Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Scalable physical deep learning using optical dynamics with state-skipping training

Yongbo Zhang

zhang@isi.imi.i.u-tokyo.ac.jp

The University of Tokyo https://orcid.org/0009-0003-8475-2145

Mitsumasa Nakajima

NTT Device Technology Laboratories

Katsuma Inoue

The University of Tokyo

Toshikazu Hashimoto

NTT Device Technology Laboratories

Yasuo Kuniyoshi

Graduate School of Information Science & Technology, The University of Tokyo https://orcid.org/0000-0001-8443-4161

Kohei Nakajima

The University of Tokyo https://orcid.org/0000-0001-5589-4054

Article

Keywords:

Posted Date: January 13th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8145052/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: There is **NO** Competing Interest.

Version of Record: A version of this preprint was published at Nature Computational Science on November 5th, 2026. See the published version at https://doi.org/10.1038/s43588-026-01075-w.

Scalable physical deep learning using optical dynamics with state-skipping training

Yongbo Zhang[1*], Mitsumasa Nakajima[2], Katsuma Inoue[1], Toshikazu Hashimoto[2], Yasuo Kuniyoshi[1,3], and Kohei Nakajima[1,3]

[1*]Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan.
[2]NTT Device Technology Labs., 3-1 Morinosato-Wakamiya, Atsugi, Kanagwa, Japan.
[3]Next Generation Artificial Intelligence Research Center, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan.

*Corresponding author(s). E-mail(s): zhang@isi.imi.i.u-tokyo.ac.jp;

Abstract

The escalating energy demands for high-performance machine learning have sparked growing interest in unconventional computing paradigms rooted in physical systems. At the core of this emerging direction is the leveraging of physical properties to design computing frameworks that integrate strong learning capabilities, efficient training strategies, and practical implementability. Here, we present a simple, efficient, and scalable framework aimed at achieving this, and validate it on an optoelectronic platform. This framework features our proposed training mechanism—state-skipping direct feedback alignment—which eliminates access to intermediate states of the system and thereby significantly simplifies the error backpropagation process, enhancing both training efficiency and practical feasibility. Compared to conventional deep neural networks, our approach substantially reduces computational costs and training parameter count while achieving comparable performance. Furthermore, we integrate our scheme into modern architectures, attaining improved performance with an approximately 40% reduction in computational resources. Notably, our approach challenges the scaling laws of conventional counterparts, underscoring its strong scalability and practical promise.

1 Introduction

Deep learning has performed remarkably well across a wide range of complex tasks as model scales continue to grow. However, this success has come with sharply rising demands for computational energy and hardware resources, raising concerns about its long-term sustainability [1–3]. Consequently, the development of computationally efficient paradigms has become a priority. Unconventional computational paradigms based on physical platforms are increasingly recognized as a promising way to overcome the computational bottlenecks of deep learning. Unlike the conventional Von Neumann architecture, which separates processing units and memory, these paradigms harness physical processes for information processing, offering inherent advantages in latency and energy efficiency [4–9]. Various physical platforms have been explored to demonstrate this potential. For instance, spintronic systems leverage electron spin to integrate memory and logic operations [10–12], while soft-body and mechanical components exploit mechanical responses to emulate neural behavior with low energy consumption [7, 13–16], other approaches, such as phase-change materials and chemical reaction networks have also shown promise [17, 18]. Among these, optical neural networks (ONNs) stand out due to their unique physical properties, including ultrafast propagation speed, inherent parallelism, and near-zero heat dissipation. These advantages make ONNs strong candidates for scalable and energy-efficient computational paradigms [19–21]. However, challenges remain—particularly the limited flexibility in adjusting the weights of optical layers, which hampers efficient training. Additionally, reliance on complex control systems and repeated photoelectric conversions further increases architectural complexity [22, 23].

Reservoir computing (RC), characterized by randomly initialized and fixed internal connectivity, offers a promising paradigm to address the aforementioned issues. Owing to its simple, weight-frozen architecture, RC can be implemented across diverse physical substrates, leveraging their dynamics to perform nonlinear information processing [24–26]. Conventional RC is driven by external inputs and excels at time-series processing. However, for classification and image processing, it typically serializes information or images into pixel streams, disrupting spatial structure and complicating wiring and control overhead [25, 27, 28]. Instead, autonomous state-propagating RC (ASP-RC) encodes static information/images as initial states and lets the system evolve to produce high-dimensional transient trajectories—an approach better suited to classification and image processing [29–31], as shown in Fig. 1a. In physical implementations, ASP-RC operates without continuous driving inputs, largely outsourcing computations to passive dynamics and thereby improving energy efficiency and simplifying deployment. To meet the demands of modern large-scale image tasks, ASP-RC is gaining attention. For instance, Refs. [32, 33] implement ASP-RC on analog circuits and photonic platforms, respectively, by exploiting chaotic dynamics to achieve high-dimensional transformations. In parallel, Refs. [29, 34] present implementations on spintronic simulators, as shown in Fig. 1b. Across these studies, training primarily relies on two strategies: readout-only training via regression and backpropagation (BP), the prevailing algorithm in software-based neural networks. The former offers implementation simplicity but struggles to achieve competitive performance. The latter improves performance but remains challenging to implement physically, due to

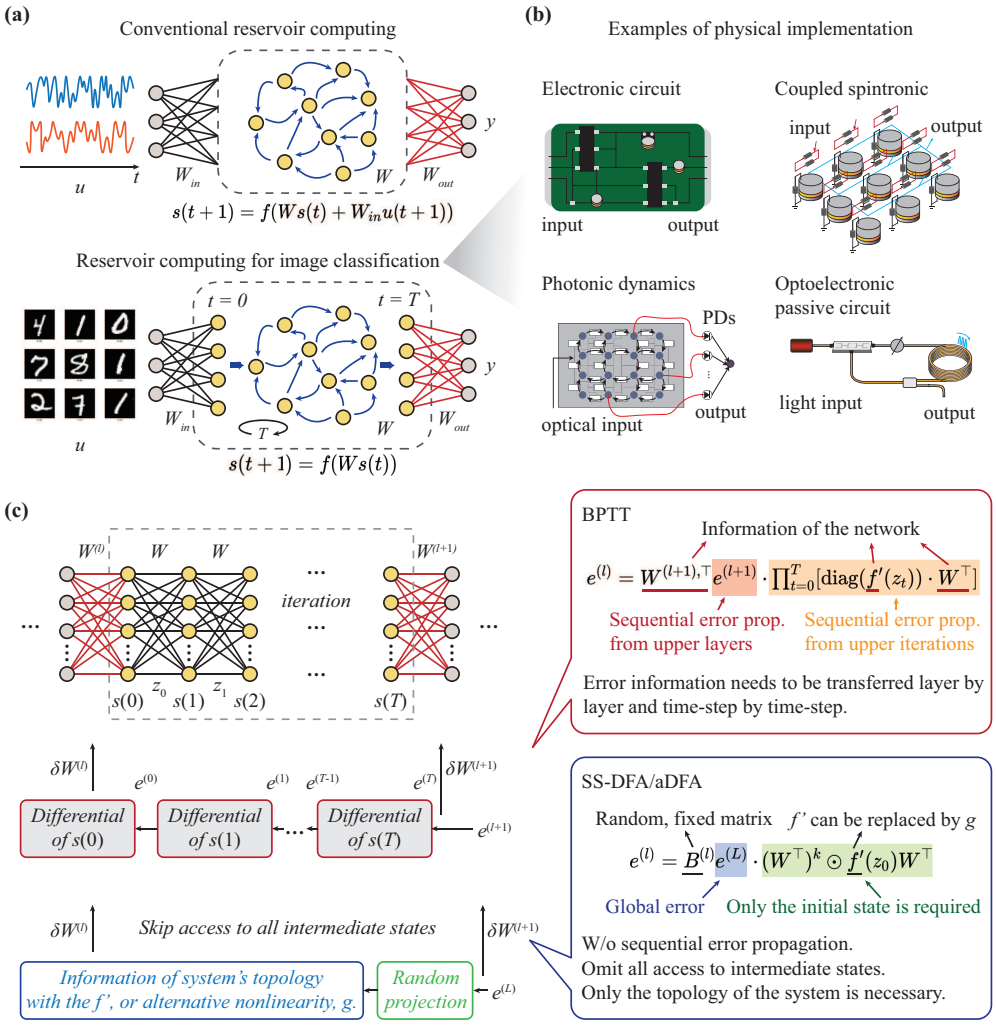


Fig. 1 (a) Schematics of conventional reservoir computing (RC) for processing temporal signals and of autonomous state-propagating RC (ASP-RC) for processing images. (b) Schematic diagram of unconventional computing paradigms on diverse physical platforms based on ASP-RC. (c) Illustrations comparing backpropagation through time (BPTT) and the proposed state-skipping direct feedback alignment (SS-DFA) used to train the model. W denotes the internal connectivity of the reservoir layer, which remains fixed throughout the entire process, while $W^{(l)}$ and $W^{(l+1)}$ represent the trainable adjacent linear layers. SS-DFA breaks the sequential error propagation, eliminates the need to access all intermediate states within the reservoir layer, enables parallel training, and supports a gradient-free variant, which is called SS-augmented DFA (SS-aDFA), by replacing the precise derivative f' with alternative nonlinear functions g.

the requirements for time-ordered error propagation, storage and retrieval of intermediate states, and high-precision computation—even when physical-layer weights

remain unchanged. Consequently, training becomes a bottleneck, hindering the realization of energy-efficient, high-performance ONNs within this paradigm, that is, optical autonomous state-propagating RC (OASP-RC).

From the neuromorphic computing perspective, direct feedback alignment (DFA) offers an alternative to BP by projecting the global error through fixed random mappings directly onto each layer, thereby eliminating the need for layer-by-layer gradient propagation [35, 36]. Recent advances have extended DFA to gradient-free training, such as augmented DFA, which replaces activation-function derivatives with alternative nonlinear functions [37, 38]. As another line of development, reservoir DFA, offers a variant that harnesses physical dynamics to generate random projections [39]. Taken together, these advances underscore DFA's hardware adaptability and practical implementability. Inspired by the attractions of DFA, we extend it from the viewpoint of state evolution over time, proposing a novel training strategy, state-skipping DFA (SS-DFA), for OASP-RC to address the aforementioned training bottleneck when implementing ONNs within this paradigm. SS-DFA eliminates the need to access all intermediate states, breaking dependence on time-ordered gradient propagation and repeated states storage/retrieval. In addition, it can be integrated with the aforementioned variants to further enhance practical realizability, thereby providing an efficient training strategy. It is worth emphasizing that, due to its ability to propagate gradients, our algorithm can be naturally extended to deep architectures, thereby offering superior scalability and an enhanced ability to tackle complex tasks.

In this work, we first validate the effectiveness of the proposed SS-DFA on a general, fully-coupled OASP-RC model. To minimize reliance on physical-layer information during the training and reduce computational cost, we then introduce the single-map architecture, a new proposed OASP-RC framework. We train it by using our algorithm and systematically evaluate its scalability in terms of network depth and width. Numerical simulations show that the single-map framework outperforms structurally matched fully connected networks and fully-coupled architectures while requiring minimal computational cost and far fewer trainable parameters. As a proof of concept, we conduct physical experiments on an optoelectronic RC platform. The results align with our simulation results, confirming the practical feasibility of both the proposed algorithm and the proposed framework. Finally, we extend the single-map framework to a modern architecture— the residual multilayer perceptron (ResMLP) [40, 41]—to challenge the scaling laws observed in conventional counterparts, thereby further highlighting its efficiency and scalability, while achieving better performance with an approximately 40% reduction in resources.

2 Results

2.1 SS-DFA and the single-map framework

Fig. 1c demonstrates the operating mechanism and training process of the ASP-RC model, details our proposed SS-DFA training strategy, and compares it with BP. During forward propagation, the output signal $a^{(l)} = W^{(l)}x^{(l)}$ from the preceding linear layer serves as the initial state $s(0)$ of the reservoir layer. The state update is described as $s(t+1) = f(z_t)$, where $z_t = Ws(t)$, and $s(t) \in \mathbb{R}^N$ denotes the state of system at

iteration t; $W \in \mathbb{R}^{N\times N}$ is the internal connectivity of the reservoir layer, which governs its underlying dynamical properties. It is randomly initialized and subsequently normalized as $W \leftarrow \rho W/\max(|\text{eigen}(W)|)$, where ρ denotes the target spectral radius. Once configured, W remains fixed throughout the entire process together with the nonlinear activation function f. After T iterations state-evolve, the final state $s(T)$ serves as the reservoir output fed to the subsequent layer. To train such networks, linear layers (e.g., $W^{(l)}$) need to be updated to minimize the prediction loss E. Although training is simplified—owing to the absence of external inputs during state evolution and fixed reservoir weights—in reservoir layers, error signals must still be propagated backward along the temporal dimension to deliver gradient information to upstream linear layers. Backpropagation through time (BPTT), as a temporal extension of the standard BP, is a widely used gradient computation method for training networks with temporal structures. It harnesses the chain rule to propagate output errors backward in temporal order, reaching all historical states and accumulating gradients at each time step [42]. The gradient of the reservoir layer state at time t, as obtained by BPTT, is given by:

$$e^{(t)} = [W^\top e^{(t+1)}] \odot f'(z_t) \tag{1}$$

where $e^{(t)} \in \mathbb{R}^N$ is defined as $e^{(t)} = \partial E/\partial s(t)$, the superscript $\top$ denotes transposition, and $\odot$ represents the Hadamard product. The error signal used to update $W^{(l)}$, that is, the cumulative error information propagated through the reservoir layer, can be further expanded as follows:

$$e^{(l)} = W^{(l+1),\top} e^{(l+1)} \cdot \textstyle\prod_{t=0}^{T}[\text{diag}(f'(z_t)) \cdot W^\top] \tag{2}$$

where $e^{(l)} \in \mathbb{R}^N$ denotes the error signal at the lth trainable linear layer, diag$(\cdot)$ is used to construct a diagonal matrix to enable element-wise operations, and $\prod_{t=0}^{T}[\cdot]$ is the multistep chain multiplication that represents the sequential accumulation of error information across time steps. From Equation 2, the gradient for each $W^{(l)}$ can be calculated as $\partial W^{(l)} = -e^{(l)} x^{(l),\top}$. It is evident that training based on Equation 2 requires propagating error signals both layer by layer and across time steps, accessing all intermediate states within the reservoir layer, and repeatedly computing the transpositions of the continually updated weight matrices in the linear layers. These requirements pose significant challenges for practical implementation and result in substantial computational cost. Therefore, despite its success in conventional neural network models, this training strategy is not well suited to OASP-RC.

Let us consider our proposed SS-DFA. First, the standard DFA strategy is adopted, where the global output error from the final layer L, $e^{(L)}$, is directly projected onto each linear layer through a fixed, randomly initialized mapping, thereby breaking the layer-by-layer error propagation mechanism. Then, the initial state of the reservoir layer and its system information accumulation are harnessed to replace the cumulative error term in Equation 2, thus eliminating the need to access all intermediate states. Based on this design, the following update rule is derived:

$$e^{(l)} = B^{(l)} e^{(L)} \cdot (W^\top)^k \odot f'(z_0) W^\top \tag{3}$$

where $B^{(l)} \in \mathbb{R}^{N \times N^L}$ is the random projection matrix for updating the lth layer, and $(W^\top)^k$ represents the accumulation of the system topology information over k iterations. In our experiments, for simplicity, k is set to $T-1$, meaning that the accumulation of topological information is fully preserved. However, in practice, k can be chosen to be smaller than $T-1$, thereby reducing the degree of accumulation (see Supplementary Information). It is worth noting that W remains fixed and acts like a scaling term, thus $(W^\top)^k$ can be precomputed offline and does not incur additional computational cost during training. Furthermore, the derivative function f' can be substituted by alternative nonlinearities g, enabling gradient-free training—SS-aDFA.

In a practical physical implementation, minimizing the precise system information required during training is a primary objective. While our approach eliminates dependence on access to intermediate states and exact derivatives, knowledge of the system topology (i.e., W) remains necessary. Further simplification reveals that the scaling factor ρ controlling the magnitude of W in Equation 3 can be omitted; however, accurate connectivity remains essential. A similar discussion can be found in [43]. In addition, due to temporal iteration, fully-coupled ASP-RC frameworks inevitably incur significant computational cost from extensive matrix – vector multiplications (MVMs). To overcome these constraints at the model level, we propose an OASP-RC framework using the single-map architecture, as shown in Fig. 2c. This framework centers on one-dimensional discrete dynamics, employing only the scalar ρ to modulate system properties. The state update is given by $s(t) = f(\rho s(t-\tau))$, where τ denotes the time delay (set to 1 for simplicity in this work). By omitting the scaling factor, its SS-DFA update rule can be expressed as:

$$e^{(l)} = B^{(l)} e^{(L)} \odot f'(z_0) \tag{4}$$

It can be observed that the single-map framework dispenses with explicit connectivity, simplifying training to dependence on the initial state alone. Simultaneously, it collapses per-iteration computation from MVMs to scalar operations, thereby markedly reducing computational cost and implementation complexity. In our experiments, to align with the nonlinear characteristics of optical systems, the activation function f is held constant as $f(x) = \sin(x + \phi)$ and, for simplicity, ϕ is set to 0.

2.2 Basic characterization of SS-DFA

First, we validate the effectiveness of the proposed SS-DFA on a fully-coupled OASP-RC model, systematically study its basic characterization, and further investigate opportunities for algorithmic simplification. In our experiments, we use the standard image classification benchmark—MNIST [44]—with the most basic architecture comprising a linear input layer, a reservoir layer with 500 nodes, and a linear output layer. Given that the reservoir behavior is governed primarily by its spectral radius ρ and iterations for the temporal evolution of the state, denoted by T, we systematically analyze the dependence of performance on ρ and T. The recording T is varied from 1 to 30, and ρ is swept over the range [0.1, 3.0]. Stochastic gradient descent (SGD) [45, 46] is employed as the optimizer with a learning rate of 0.005 and a batch size of 64. Additional training details are summarized in the Supplementary Information.

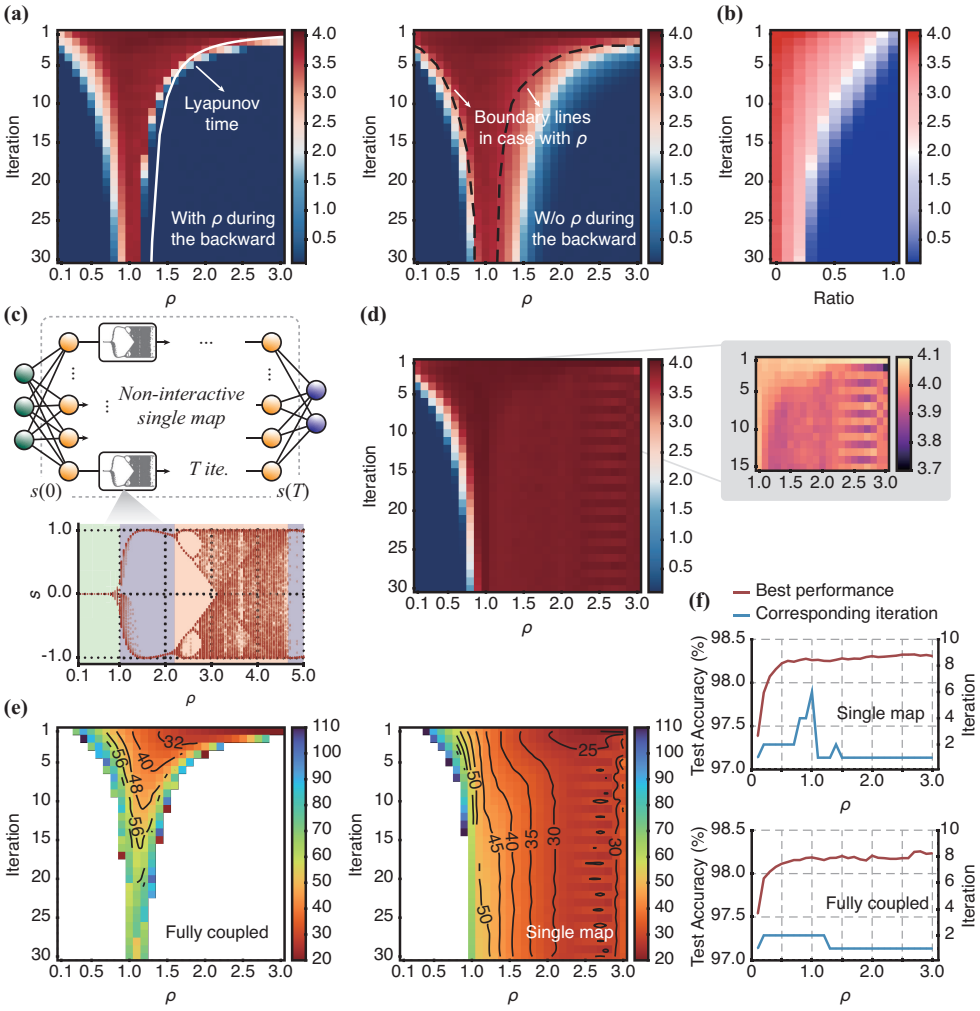


Fig. 2 (a) Performances on the MNIST dataset for the general, fully-coupled OASP-RC trained with SS-DFA. The x- and y-axes denote, respectively, the spectral radius ρ and the number of iterations of the reservoir layer. $|\log(\epsilon)|$ is used as a proxy for test accuracy (higher values indicating better performance), where ϵ denotes the prediction error. Left: training with ρ; the white curve denotes the Lyapunov time of the system, defining the effective prediction horizon. Right: ρ is omitted during training; black lines mark the boundary of the effective operating region inferred from the left panel. (b) Robustness to randomized reservoir connectivity during training ($\rho = 1$). In the backward process, proportionally selected elements in $W^\top$ are replaced with random values (the rest remain unchanged). The heatmap shows performance as a function of the replacement ratio and the reservoir iterations. (c) Schematic of the proposed single-map architecture and its bifurcation diagram, showing the (long-term) node state s as a function of the scaling factor ρ. The green region denotes the single fixed-point regime, the blue region is the dual-fixed-point (bistable) regime, and period doubling bifurcation is found in the red region. (d) Performance of the single-map framework trained with SS-DFA. The right panel shows an enlarged view of the performance in the transient expanding/chaotic regime. (e) Convergence speed of fully-coupled and single-map frameworks as a function of ρ and iterations. (f) The best performance achieved at different ρ, and the required iterations to achieve this performance.

Fig. 2a (left panel) illustrates the performance of it as a function of ρ and T. For clarity, $|\log(\epsilon)|$ is harnessed as a proxy for test accuracy, with higher values indicating better performance, where ϵ is the classification error. As can be seen, the test accuracy achieved is highly dependent on the choice of ρ and T. When $\rho < 1$, the reservoir exhibits contraction dynamics, where the states decay with increasing T and converge to a fixed point, progressively erasing information and collapsing performance. When $\rho > 1$, the system enters the expanding regime in which nonlinearity and sensitivity to initial conditions intensify, producing stretching dynamics in state space that enhance feature separability and yield superior performance. This expansion-driven improvement in representation has been observed across various deep learning models [47–54]. Although one of the optimal regions is in the expanding regime, taking too large a value for T drives the system into excessive chaotic dynamics, nearly completely losing its memory of the initial input, thus making predictions unstable or even meaningless. The available horizon is theoretically bounded by the Lyapunov time, which refers to the characteristic timescale over which an infinitesimal perturbation grows exponentially, as shown by the white curve in the panel. The above SS-DFA results accord with the bifurcation analysis and are consistent with the overall trends revealed by training with BP and linear regression (see Supplementary Information), thereby validating the effectiveness of the proposed training strategy. To reduce the dependence of training on precise system information, we investigate algorithmic simplifications. Fig. 2a (right panel) shows results for the simplified SS-DFA obtained by omitting the scaling factor during the backward process (i.e., sets ρ of W in Equation 3 to 1). The simplified method remains effective; moreover, its effective operating region becomes even broader because error signals are no longer excessively attenuated or amplified by contractive or expanding dynamics. Furthermore, to assess the sensitivity of the training to precise connectivity, we randomly select a controlled proportion of elements in $W^\top$ (Equation 3) and replace them with random values while fixing $\rho = 1$. Performance as a function of the replacement ratio and T is shown in Fig. 2b. The results indicate that introducing random substitutions degrades performance, with the effect intensifying as T increases, highlighting the critical role of precise connectivity. In summary, in the training stage of a fully-coupled OASP-RC with SS-DFA, the scaling of W by ρ can be omitted, whereas precise connectivity remains indispensable.

We then evaluate the proposed single-map OASP-RC, which removes internal connectivity and thus eliminates reliance on precise topological information during training. Its structural schematic and bifurcation diagram are shown in Fig. 2c. The bifurcation analysis indicates that within the range of ρ shown in the figure: for $\rho < 1$, states of the system converge to the fixed point zero; for $1 < \rho \lesssim 2.2$, states converge to two symmetric fixed points ($\pm$); for $\rho \gtrsim 2.3$, the system starts to show period-doubling bifurcation toward chaos; and as ρ further increases to ~ 4.8, the system returns to the bistable (dual-fixed-point) regime [55]. Fig. 2d presents the results of a single-map framework trained with updated SS-DFA defined in Equation 4 on the MNIST dataset. When the system resides in the single fixed-point region, its performance trend aligns with that of the fully-coupled framework; upon entering the dual-fixed-point region, information is compressed, yet the dynamics preserve the sign of initial condition by mapping it to distinct basins, thereby enabling the framework

to retain discriminative capability; and when $\rho \gtrsim 2.3$, the framework remains effective because excessively chaotic dynamics have not yet been reached. A finer resolution and a broader sweep in ρ, providing a more detailed characterization, are summarized in the Supplementary Information. Particularly noteworthy is that optimal regions emerge at the bifurcation point ($\rho \sim 1$) and for small T with $\rho > 1$; the latter stems from the transient expanding/chaotic dynamics of the system. Since bifurcation reflects the system's global property on a long-term scale, this transient characteristic is not directly visible from bifurcation analysis. Instead, this local instability on a finite timescale requires quantitative interpretation using the finite-time maximum Lyapunov exponents (FTMLE). Detailed derivations and FTMLE analysis are provided in the Supplementary Information. Fig. 2e shows and compares the convergence speed of a fully-coupled and the proposed single-map frameworks under SS-DFA, measured by the number of epochs required for the training error to drop to 10^{-2}. Similar to performance trends, the transient expanding/chaotic regime exhibits the fastest convergence for both frameworks. Compared to the fully-coupled framework, the proposed single-map framework streamlines training, offers a wider operating range, converges faster overall, and shows better performance. In particular, it achieves the highest test accuracy of 98.33% at $\rho = 2.7, T = 1$, whereas the fully-coupled framework achieves 98.26% at $\rho = 2.8, T = 1$. Furthermore, we could successfully extend the proposed approach to the gradient-free variant, SS-aDFA, thereby completely eliminating reliance on precise system information (see Supplementary Information).

Fig. 2f illustrates the optimal performance of both frameworks across different values of ρ, together with the required T to achieve that performance. As ρ increases, performance steadily improves and converges toward the optimal operating range. Once in this high-performance region, the required T decreases with further increases in expansion. Given that state iterations incur additional computational cost, strategically increasing ρ leverages the advantages of transient expanding/chaotic dynamics to improve performance while reducing the computational burden, thus offering guidance for the practical application of our approach.

2.3 Scalability and resource efficiency

Here, we assess the scalability (in network width and depth) and resource efficiency of the single-map OASP-RC. Fig. 3a shows the performance of the aforementioned basic architecture trained with SS-DFA as network size varies under various (ρ, T) settings. As a comparison, a size-matched three-layer baseline MLP trained with BP is employed. Label smoothing [56] is applied on both tasks, and for the CIFAR-10 dataset [57], mild data augmentation [58] is additionally incorporated to mitigate overfitting. Figs.3b and 3c compare corresponding computational cost and trainable parameter counts, including fully-coupled OASP-RC for comparison. Following Refs. [59, 60], computational cost is quantified by the floating point operations (FLOPs) per epoch of training, primarily driven by MVMs. Results demonstrate that, across most settings, our framework outperforms the baseline MLP and achieves superior resource efficiency and scalability. Specifically, on MNIST and CIFAR-10, even at the largest T setting, the maximum reductions in FLOPs and trainable parameter counts reach around 90% and 75%, respectively.

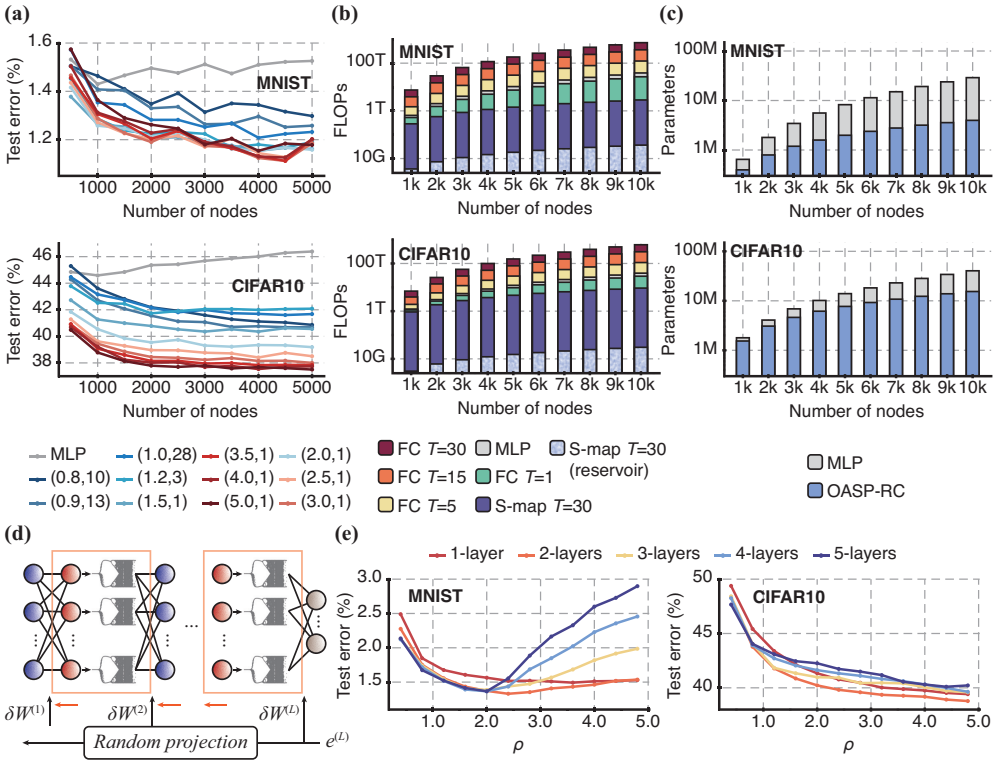


Fig. 3 (a) Test error of the single-map framework as a function of the number of reservoir nodes across various (ρ, T) combinations. Gray lines show results of the baseline fully connected network—i.e., a multilayer perceptron (MLP) with identical architecture. (b) Comparison of computational cost—floating point operations (FLOPs) per epoch of training—between the single-map and the fully-coupled frameworks, reported relative to the baseline MLP across various state iterations T. FC, S-map denote the fully-coupled and single-map frameworks, respectively. It should be noted that the single-map framework primarily operates as scalar computations during state iteration, incurring minimal external computational cost; accordingly, FLOPs at $T = 30$ are nearly the same as for smaller T. For a conservative assessment, we report the FLOPs of the single-map framework based on the maximum T for comparison. "(reservoir)" denotes the FLOPs contributed solely by the reservoir layer, which account for only a small fraction of the single-map framework's total FLOPs. Specifically, for MNIST, this corresponds to 1.26% of the total FLOPs, and for CIFAR-10, 0.33%. (c) Comparison of trainable parameter counts between the baseline MLP and the employed OASP-RC. (d) Schematic of the deep architectures with embedded single-map modules trained with SS-DFA. (e) Test error of deep OASP-RCs with varying depths as a function of ρ; the legend indicates the layer counts of the embedded single-map module.

Fig. 3d presents the schematic of a deep network embedding the single-map module. Fig. 3e shows its performance as a function of ρ across different depth configurations, with $T = 1$. We observe that deep networks with the single-map modules operate effectively; as ρ increases, performance improves (on MNIST, the decline in deeper networks is attributable to overfitting), with shallower networks benefiting more significantly. This suggesting that moderately enhancing the system's bifurcation parameter

ρ not only reduces the required T but also allows shallower networks to match or even surpass deeper counterparts, thereby further decreasing overall computational cost.

2.4 Physical implementation

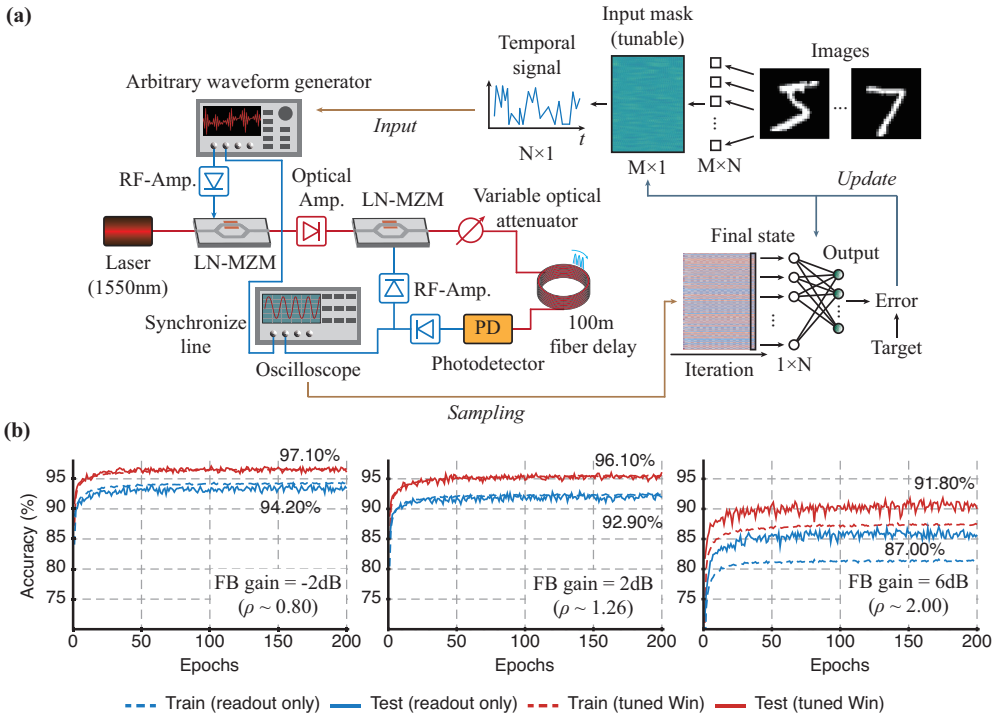


Fig. 4 (a) Schematic of the single-map framework implemented on an optoelectronic RC platform. The input is masked by a random matrix in a digital processor and then converted to the optical domain by an optical intensity modulator based on $LiNbO_3$ Mach–Zehnder modulator (LN-MZM). The optical signal is injected into a 200-meter-long, fiber delay with a single optoelectronic nonlinearity and operates in a time-multiplexed manner, forming the reservoir layer. A variable optical attenuator is employed to adjust the feedback gain α, thereby controlling ρ in the single-map framework and modulating the system's transient properties. The state $s(t)$ is converted to a digital signal by a photodetector (PD), amplified by a radio-frequency amplifier (RF-Amp.), and measured with an oscilloscope. The state at the final step is fed into the digital readout layer to produce a prediction. The mask matrix and readout are updated via SS-DFA. (b) Training and test accuracy as functions of epochs at different α on the MNIST dataset; readout-only training results are included as comparisons. The best test accuracy is indicated respectively.

As a demonstration of our proposed framework, we implement the aforementioned basic single-map framework trained with SS-DFA on a delay-based optoelectronic reservoir computer. Fig. 4a illustrates the experimental platform. In this setup, input images are masked by $W^{\text{in}} \in \mathbb{R}^{N\times N'}$ in a digital processor (where N' is the input size) and then converted to optical signals by an optical intensity modulator with

time interval θ. Thus, the input signals for a single image are elongated to duration $T_{RC} = N\theta$. The signals are subsequently injected into a 200-meter-long spiral-based delay ring system with a single optoelectronic nonlinearity, functioning as the reservoir layer. A variable optical attenuator adjusts the feedback gain α of the system, thereby controlling ρ in the single-map framework and modulating the system's transient properties. As the nonlinear element, we employ a Mach–Zehnder modulator (MZM), providing the activation $f(x) = \sin(x + \varphi_{\text{bias}})$, where φ_{bias} is the MZM bias voltage. The resulting system state can then be expressed as:

$$s(t) = \sin\{\alpha s(t - \tau) + \varphi_{\text{bias}}\} \tag{5}$$

where φ_{bias} is set to 0. A typical delay line RC approach [61, 62] is adopted, where the length of the delay ring τ is set such that $\tau \sim T_{RC}$ to couple with previous state of itself, thus implementing single-map functionality.

Models		**Training strategy**	**MNIST**	**CIFAR-10**	**Sim./Exp.**
Autonomous state-propagating RC	Analog circuit [32]	Linear regression	95.42%	-	Sim.
	Spin-torque oscillators [34]	Gradient-based optimization	98.49%	53.49%	
	Feedforward ESN [29]	BPTT	98.23%	-	
ONNs	Continuous-time system [63]	DFA-adjoint	97.30%	46.60%	Sim.
			96.40%	-	Exp.
	Deep reservoir computing [37]	aDFA	98.96%	55.80%	Sim.
			97.80%	47.83%	Exp.
	Pre-sensor ONN [64]	Forward-Forward	94.00%	-	Sim.
			93.00%	-	Exp.
	Quasi-convolution deep RC [65]	Ridge regression	98.93%	-	Sim.
			96.28%	-	Exp.
	Optical multimode fiber [9]	PhyLL	96.36%	-	Exp.
Baseline	MLP	BP	98.16%	48.39%	Sim.
Ours	**Basic architecture**	**SS-DFA**	**98.33%**	**54.85%**	**Sim.**
			97.10%	**-**	**Exp.**
	Scaled architecture* (hidden nodes)		**98.89% (4500)**	**62.50% (5000)**	**Sim.**

Table 1 Comparison of various models on MNIST and CIFAR-10 (simulation and experiment results). The scaled architecture uses label smoothing on both MNIST and CIFAR-10; for CIFAR-10 it further employs mild data augmentation.

We first conduct readout-only experiments to verify consistency between the physical implementation and numerical simulations of the proposed single-map framework, the results exhibit consistent trends (see Supplementary Information). We then perform end-to-end training with $T = 1$ across different α configurations. To reduce experimental duration and operational overhead, we adopt a two-stage strategy: (i) update W^{in} and W^{out} for 5 epochs under real-time optimization conditions to rapidly improve performance; and (ii) train only W^{out} for a further 195 epochs to achieve robust convergence. The results are shown in Fig. 4b. It can be seen that the proposed framework works effectively on the physical platform; compared to the readout-only

reference, performance improves substantially when W^{in} is trained, further validating the success of the SS-DFA training strategy. Notably, in our experiments, $\alpha = -2$dB (corresponding to $\rho \simeq 0.8$)—in other words, around the bifurcation point corresponding to the transition from single fixed point to bistable states—yields the best performance. The transient expanding regime performs well yet remains suboptimal; we attribute this to amplification of hardware noise when the system operates in the expanding regime. Table 1 presents our simulation and experiment results, and compares them to existing research. These results show that our framework not only aligns with numerical simulations in its physical implementation but also exhibits competitive performance—in both simulation and experiment—within the ASP-RC model and ONNs domain.

2.5 Embedding into modern architecture and challenging scaling laws

To further evaluate the applicability and efficiency of the proposed single-map module, we integrate it into a modern deep architecture, ResMLP, a great baseline with minimal induction bias. As shown in Fig. 5a, in the no-normalization baseline model, the nonlinearity and linear readout within each block are replaced by the single-map module. Given that computational cost is driven primarily by architecture rather than training strategies, we train on CIFAR-10 with a 1024-node hidden dimension using BP for simplicity and to emphasize the module's inherent advantages. To fully unlock the models' potential, the LION optimizer [66], label smoothing, and strong data augmentation (including Mixup [67]) are employed. Fig. 5b and 5d show the training curves, test accuracies, computational cost, and trainable parameter counts for six-block baseline and proposed frameworks at $T = 1$ over various ρ, alongside a lightweight baseline variant that omits the linear readout to mirror the proposed block design. The results demonstrate that ResMLP with embedded single-map modules works effectively: upon entering the transient expanding regime, it achieves faster convergence and superior performance with nearly 40% reduction of the computational cost and trainable parameters of the baseline.

Fig. 5c summarizes the scaling laws of the baseline and proposed frameworks ($\rho = 1.2$, $T = 1$). In deep learning, scaling laws relate performance to compute via a power law, typically expressed as $E(C) = aC^{-P}$ [68, 69], where E is test error and C denotes computational cost; a larger exponent P indicates better scaling behavior, showing lower computational consumption, while a smaller coefficient a reflects improved offset. In our experiments, we systematically scale frameworks by varying the hidden-layer width (512, 1024, 2048) and the number of blocks (2, 4, 6). The minimum test error is recorded every 10 epochs to track how performance improves as compute increases. Under power-law fitting, the baseline fits as $E(C) = (1/6.50 \times 10^{-38})C^{-3.45}$, while our framework follows $E(C) = (1/2.20 \times 10^{-37})C^{-4.30}$. This breaks the scaling laws exhibited by the baseline—at equal resources, our framework exceeds conventional expectations—thus establishing a new performance–compute relationship, and offering a new perspective for developing efficient and scalable computing paradigms.

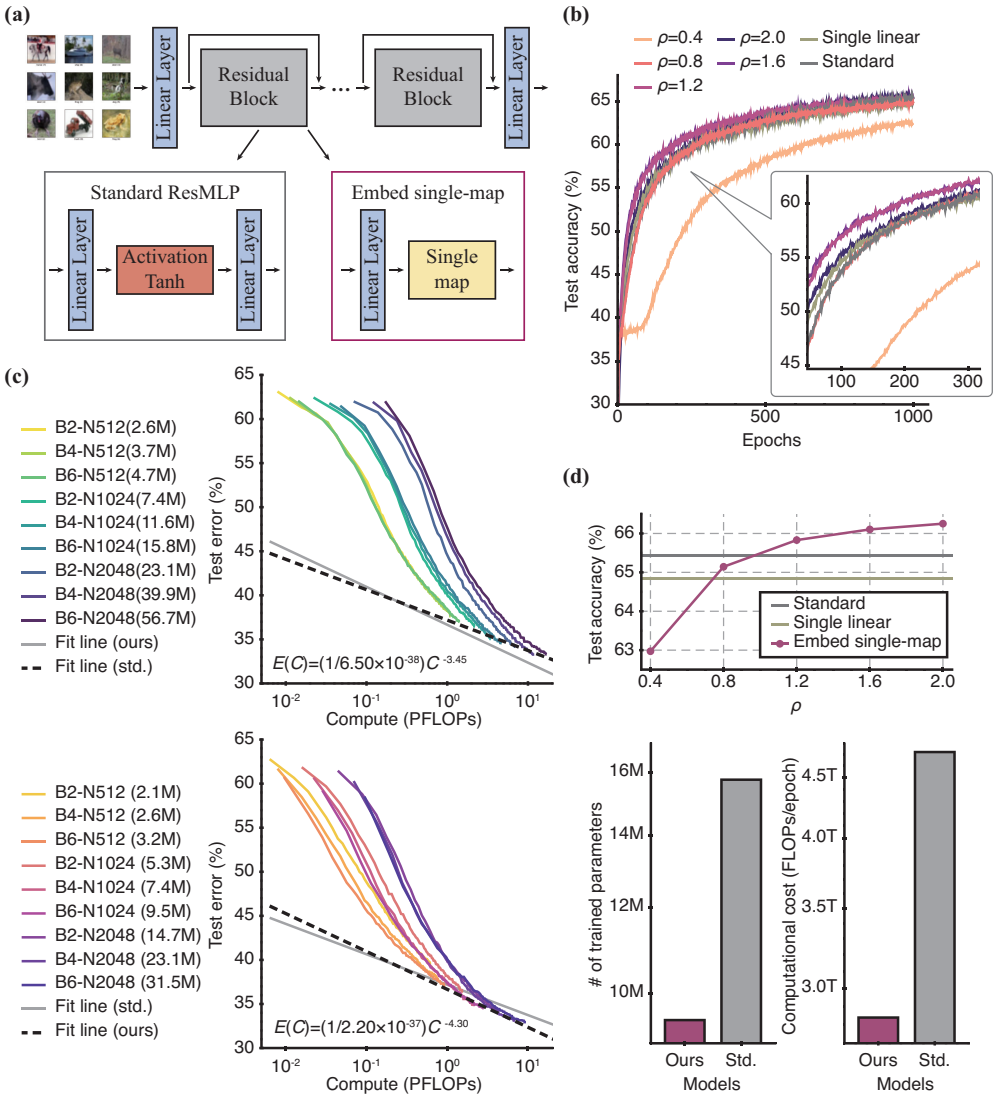


Fig. 5 (a) Schematics of the baseline ResMLP and the proposed framework incorporating single-map modules, in which the original nonlinearity and the linear readout within each block are replaced by a single-map module. (b) Test accuracy on CIFAR-10 dataset of the proposed framework as a function of training time at various transient expanding levels (ρ). The baseline framework serves as the reference, alongside the lightweight variant (single linear) that retains one linear layer and a nonlinearity within each block. (c) Scaling laws and corresponding power-law fits for the baseline and proposed frameworks. Bx–Nk denotes a model with x blocks and a hidden layer dimension of k; the number of trainable parameters is given in parentheses. (d) Comparison of test accuracy, computational cost, and number of trainable parameters.

3 Discussion

In this study, we introduce a novel unconventional computing framework based on ASP-RC, which achieves strong expressiveness and high training efficiency, and validate it on an optoelectronic platform. Training is simplified via SS-DFA, removing reliance on precise system information and on access to intermediate states, which ensures practical implementability. Benefiting from the single-map architecture, requiring no training, and featuring no internal connections, the proposed framework significantly reduces computational cost. Interestingly, the transient expanding property can be leveraged to enhance performance and accelerate convergence. When integrated into a modern deep architecture, it challenges conventional scaling laws, demonstrating outstanding scalability and adaptability.

Beyond the scope of this study, the autonomous state-propagating paradigm deserves further exploration on two fronts: embedding it within more complex modern architectures and implementing and evaluating it across diverse physical platforms, thereby offering a general route to more efficient computation. Current limitations should be acknowledged: Systematic investigation across larger-scale, multimodal models, and more complex tasks remains lacking. Fortunately, the block-level, pluggable structural characteristic demonstrated here provides a potential pathway for transitioning to such complex scenarios. Regarding physical implementation, while the framework exhibits robustness to noise (see Supplementary Information), fully exploiting the benefits of transient expanding property in hardware may require further analysis and calibration, as hardware-level noise in the expanding regime potentially differs from conventional assumptions. Although we have validated the approach on an optoelectronic platform, cross-platform generalization—including analog circuits, spintronic systems, or compute-in-memory devices—remains an avenue for future investigation.

Methods

Lyapunov exponent

The Lyapunov exponent is an important tool for measuring the sensitivity of a dynamical system to initial conditions. It characterizes the average rate of divergence of two initially nearly identical trajectories as they evolve over time. If small perturbations grow at an exponential rate, the system exhibits chaotic behavior [70]. For a one-dimensional continuous system, the system has only one direction of evolution, so the Lyapunov exponent itself represents the maximal value. This maximum Lyapunov exponent (MLE) is derived as:

$$\lambda = \lim_{t\to\infty} \lim_{|\delta_0|\to 0} \frac{1}{t} \ln \frac{|\delta(t)|}{|\delta_0|} \tag{6}$$

where δ_0 is the initial perturbation and $\delta(t)$ denotes the evolution of the perturbation after time t. The condition $|\delta_0| \to 0$ guarantees the validity of the linear approximation at any time. For discrete-time systems (e.g., maps or recursive model updating as in

our model) defined by $x_{n+1} = f(x_n)$ for an orbit starting with x_0, this definition can be translated as:

$$\lambda(x_0) = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x_i)| \tag{7}$$

where $f'(x_i)$ denotes the derivative of the system at x_i. Lyapunov time, T_L, is a timescale used to describe the sensitivity of a dynamical system to initial conditions, reflecting the effective time window for prediction in chaotic systems. It represents the amount of time after which the system completely loses memory of its initial perturbations, rendering future states unpredictable (Here we use a one-dimensional system as an example; high-dimensional systems will be demonstrated later). T_L is typically defined as the reciprocal of the MLE, shown as:

$$T_L = \frac{1}{\lambda} \tag{8}$$

However, the MLE and T_L define the global dynamics of the system based on an infinite timescale and may mask transient instabilities as well as local, finite-time expanding and chaotic behavior. Therefore, for our model, it is more appropriate to consider a local-time version of the MLE, known as the finite time maximum Lyapunov exponent (FTMLE). It is defined as follows:

$$\lambda_T(x_0) = \frac{1}{T} \sum_{i=0}^{T-1} \ln |f'(x_i)| \tag{9}$$

where T denotes the evolution time of the system. Extending to high-dimensional systems, the propagation of perturbations depends not only on the stretching strength at the current position but also closely on the direction of the perturbation. Therefore, the Jacobian matrix needs to be introduced to characterize the linear approximate transformation at each step. Consider introducing a unit perturbation vector v_0 at the initial point x_0 of the system. Its propagation during the T steps evolution process can be expressed as the following chain product:

$$\delta x_T = J(x_{T-1}) \cdots J(x_0) v_0 \tag{10}$$

where $J(x_i) = \partial f(x_i)/\partial x$ is the Jacobian matrix of the system at step i. In this high-dimensional system, the FTMLE is defined as the growth rate of perturbations in the most unstable direction, represented by the maximum singular value of the chain product matrix as:

$$\lambda_T^{max}(x_0) = \frac{1}{T} \ln \sigma_1(J(x_{T-1}) \cdots J(x_0)) \tag{11}$$

where $\sigma_1(\cdot)$ denotes the maximum singular value. In actual calculations, we use singular value decomposition (SVD) to compute it. On a finite timescale, an FTMLE greater than 0 indicates that the orbit is sensitive to initial conditions in the short term, demonstrating expanding and chaotic behavior or transient instability. When the FTMLE is less than 0, the orbit tends toward an attractor or stable periodic orbit, indicating that the system has good local stability and is controllable.

Computational cost and floating point operations (FLOPs)

Computational cost typically refers to the amount of various resources required during training or inference (such as time, computing power, etc.). In deep learning, the computational cost of a model is roughly measured using floating point operations (FLOPs) or multiply–accumulate operations (MACs). FLOPs refer to the total number of floating-point addition, subtraction, multiplication, and division operations performed during a single forward propagation of a model. It is a hardware-independent theoretical metric that can be used to compare the computational complexity and efficiency of different model architectures. When calculating FLOPs, the number of operations in each layer is usually accumulated layer by layer. In this work, we consider smaller models; so, in addition to matrix operations, we also take into account the operations of the activation function. The FLOPs of the fully connected linear layer with the activation function are calculated as follows:

$$C_{\mathrm{FC}} = C_{\mathrm{MVM}} + C_{\mathrm{act}} \tag{12}$$
$$= 2 \times N_{\mathrm{in}} \times N_{\mathrm{out}} + N_{\mathrm{out}} \tag{13}$$

where C_{MVM} denotes the FLOPs for MVMs, C_{act} denotes the FLOPs for the activation function operations, and N_{in} and N_{out} represent the input and output dimensions of this layer, respectively. The coefficient of 2 indicates that during MVM operations, each output neuron performs one multiplication on N_{in} inputs and then accumulates the results. For our model, we considered two topologies of the reservoir layer. The first is fully-coupled, whose FLOPs can be calculated as follows:

$$C_{\text{fully-coupled}} = T \times (C_{\mathrm{MVM}} + C_{\mathrm{act}}) \tag{14}$$
$$= T \times (2 \times N^2 + N) \tag{15}$$

where T denotes the iterations of system state evolution, N represents the number of nodes in this reservoir layer. The second is the single-map framework, and its FLOPs are calculated as:

$$C_{\text{single-map}} = T \times (C_{\mathrm{MS}} + C_{\mathrm{act}}) \tag{16}$$
$$= 2 \times T \times N \tag{17}$$

where $C_{\mathrm{MS}} = N$ denotes the matrix scalar multiplication derived from $s(t) = f(\rho s(t - \tau))$. It should be noted that in the single-map framework, MVMs are not included. These operations are essentially implemented through memory reordering or delay-line buffering—that is, data transfer rather than floating-point computation—and do not involve multiplication or addition; thus they are not counted as FLOPs.

Next, consider the total computational cost required for these layers in one epoch of training. Following [59, 60, 69], the total computational cost C_{total} incurred from

training the fully connected linear layer on E examples for one epoch with BP is as follows:

$$C_{\text{total_FC}} = 3 \times C_{\text{FC}} \times E \tag{18}$$

where the coefficient of 3 accounts for the two main stages of the training process, forward propagation and backward propagation of the error. The FLOPs of backward propagation are approximately twice those of the forward pass, since gradients must be computed for both the layer activations (to propagate the error signal upstream) and the weights (to perform parameter updates). Meanwhile, in the ASP-RC, since the reservoir layer parameters remain fixed during training, the computational cost incurred by parameter updates can be ignored. Thus, the FLOPs of the ASP-RC trained with BP-based methods can be expressed as:

$$C_{\text{total_RC}} = 2 \times C_{\text{forward}} \times E \tag{19}$$

where C_{forward} denotes the FLOPs incurred during the forward pass.

At the same time, our SS-DFA-trained ASP-RC avoids the accumulated derivative calculation in error backpropagation by eliminating the access to intermediate states (as shown in Equation 3), thereby further simplifying the computational cost. For fully-coupled topology, the C_{total} can be simplified to:

$$C_{\text{total_fully-coupled}} = (C_{\text{fully-coupled}} + T \times 2 \times N^2 + N) \times E \tag{20}$$

For the single-map framework, the C_{total} is further updated to:

$$C_{\text{total_single-map}} = (C_{\text{single-map}} + N) \times E \tag{21}$$

where the accumulation of the scaling factor is eliminated further; therefore, the matrix scalar multiplications are not even required and only the computational cost of $f'(\cdot)$ operation needs to be considered. From the above equations, it can be seen that the single-map version trained with SS-DFA can further markedly reduce the computational cost by completely decoupling the backward pass from the temporal evolution process. In addition, in our framework, T plays an important role in FLOPs. When adopting the large ρ and small T strategy mentioned in Section 2.2, the computational cost of our framework can be much lower than that of conventional fully connected networks.

Optoelectronic benchtop

In the setup of Fig. 4a, datasets on a host computer are streamed over a USB link to an arbitrary waveform generator (AWG; Keysight M8195A; 18 GHz bandwidth; 60 GSa/s). The matrix operation of images and input mask are executed on the standard electronic computer. The resulting electrical waveform drives a first $LiNbO_3$ Mach–Zehnder modulator (LN-MZM, Thorlabs LN05FC; 32 GHz), converting it to optical intensity that propagates through a single mode fiber-based delay line and is detected by a photodetector (Finisar XPRV2022; 33 GHz). The detected signal is amplified by a radio frequency (RF) amplifier (SHF-S807C; 50 GHz). Internal dynamics are tapped via a 50:50 optical splitter and routed to the photodetector (PD)/RF

chain. Then, the RF signals are again converted to optical signals by second LN-MZM, making the nonlinear recurrent connection within the optical circuit. Finally, the electrical output is digitized by a real-time oscilloscope (Teledyne, Labmaster 10, 20GHz; 80GSa/s). The feedback gain α can be controlled by changing the variable optical attenuator value. We set the bias voltage of the first LN-MZM to zero, which sets the optical intensity within the fiber ring to zero in the absence of input signals. This enables the initial state to be fixed within the optical reservoir, even in the chaotic region with high feedback gain. We oversample the input/output signals to reduce the signal distortion and noise. Therefore, the effective clock rate of the computation is 30GSa/s (half of the sampling rate of the AWG). All the above-described processes are implemented through a PyTorch-compatible software interface.

Numerical simulation

The benchmark datasets used in this study are publicly available: MNIST (http://yann.lecun.com/exdb/mnist/); CIFAR-10 (https://www.cs.toronto.edu/~kriz/cifar.html). The numerical experiments were performed on a dual-socket AMD EPYC 7702 server (2 × 64-core, 128 logical CPUs; base 2.0 GHz, boost enabled; 2 NUMA nodes; 512 MB total L3 cache) equipped with two NVIDIA A100 (40 GB) GPUs. Most experiments were implemented using our own PyTorch/Python codebase.

Data availability

The benchmark datasets used in this study are publicly available via PyTorch or TensorFlow. All data and methods necessary to reproduce and evaluate our results are provided in the main text and the Supplementary Information.

Code availability

Code that is used in this paper is available from the corresponding author on reasonable request.

Acknowledgements

K.N. was supported by JST CREST Grant No. JPMJCR2014 and by a project, JPNP14004, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). K. I. was supported by JSPS KAKENHI Grant Number JP22K21295.

Author contributions

Y.Z., M.N., and K.N. conceived the basic concept of the presented physical deep learning method. Y.Z. performed the numerical simulations with supports from K.I. M.N. constructed the optoelectronic benchtop and executed the optical experiment. T.H., Y.K., and K.N. supervised the project. Y.Z. wrote the initial draft of the manuscript. All the authors discussed the results and contributed to writing the manuscript.

References

[1] Strubell, E., Ganesh, A. & McCallum, A. Energy and policy considerations for modern deep learning research. *Proceedings of the AAAI conference on artificial intelligence* **34**, 13693–13696 (2020).

[2] Thompson, N. C., Greenewald, K., Lee, K., Manso, G. F. *et al.* The computational limits of deep learning. *arXiv preprint arXiv:2007.05558* **10** (2020).

[3] Chen, S. How much energy will ai really consume? the good, the bad and the unknown. *Nature* **639**, 22–24 (2025).

[4] Marković, D., Mizrahi, A., Querlioz, D. & Grollier, J. Physics for neuromorphic computing. *Nature Reviews Physics* **2**, 499–510 (2020).

[5] Kudithipudi, D. *et al.* Neuromorphic computing at scale. *Nature* **637**, 801–812 (2025).

[6] Muir, D. R. & Sheik, S. The road to commercial success for neuromorphic technologies. *Nature Communications* **16**, 3586 (2025).

[7] Wright, L. G. *et al.* Deep physical neural networks trained with backpropagation. *Nature* **601**, 549–555 (2022).

[8] Momeni, A. *et al.* Training of physical neural networks. *Nature* **645**, 53–61 (2025).

[9] Momeni, A., Rahmani, B., Malléjac, M., Del Hougne, P. & Fleury, R. Backpropagation-free training of deep physical neural networks. *Science* **382**, 1297–1303 (2023).

[10] Grollier, J. *et al.* Neuromorphic spintronics. *Nature electronics* **3**, 360–370 (2020).

[11] Marrows, C. H., Barker, J., Moore, T. A. & Moorsom, T. Neuromorphic computing with spintronics. *npj Spintronics* **2**, 12 (2024).

[12] Roy, K. *et al.* Spintronic neural systems. *Nature Reviews Electrical Engineering* 1–16 (2024).

[13] Nakajima, K., Hauser, H., Li, T. & Pfeifer, R. Information processing via physical soft body. *Scientific reports* **5**, 10487 (2015).

[14] Garrad, M., Soter, G., Conn, A., Hauser, H. & Rossiter, J. A soft matter computer for soft robots. *Science Robotics* **4**, eaaw6060 (2019).

[15] Tomioka, H., Inoue, K., Kuniyoshi, Y. & Nakajima, K. Backpropagation through soft body: Investigating information processing in brain-body coupling systems. *arXiv preprint arXiv:2503.05601* (2025).

[16] Li, S. & Mao, X. Training all-mechanical neural networks for task learning through in situ backpropagation. *Nature Communications* **15**, 1–12 (2024).

[17] Zhang, W., Mazzarello, R., Wuttig, M. & Ma, E. Designing crystallization in phase-change materials for universal memory and neuro-inspired computing. *Nature Reviews Materials* **4**, 150–168 (2019).

[18] Baltussen, M. G., de Jong, T. J., Duez, Q., Robinson, W. E. & Huck, W. T. Chemical reservoir computation in a self-organizing reaction network. *Nature* **631**, 549–555 (2024).

[19] Shastri, B. J. *et al.* Photonics for artificial intelligence and neuromorphic computing. *Nature Photonics* **15**, 102–114 (2021).

[20] Farmakidis, N., Dong, B. & Bhaskaran, H. Integrated photonic neuromorphic computing: opportunities and challenges. *Nature Reviews Electrical Engineering* **1**, 358–373 (2024).

[21] Fu, T. *et al.* Optical neural networks: progress and challenges. *Light: Science & Applications* **13**, 263 (2024).

[22] Yang, Q., Bai, B., Hu, W. & Wang, X. Strategies for training optical neural networks. *National Science Open* **1**, 20220041 (2022).

[23] Wang, D., Nie, Y., Hu, G., Tsang, H. K. & Huang, C. Ultrafast silicon photonic reservoir computing engine delivering over 200 tops. *Nature Communications* **15**, 10841 (2024).

[24] Tanaka, G. *et al.* Recent advances in physical reservoir computing: A review. *Neural Networks* **115**, 100–123 (2019).

[25] Nakajima, K. & Fischer, I. *Reservoir computing* (Springer, 2021).

[26] Nakajima, K. Physical reservoir computing—an introductory perspective. *Japanese Journal of Applied Physics* **59**, 060501 (2020).

[27] Jaeger, H. & Haas, H. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science* **304**, 78–80 (2004).

[28] Maass, W., Natschläger, T. & Markram, H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation* **14**, 2531–2560 (2002).

[29] Liu, S., Akashi, N., Huang, Q., Kuniyoshi, Y. & Nakajima, K. Exploiting chaotic dynamics as deep neural networks. *Physical Review Research* **7**, 033031 (2025).

[30] Cernansky, M. & Makula, M. Feed-forward echo state networks. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* **3**, 1479–1482

(2005).

[31] Negoro, M., Mitarai, K., Fujii, K., Nakajima, K. & Kitagawa, M. Machine learning with controllable quantum dynamics of a nuclear spin ensemble in a solid. *arXiv preprint arXiv:1806.10910* (2018).

[32] Kesgin, B. U. & Teğin, U. Implementing the analogous neural network using chaotic strange attractors. *Communications Engineering* **3**, 99 (2024).

[33] Kesgin, B. U. & Teğin, U. Photonic neural networks at the edge of spatiotemporal chaos in multimode fibers. *Nanophotonics* **X** (2025).

[34] Imai, Y., Liu, S., Akashi, N. & Nakajima, K. Gradient-based optimization of spintronic devices. *Applied Physics Letters* **126** (2025).

[35] Nøkland, A. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems* **29** (2016).

[36] Launay, J., Poli, I., Boniface, F. & Krzakala, F. Direct feedback alignment scales to modern deep learning tasks and architectures. *Advances in neural information processing systems* **33**, 9346–9360 (2020).

[37] Nakajima, M. *et al.* Physical deep learning with biologically inspired training method: gradient-free approach for physical hardware. *Nature communications* **13**, 7847 (2022).

[38] Zhang, Y. *et al.* Training spiking neural networks via augmented direct feedback alignment. *arXiv preprint arXiv:2409.07776* (2024).

[39] Nakajima, M. *et al.* Reservoir direct feedback alignment: deep learning by physical dynamics. *Communications Physics* **7**, 411 (2024).

[40] Ma, X., Qin, C., You, H., Ran, H. & Fu, Y. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123* (2022).

[41] Touvron, H. *et al.* Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE transactions on pattern analysis and machine intelligence* **45**, 5314–5321 (2022).

[42] Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* **78**, 1550–1560 (1990).

[43] Liao, Q., Leibo, J. & Poggio, T. How important is weight symmetry in backpropagation? *Proceedings of the AAAI Conference on Artificial Intelligence* **30** (2016).

[44] LeCun, Y., Cortes, C. & Burges, C.-J. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* **2** (2010).

[45] Robbins, H. & Monro, S. A stochastic approximation method. *The annals of mathematical statistics* 400–407 (1951).

[46] Duchi, J., Hazan, E. & Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* **12** (2011).

[47] Ishihara, S. & Kaneko, K. Magic number 7±2 in networks of threshold dynamics. *Physical review letters* **94**, 058102 (2005).

[48] Saxe, A. M., McClelland, J. L. & Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120* (2013).

[49] Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J. & Ganguli, S. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems* **29** (2016).

[50] Lin, H., Wang, C., Yao, W. & Tan, Y. Chaotic dynamics in a neural network with different types of external stimuli. *Communications in Nonlinear Science and Numerical Simulation* **90**, 105390 (2020).

[51] Keup, C., Kühn, T., Dahmen, D. & Helias, M. Transient chaotic dimensionality expansion by recurrent networks. *Physical Review X* **11**, 021064 (2021).

[52] Inoue, K., Ohara, S., Kuniyoshi, Y. & Nakajima, K. Transient chaos in bidirectional encoder representations from transformers. *Physical Review Research* **4**, 013204 (2022).

[53] Engelken, R., Wolf, F. & Abbott, L. F. Lyapunov spectra of chaotic recurrent neural networks. *Physical Review Research* **5**, 043044 (2023).

[54] Storm, L., Linander, H., Bec, J., Gustavsson, K. & Mehlig, B. Finite-time lyapunov exponents of deep neural networks. *Physical Review Letters* **132**, 057301 (2024).

[55] Lalescu, C. C. Patterns in the sine map bifurcation diagram. *arXiv preprint arXiv:1011.6552* (2010).

[56] Müller, R., Kornblith, S. & Hinton, G. E. When does label smoothing help? *Advances in neural information processing systems* **32** (2019).

[57] Krizhevsky, A., Hinton, G. *et al.* Learning multiple layers of features from tiny images (2009).

[58] Shorten, C. & Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data* **6**, 1–48 (2019).

[59] Bachmann, G., Anagnostidis, S. & Hofmann, T. Scaling mlps: A tale of inductive bias. *Advances in Neural Information Processing Systems* **36**, 60821–60840 (2023).

[60] Open AI. AI and compute (2018).

[61] Torrejon, J. *et al.* Neuromorphic computing with nanoscale spintronic oscillators. *Nature* **547**, 428–431 (2017).

[62] Toprasertpong, K. *et al.* Reservoir computing on a silicon platform with a ferroelectric field-effect transistor. *Communications Engineering* **1**, 21 (2022).

[63] Sunada, S. *et al.* Blending optimal control and biologically plausible learning for noise-robust physical neural networks. *Physical Review Letters* **134**, 017301 (2025).

[64] Huang, Z. *et al.* Brain-like training of a pre-sensor optical neural network with a backpropagation-free algorithm. *Photonics Research* **13**, 915–923 (2025).

[65] Zhou, C. *et al.* Deep photonic reservoir computing based on a distributed feedback laser array. *APL Photonics* **10** (2025).

[66] Chen, X. *et al.* Symbolic discovery of optimization algorithms. *Advances in neural information processing systems* **36**, 49205–49233 (2023).

[67] Zhang, H., Cisse, M., Dauphin, Y. N. & Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412* (2017).

[68] Kaplan, J. *et al.* Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361* (2020).

[69] Filipovich, M. J., Cappelli, A., Hesslow, D. & Launay, J. Scaling laws beyond backpropagation. *arXiv preprint arXiv:2210.14593* (2022).

[70] Vulpiani, A. *Chaos: from simple models to complex systems* Vol. 17 (World Scientific, 2010).

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- Supplementaryinformation.pdf