moving average windows of 200 steps reached approximately 1.1×10^{-1} after 2500 steps [51], whereas our system reaches the same MSE value relatively faster, after 820 steps [e.g., case 4 in Fig. 4(c)]. Since our system exploits only 70 ($N = 7$ qubits and $V = 10$ virtual nodes) computational nodes in total in this demonstration, it would be worth investigating how our system performance behaves according to the increase in computational nodes in future work. Furthermore, checking a two-dimensional plot by plotting points (y_k, y_{k+15}), it appears that the learned model has captured the essential structure of the original attractor (e.g., when $\tau_{\mathrm{MG}} = 17$, the model actually demonstrates chaos). In both tasks, the seven-qubit QR systems generally perform better than the six-qubit QR systems. Further details of the statistical analyses of the performance and the analysis of the largest Lyapunov exponent for the cases in Fig. 4(c) are provided in Appendix A 3.

IV. PERFORMANCE ANALYSES

We perform detailed analyses on the computational capabilities of the five-qubit QRs focusing on the two popular benchmark tasks of Boolean function emulations over a binary input sequence (see, e.g., Refs. [34,57]), which we name the *short-term memory* (STM) task and *parity check* (PC) task. The former task is intended to emulate a function that outputs a version of the input stream delayed by τ_B time steps, whereas the latter is intended to emulate an τ_B-bit parity checker. Both tasks require memory to be emulated, and the PC task requires non-linearity in addition, because the parity-checker function performs nonlinear mapping. Hence, the STM task can evaluate the memory capacity of systems, and the PC task can additionally evaluate the amount of nonlinearity within systems.

The function for the STM task can be expressed as follows:

$$y_k = s_{k-\tau_B},$$

where s_k is a binary sequence and τ_B represents the delay. The function for the PC task is expressed as follows:

$$y_k = Q\left(\sum_{m=0}^{\tau_B} s_{k-m}\right),$$

$$Q(x) = \begin{cases} 0 & [x \equiv 0 (\mathrm{mod}\ 2)] \\ 1 & (\mathrm{otherwise}). \end{cases}$$

We investigate both tasks thoroughly by applying a random input sequence for the tasks such that there is no external source to provide temporal coherence to the system. In these tasks, one trial consists of 5000 time steps. The first 1000 time steps are discarded, the next 3000 time steps are used for training, and the last 1000 time steps are used for system evaluation. We evaluated the system performance with the target output for each given τ_B by using the measure known as τ_B-delay capacity $C(\tau_B)$ expressed as

$$C(\tau_B) = \frac{\mathrm{cov}^2(y_k, \bar{y}_k)}{\sigma^2(y_k)\sigma^2(\bar{y}_k)}.$$

In the main text, τ_B-delay capacities for the STM task and the PC task are termed τ_B-delay STM capacity $C_{\mathrm{STM}}(\tau_B)$ and τ_B-delay PC capacity $C_{\mathrm{PC}}(\tau_B)$, respectively. Note that, in the analyses, to reduce a bias due to the effect of the finite data length, we subtract $C(\tau_B^{\mathrm{max}})$ from $C(\tau_B)$, where τ_B^{max} is a substantially long delay. The capacity C is defined as
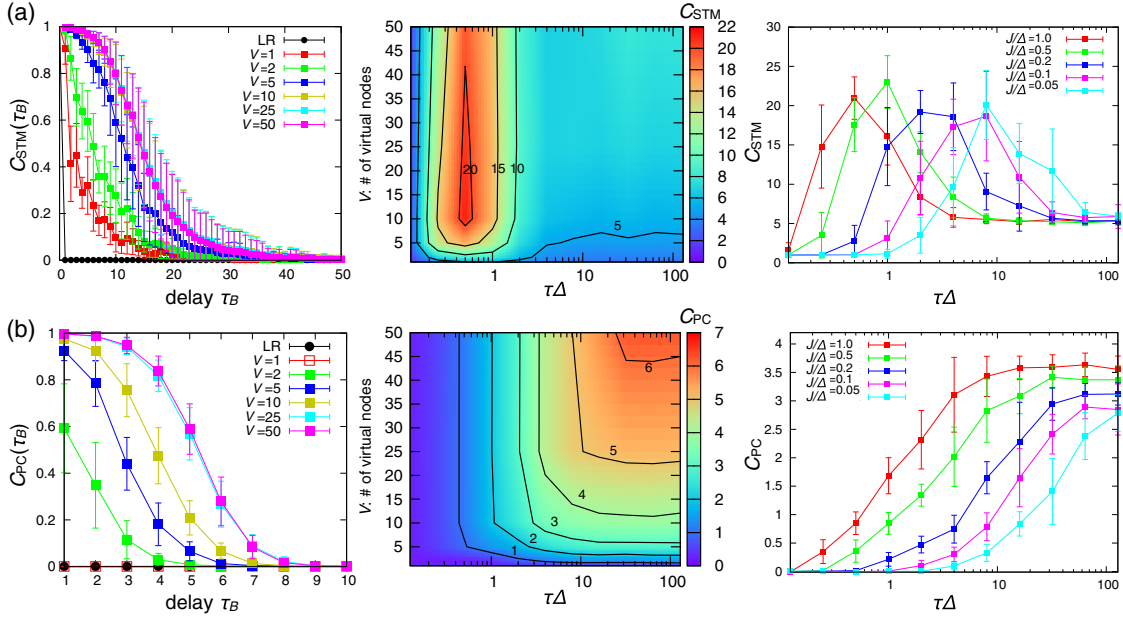
FIG. 5.   Performance analyses of the five-qubit QRs. (a) (Left) STM curve $C_{\text{STM}}(\tau_B)$ plotted as a function of the delay τ_B for $J/\Delta = 2h/\Delta = 1$, $\tau\Delta = 1$, and $V = 1$–50. (Middle) STM capacity C_{STM} plotted as a function of the number of virtual nodes V with the same QR settings for $\tau\Delta = 0.5$–128. (Right) STM capacity C_{STM} plotted as a function of $\tau\Delta$ with couplings $J/\Delta = 0.05$–1.0 and $h/\Delta = 0.5$. (b) PC curve $C_{\text{PC}}(\tau_B)$ and capacity C_{PC} plotted with the same settings as (a). The error bars in the left and right panels indicate the standard deviations of the capacities evaluated on 20 samples of the QRs with respect to the random couplings.

$$C = \sum_{\tau_B=0}^{\tau_B^{\max}} C(\tau_B),$$

where $\tau_B^{\max}$ is 500 throughout our experiments. The capacities for the STM task and the PC task are referred to as the STM capacity C_{STM} and the PC capacity C_{PC}, respectively. For each task, 20 samples of the QRs are randomly generated, and the average values of the τ_B-delay capacities and the capacities are obtained.

In Fig. 5(a) (left), $C_{\text{STM}}(\tau_B)$ is plotted as a function of τ_B for $V = 1, \ldots, 50$, where $\tau\Delta = 1$ and $J/\Delta = 1.0$ are set. The abrupt decay exhibited by the curve is improved when the number of virtual nodes is increased. In Fig. 5(a) (middle), the STM capacity is plotted as a function of the number of virtual nodes V and the time interval $\tau\Delta$. It shows that the STM capacity becomes saturated around $V = 10$. The five-qubit QRs with $\tau\Delta = 0.5$ and 1.0 exhibit a substantially high STM capacity of approximately 20, which is much higher than that of the ESNs of 500 nodes (see Sec. IV A for details). A plot of the STM capacity as a function of τ for a fixed number of virtual nodes $V = 10$ does not exhibit monotonic behavior, as shown in Fig. 5 (right). This behavior is understood as follows. In the limit of $\tau \to 0$, the dynamics approach an identity map and hence become less attractive, and this is more desirable to maintain the separation among different inputs. At the same time, a shorter τ implies less information is embedded in the present input setting. In the limit of larger τ, on the other hand, the input sequence is injected effectively;

however, the dynamics become attractive, and the separation fades rapidly. Originating from these two competing effects, there is an optimal time interval τ for which the STM capacity is maximized.

In Fig. 5(b) (left), $C_{\text{PC}}(\tau_B)$ is plotted as a function of τ_B for $V = 1, \ldots, 50$. Specifically, $C_{\text{PC}}(\tau_B)$ is exactly zero when $V = 1$. This clearly shows that the virtual nodes, which spatialize the real-time dynamics during the interval τ, are important to extract nonlinearity. In Fig. 5(b) (middle), the PC capacity is plotted as a function of the number of virtual nodes V and the time interval $\tau\Delta$. As expected, the longer the time interval τ is, the higher the PC capacity exhibited by the QR, as shown in Fig. 5 (middle and right). This is because the true nodes are able to increase communication with the virtual nodes. The number of virtual nodes required for the saturation of the PC capacity is also increased in the case of a longer τ.

A. Characterizations of QRs

Let us clarify the unique properties of the QRs in terms of the STM and PC capacities. We plot $(C_{\text{STM}}, C_{\text{PC}})$ for the five-qubit QRs with various coupling settings in Fig. 6(a), which include a restricted type of QR with one-dimensional nearest-neighbor (1DNN) couplings, i.e., $J_{ij} \neq 0$ only for $j = i + 1$ in Eq. (16). In this case, the transversal-field Ising model becomes integrable, that is, exactly solvable by mapping it into a free-fermionic model via the Jordan-Wigner transformation. Because the effective dimension of the state space is reduced from 2^{2N} to $2N$, the amplitudes of
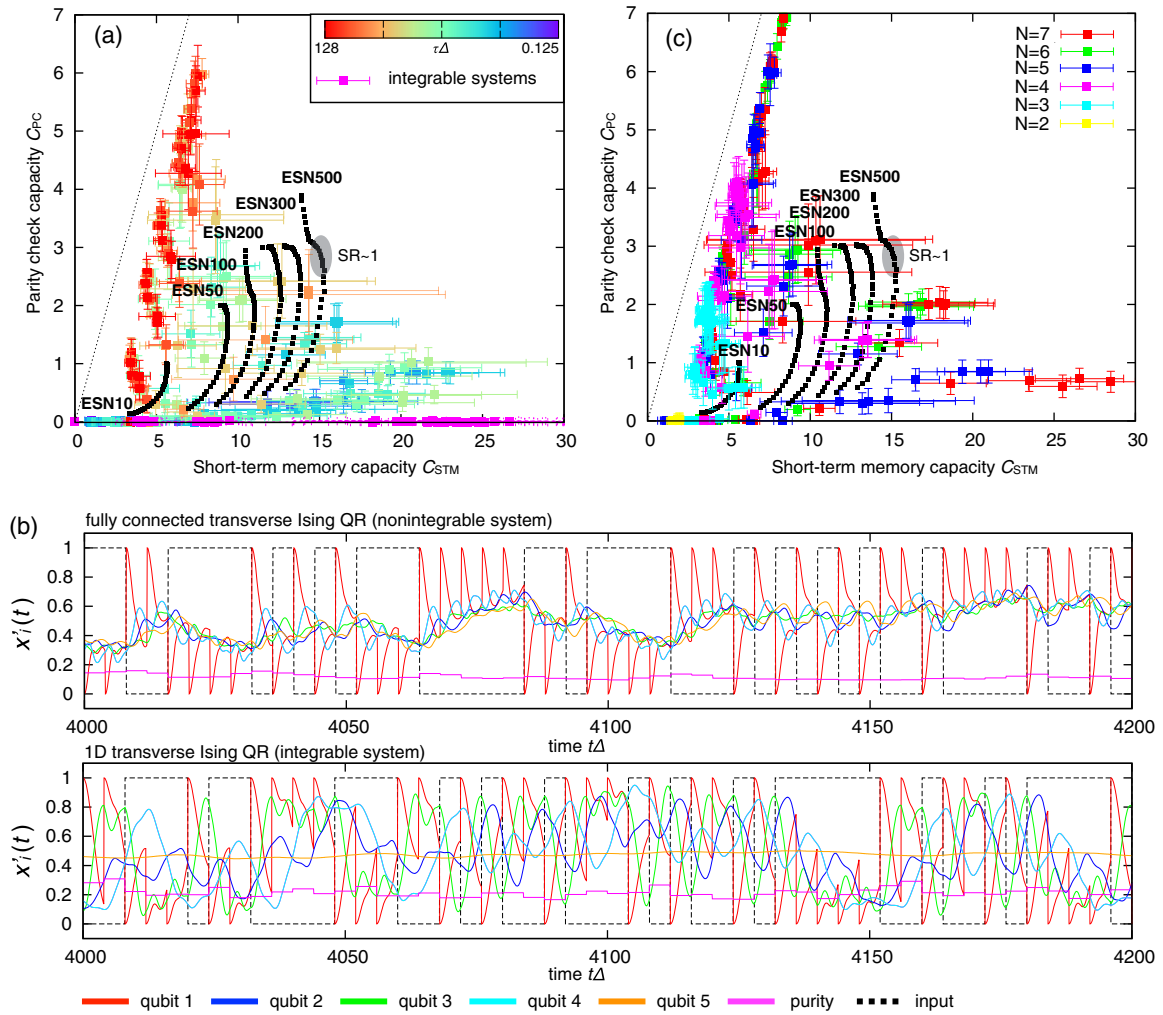
FIG. 6.   STM and PC capacities under various settings. (a) Performance dependence on the time interval τ. Capacities for the five-qubit QRs plotted with various parameters $\tau\Delta = 0.125$–128, $J/\Delta = 0.05$–1.0, $h/\Delta = 0.5$, and $V = 1$–50. Specifically, different cases of τ are colored differently. Integrable cases with 1DNN couplings are shown as "integrable systems." For each setting, the capacities are evaluated as an average on 20 samples, and the standard deviations are shown by the error bars. (b) Typical dynamics with fully connected (upper) and 1DNN couplings (lower) are shown with the signals from each qubit. The input sequence and purity, a measure of quantum coherence, are shown by dotted and solid lines, respectively. (c) Performance dependence on the number of qubits. QRs with $J/\Delta = 2h/\Delta = 1$ and $0.125 \leq \tau\Delta \leq 128$ for $N = 2$–7. For each setting, the capacities are evaluated as an average on 20 samples, and the standard deviations are shown by the error bars. Specifically, different cases of the number of qubits are colored differently. The ESNs with 10–500 nodes are shown as references. From top to bottom, the spectral radius (SR) is changed from 0.5 to 2.0. The spectral radius around one is also shown by the areas colored by gray.

the oscillations are larger for the 1DNN case as shown in Fig. 6(b). From the real-time dynamics, one might expect a rich computational capability even for the integrable dynamics. Although this is true for the STM capacity, it does not hold for the PC capacity. As shown in Fig. 7, the STM capacity of the 1DNN QRs is above 20, rather higher than that of fully connected QRs. However, the PC capacity is substantially poor, which cannot improve even if the time intervals τ or the number of virtual nodes are changed. This is a natural consequence of the inability of the 1DNN model to fully employ exponentially large state spaces. In this way, the computational capacity of QRs, especially their nonlinear capacity, has a close connection with the

nonintegrability of the underlying QR dynamics. This implies that the computational capacity as a QR provides a good metric of the integrability of quantum dynamics. A nonintegrable quantum system is identified as quantum chaos, which is specified by the Wigner-Dyson distribution of the energy eigenstate spacing. The operational metric of the integrability of quantum dynamics would be useful to build a modern operational understanding of quantum chaos by relating it to the emulatability of classical chaos.

Next we investigate the scaling of the STM and PC capacities against the number of the qubits N in the QRs. In Fig. 8, the STM and PC capacities are plotted against the number of qubits for the virtual nodes $V = 1, 2, 5, 10, 25$,
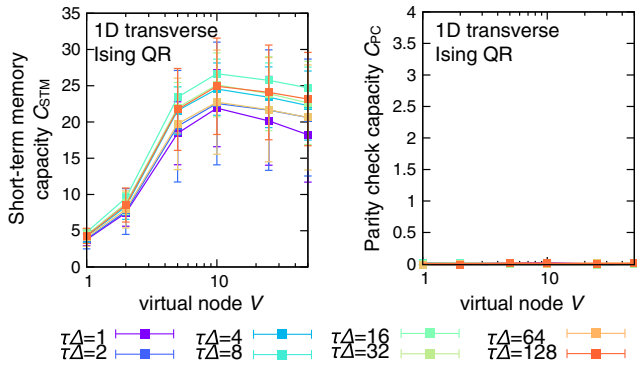
FIG. 7. STM (left) and PC (right) capacities for the 1D transversal Ising model. The error bars show the standard deviations evaluated on 20 samples of the QRs with respect to the random couplings.

and 50. First, both capacities monotonically increase in the number of the qubits N and the virtual nodes V. Thus, by increasing the time resolution and size of the QR, we can enhance its computational capability. The STM capacity is improved by increasing the number of virtual nodes V especially for optimally chosen time intervals τ. The improvement saturates around $V = 10$. The scaling behavior of the STM capacity seems to be different for $N = 2$–4 and $N = 4$–7 when the virtual nodes are introduced. For

optimally chosen time intervals, the STM capacity seems to increase linearly in terms of the number of qubits.

The PC capacity also increases in terms of the number of virtual nodes V, but its saturation highly depends on the choice of the time interval τ. For a short interval $\tau\Delta = 1$, the PC capacity saturates around $V = 10$. However, for $\tau\Delta = 128$, it seems not to saturate even with $V = 50$. In any case, the PC capacity seems to increase linearly in terms of the number of the qubits N. Interestingly, at the large τ and large V limits, the PC capacity saturates the line defined by $C_{PC} = 2(N - 2)$. The origin of this behavior is completely unknown at this moment.

In Fig. 6(c), the STM and PC capacities are plotted for the QRs from $N = 2$ to $N = 7$. The seven-qubit QRs, for example, with $\tau\Delta = 2$, $J/\Delta = 2h/\Delta = 1$, and $V = 10$–50, are as powerful as the ESNs of 500 nodes with the spectral radius around 1.0. Note that, even if the virtual nodes are included, the total number of computational nodes $NV = 350$ is less than 500.

B. Robustness against imperfections

We here investigate the effect of decoherence (noise) to validate the feasibility of QRC. We consider two types of noise: The first is decoherence, which is introduced by an undesired coupling of QRs with the environment, thereby
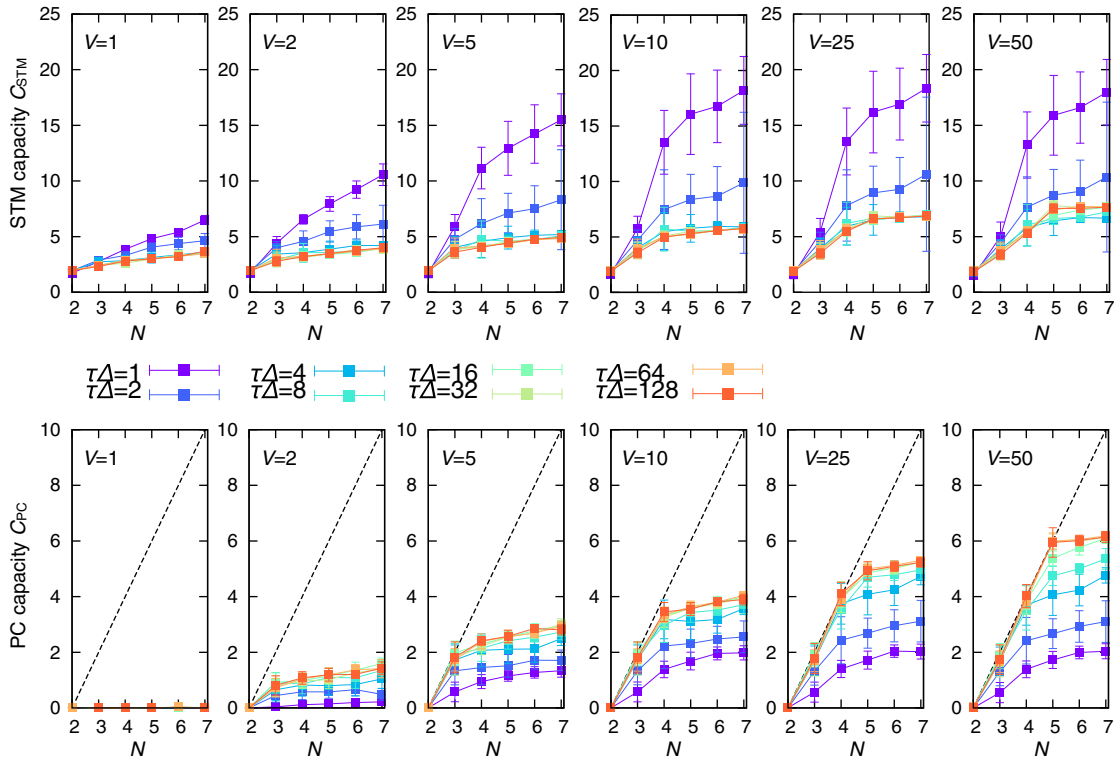

FIG. 8. Scaling of the STM and PC capacities against the number of the qubits. (Top) The STM capacity C_{STM} is plotted against the number of qubits N for each number of virtual nodes ($V = 1, 2, 5, 10, 25, 50$ from left to right). (Bottom) The PC capacity C_{PC} is plotted against the number of qubits N for each number of virtual nodes ($V = 1, 2, 5, 10, 25, 50$ from left to right). $C_{PC} = 2(N - 2)$ is shown by dotted lines. The error bars show the standard deviations evaluated on 20 samples of the QRs with respect to the random couplings.

resulting in a loss of quantum coherence, and the other is a statistical error on the observed signals from QRs. The former is more serious, because quantum coherence is, in general, fragile against decoherence, which is the most difficult barrier for realizations of quantum information processing.

We employ the dephasing noise as decoherence, which is a simple yet experimentally dominant source of noise. In the numerical simulation, the time evolution is divided into a small discrete interval δt, and qubits are exposed to the single-qubit phase-flip channel with probability $(1 - e^{-2\gamma\delta t})/2$ for each time step:

$$\mathcal{E}(\rho) = \frac{1 + e^{-2\gamma\delta t}}{2}\rho + \frac{1 - e^{-2\gamma\delta t}}{2}Z\rho Z. \qquad (19)$$

This corresponds to a Markovian dephasing with a dephasing rate γ and destroys quantum coherence, i.e., off-diagonal elements in the density matrix. Apart from the dephasing in the z direction, we also investigate the dephasing in the x direction, where the Pauli Z operator is replaced by X. In Fig. 9, the STM and PC capacities $(C_{\mathrm{STM}}, C_{\mathrm{PC}})$ are plotted for $\tau\Delta = 0.5$, 1.0, 2.0, and 4.0 (from left to right) with $V = 1, 2, 5, 10, 25,$ and 50 and $\gamma = 10^{-1}, 10^{-2},$ and 10^{-3}. The results show that dephasing of the rates 10^{-1}–10^{-3}, which is within an experimentally feasible range, does not degrade computational capabilities. A subsequent increase in the dephasing rate causes the

STM capacity to become smaller, especially for the case with a shorter time interval $\tau\Delta = 0.5$. On the other hand, the PC capacity is improved by increasing the dephasing rate. This behavior can be understood as follows. The origin of quantum decoherence is the coupling with the untouchable environmental degree of freedom, which is referred to as a reservoir in the context of open quantum systems. Thus, decoherence implies an introduction of another dynamics with the degree of freedom in the reservoir computing framework. This leads to the decoherence-enhanced improvement of nonlinearity observed in Fig. 9, especially for a shorter τ with less rich dynamics. Of course, for a large decoherence limit, the system becomes classical, preventing us from fully exploiting the potential computational capability of the QRs. This appears in the degradation of the STM capacity. By attaching the environmental degree of freedom, the spatialized temporal information is more likely to leak outside the true nodes. Accordingly, we cannot reconstruct a past input sequence from the signals of the true nodes. In other words, quantum coherence is important to retain information of the past input sequence within the addressable degree of freedom. In short, in the QRC approach, we do not need to distinguish between coherent dynamics and decoherence; we can exploit any dynamics on the quantum system as it is, which is monitored only from the addressable degree of freedom of the quantum system.
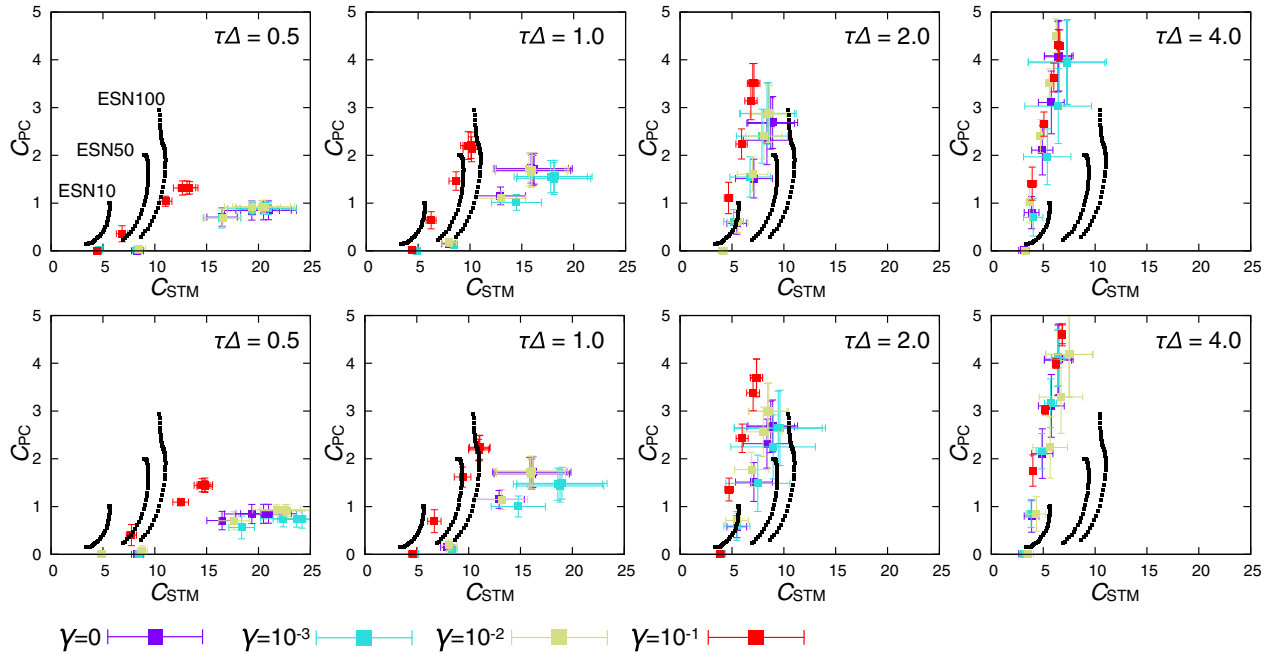

FIG. 9. STM and PC capacities under decoherence investigated for the five-qubit QRs. The parameters are set as $\tau\Delta = 0.5, 1.0, 2.0,$ 4.0 and $V = 1, 2, 5, 10, 25, 50$. (Top) Capacities $(C_{\mathrm{STM}}, C_{\mathrm{PC}})$ under dephasing in the z axis are plotted for $\gamma = 10^{-1}, 10^{-2}, 10^{-3}$. (Bottom) Capacities $(C_{\mathrm{STM}}, C_{\mathrm{PC}})$ under dephasing in the x axis are plotted for $\gamma = 10^{-1}, 10^{-2}, 10^{-3}$. The error bars show the standard deviations evaluated on 20 samples of the QRs with respect to the random couplings. The ESNs with 10, 50, and 100 nodes are shown as references. From top to bottom, the spectral radius is changed from 0.5 to 2.0.
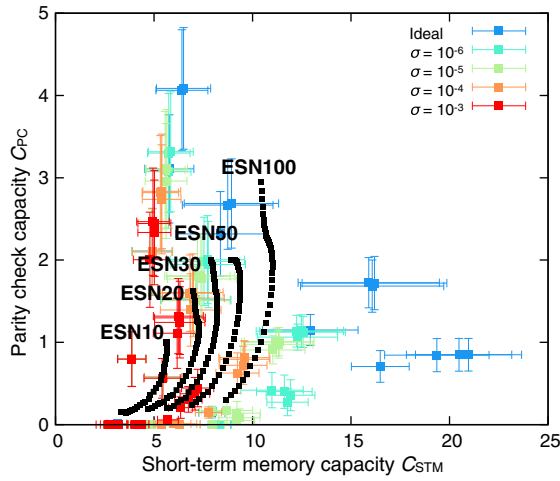
FIG. 10. Effect of the statistical error on the observed signals investigated for the five-qubit QRs. The parameters are set as $\tau\Delta = 0.5, 1.0, 2.0, 4.0$ and $V = 1, 2, 5, 10, 25, 50$. Specifically, different cases of the amount of the statistical error are colored differently. The STM and PC capacities (C_{STM}, C_{PC}) are plotted for Gaussian noise with zero mean and variance $\sigma = 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$. The error bars show the standard deviations evaluated on 20 samples of the QRs with respect to the random couplings. The performances for the ESNs are calculated without adding observational noise. The ESNs with 10, 20, 30, 50, and 100 nodes are shown as references. From top to bottom, the spectral radius is changed from 0.5 to 2.0.

Next, we consider the statistical noise on the observed signal from the QRs. We investigate the STM and PC capacities by introducing Gaussian noise with zero mean and variance σ on the output signals as shown in Fig. 10. The introduction of statistical noise leads to a gradual degradation of the computational capacities. However, the degradation is not abrupt, which means that QRC would be able to function in a practical situation. In the small τ region, the STM capacity is sensitive to the statistical observational noise. This is because, in such a region, the dynamic range of the observed signals becomes narrow. For example, when $\tau\Delta = 0.5$ and $\tau\Delta = 4$, the dynamic ranges are approximately 0.01 and approximately 0.5, respectively. While, in the ideal case, the performances of the five-qubit QRs are comparable to the ESNs of 100 nodes, their performances under the statistical observational noise of the order of 10^{-3} against the dynamic ranges is still comparable to the ESNs of 50 nodes without any noise. Moreover, as we see in the demonstration of the chaotic time series prediction, we even introduce statistical noise to the observed signals with the aim of stabilizing the learning process. This implies that in some situation we can positively exploit the natural observational noise in our framework.

These tolerances against imperfections indicate that the proposed QRC framework soundly functions in realistic experimental setups as physical reservoir computing.

V. DISCUSSION

The QRC approach enables us to exploit any kind of quantum systems, including quantum simulators and quantum annealing machines, provided their dynamics are sufficiently rich and complex to allow them to be employed for information processing. In comparison to the standard approach for universal quantum computation, QRC does not require any sophisticatedly synthesized quantum gate, but natural dynamics of quantum systems is enough. Therefore, QRC exhibits high feasibility in spite of the fact that its applications are broad for temporal learning tasks.

The conventional software approach for recurrent neural networks takes time, which depends on the size of the network, to update the network states. In contrast, in the case of QRC, the time evolution is governed by natural physical dynamics in a physically parallelized way. For example, liquid- and solid-state NMR systems with nuclear and electron spin ensembles [47,48] are favorable for implementing QRC. These systems enable us to obtain the output signals in real time via the radio-frequency coil by virtue of its huge number of ensembles. Note that we employ the simplest model and that no optimization of the QRs has been done yet. More study is necessary to optimize the QRs with respect to a Hamiltonian, network topology, the way of injecting the input sequences, and the readout observables.

Notwithstanding its experimental feasibility, controllability, and robustness against decoherence, the QRC framework would also be useful to analyze complex real-time quantum dynamics from an operational perspective. The computational capabilities provide operational measures for quantum integrable and chaotic dynamics. Recently, delocalization (and localization) of quantum information by unitary dynamics in closed quantum systems has been attracting much attention in various fields of physics, such as fast scrambling in a black hole and thermalization in statistical physics of closed quantum systems. Our framework shares the same setting, time-independent Hamiltonian dynamics monitored from local observables. Since our framework allows us to evaluate the computational capability of the signals obtained from such systems, it provides an operational way to understand these phenomena related to complex unitary dynamics. Apparently, the STM is closely related to time correlation in many-body quantum physics and the thermalization of closed quantum systems. Moreover, the chaotic behavior of quantum systems has been investigated in an attempt to understand the fast scrambling nature of black holes [58,59]. It would be intriguing to measure the computational capabilities of such black hole models. We believe that QRC for universal real-time quantum computing, which bridges quantum information science, machine learning, quantum many-body physics, and high-energy physics coherently, provides an alternative paradigm to quantum digital computing.

APPENDIX: EXPERIMENTAL SETTINGS AND EXTENDED ANALYSES

This section describes detailed settings for the task experiments mentioned in the main text and provides extended analyses. We maintain the notation for symbols used in the main text.

1. The timer task

The timer task is one of the simplest yet most important benchmark tasks to evaluate the memory capacity of a system (see, e.g., Ref. [53]). As explained in the main text, our goal for the demonstration of QRC is to emulate the function of a timer [Fig. 4(a) in the main text]. The I/O relation for a timer can be expressed as follows:

$$s_k = \begin{cases} 1 & (k \geq k'), \\ 0 & (\text{otherwise}), \end{cases}$$

$$y_k = \begin{cases} 1 & (k = k' + \tau_{\text{timer}}), \\ 0 & (\text{otherwise}), \end{cases}$$

where k' is a time step for launching the cue to the system and τ_{timer} is a delay for the timer. Our aim is to emulate this timer by exploiting the QR dynamics generated by the input projected to the first qubit in the QR system.

A single experimental trial of the task consists of 800 time steps, where the first 400 time steps are discarded as initial transients. At time step 500, the input is switched from 0 to 1 (i.e., $k' = 500$), and the system continues to run for another 300 time steps. For the training procedure, using a six-qubit QR system with $\tau\Delta = 1$, we iterate this process over five trials, starting from different initial conditions, and collect the corresponding QR time series for each time step from time step 400 to time step 800 as training data. We optimize the linear readout weights using these collected QR time series with a linear regression to emulate the target output for the given delay τ_{timer} and the setting of the number of virtual nodes V in QR systems. We evaluate the performance of the system with the optimized weights by running five additional trials (evaluation trials) and compare the system outputs to the target outputs in the time region from time step 400 to time step 800.

Here, we aim to analyze the performance of the timer task further. We prepare ten different six-qubit QR systems, whose coupling strengths are assigned differently, and for each setting of (τ_{timer}, V), we iterate the experimental trials

as explained above over these ten different systems. To effectively evaluate the system's performance against the target outputs $\bar{y}_k$, given the setting of τ_{timer}, we define a measure $C(\tau_{\text{timer}})$, which is expressed as

$$C(\tau_{\text{timer}}) = \frac{\text{cov}^2(y_k, \bar{y}_k)}{\sigma^2(y_k)\sigma^2(\bar{y}_k)},$$

where $\text{cov}(x, y)$ and $\sigma(x)$ express the covariance between x and y and the standard deviation of x, respectively. In short, this measure evaluates the association between two time series and takes a value from 0 to 1. If the value is 1, it means that the system outputs and the target outputs completely overlap, which implies that the learning is perfect. At the other extreme, if the value is 0, it implies that the learning completely fails. Evaluation trials are used to actually calculate this measure. Now, we further define a measure, capacity C, which is expressed as a simple summation of $C(\tau_{\text{timer}})$ over the entire delay:

$$C = \sum_{\tau_{\text{timer}}=0}^{\tau_{\text{timer}}^{\max}} C(\tau_{\text{timer}}),$$

where $\tau_{\text{timer}}^{\max}$ is set to 300 in our experiments.

By using these two measures, $C(\tau_{\text{timer}})$ and C, we evaluate the performance of the timer tasks of six-qubit QR systems. Figure 11 plots the results. Figure 11(a) clearly indicates that larger values of V can perform the timer task reliably for a longer delay, which shows a characteristic curve for each setting of V. This point is also confirmed by checking the plot of C according to the value of V, where C increases almost linearly with an increase in V [see Fig. 11(b)]. These results are consistent with the result demonstrated in the main text. We compare the performance with ESN, where the experimental conditions are set the same as the experiment with QR (see Appendix A 4 for the basic settings of ESN). The spectral radius of the ESN is fixed to 0.95, and we vary the number of nodes as $M = 12, 30, 60, 100,$ and 200, and, for each number of nodes, we perform ten cycles of experimental runs using different ESN internal connection weights. As a result, we obtain $C = 4.48 \pm 0.53$ for $M = 12$, $C = 11.15 \pm 0.52$ for $M = 30$, $C = 20.89 \pm 1.35$ for $M = 60$, $C = 28.16 \pm 1.93$ for $M = 100$, and $C = 31.88 \pm 1.67$ for $M = 200$, suggesting that QR compares favorably to ESN.

2. The NARMA task

As explained in the main text, the emulation of NARMA systems is a challenge for machine-learning systems, in general, because it requires nonlinearity and memory [22]. Thus, the emulation task has been a benchmark for evaluating the amount of nonlinearity and memory to be exploited in the system [20,24,31–33,35,54,55]. These tasks appear as the second demonstration of QRC in the
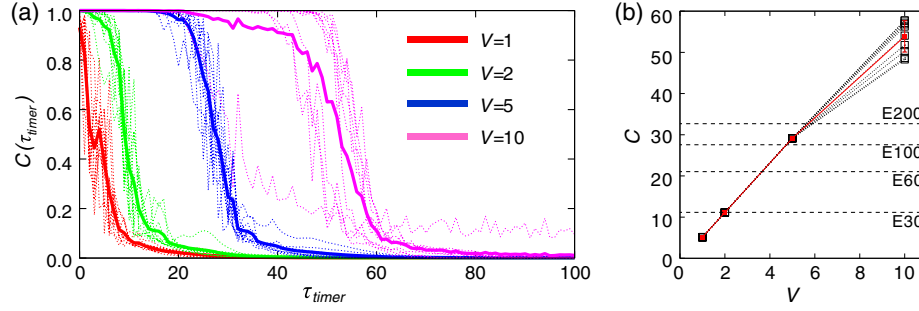
FIG. 11.   Analyses of the memory capacity based on $C(\tau_{\text{timer}})$ and C for a six-qubit QR system. (a) Plot showing $C(\tau_{\text{timer}})$ according to τ_{timer} for $V = 1, 2, 5,$ and 10. For each setting of V, ten different six-qubit QR systems are prepared, and for each system, the average of $C(\tau_{\text{timer}})$ is calculated using five trials according to τ_{timer} (dashed lines: the cases for ten different QR systems are overlaid), and the averaged values for these averaged $C(\tau_{\text{timer}})$ over ten different QR systems are overlaid as solid lines. (b) Plot of C as a function of V. Similar to (a), the averaged C for each different QR system for each number of V is overlaid as a dashed line, and the averaged values for these averaged C over ten different QR systems are overlaid as solid lines. The error bars represent the standard deviations. For comparisons, the dashed lines show the performance of ESN with M nodes (e.g., E100 represents the performance of ESN with 100 nodes). See the text for details on analyses.

main text [Fig. 4(b) in the main text]. Here, we explain the experimental procedures in detail and present extended analyses for these tasks.

We use a superimposed sine wave for the input to the NARMA systems, which is expressed as follows:

$$s_k = 0.1 \left[\sin\left(\frac{2\pi\alpha k}{T}\right) \sin\left(\frac{2\pi\beta k}{T}\right) \sin\left(\frac{2\pi\gamma k}{T}\right) + 1 \right],$$

where $(\alpha, \beta, \gamma) = (2.11, 3.73, 4.11)$ and $T = 100$. Note that s_k is in the range $[0, 0.2]$ with the aim of stabilizing the behavior of the NARMA systems (to prevent divergence). Similar types of input sequences for NARMA systems can be found in Refs. [31–33,35]. The input range is rescaled to $[0, 1]$ when projected to the first qubit of the QR system. The experimental trial consists of 5000 time steps, where the first 1000 time steps are used for the washout, the following 3000 time steps are used for the training phase, and the final 1000 time steps are used for the evaluation phase. Note that, when the input is a superimposed sine wave, we should be careful to prevent the same input and target output time series in the training phase from appearing again in the evaluation phase, because this would not enable us to characterize the generalization capability of the system effectively. Our setting of the length of the training and evaluation phases is confirmed to be safe on this point. By collecting the QR time series and the corresponding target outputs for each task in the training phase, we train the linear readouts for five outputs, which correspond to the five target NARMA systems, by using the scheme explained in the main text. The trained linear readouts are used to generate system outputs for the evaluation phase.

The contribution of the QR system is characterized explicitly by comparing the task performance with a simple linear-regression (LR) model, $y_{k+1} = w'_1 \times s_k + w'_0$, where

w'_0 and w'_1 are trained using the same time series as in the training phase. Note that this corresponds to the case in which no QR system is available, and only the raw input remains for LR. This comparison enables us to conclude that, for any system performance exceeding that of this model, the QR system has contributed to the emulation task [31–35].

We evaluate the performance of the system output in the evaluation phase by calculating the normalized mean-squared error (NMSE) with the target output:

$$\text{NMSE} = \frac{\sum_{k=L+1}^{M-L} (\bar{y}_{k+1} - y_{k+1})^2}{\sum_{k=L+1}^{M-L} \bar{y}_{k+1}^2}, \tag{A1}$$

where L represents the time steps for the washout and training phase, of which the duration is 4000 time steps in this experiment, and M is the time steps for the evaluation phase, which requires 1000 time steps. Table I lists the NMSE for each of the experimental conditions. We can confirm that our six-qubit QR system outperforms the LR system in any setting of V for each NARMA task, which implies that the QR system has contributed to the task performance. Furthermore, we can see that, by increasing V, the performance improves in all the NARMA tasks, which is consistent with the plots presented in Fig. 4(b) in the main text.

Here we aim to further analyze the information processing capacity of our QR system based on the NARMA tasks. We adopt the same task settings as for the previous case except for the input settings. The input stream is generated by using white noise with a range of $[0, 0.2]$ for the same reason as in the previous experiment, rather than using a superimposed sine wave. This choice of input stream is commonly used [20,24,55] and is determined not to add additional temporal coherence originating from external

TABLE I. Performance of the six-qubit QR systems in terms of NMSE for NARMA tasks using the superimposed sine wave.

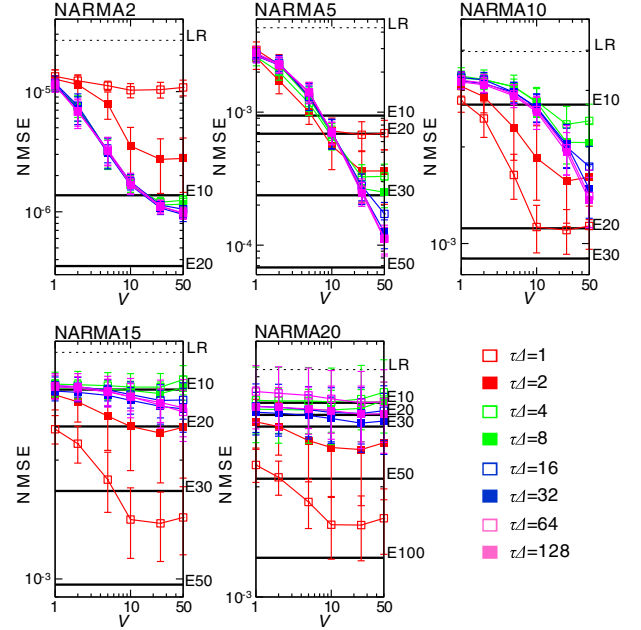| Task	System	Error (NMSE)
NARMA2	LR	1.7×10^{-5}
	QR ($V = 1$)	1.0×10^{-5}
	QR ($V = 2$)	4.7×10^{-6}
	QR ($V = 5$)	1.7×10^{-7}
	QR ($V = 10$)	4.9×10^{-8}
NARMA5	LR	3.0×10^{-3}
	QR ($V = 1$)	4.6×10^{-4}
	QR ($V = 2$)	7.1×10^{-5}
	QR ($V = 5$)	2.8×10^{-5}
	QR ($V = 10$)	7.6×10^{-6}
NARMA10	LR	2.6×10^{-3}
	QR ($V = 1$)	2.0×10^{-4}
	QR ($V = 2$)	9.2×10^{-5}
	QR ($V = 5$)	3.0×10^{-5}
	QR ($V = 10$)	1.3×10^{-5}
NARMA15	LR	2.7×10^{-3}
	QR ($V = 1$)	6.7×10^{-4}
	QR ($V = 2$)	3.1×10^{-4}
	QR ($V = 5$)	1.2×10^{-4}
	QR ($V = 10$)	4.0×10^{-5}
NARMA20	LR	2.3×10^{-3}
	QR ($V = 1$)	1.2×10^{-3}
	QR ($V = 2$)	2.6×10^{-4}
	QR ($V = 5$)	1.3×10^{-4}
	QR ($V = 10$)	3.8×10^{-5}

FIG. 12. Performance of five-qubit QR systems with several $\tau\Delta$ settings for the NARMA tasks with a random input stream. For each plot, the vertical and horizontal axes show the NMSE and the number of virtual nodes V, respectively, and both axes are in logarithmic scales. The error bars show the standard deviations evaluated on 20 samples of the QRs with respect to the random couplings. For comparisons, the dashed line shows the performance of an LR system in terms of NMSE, and the solid lines show the performance of an ESN with M nodes (e.g., "E10" represents the performance of ESN with 10 nodes). See the text for details on analyses and experimental procedures.

input to the system and to evaluate the pure computational power contributed only by the QR systems. As this input setting perceivably complicates the performance evaluation, we quantify the task performance in terms of NMSE. For each NARMA task, we test the relevance of τ and V in terms of the task performance and vary them for $\tau\Delta = 1, 2, 4, 8, 16, 32, 64,$ and 128, and $V = 1, 2, 5, 10, 25,$ and 50, respectively. Using a five-qubit QR system, 20 samples of the QRs are randomly generated, and for each $(\tau\Delta, V)$ setting, the average values of NMSEs are obtained.

The performance of the QR systems is characterized by again using the previously mentioned LR system for comparison. Furthermore, we use a conventional ESN as a candidate for the standard machine-learning system and use it to compare our task performance. The basic settings of the ESN are described in Appendix A 4. To allow for fair comparisons, 100 samples of ESN with M nodes are generated by assigning the same NARMA tasks with the same experimental settings explained above, and the average values of NMSEs for the ESN are obtained (by varying the spectral radius of the ESN internal weight matrix from 0.05 to 1.95 in increments of 0.1; the case of the smallest NMSE, which provides the best performance, is used for comparison). The number of nodes M is varied for $M = 10, 20, 30, 50,$ and 100 for each NARMA task.

Figure 12 depicts the results for the performance of the five-qubit QR system for the NARMA tasks with a random input stream. First, we can see that, for all the NARMA tasks, our five-qubit QR system outperforms the LR system, which means that the QR system actually contributes to the task performance. In general, we can confirm that the increase in the number of the virtual nodes V leads to an improved performance. The effect of the number of $\tau\Delta$ on the task performance depends on the type of task. For the NARMA2 and NARMA5 tasks, an increase in the number of $\tau\Delta$ tends to improve the performance. In particular, for the NARMA5 task, when $V = 50$ and $\tau\Delta = 128$, the five-qubit QR system performs at a level in between the performance of the ESN with $M = 30$ and $M = 50$ (see Fig. 12). For NARMA10, NARMA15, and NARMA20, this tendency does not hold, in general, and the use of a small number of $\tau\Delta$ is found to improve the performance. In particular, for the NARMA20 task, when $\tau\Delta = 1$ and $V > 5$, the five-qubit QR system performs at a level in between the performance of the ESN with $M = 50$ and $M = 100$ (see Fig. 12). Interestingly, this implies that the five-qubit QR system can deliver a performance similar to that of an ESN with fewer computational nodes (e.g., when $\tau\Delta = 1$ and $V = 5$, the five-qubit QR system has 25

computational nodes and the performance exceeds that of an ESN with 50 nodes). These outcomes of the task performance are induced by the balancing of memory and nonlinearity, which can be exploited by the system and which is required to perform the task; this is closely related to the results mentioned in the main text. Further analyses will be included in our future work.

3. The Mackey-Glass prediction task

Chaotic attractor learning is a popular test for learning dynamical systems [18,56]. One of the well-known systems used for the target of learning is the MG delay differential equation

$$\dot{y}(t) = \frac{\alpha y(t - \tau_{\text{MG}})}{1 + y(t - \tau_{\text{MG}})^\beta} - \gamma y(t),$$

where the parameters are set to $\alpha = 0.2$, $\beta = 10$, and $\gamma = 0.1$. The system has a chaotic attractor if $\tau_{\text{MG}} > 16.8$. In the majority of studies, $\tau_{\text{MG}} = 17$ is used, which yields a chaotic attractor. In our experiments, we also use this parameter setting of $\tau_{\text{MG}} = 17$. Additionally, we use the case of $\tau_{\text{MG}} = 16$ for comparison, as this setting does not exhibit chaos.

The discrete time version of the MG system is often used to prepare the training sequences [18] through

$$y_{k+1} = y_k + \sigma \left(\frac{0.2 y_{k - \frac{\tau_{\text{MG}}}{\sigma}}}{1 + y_{k - \frac{\tau_{\text{MG}}}{\sigma}}^{10}} - 0.1 y_k \right),$$

with a step size of $\sigma = 1/10$, and then subsampled by 10. One step from k to $k + 1$ in the resulting sequences corresponds to a unit time interval $[t, t + 1]$ of the original continuous system. In our numerical experiments, the target time series is linearly scaled to [0, 1] and used in the actual experiments.

For each setting of τ_{MG}, we generate the above system for a while as a washout, and then a length of 12 000 time steps (already subsampled) is collected for the experiment. We use 10 000 time steps for the training phase and the remaining 2000 time steps for the evaluation phase. The task is to train the QR system by using these training sequences, which after training should regenerate the corresponding chaotic or nonchaotic attractors.

Because this task requires feedback to the system, the training procedure is different from the previous cases. During the training phase, we clamp the feedback from the system output and provide the target outputs as inputs, which means we set $s_{k+1} = \bar{y}_k$. Thus, the training phase is carried out with an open loop, such that the system is forced into the desired operative state by the target signals (this approach is typically referred to as *teacher forcing*). The robustness of the learning is improved by adding a slight amount of noise in the range of $[-\sigma, \sigma]$ in the training phase.

TABLE II. Experimental settings and prediction errors (NMSE) for the Mackey-Glass prediction tasks in the main text.

τ_{MG}	Case	Qubit	$\tau\Delta$	Noise strength (σ)	Error (NMSE)
16	1	6	3	1.0×10^{-4}	4.7×10^{-3}
	2	7	2	1.0×10^{-4}	3.9×10^{-3}
17	1	6	3	1.0×10^{-4}	1.6×10^{-1}
	2	7	3	1.0×10^{-4}	2.5×10^{-2}
	3	7	4	1.0×10^{-5}	4.9×10^{-2}
	4	7	2	1.0×10^{-5}	1.7×10^{-2}

When the evaluation phase starts, we switch the inputs to the system output generated by the trained readout weights [this phase is expressed as the autonomous phase in Fig. 4(c) in the main text] and check whether the system is able to embed the corresponding MG system.

Table II summarizes the experimental conditions and the prediction errors for the QR system used in the main text. We calculate the errors in NMSE by using the entire time series in the evaluation phase.

We test whether the trained network indeed generates a chaotic time series by empirically estimating the largest Lyapunov exponent of the network-generated output signal by using a procedure similar to that introduced in Ref. [56]. For the trained network, we analyze the previous four cases (cases 1–4) in the $\tau_{\text{MG}} = 17$ setting. When the network is switched from the teacher-forcing condition to the closed-loop mode at time step 10 000, the reservoir signals are perturbed by a uniform noise vector, the network is left running freely, on this occasion starting from the perturbed state for the entire 2000 steps of the evaluation phase, and the resulting output sequence is recorded. The exponential divergence rate λ between this perturbed sequence y'_k and the original sequence y_k is estimated by computing

$$d_k = \|[y_{10001+k} \cdots y_{10017+k}] - [y'_{10001+k} \cdots y'_{10017+k}]\|,$$

$$\lambda = \frac{\log(d_{500}) - \log(d_0)}{500},$$

where the subsequent 17 time steps that are used for the computation of d_k are chosen because they correspond to approximately one full "loop" of the attractor. Figure 13 plots the behavior of d_k for four cases. We can see that all four cases have a positive λ value, which implies that their output sequences are chaotic.

4. Echo-state network settings for comparisons

We further characterized the computational power of our QR system by comparing its task performance with that of a conventional ESN [18,56,61,62] (the comparisons of ESN performance with that of our systems appear in Sec. IV A and also in the analyses of the NARMA tasks demonstrated above). The ESN is a type of recurrent neural network, which has M internal network units, input units, and output
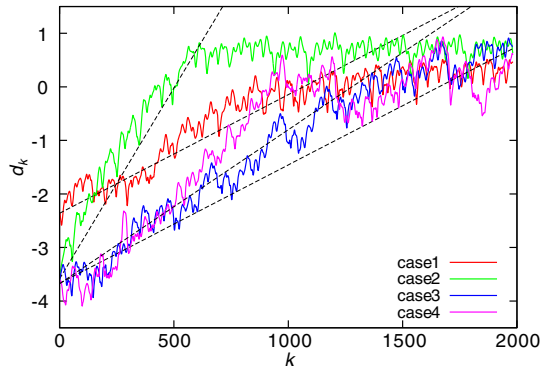
FIG. 13.   Plot showing the time series of d_k. The results for cases 1–4 are overlaid. Note that the vertical axis is in a logarithmic scale. The estimated largest Lyapunov exponent λ is 0.0022, 0.0071, 0.0022, and 0.0029 for case 1, 2, 3, and 4, respectively. Note that, according to Ref. [60], it has been investigated that a Mackey-Glass system that has the same parameter settings has the largest Lyapunov exponent of 0.007, which is similar in magnitude to our system's results.

FIG. 14.   Capacities $(C_{\mathrm{STM}}, C_{\mathrm{PC}})$ of the ESNs with two types of input sequence. Cases I and II represent the input type with $\{-1, 1\}$ (colored blue) and with $\{0, 1\}$ (colored red), respectively. Plots show the results for the spectral radius from 0.05 to 2.00. The error bars show the standard deviations evaluated on 100 samples of the randomly generated ESNs. From top to bottom, the spectral radius is changed from 0.5 to 2.0.

units. Activation of the ith internal unit at time step k is x_k^i ($i = 1, \ldots, M$). We use the same I/O setting for the ESN as with our system for each task concerned to enable us to efficiently and directly compare the performance. The number of trials, the lengths of the washout, training, and evaluation phases, and the evaluation procedures are also kept the same. The connection weights for the $M \times M$ internal network connecting the ith unit with the jth unit are denoted as w_{ij}, and the input weights proceeding from the input unit into the ith internal unit are denoted as w_{in}^i. The readout weights w_{out}^i proceed from M internal units and one bias to the output unit (where $x_k^0 = 1$ and w_{out}^0 for a bias term). The readout weights w_{out}^i are trained using the procedure explained for each task; the internal weights w_{ij} and the input weights w_{in}^i are randomly assigned from the range $[-1.0, 1.0]$ and fixed beforehand. The activation of the internal units and the output unit are updated as

$$x_k^i = f\left(\sum_{j=1}^{M} w_{ij} x_{k-1}^j + w_{\mathrm{in}}^i s_k\right),$$

$$y_k = \sum_{i=0}^{M} w_{\mathrm{out}}^i x_k^i,$$

where f is a tanh function. It is reported that the computational power of ESN can be well characterized by the spectral radius of the internal connection weight matrix [18,20,56,61,62]. In each comparative experiment, by incrementally varying the spectral radius, we observe the ESN performance. Detailed experimental conditions are given for each of these comparisons.

Here we present the ESN settings for the comparisons with QR systems that appear in Sec. IV A. The experimental
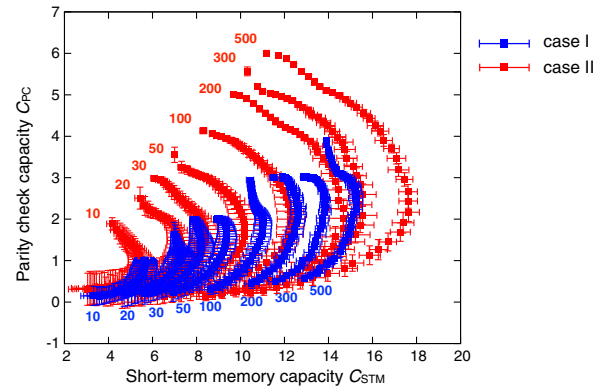
settings including the length of training and evaluation phases are kept the same with the QR system for a fair comparison. For the random binary input sequence, we adopt two cases. In the first case (case I), we change the actual input value to "−1" only if $s_k = 0$. For the second case (case II), we directly project the $\{0, 1\}$-binary state input s_k to the internal network units. In the ESN, if $s_k = 0$, the internal units receive no external input and, therefore, are expected to introduce an asymmetry into the network performance. We test these two cases (case I and case II) in Fig. 14. As can be seen from the plot, both cases show different modalities of performances in terms of C_{PC} and C_{STM}, which are due to the asymmetry introduced by the input settings. We present the results for case I in Sec. IV A, but the same explanations hold for both cases.

[1] R. P. Feynman, Simulating physics with computers, Int. J. Theor. Phys. **21,** 467 (1982).

[2] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2010).

[3] K. Fujii, *Quantum Computation with Topological Codes— From Qubit to Topological Fault-Tolerance*, Springer Briefs in Mathematical Physics (Springer-Verlag, Berlin, 2015).

[4] P. W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (IEEE, New York, 1994), p. 124.

[5] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, C. Neill, P. OMalley, P. Roushan, A. Vainsencher, J. Wenner, A. N. Korotkov, A. N. Cleland, and J. M. Martinis, Superconducting

quantum circuits at the surface code threshold for fault tolerance, Nature (London) **508,** 500 (2014).

[6] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, I.-C. Hoi, C. Neill, P. J. J. OMalley, C. Quintana, P. Roushan, A. Vainsencher, J. Wenner, A. N. Cleland, and J. M. Martinis, State preservation by repetitive error detection in a superconducting quantum circuit, Nature (London) **519,** 66 (2015).

[7] J. I. Cirac and P. Zoller, Goals and opportunities in quantum simulation, Nat. Phys. **8,** 264 (2012).

[8] I. Bloch, J. Dalibard, and S. Nascimbène, Quantum simulations with ultracold quantum gases, Nat. Phys. **8,** 267 (2012).

[9] I. M. Georgescu, S. Ashhab, and F. Nori, Quantum simulation, Rev. Mod. Phys. **86,** 153 (2014).

[10] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58,** 5355 (1998).

[11] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem, Science **292,** 472 (2001).

[12] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer, Defining and detecting quantum speedup, Science **345,** 420 (2014).

[13] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, Evidence for quantum annealing with more than one hundred qubits, Nat. Phys. **10,** 218 (2014).

[14] T. Morimae, K. Fujii, and J. F. Fitzsimons, Hardness of Classically Simulating the One-Clean-Qubit Model, Phys. Rev. Lett. **112,** 130502 (2014).

[15] K. Fujii, H. Kobayashi, T. Morimae, H. Nishimura, S. Tamate, and S. Tani, in *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)* (Schloss Dagstuhl Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2016), pp. 13:1–13:14.

[16] K. Fujii and S. Tamate, Computational quantum-classical boundary of complex and noisy quantum systems, Sci. Rep. **6,** 25598 (2016).

[17] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, A million spiking-neuron integrated circuit with a scalable communication network and interface, Science **345,** 668 (2014).

[18] H. Jaeger and H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science **304,** 78 (2004).

[19] W. Maass, T. Natschläger, and H. Markram, Real-time computing without stable states: A new framework for neural computation based on perturbations, Neural Comput. **14,** 2531 (2002).

[20] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, An experimental unification of reservoir computing methods, Neural Netw. **20,** 391 (2007).

[21] M. Rabinovich, R. Huerta, and G. Laurent, Transient dynamics for neural processing, Science **321,** 48 (2008).

[22] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, Information processing capacity of dynamical systems, Sci. Rep. **2,** 514 (2012).

[23] C. Fernando and S. Sojakka, Pattern recognition in a bucket, in *European Conference on Artificial Life* (Springer, Berlin, Heidelberg, 2003), pp. 588–597.

[24] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, Information processing using a single dynamical node as complex system, Nat. Commun. **2,** 468 (2011).

[25] D. Woods and T. J. Naughton, Photonic neural networks, Nat. Phys. **8,** 257 (2012).

[26] L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutierrez, L. Pesquera, C. R. Mirasso, and I. Fischer, Photonic information processing beyond Turing: An optoelectronic implementation of reservoir computing, Opt. Express **20,** 3241 (2012).

[27] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar, Optoelectronic reservoir computing, Sci. Rep. **2,** 287 (2012).

[28] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, Parallel photonic information processing at gigabyte per second data rates using transient states, Nat. Commun. **4,** 1364 (2013).

[29] K. Vandoorne, P. Mechet, T. V. Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, Experimental demonstration of reservoir computing on a silicon photonics chip, Nat. Commun. **5,** 3541 (2014).

[30] A. Z. Stieg, A. V. Avizienis, H. O. Sillin, C. Martin-Olmos, M. Aono, and J. K. Gimzewski, Emergent criticality in complex turing B-type atomic switch networks, Adv. Mater. **24,** 286 (2012).

[31] H. Hauser, A. J. Ijspeert, R. M. Füchslin, R. Pfeifer, and W. Maass, Towards a theoretical foundation for morphological computation with compliant bodies, Biol. Cybern. **105,** 355 (2011).

[32] K. Nakajima, H. Hauser, R. Kang, E. Guglielmino, D. G. Caldwell, and R. Pfeifer, in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, New York, 2013), p. 1496.

[33] K. Nakajima, H. Hauser, R. Kang, E. Guglielmino, D. G. Caldwell, and R. Pfeifer, A soft body as a reservoir: Case studies in a dynamic model of octopus-inspired soft robotic arm, Front. Comput. Neurosci. **7,** 1 (2013).

[34] K. Nakajima, T. Li, H. Hauser, and R. Pfeifer, Exploiting short-term memory in soft body dynamics as a computational resource, J. R. Soc. Interface **11,** 20140437 (2014).

[35] K. Nakajima, H. Hauser, T. Li, and R. Pfeifer, Information processing via physical soft body, Sci. Rep. **5,** 10487 (2015).

[36] K. Caluwaerts, J. Despraz, A. Işçen, A. P. Sabelhaus, J. Bruce, B. Schrauwen, and V. SunSpiral, Design and control of compliant tensegrity robots through simulations and hardware validation, J. R. Soc. Interface **11,** 20140520 (2014).

[37] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521,** 436 (2015).

[38] H. J. Briegel and G. De las Cuevas, Projective simulation for artificial intelligence, Sci. Rep. **2**, 400 (2012).

[39] G. D. Paparo, V. Dunjko, A. Makmal, M. A. Martin-Delgado, and H. J. Briegel, Quantum Speedup for Active Learning Agents, Phys. Rev. X **4**, 031002 (2014).

[40] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum Support Vector Machine for Big Data Classification, Phys. Rev. Lett. **113**, 130503 (2014).

[41] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. **10**, 631 (2014).

[42] N. Wiebe, A. Kapoor, and K. M. Svore, Quantum deep learning, arXiv:1412.3489.

[43] J. C. Adcock, E. Allen, M. Day, S. Frick, J. Hinchliff, M. Johnson, S. Morley-Short, S. Pallister, A. Price, and S. Stanisic, Advances in quantum machine learning, arXiv:1512.02900.

[44] In this paper, to evaluate the performance of our proposed system for several tasks, we compare the task performance of our platform with the conventional ESN. On this point, we stress that this comparison is not based on the intention of claiming that our platform is superior to ESN, in general, but it is introduced as a reference to grasp a sense of the information-processing capability of our system. All of the detailed settings of ESN used for the comparisons are given in the Appendix in order to guarantee the reproducibility in a statistical sense.

[45] E. P. Wigner, On the Statistical Distribution of the Widths and Spacings of Nuclear Resonance Levels, *Mathematical Proceedings of the Cambridge Philosophical Society* Vol. 47 (Cambridge University Press, Cambridge, England, 1951).

[46] T. Guhr, A. MüllerGroeling, and H. A. Weidenmüller, Random-matrix theories in quantum physics: Common concepts, Phys. Rep. **299**, 189 (1998).

[47] D. G. Cory, R. Laflamme, E. Knill, L. Viola, T. F. Havel, N. Boulant, G. Boutis, E. Fortunato, S. Lloyd, R. Martinez, C. Negrevergne, M. Pravia, Y. Sharf, G. Teklemariam, Y. S. Weinstein, and W. H. Zurek, NMR based quantum information processing: Achievements and prospects, Fortschr. Phys. **48**, 875 (2000). http://onlinelibrary.wiley.com/doi/10.1002/1521-3978(200009)48:9/11<875::AID-PROP875>3.0.CO;2-V/abstract

[48] J. A. Jones, Quantum computing with NMR, Prog. Nucl. Magn. Reson. Spectrosc. **59**, 91 (2011).

[49] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, in *Proceedings of the 35th ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, 2003), pp. 59–68.

[50] A. M. Childs, Universal Computation by Quantum Walk, Phys. Rev. Lett. **102**, 180501 (2009).

[51] P. Antonik, M. Haelterman, and S. Massar, Brain-Inspired Photonic Signal Processor for Generating Periodic Patterns and Emulating Chaotic Systems, Phys. Rev. Applied **7**, 054014 (2017).

[52] B. Georgeot and D. L. Shepelyansky, Quantum chaos border for quantum computing, Phys. Rev. E **62**, 3504 (2000).

[53] R. Laje and D. V. Buonomano, Robust timing and motor patterns by taming chaos in recurrent neural networks, Nat. Neurosci. **16**, 925 (2013).

[54] S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural Comput. **9**, 1735 (1997).

[55] A. F. Atiya and A. G. Parlos, New results on recurrent network training: Unifying the algorithms and accelerating convergence, IEEE Trans. Neural Networks **11**, 697 (2000).

[56] H. Jaeger, GMD Report No. 148, German National Research Institute for Computer Science, 2001.

[57] N. Bertschinger and T. Natschläger, Real-time computation at the edge of chaos in recurrent neural networks, Neural Comput. **16**, 1413 (2004).

[58] P. Hayden and J. Preskill, Black holes as mirrors: Quantum information in random subsystems, J. High Energy Phys. 09 (2007) 120.

[59] P. Hosur, X.-L. Qi, D. A. Roberts, and B. Yoshida, Chaos in quantum channels, J. High Energy Phys. 02 (2016) 004.

[60] J. D. Farmer, Chaotic attractors of an infinite-dimensional dynamical system, Physica (Amsterdam) **4D**, 366 (1982).

[61] H. Jaeger, GMD Report No. 159, German National Research Center for Information Technology, 2002.

[62] H. Jaeger, GMD Report No. 152, German National Research Center for Information Technology, 2001.