NEUROSCIENCE

A brain-inspired algorithm that mitigates catastrophic forgetting of artificial and spiking neural networks with low computational cost

Tielin Zhang[1,2,3]*†, Xiang Cheng[1,2]†, Shuncheng Jia[1,2]†, Chengyu T Li[3,4], Mu-ming Poo[3,4], Bo Xu[1,2]*

Neuromodulators in the brain act globally at many forms of synaptic plasticity, represented as metaplasticity, which is rarely considered by existing spiking (SNNs) and nonspiking artificial neural networks (ANNs). Here, we report an efficient brain-inspired computing algorithm for SNNs and ANNs, referred to here as neuromodulation-assisted credit assignment (NACA), which uses expectation signals to induce defined levels of neuromodulators to selective synapses, whereby the long-term synaptic potentiation and depression are modified in a nonlinear manner depending on the neuromodulator level. The NACA algorithm achieved high recognition accuracy with substantially reduced computational cost in learning spatial and temporal classification tasks. Notably, NACA was also verified as efficient for learning five different class continuous learning tasks with varying degrees of complexity, exhibiting a markedly mitigated catastrophic forgetting at low computational cost. Mapping synaptic weight changes showed that these benefits could be explained by the sparse and targeted synaptic modifications attributed to expectation-based global neuromodulation.

INTRODUCTION

Many synaptic plasticity rules found in natural neural networks have been used to determine the credit assignment (i.e., the extent of local synaptic weight modification) of neural networks to correct global output errors (*1*). Prior efforts to introduce biologically plausible plasticity rules into spiking (SNNs, see table S1 for more abbreviations) and nonspiking artificial neural networks (ANNs) have focused mainly on short-term plasticity (*2*), Hebbian learning (*3*), and spike timing–dependent plasticity (STDP) (*4*). While Hebbian learning emphasizes the correlation of neuronal activity in modifying synapses, STDP runs further by considering the temporal order of pre- and postsynaptic spiking (*5*). In both cases, synaptic plasticity rules depend only on local neuronal activity without properly reflecting global instructive signals. However, global modulation of synapses occurs during reward-associated learning via neuromodulators, such as dopamine (*6*), noradrenaline (*7*), serotonin (*8*), and acetylcholine (*9*); those act at multiple synapses and derive from dispersed axons of specific neuromodulatory neurons (Fig. 1A). These neuromodulators could modify the capacity and property of synaptic plasticity (Fig. 1B), known as metaplasticity (*10*, *11*), that endows neural circuits with dynamic learning capabilities. For example, dopamine markedly modifies the amplitude, polarity, and temporal specificity of long-term potentiation (LTP) and depression (LTD) in STDP (*11*–*13*), which varies among neuronal and synaptic types (*11*, *14*), resulting in potential modification of original temporal-specific LTD/LTP into other STDP forms those exhibit various combinations of LTD, LTP, and no modification at negative and positive time windows (Fig. 1C). Our previous work has discussed the role of a mesoscale plasticity rule named self-backpropagation (*15*) of STDP for the efficient classification in SNNs and ANNs. In this study, we run further by considering the possibility that the amplitude and polarity of LTP and LTD could be modified in SNNs and ANNs by the neuromodulator level in a nonlinear manner, analogous to some potential forms of STDP endowed by the global dopamine (*11*) and local calcium level (*16*, *17*). By assigning specific neuromodulator levels to subpopulations of synapses during the training process, we are able to achieve an efficient expectation-based credit assignment in both SNNs and ANNs.

Another area for improvement faced by ANNs is catastrophic forgetting in continuous learning. Previous algorithms designed to reduce catastrophic forgetting can be divided into two categories. The first is the replay-based approach, whereby earlier memories are stored elsewhere in multiple manners, such as the replay sample buffer (*18*), sample generation network (*19*), or higher-dimensional space (*20*) for future replay. The second is the selection-based approach (*21*), whereby different network parameters for threshold (*22*), attention (*23*), or context (*24*) are selected during continuous learning to reduce the overlap of sequential weight modifications that erase previously learned memories. The computational costs for replay-based algorithms are generally much higher than those for selection-based algorithms, but the performance in continuous learning is poorer for the latter. In the biological nervous system, neuromodulation may not only directly manipulate STDP but also influence the synaptic modifications by altering the neuronal excitability and spiking dynamics. In this study, we showed that furtherly introducing this formulation of neuromodulation which conveyed input signal–based expectations to excitability could markedly mitigate the problem of catastrophic forgetting during class continuous learning (class-CL). Together with the neuromodulation on synaptic modification, we devised a new algorithm termed "neuromodulation-assisted credit assignment" (NACA).

Biological neuromodulation mechanisms have enlightened several plasticity algorithms in neural network models. For

[1]Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. [2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China. [3]Shanghai Center for Brain Science and Brain-inspired Technology, Lingang Laboratory, Shanghai 200031, China. [4]Center for Excellence in Brain Science and Intelligence Technology, Institute of Neuroscience, Chinese Academy of Sciences, Shanghai 200031, China.
*Corresponding author. Email: tielin.zhang@ia.ac.cn (T.Z.); xubo@ia.ac.cn (B.X.)
†These authors contributed equally to this work.

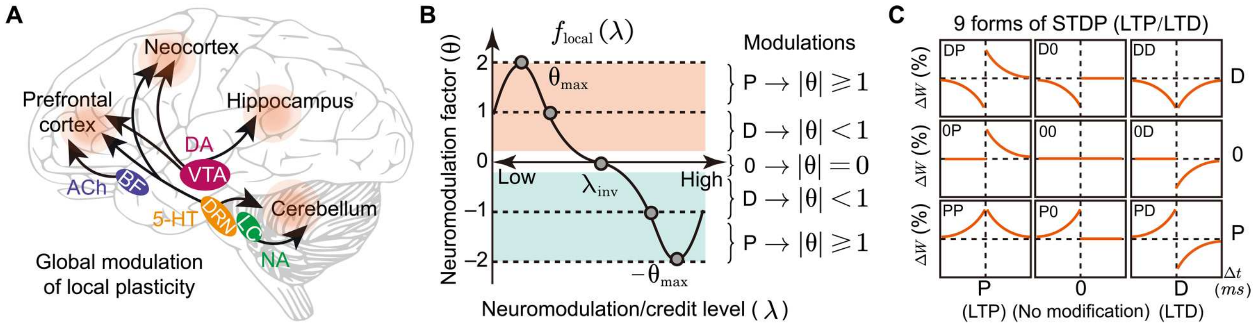

A
Neocortex
Prefrontal cortex
Hippocampus
DA
VTA
ACh
BF
5-HT
DRN
LC
Cerebellum
NA
Global modulation of local plasticity
B
$f_{\text{local}}(\lambda)$
$\theta_{\max}$
λ_{inv}
$-\theta_{\max}$
Low
High
Neuromodulation factor (θ)
Neuromodulation/credit level (λ)
Modulations
$P \rightarrow |\theta| \geqslant 1$
$D \rightarrow |\theta| < 1$
$0 \rightarrow |\theta| = 0$
$D \rightarrow |\theta| < 1$
$P \rightarrow |\theta| \geqslant 1$
C
9 forms of STDP (LTP/LTD)
DP
D0
DD
0P
00
0D
PP
P0
PD
ΔW (%)
Δt (ms)
P (LTP)
0 (No modification)
D (LTD)

Fig. 1. Global neuromodulation in the brain. (**A**) Schematic diagram depicting global neuromodulation of local plasticity via neuromodulators from dispersed axons of neuromodulatory neurons, such as dopamine (*6*) from ventral tegmental area (VTA), noradrenaline (*7*) from locus coeruleus (LC), serotonin (*8*) from dorsal raphe nucleus (DRN), and acetylcholine (*9*) from basal forebrain (BF). (**B**) Diagram depicting the nonlinear dependence of local modulation on the neuromodulator level (λ). $\lambda \in [0, 1]$ represent low and high dopamine concentrations, respectively (*29*). As λ increases, the neuromodulation factor ($\theta \in [-2, 2]$) sets the extent of LTP/LTD amplitude modulation. For a particular pre- and postsynaptic spike timing, no modulation occurs at $\lambda = 0$. The modulation factor first increases with λ until it reaches the maximal amplitude ($\theta_{\max}$) and then decreases with λ until λ_{inv}, where no modification occurs again. Above λ_{inv}, the neuromodulation reverses the direction of LTP/LTD. (**C**) In the biological nervous system, the common STDP form of spike timing–dependent LTD/LTP (DP) can be reshaped into eight other forms that exhibit various combinations of LTD (D), LTP (P), and no modification (0) at negative and positive time windows: complete temporal inversion (PD), partial inversion of DP or PD (PP and DD), partial elimination of DP (0P and D0) and PD (0D and P0), and no synaptic modification (00).

example, the Hebbian rule has inspired the three-factor rule for reinforcement learning (*25*, *26*), which uses pre- and postsynaptic neuronal activity as the first two factors and distal reward-dependent neuromodulator levels as the third factor, with a substantial time delay between the Hebbian modification and the reward. The eligibility trace model (*27*) accumulates a temporal trace of previous coincident pre- and postsynaptic spiking to allow delayed reward-dependent synaptic modifications. Although models in computational neuroscience have used the neuromodulator level for determining the amplitude and polarity of synaptic modification (*28*), such mechanisms have not been incorporated into either ANNs or SNNs. Compared to these plasticity algorithms, the NACA algorithm has not only achieved higher recognition accuracy and a lower computational cost during the supervised learning of image and speech recognition but also markedly mitigated the problem of catastrophic forgetting during class-CL. Further mapping of synaptic weight changes in the hidden layer showed that, unlike other backpropagation (BP)–based algorithms, NACA yielded a distribution of weight changes that avoided excessive synaptic potentiation or depression, preserving a large number of synapses with minimal modifications. Together, our results provide a new brain-inspired algorithm for expectation-based global neuromodulation of synaptic plasticity that enables neural network performance with high accuracy and low computational cost for various recognition and continuous learning tasks.

RESULTS

Introducing brain-inspired NACA into SNNs and ANNs

During network training with the NACA algorithm, we defined neuromodulator levels at subpopulations of synapses in the hidden and output layers based on the input type and the output error, respectively. At each synapse in SNNs, modulation of LTP and LTD amplitude and polarity depended on the neuromodulator level in a nonlinear manner (Fig. 1B), as inspired by the dependence of synaptic efficacy on the level of neuromodulators or calcium (*29*, *30*). For example, dopamine binding to synapses containing D1-like or D2-like receptors could differentially activate intracellular signaling cascades, leading to the modulation of activity-induced LTP or LTD (*12*, *31*, *32*). Specifically, we used the input type–based expectation matrix (N_{in}) to assign the neuromodulator level (ranged from 0 to 1) at each hidden layer synapse. The neuromodulator values in the N_{in} matrix were randomly dispersed values based on a uniform probability distribution, which was congruent to the discovered distribution of biological reward expectations for generating dopamine signals in the mouse brain (*33*). The determination of expectation values (E) for each input type is illustrated by the example of input digit classes two and seven in the Modified National Institute of Standards and Technology (MNIST) dataset (*34*). We used two normal distributions with the same SD configuration but located randomly (see Materials and Methods) and determined E for each input type (e.g., E_{2a} and E_{2b} for digit class two) based on the intersection with two normal distributions. This approach of determining E allows dispersed assignment of input-specific credit values in the N_{in} matrix, resulting in input type–specific coding of neuromodulator levels at selective subpopulations of hidden layer synapses (Fig. 2A). The neuromodulator level at each synapse was then used to determine the modulation of LTP/LTD amplitude and polarity based on the nonlinear function $f_{\text{local}}(\cdot)$.

In parallel, we also set the neuromodulator levels at output layer synapses. We used the vector difference between the output expectation and actual output value as the output error to determine the neuromodulator level, where an output error–based expectation matrix (N_{out}) was designed with error values only in diagonal units (Fig. 2A). The local plasticity for output layer synapses also depended on the nonlinear function of $f_{\text{local}}(\cdot)$ (Fig. 2B). Together, this NACA algorithm allows global expectation-dependent presetting of local neuromodulator levels at selective subpopulations of hidden layer and output layer synapses (Fig. 2C).

For applying NACA to SNNs, feedforward-only networks were used (e.g., a three-layer SNN in Fig. 2D). Neurons in the first (input) layer received input spike trains, which encoded input signals from the dataset via comparison with a train of computer-generated random numbers (see Materials and Methods). The fully connected

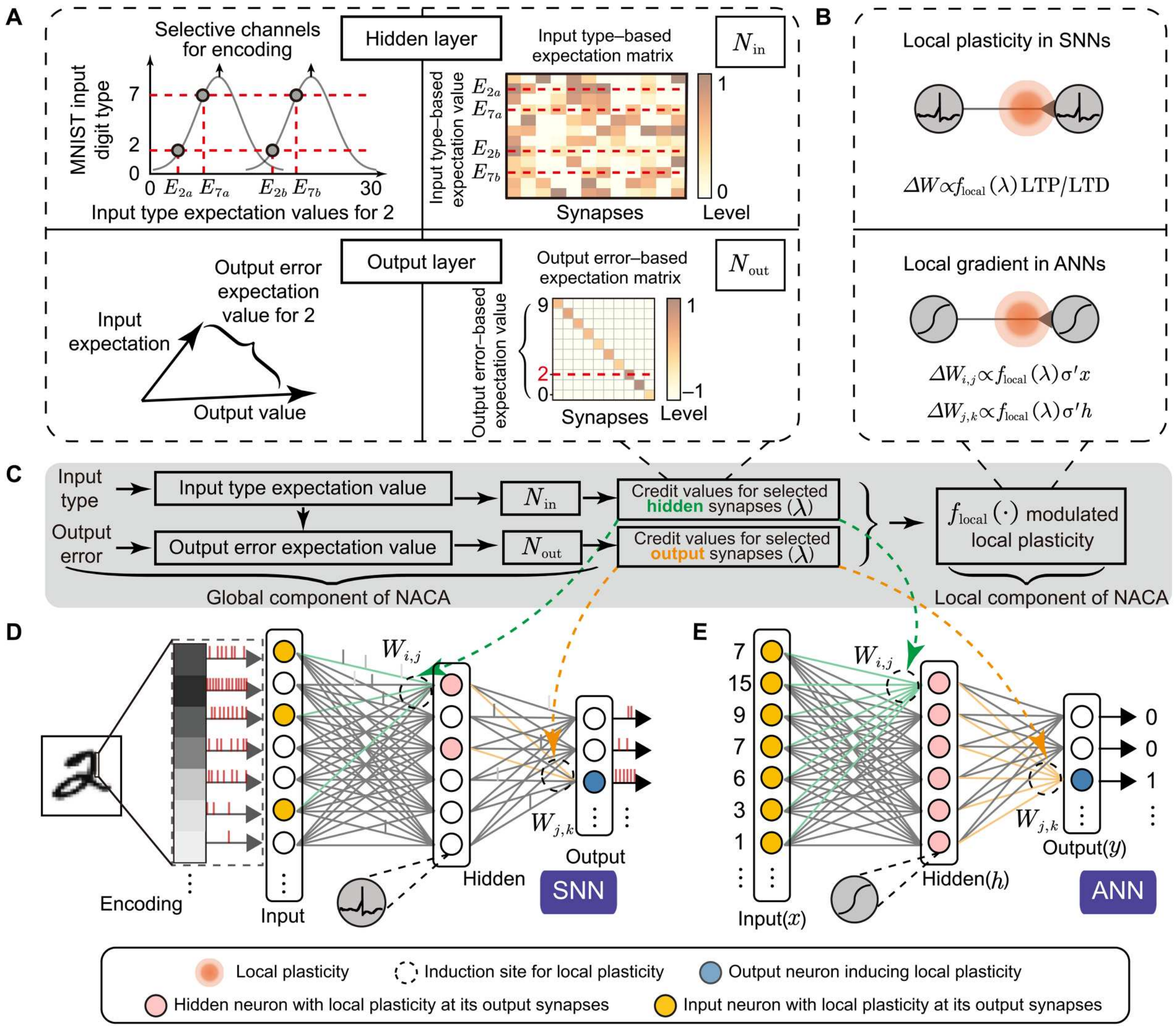

A
Selective channels for encoding
MNIST input digit type
Input type expectation values for 2
Hidden layer
Input type–based expectation matrix
N_{in}
Input type–based expectation value
Synapses
Level
Output layer
Output error–based expectation matrix
N_{out}
Output error–based expectation value
Output error expectation value for 2
Input expectation
Output value
B
Local plasticity in SNNs
$\Delta W \propto f_{local}(\lambda)$ LTP/LTD
Local gradient in ANNs
$\Delta W_{i,j} \propto f_{local}(\lambda)\sigma' x$
$\Delta W_{j,k} \propto f_{local}(\lambda)\sigma' h$
C
Input type
Output error
Input type expectation value
Output error expectation value
Credit values for selected hidden synapses (λ)
Credit values for selected output synapses (λ)
$f_{local}(\cdot)$ modulated local plasticity
Global component of NACA
Local component of NACA
D
Encoding
Input
Hidden
Output
SNN
E
Input(x)
Hidden(h)
Output(y)
ANN
Local plasticity
Induction site for local plasticity
Output neuron inducing local plasticity
Hidden neuron with local plasticity at its output synapses
Input neuron with local plasticity at its output synapses

Fig. 2. Introducing the NACA algorithm into SNNs and ANNs. (**A**) Expectations based on the input type and output error are encoded by the neuromodulator level (credit value) at selected populations of synapses via expectation-based neuromodulator value matrices N_{in} and N_{out}, which set input type credit values at hidden layer synapses and output error credit values at output layer synapses, respectively. The values in N_{in} are randomly generated (with a uniform probability distribution), inspired by the biological discovery (*33*), and those in N_{out} are output error–dependent values only for diagonal units. $E_{2a/b}$ and $E_{7a/b}$ are the input type credit values after encoding with two normal distributions (see Materials and Methods). (**B**) Nonlinear function for local credit assignment $f_{local}(\bullet)$ based on the neuromodulator level dependence of modulation factors (θ in Fig. 1B). (**C**) Schematic diagram depicting the pipeline of the NACA algorithm that implements global neuromodulation of local plasticity, including linear encoding of expectation values into neuromodulator levels globally at selective subpopulations of synapses and nonlinear neuromodulator level–dependent modulation of local plasticity. (**D** and **E**) Implementing the NACA algorithm in a standard three-layer SNN using leaky integrate-and-fire neurons (D) or ANN using sigmoid-like activation function (E). Input neurons (yellow) receive input type–specific expectations to activate hidden neurons (pink) and to generate local plasticity at hidden layer synapses ($W_{i,j}$). The hidden neurons, in turn, activate output neurons (blue) to generate LTP/LTD in SNNs or local gradients (*52*) in ANNs at output layer synapses ($W_{j,k}$), both in a manner that depends on the neuromodulator level assigned to each synapse.

hidden layer consisted of leaky integrate-and-fire (LIF) neurons that exhibited a refractory period, nonlinear integration, and nondifferentiable membrane potential. The output layer consisted of LIF neurons that received spiking signals from hidden neurons. During network training, input spiking signals first propagated to the hidden neurons, and the input type–dependent modification of LTP/LTD was then introduced at various hidden layer synapses. Then, spikes of hidden neurons further propagated to output neurons, resulting in synaptic modification at output layer synapses, based on the output error–dependent setting of the local plasticity. Thus, hidden layer synapses ($W_{i,j}$) and output layer synapses ($W_{j,k}$) underwent potentiation or depression of their efficacy ($\Delta W_{i,j}$ and $\Delta W_{j,k}$) in a manner that depended on the input type and output error expectations. To introduce some specificity in the magnitude and polarity of synaptic modification, we set an inversion factor ($\lambda_{inv} \in [0, 1]$) and a maximal modulation factor ($\theta_{max} \in [0, 2]$; Fig. 1B).

Feedforward-only ANNs were also used to examine the performance of the NACA algorithm (e.g., a three-layer ANN in Fig. 2E). Neurons in this ANN operated with tanh and sigmoid activation functions in hidden and output layers, respectively. The learning procedure of ANNs was similar to that of SNNs, where the local gradient propagation replaced the local plasticity of LTP/LTD. For both ANNs and SNNs, the NACA algorithm can be described as $\Delta W^l_{\mathrm{NACA}} \propto f_{\mathrm{local}}(N_{\mathrm{in/out}}E, \lambda_{\mathrm{inv}}, \theta_{\max})\Delta W^l_{\mathrm{local}}$ at layer l, in which global linear function (N_{in} and N_{out}) and local nonlinear function $f_{\mathrm{local}}(\cdot)$ were both implemented (see Materials and Methods).

SNNs or ANNs using NACA handled recognition tasks efficiently

We next examined the effect of introducing the NACA algorithm on the recognition accuracy and computational cost of SNNs and ANNs in supervised learning of standard image and speech recognition. Two benchmark tasks were examined: (i) recognition of handwritten digits, using the MNIST dataset (fig. S1, A and B) (*34*) and (ii) recognition of spoken digits, using the TIDigits dataset (fig. S1, A and C) (*35*). The algorithmic complexity (*15*) was selected as the computational cost indicator with the unit of floating-point operations (FLOPs), calculated as the product of the mean number of training epochs (for achieving some predefined recognition accuracy levels, Fig. 3A) and the algorithmic complexity per epoch (Fig. 3B), with comparable numbers of parameters used for SNNs and ANNs (tables S2 to S4).

Learning hand-digit recognition using MNIST dataset

For this image recognition task, we used a feedforward SNN comprising input (784 neurons), two convolutional, one full connection (1000 neurons), and output layers (table S2). We trained the SNN with a subset (60,000 samples, 28 × 28 pixels resolution) of the MNIST dataset and tested its recognition accuracy using the remaining MNIST data (10,000 samples). Input-specific spike trains within a time window (T_w) propagated through the network, and normal probability distribution with a full matrix rank size was selected as the standard configuration of the matrix N_{in}. The latter choice was made after a systematic comparison of network performance for three different probability distributions of matrix values (fig. S2) and was used throughout the rest of this study. Under the normal probability distribution, the standard values of $\lambda_{\mathrm{inv}}= 0.5$ were chosen after a range of values was tested for the network performance (Fig. 3C, top), representing an optimal value for modulating the polarity of local plasticity. Similarly to it, the maximal modulation factor $\theta_{\max}= 1.2$ was chosen to modulate the magnitude of local plasticity. In these experiments, λ_{inv} also displayed more importance compared to $\theta_{\max}$, which was congruent to the discovery of artificial networks where the gradient direction (e.g., a similar role of the λ_{inv}) usually contributed to classification performance more than the gradient amplitude (e.g., a similar role of the $\theta_{\max}$) during network learning (*36*). We found that the test accuracy converged to 99.03 ± 0.06% (SEM, $n = 5$, repeats with different random seeds) after the 100th epoch for the NACA algorithm. This accuracy level was slightly higher than that achieved by using two state-of-the-art (SOTA) algorithms, namely, biologically plausible reward propagation (BRP) (*37*) (98.79 ± 0.12%, SEM, $n = 5$, $P = 0.017$, t test) and eligibility propagation (e-prop) (*27*) (98.18 ± 0.15%, SEM, $n = 5$, $P < 0.01$, t test; Fig. 3C, middle), under the same SNN configurations and parameters. Notably, the computational cost for training the SNN (see Materials and Methods) using NACA algorithm was much lower ($0.02 \pm 0.00 \times 10^8$ FLOPs, SEM, $n = 5$) than those using BRP ($0.17 \pm 0.02 \times 10^8$ FLOPs, SEM, $n = 5$, $P < 0.01$, t test) and e-prop ($1.41 \pm 0.01 \times 10^8$ FLOPs, SEM, $n = 5$, $P < 0.01$, t test). Thus, the NACA algorithm has achieved a little higher level of recognition accuracy (up to 0.85% accuracy increase) with a marked reduction of computational cost (up to 98% cost reduction; Fig. 3C, bottom).

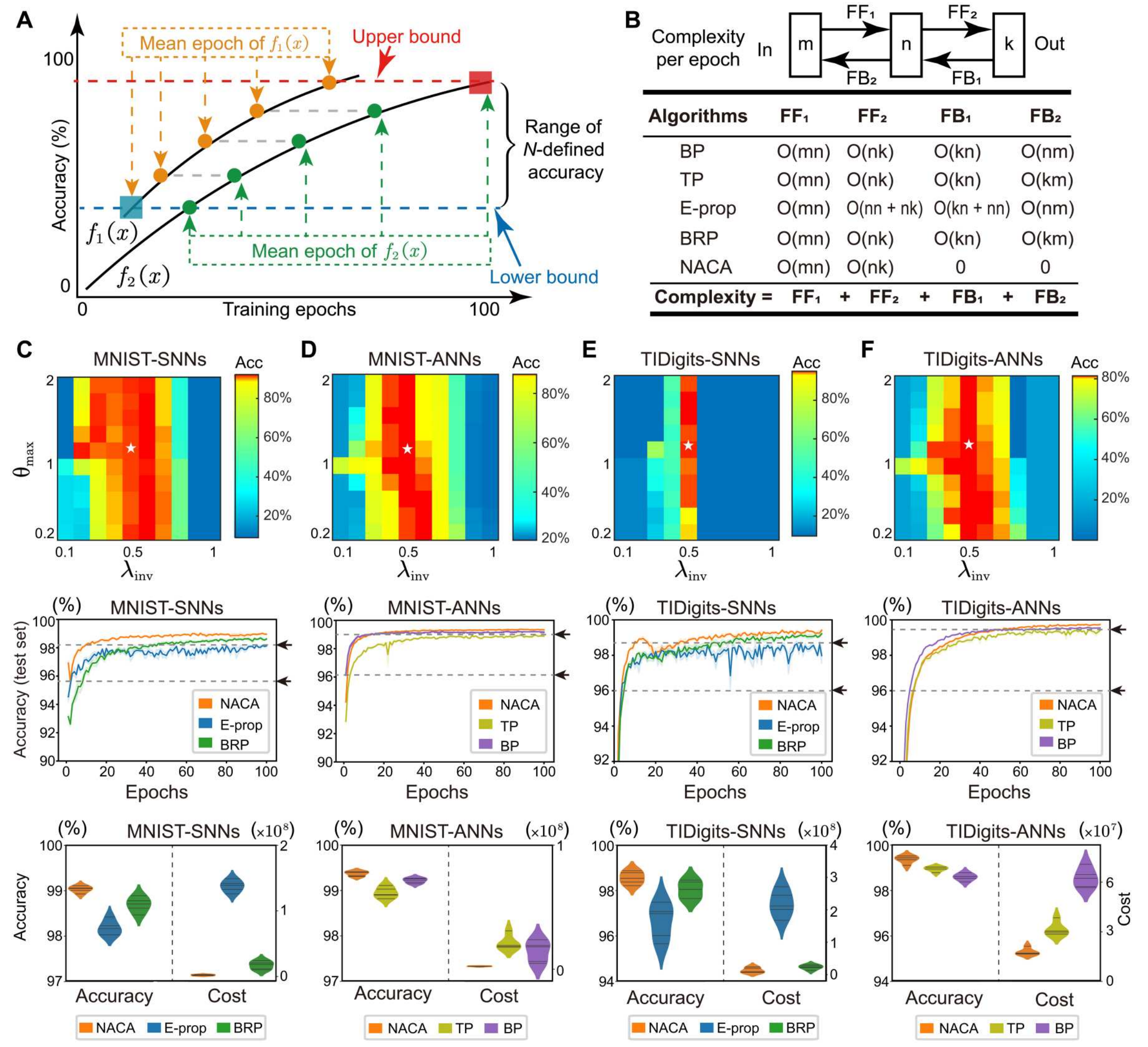
Similar elevated performance was also achieved using the NACA algorithm in ANNs with the same network configuration as the SNN above (see tables S2 and S4). At $\lambda_{\mathrm{inv}} = 0.5$ and $\theta_{\max} = 1.2$ (Fig. 3D, top), a slightly higher accuracy (99.37 ± 0.04%, SEM, $n = 5$) and a much lower computational cost ($0.23 \pm 0.01 \times 10^7$ FLOPs, SEM, $n = 5$) were also found, as compared with those obtained by using two SOTA algorithms for the ANN, namely, target propagation (*38*, *39*) (TP; accuracy, 98.95 ± 0.10%; cost, $2.09 \pm 0.22 \times 10^7$ FLOPs) and BP (accuracy, 99.22 ± 0.05%; cost, $1.43 \pm 0.34 \times 10^7$ FLOPs; Fig. 3D, middle), corresponding to up to 0.42% increase in accuracy and up to 88% reduction in computational cost (Fig. 3D, bottom).

Learning spoken digit recognition using TIDigits dataset

For this speech recognition task, we also used a feedforward SNN comprising 900, 1000, and 10 neurons in the input, full-connection hidden, and output layers, respectively (tables S2 and S3), with the standard $\lambda_{\mathrm{inv}} = 0.5$ and $\theta_{\max} = 1.2$ (Fig. 3E, top). The auditory stimulus inputs were sampled at 20 kHz and processed with Mel-scale frequency cepstrum coefficients (MFCCs) into 30 frames and 30 bands. We trained the SNN with a subset (2900 samples) of the TIDigits dataset and tested the speech recognition accuracy using the remaining TIDigits data (1244 samples). We found that the NACA algorithm achieved a test accuracy slightly higher (98.56 ± 0.24%, SEM, $n = 5$) than those achieved by BRP (*37*) (98.04 ± 0.35%, SEM, $n = 5$, $P < 0.01$) and e-prop (*27*) (96.64 ± 0.69%, SEM, $n = 5$, $P < 0.01$), under the same network configuration and parameters (up to 1.92% accuracy increase; Fig. 3E, middle). The computational cost for the SNN training with the NACA algorithm ($0.13 \pm 0.03 \times 10^8$ FLOPs, SEM, $n = 5$) was much lower than that with BRP ($0.23 \pm 0.03 \times 10^8$ FLOPs, SEM, $n = 5$, $P < 0.01$) and e-prop ($2.19 \pm 0.16 \times 10^8$ FLOPs, SEM, $n = 5$, $P < 0.01$), representing up to a 94% cost reduction (Fig. 3E, bottom).

Similarly, using the NACA algorithm (with $\lambda_{\mathrm{inv}} =0.5$ and $\theta_{\max} =1.2$; Fig. 3F, top) in the ANN achieved slightly higher recognition accuracy (99.37 ± 0.14%, SEM, $n = 5$) and reduced computational cost ($1.78 \pm 0.08 \times 10^7$ FLOPs, SEM, $n = 5$), as compared with those achieved by TP (accuracy, 98.59 ± 0.13%; cost, $3.14 \pm 0.16 \times 10^7$ FLOPs) and BP (accuracy, 98.95 ± 0.09%; cost, $6.23 \pm 0.23 \times 10^7$ FLOPs; Fig. 3F, middle), representing up to a 0.78% accuracy increase and a 71% cost reduction (Fig. 3F, bottom).

Last, we examined the benefit of NACA for SNNs and ANNs that were only composed of fully connected hidden layers in learning MNIST (fig. S1B) and TIDigits (fig. S1C) recognition tasks when the number of hidden layers was increased from one to three (fig. S3, A and B). We found that increasing the number of layers had little effect on the recognition accuracy, but computational costs became notably higher than those of three-layer networks. Thus, the shallow SNNs and ANNs appeared to be the optimal network configuration that yielded benefits in both recognition accuracy and computational cost (up to 1.92% accuracy increase and 98% cost reduction).



Algorithms	FF₁	FF₂	FB₁	FB₂
BP	O(mn)	O(nk)	O(kn)	O(nm)
TP	O(mn)	O(nk)	O(kn)	O(km)
E-prop	O(mn)	O(nn + nk)	O(kn + nn)	O(nm)
BRP	O(mn)	O(nk)	O(kn)	O(km)
NACA	O(mn)	O(nk)	0	0
Complexity =	FF₁ +	FF₂ +	FB₁ +	FB₂

Fig. 3. The NACA algorithm enhanced the performance of SNNs and ANNs for two recognition tasks. (**A**) Schematic diagram depicting the calculation of the mean epoch in *N* training epochs (*N* = 5) for learning accuracy curves of $f_1(x)$ and $f_2(x)$ between an upper bound and a lower bound (see Materials and Methods). (**B**) The algorithmic complexity per epoch was compared for BP, target propagation (TP) (*38*), biologically plausible reward propagation (BRP) (*37*), eligibility propagation (e-prop) (*27*), and NACA. The total complexity was obtained by summing the complexity $O(\cdot)$ for feedforward (FF) and feedback (FB) steps across multiple layers, where *m*, *n*, and *k* are the numbers of neurons in different layers. (**C**) Performance of SNNs for the handwritten digit recognition task using the MNIST dataset. Top: Accuracies achieved by NACA, using various combinations of inversion factor (λ_{inv}) and maximal modulation factor (θ_{max}). The parameter pair $\lambda_{inv} = 0.5$ and $\theta_{max} = 1.2$ (marked by a white star) was chosen for its relatively good performance. The accuracy is coded in color by the scale shown on the right. Middle: Accuracy values during the training process for the chosen values of λ_{inv} (0.5) and θ_{max} (1.2). Dashed lines (marked by arrows) represent the upper and lower bounds of accepted accuracy levels. Bottom: Violin diagram depicting accuracies at the end of training and computational costs for three different algorithms. (**D**) Performance of ANNs on the same MNIST dataset, presented in the same manner as those in (C). (**E** and **F**) Performance of SNNs and ANNs for the spoken digit recognition task using the TIDigits dataset, presented as those in (C).

NACA mitigated catastrophic forgetting in continuous learning

We next examined the effect of introducing the NACA algorithm on catastrophic forgetting of SNNs and ANNs in continuous supervised learning of three types of benchmark class-CL tasks: (i) 10-class-CL of handwritten digits, using image samples in the MNIST dataset (fig. S1D) (*34*); (ii) 26-class-CL of handwritten alphabets, using image samples in the Alphabet dataset (fig. S1E) (*40*); and (iii) 46-class-CL of mathematic Greek alphabet symbols, using image samples in the MathGreek dataset (fig. S1F) (*41*). Inspired by the multiscale neuromodulation in the brain (Fig. 4A), the NACA in CL tasks contained not only synaptic modulation but also neuronal modulation for the mask-like neuronal selection (Fig. 4B; see Materials and Methods). The learning capability of SNNs and ANNs using the NACA algorithm was compared with those using BP and elastic weight consolidation (EWC) (*21*) algorithms. The classification accuracy and computational cost after continuous learning of *c* classes were evaluated by averaging all accuracy and cost values, represented as $acc^{\leq c}$ and $cost^{\leq c}$, respectively, for classifying test set samples for all *c* classes (see Materials and Methods).

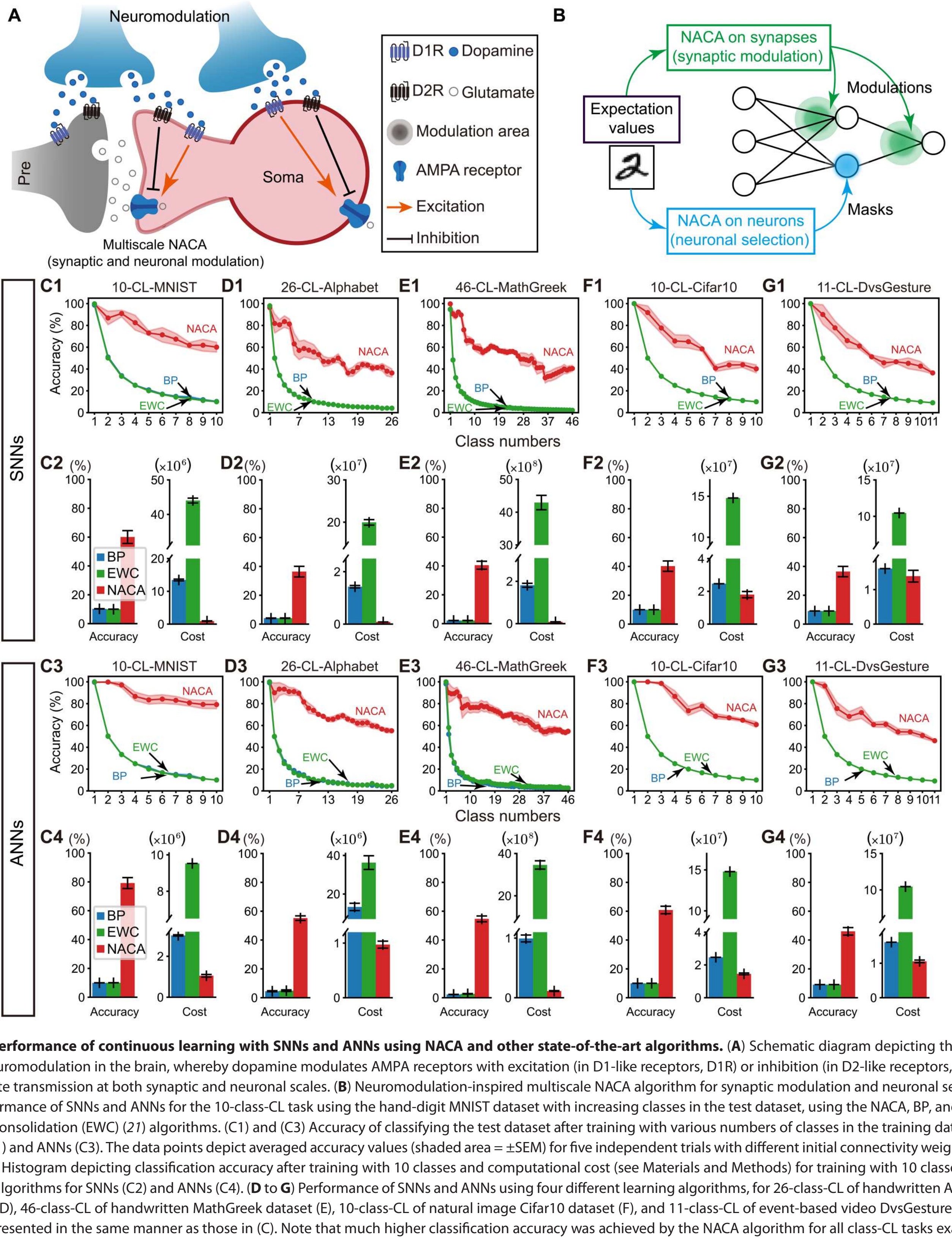


Fig. 4. Performance of continuous learning with SNNs and ANNs using NACA and other state-of-the-art algorithms. (**A**) Schematic diagram depicting the multiscale neuromodulation in the brain, whereby dopamine modulates AMPA receptors with excitation (in D1-like receptors, D1R) or inhibition (in D2-like receptors, D2R) of glutamate transmission at both synaptic and neuronal scales. (**B**) Neuromodulation-inspired multiscale NACA algorithm for synaptic modulation and neuronal selection. (**C**) Performance of SNNs and ANNs for the 10-class-CL task using the hand-digit MNIST dataset with increasing classes in the test dataset, using the NACA, BP, and elastic weight consolidation (EWC) (*21*) algorithms. (C1) and (C3) Accuracy of classifying the test dataset after training with various numbers of classes in the training dataset for SNNs (C1) and ANNs (C3). The data points depict averaged accuracy values (shaded area = ±SEM) for five independent trials with different initial connectivity weights. (C2) and (C4) Histogram depicting classification accuracy after training with 10 classes and computational cost (see Materials and Methods) for training with 10 classes using various algorithms for SNNs (C2) and ANNs (C4). (**D** to **G**) Performance of SNNs and ANNs using four different learning algorithms, for 26-class-CL of handwritten Alphabet dataset (D), 46-class-CL of handwritten MathGreek dataset (E), 10-class-CL of natural image Cifar10 dataset (F), and 11-class-CL of event-based video DvsGesture dataset (G), all presented in the same manner as those in (C). Note that much higher classification accuracy was achieved by the NACA algorithm for all class-CL tasks examined.

Learning 10-class-CL on MNIST dataset

We used a three-layer SNN comprising 784, 1000, and 10 neurons in the input, hidden, and output layers, respectively (λ_{inv} = 0.5 and $\theta_{\max}$ = 1.2; see table S2). The SNN was trained using 10 classes of MNIST training data subset (each class, 6000 samples) continuously without iteration (fig. S1D) and was tested for its classification accuracy and computational cost (see Materials and Methods) using the 10 classes of MNIST test data subset (10 classes, a total of 10,000 samples). We found that the NACA achieved an accuracy ($\text{acc}^{\leq 10}$) of 60.06 ± 4.45% (SEM, n = 5), much higher than those achieved by EWC (10.01 ± 0.02%, SEM, n = 5, $P < 0.01$, t test) and BP (10.26 ± 0.10%, SEM, n = 5, $P < 0.01$, t test) algorithms, representing a striking up to 50% increase in accuracy (Fig. 4C1). Similarly, the computational cost ($\text{cost}^{\leq 10}$) of SNNs using NACA was much lower (0.92 ± 0.07 × 10^6 FLOPs, SEM, n = 5) than those using EWC (44.01 ± 0.70 × 10^6 FLOPs, SEM, n = 5, $P < 0.01$) and BP (13.47 ± 0.26 × 10^6 FLOPs, SEM, n = 5, $P < 0.01$), representing up to a 97% cost reduction (Fig. 4C2). Similar results were achieved by the NACA for ANNs, where much higher classification accuracy and lower computational cost were also found (Fig. 4C, 3 and 4).

Learning 26-class-CL on Alphabet dataset

To increase the number of classes in the class-CL, we trained a three-layer SNN with 26 classes of Alphabet (*40*) training set (26 classes, a total of 297,960 samples) continuously without iteration, and then tested its performance in class recognition using the 26 classes of Alphabet test set (26 classes, a total of 74,490 samples). The SNN configuration was similar to that for the 10-class-CL (see table S2), comprising 784, 1000, and 26 neurons in the input, hidden, and output layers, respectively. We found that the NACA achieved an accuracy ($\text{acc}^{\leq 26}$) of 36.40 ± 3.69% (SEM, n = 5), much higher than those achieved by EWC (4.09 ± 0.15%, SEM, n = 5, $P < 0.01$, t test) and BP (4.07 ± 0.15%, SEM, n = 5, $P < 0.01$, t test) algorithms, representing a striking up to 32% increase in accuracy (Fig. 4D1). Similarly, the computational cost ($\text{cost}^{\leq 26}$) of SNNs using NACA was much lower (0.08 ± 0.01 × 10^7 FLOPs, SEM, n = 5) than those using EWC (19.96 ± 0.65 × 10^7 FLOPs, SEM, n = 5, $P < 0.01$) and BP (1.43 ± 0.05 × 10^7 FLOPs, SEM, n = 5, $P < 0.01$), representing up to a 99% reduction of the computational cost (Fig. 4D2). Similar results were achieved by the NACA for ANNs, representing mitigated catastrophic forgetting and much lower computational cost (Fig. 4D, 3 and 4).

Learning 46-class-CL on MathGreek dataset

To further increase the number and complexity of the classes, we also examined continuous learning of 46-class classification, using the MathGreek (*41*) training dataset (46 classes, a total of 18,224 samples) and the remaining test dataset (46 classes, a total of 12,145 samples). We also used a three-layer SNN similar to that for 10-class-CL (see table S2), comprising 2025, 3000, and 46 neurons in the input, hidden, and output layers, respectively. We found that NACA achieved an accuracy ($\text{acc}^{\leq 46}$) of 40.51 ± 2.76% (SEM, n = 5), much higher than those achieved by EWC (2.17 ± 0.03%, SEM, n = 5, $P < 0.01$, t test) and BP (2.16 ± 0.02%, SEM, n = 5, $P < 0.01$, t test), representing up to a 38% increase in accuracy (Fig. 4E1). Similarly, the computational cost ($\text{cost}^{\leq 46}$) of the SNN using NACA was much lower (0.08 ± 0.01 × 10^8 FLOPs, SEM, n = 5) than those using EWC (42.94 ± 2.20 × 10^8 FLOPs, SEM, n = 5, $P < 0.01$) and BP (1.81 ± 0.09 × 10^8 FLOPs, SEM, n = 5, $P < 0.01$), representing up to a 99% reduction (Fig. 4E2). The NACA achieved similar results for ANNs, with much higher classification accuracy and lower computational cost (Fig. 4E, 3 and 4).

In addition to the above three tests, we also tested these three-layer SNNs and ANNs in learning 10-class-CL on the Cifar10 dataset (fig. S1G) (*42*) that contained more complex natural images, as well as 11-class-CL on DvsGesture (fig. S1H) (*43*) comprising with event-based video records of dynamic images. The results showed that three-layer SNNs and ANNs using the NACA algorithm achieved higher classification accuracy and lower computational cost than those using EWC and BP (Fig. 4, F and G, and tables S2 and S3). The high classification accuracy in performing all five class-CL tasks indicates that the catastrophic forgetting problem has been significantly mitigated by using the NACA algorithm (up to 50% accuracy increase and 99% cost reduction).

The synaptic mechanism for mitigating catastrophic forgetting during class-CL

To understand the mechanism underlying the effectiveness of the NACA algorithm in mitigating catastrophic forgetting during class-CL tasks, we examined weight changes in hidden layer synapses in the three-layer SNN using various training algorithms. Before learning, the synaptic weights of all hidden layer synapses were randomly assigned with initial values between −0.1 and +0.1 without any synaptic modifications. We first mapped all synaptic weight changes ($\Delta W_{i,j}$) after training with the 10th class in the 10-class-CL of the MNIST dataset, using pure local plasticity without neuromodulation (i.e., NACA with λ= 0). We found that the histogram distribution of synaptic weight changes after learning became highly polarized into maximally potentiated and depressed states, with values clustered near −1 and +1 (Fig. 5A), indicating excessive potentiation or depression of most synapses. In addition, after training with the 10th class using the standard NACA algorithm (i.e., with λ_{inv} = 0.5 and $\theta_{\max}$ = 1.2), synaptic weight changes were found to be normally distributed between −1 and +1 (Fig. 5B), without significant bias toward either potentiated or depressed states. Notably, the persistently high number of synapses at position zero shows no weight change throughout continuous learning, reflecting the selective synapse subpopulations using modified forms of local plasticity with no modulation. In contrast, after training with EWC and BP algorithms, most synaptic weight changes were biased toward depressed states under a normal-like distribution between −0.7 and +0.1 (Fig. 5, C and D).

To further examine the evolution of synaptic weight changes during class-CL, we plotted the mean distribution (with five repeated episodes) before (0th) and after learning of the 1st, 3rd, or 10th sample class (Fig. 5E). For learning without neuromodulation (λ = 0), the polarized distribution of synaptic weight changes became apparent after learning the first class, and most changes assumed maximally depressed or potentiated states after learning the 10th class. In contrast, the standard NACA algorithm allowed the evolution of synaptic weight changes during continuous learning toward both a normal-like distribution and the persistently high number of synapses with minimal modifications, reminiscent of that observed during the joint learning using an additional weight regulation (*44*). During class-CL with EWC and BP algorithms, we found that the number of highly potentiated synapses progressively declined, leading to a normally increased number of synapses with varying degrees of depression. In summary, the wide range of synaptic modifications, with most synapses assuming moderate

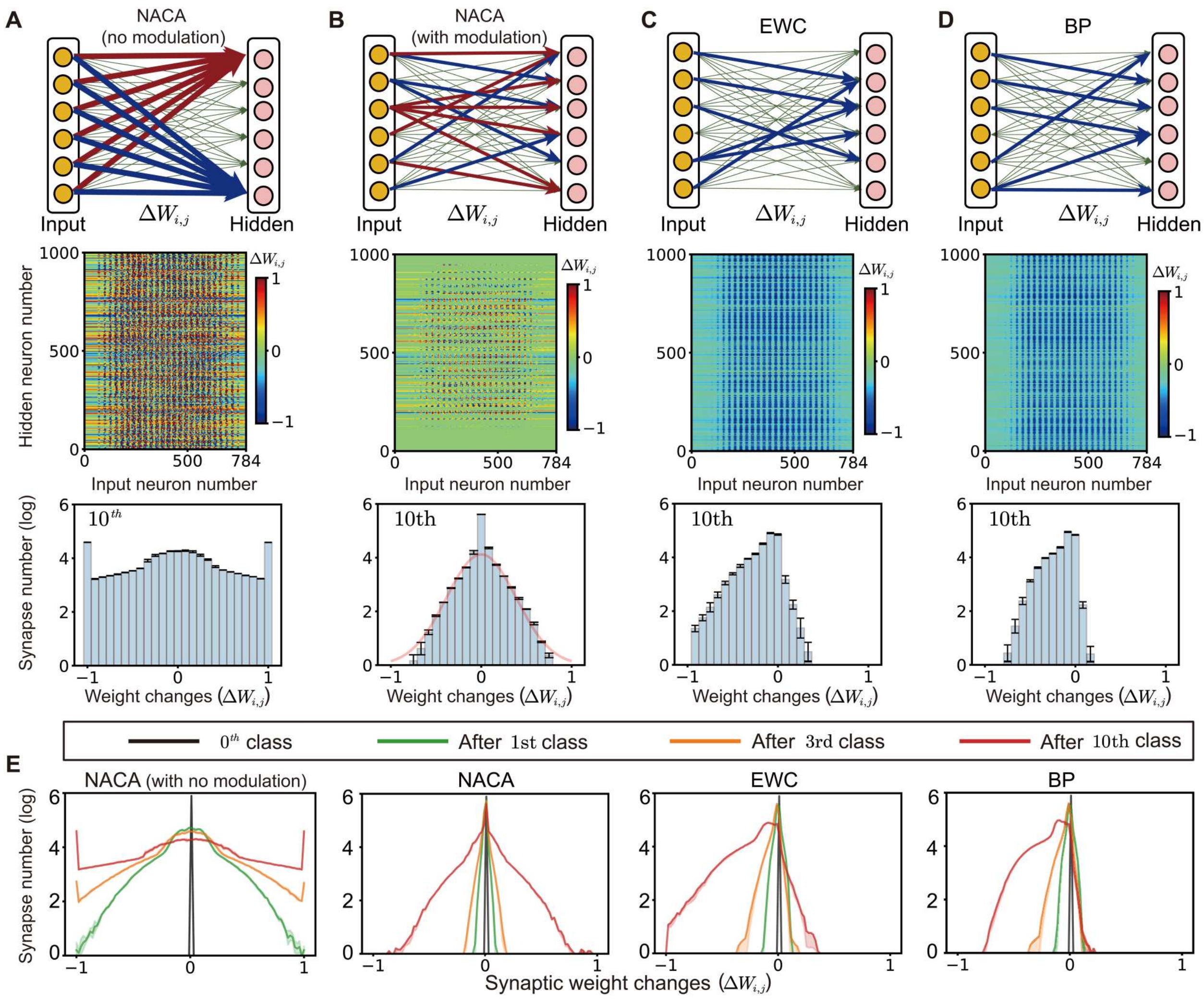


Fig. 5. Synaptic weight changes underlying 10-class-CL of MNIST dataset for different learning algorithms. (**A**) Weight changes at all hidden layer synapses in SNNs after learning the 10th class in a 10-class-CL task using pure local plasticity of LTP and LTD without neuromodulation (NACA with λ = 0). Top: Circuit diagram depicting the synaptic weight changes (ΔW_{ij}) between −1 and +1, representing maximally depressed and potentiated values, respectively. Middle: Weight change values are color-coded in the matrix (with the scale shown on the right) for all synapses between 784 input neurons (horizontal) and 1000 hidden neurons (vertical). Bottom: Histogram of the total number of synapses (in log unit) exhibiting various weight change values. Note that most synapses were polarized toward maximally potentiated or depressed states. (**B**) Synaptic weight changes after learning the 10th class with the NACA algorithm (λ_{inv} = 0.5, θ_{max} = 1.2), exhibiting a normal-like distribution of synapses with various weight change values. Note that weight changes of a large number of synapses remained unchanged (peak at 0). Three panels were presented in the same manner as in (A). (**C** and **D**) Synaptic weight changes after learning the 10th class, using EWC and BP algorithms. (E) Evolution of synaptic weight changes during class-CL, as shown by weight distributions before (0th) and after learning the 1st, 3rd, and 10th class, using pure local plasticity of LTP and LTD without neuromodulation (NACA with λ = 0), standard NACA, EWC, and BP algorithms, respectively. All figures were averaged over five learning episode repeats using different initial random seeds.

potentiation or depression, may reflect the effective credit assignment of the NACA algorithm, underlying the mechanism for mitigating catastrophic forgetting (*44*).

DISCUSSION

In this study, we have developed a learning algorithm NACA for SNNs and ANNs that incorporates neuromodulation-dependent synaptic plasticity and found elevated accuracy and markedly reduced computational cost of the network in performing standard image and voice recognition tasks. The NACA algorithm also significantly mitigated catastrophic forgetting associated with continuous learning of five different class-CL tasks with varying degrees of complexity. Although some neuromodulation-inspired network learning algorithms have been proposed, such as global neuronal workspace theory in SNNs (*45*) and neuromodulation of dropout probability (or learning rate decay) in ANNs (*46*), three key features set NACA apart and may underlie its effectiveness. First, input type and output error are used as expectations for setting the neuromodulator level at selective neurons and synapses in the hidden and output layers. Second, the local synaptic plasticity of LTP or LTD is modified in a manner that depends nonlinearly on the neuromodulator level. Third, network learning depends only on local plasticity without involving the global BP of error signals.

Compared with other learning algorithms, the NACA algorithm markedly reduced the computational cost in all tasks examined. NACA mitigated catastrophic forgetting known to plague continuous learning. Further mapping of synaptic weight modifications at hidden layer synapses during class-CL showed that NACA led to rather normally distributed synaptic weight changes without excessive potentiation or depression and preserved a large number of synapses with minimal modification throughout class-CL. This distribution of synaptic weight changes may underlie the effectiveness of NACA in mitigating catastrophic forgetting.

There are several potential causes for the reduction of computational cost by NACA. First, credit assignment into neuromodulator levels at selective synapses allowed effective synaptic weight modifications that reduced the mean number of epochs required for training. Second, local synaptic modification (e.g., different STDP forms supported by the reversed LTP or LTD) did not involve some extensive computation usually required for the global propagation of gradient-descent signals in other algorithms. The elevated accuracy in the network performance using NACA could be attributed to the reduced overlap of synaptic modifications during network learning, resulting from the expectation-based selection of synapses and neurons. In other BP-based algorithms, there is an extensive overlap of gradient-descent calculations that used global loss functions interleaved between feedforward and BP of signals.

The NACA algorithm performs neuromodulation of local plasticity (e.g., LTP and LTD) in SNNs and local gradient in ANNs in a manner that depends nonlinearly on the neuromodulator level assigned to each synapse. This modulation alters the capability and pattern of LTP and LTD, resulting in synapse subpopulation-specific metaplasticity that confers differential STDP forms at specific synapses in accordance to input and output expectation values. Diversifying local plasticity could prevent excessive synapse potentiation or depression, as evidenced by our results on the distribution of synapse weight changes in the hidden layer. We note that the advantage of NACA in class-CL appeared soon after learning a few sample classes, as shown by much higher classification accuracy than that achieved by other algorithms (Fig. 4). The mapping of synaptic weight changes in the hidden layer during class-CL also showed the early emergence of both normally distributed synaptic weight changes and the persistently high number of synapses showing no weight change throughout continuous learning (Fig. 5). Thus, the reduction of catastrophic forgetting by NACA could be explained by effective control of weight changes at most synapses toward moderate potentiation or depression, as well as different subpopulation selections of neurons and synapses.

There are also some limitations of the proposed NACA algorithm. First, the NACA algorithm exhibits a little nonstability during the neuromodulation of synaptic modifications, especially in deeper neural networks. For example, test accuracy encounters a systematic transient drop during the first few epochs (fig. S2), partly caused by the parallel neuromodulation at multilayer synapses (see the subfigure of fig. S2 for more details). Second, the NACA algorithm is hard to be integrated with the conventional BP algorithm since the global neuromodulation in NACA is parallel or even in advance of the local spike propagation, with the spirit of predictive coding. In contrast, the global error gradient in BP is calculated only after the sequential propagation of multilayer spike trains. Third, only a single type of neuromodulator and excitatory LIF neuron is introduced and examined in NACA, without further discussing the interactions of neuromodulations from different types of neuromodulators (e.g., the noradrenaline, serotonin, and acetylcholine that can bind several different kinds of local GPCRs and trigger diverse following biochemical reactions to support various effects on synaptic plasticity) and more neuron types (e.g., some inhibitory neurons).

In summary, the NACA algorithm could guide network learning for SNNs and ANNs by incorporating biologically plausible learning rules without using global BP-like gradient descent computations. It exemplifies that using brain-inspired mechanisms can achieve high efficiency and low computational cost in machine learning. Implementing the NACA algorithm into neuromorphic devices may facilitate the development of computing systems capable of power-efficient online continuous learning. From a computational neuroscience perspective, the effectiveness of NACA also suggests that metaplasticity-based diversification of local plasticity may underlie the efficiency of neural circuits in the brain for continuous learning.

MATERIALS AND METHODS

Dynamic neurons in SNNs

We use standard LIF as a basic model of spiking neurons, shown as follows:

$$\sigma_l^{\text{lif}}(\boldsymbol{S}_t^{l-1}) = \begin{cases} C\frac{d\boldsymbol{V}_t^l}{dt} = g_l(\boldsymbol{V}_t^l - V_r)(1-\boldsymbol{S}_t^l) + \boldsymbol{W}^l\boldsymbol{S}_t^{l-1} \\ \boldsymbol{V}_t^l = V_r, \boldsymbol{S}_t^l = 1 \quad if(\boldsymbol{V}_t^l = V_{th}) \end{cases} \tag{1}$$

where $\boldsymbol{V}_t^l$ is the membrane potential of layer l at time step t, $\boldsymbol{W}^l$ is the weight of synapses onto layer l neurons, C is the membrane capacitance, g_l is the conductance, V_r is the rest and reset potential, V_{th} is the firing threshold, and $\boldsymbol{S}_t^l$ is the spiking state in layer l neurons at time t. More detailed parameters are shown in table S2.

Encoding of the expectation value *E*

The input type and output error encode the expectation values for hidden and output layers, respectively. The input type $\boldsymbol{E}$ is encoded by two normal distributions with the same SDs (set value as the number of $\boldsymbol{c}$ classes) but different means (randomly generated in the range of $\boldsymbol{c}$ and $3\boldsymbol{c}$). The two input type expectation values are then rounded for synapse selections in the expectation matrix $\boldsymbol{N}_{\mathbf{in}}$. The output error $\boldsymbol{E}$ is the error difference between the output expectation and actual output values, which is then organized as diagonal units in the expectation matrix with error values.

Preprocessing of datasets

For SNNs and ANNs, the raw data of datasets in both recognition tasks and class-CL tasks are preprocessed with data normalization (by subtracting the minimum and dividing by the data range). For SNNs, the normalized raw data are then repeated T_w times to generate spike trains (see the following NACA-SNN for more details). However, this additional rate-to-spike conversion is unnecessary for some special paradigms, such as the DvsGesture task, where the raw data are already event-based spikes, or classification tasks using ANNs, where only raw fire-rate data are required.

NACA for SNNs

The NACA algorithm for SNNs consists of five parts: (i) encoding of spike trains; (ii) setting neuromodulator levels by using input type–based expectation matrix $\boldsymbol{N}_{\mathbf{in}}$ and output error–based expectation matrix $\boldsymbol{N}_{\mathbf{out}}$; (iii) update of membrane potentials $\boldsymbol{V}_{\boldsymbol{t}}^{\boldsymbol{l}}$ and spikes $\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l}}$; (iv) neuromodulator level–dependent shaping of local synaptic plasticity by using a local nonlinear function $\boldsymbol{f}_{\mathbf{local}}(\cdot)$; (v) decoding of spike trains. The procedure of information processing and expectation-based synaptic modification in SNNs is depicted as follows

$$\underbrace{\sigma_l^{\mathrm{enc}}(\boldsymbol{X})}_{\text{Encoding}} \rightarrow \underbrace{\lambda^l = \boldsymbol{N}_{\mathbf{in}}^{\boldsymbol{l}} \times \boldsymbol{E}}_{\text{Global neuromodulation}} \rightarrow \underbrace{\sigma_l^{\mathrm{lif}}(\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l-1}})}_{\text{Update } V_t^l, S_t^l}$$
$$\rightarrow \underbrace{\Delta \boldsymbol{W}_{\mathbf{NACA}}^{\mathbf{SNN},\boldsymbol{l}} = -\eta f_{\mathrm{local}}(\lambda^l)\Delta \boldsymbol{W}_{\mathbf{local}}^{\mathbf{SNN},\boldsymbol{l}}}_{\text{Modulated local plasticity}} \rightarrow \underbrace{\sigma_l^{\mathrm{dec}}(\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}})}_{\text{Decoding}} \tag{2}$$

First, the visual and auditory signals for the SNN input should be preprocessed into spike trains before being given to the input neurons of SNNs. We use simple random sampling to generate spike trains in a time window T_w by comparing the raw pixels with a random number value, shown as follows

$$\boldsymbol{S}_t^0 = \sigma_l^{\mathrm{enc}}(\boldsymbol{X}) = \boldsymbol{X} > \mathrm{Rand}_{N(0,1)}, t \in T_w \tag{3}$$

where $\boldsymbol{S}_{\boldsymbol{t}}^{\mathbf{0}}$ is the generated spike train, $\boldsymbol{X}$ is the raw data, $\mathrm{Rand}_{N(0,1)}$ is a random number generator with a standard uniform distribution with a mean of 0.5 and a variance of 0.5, and σ_l^{enc} is an encoding function. In some trials, spiking convolutional layers follow the input layer to achieve higher accuracy.

Second, the local neuromodulator level λ^l is generated by the linear global neuromodulation propagation function, calculated as follows

$$\begin{cases} \lambda^l = \boldsymbol{N}_{\mathbf{in}}^{\boldsymbol{l}} \times \boldsymbol{E} \\ \lambda^L = \boldsymbol{N}_{\mathbf{out}}^{\boldsymbol{L}} \times (\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}} - \boldsymbol{E}) \end{cases} \tag{4}$$

where $\boldsymbol{N}_{\mathbf{in}}^{\boldsymbol{l}}$ is the input type–based expectation matrix at hidden layer $\boldsymbol{l}$, $\boldsymbol{N}_{\mathbf{out}}^{\boldsymbol{L}}$ is the output error–based expectation matrix at output layer L, $\boldsymbol{E}$ is the encoded expectation value, and $\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}}$ is the output value. The modulation of local plasticity in the function $f_{\mathrm{local}}(\cdot)$ could be calculated as follows

$$f_{\mathrm{local}}(\lambda) = \begin{cases} -\frac{(\theta_{\max}-1)(2\lambda-\lambda_1)^2}{\lambda_1{}^2} + \theta_{\max} & \text{if } (0 \le \lambda \le \lambda_1) \\ \frac{(\lambda-\lambda_{\mathrm{inv}})^2}{(\lambda_1-\lambda_{\mathrm{inv}})^2} & \text{if } (\lambda_1 \le \lambda \le \lambda_{\mathrm{inv}}) \end{cases} \tag{5}$$

where λ_{inv} and $\theta_{\max}$ are polarity inversion and maximal modulation factors in the NACA algorithm, respectively, λ_1 is set as $0.5\lambda_{\mathrm{inv}}$, and $f_{\mathrm{local}}(\lambda)$ is a piecewise function containing two parabolic functions inspired by biological findings (*29*, *30*). For $\lambda_{\mathrm{inv}} \le \lambda \le 2\lambda_{\mathrm{inv}}$, we defined $f_{\mathrm{local}}(\lambda) = -f_{\mathrm{local}}(2\lambda_{\mathrm{inv}} - \lambda)$.

Third, the feedforward information is propagated sequentially from the input to the output layer, and the update of the membrane potential is obtained as follows

$$\begin{cases} \boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l}} = \sigma_l^{\mathrm{lif}}(\boldsymbol{W}^{\boldsymbol{l}}\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l-1}}) \\ \boldsymbol{r}^{\boldsymbol{L}} = \sigma_L^{\mathrm{dec}}(\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}}) \end{cases} \tag{6}$$

where the activation function in the hidden layer $l \in [2, L]$ is $\sigma_l^{\mathrm{lif}}(\cdot)$, and an additional spike decoding function $\sigma_L^{\mathrm{dec}}(\cdot)$ is given only at the output layer L to calculate the output firing rate.

Fourth, an STDP-like local synaptic plasticity of LTP and LTD was used to guide synaptic modifications, shown as follows

$$\Delta \boldsymbol{W}_{\mathbf{local}}^{\boldsymbol{l}} = \begin{cases} -\boldsymbol{S}_{\boldsymbol{t-1}}^{\boldsymbol{l}}\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l-1}} & \text{if } (\boldsymbol{V}_{\boldsymbol{t}}^{\boldsymbol{l}} < V_{th} - V_{\mathrm{lens}}) \\ (1-\boldsymbol{S}_{\boldsymbol{t-1}}^{\boldsymbol{l}})\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{l-1}} & \text{if } (\boldsymbol{V}_{\boldsymbol{t}}^{\boldsymbol{l}} \ge V_{th} - V_{\mathrm{lens}}) \end{cases} \tag{7}$$

where V_{lens} represents a V_{th} deviation for identifying two basic STDP forms. The global neuromodulation described in Eq. 5 and the local plasticity in Eq. 7 could be calculated as follows

$$\Delta \boldsymbol{W}_{\boldsymbol{NACA}}^{\mathbf{SNN},\boldsymbol{l}} = -\eta f_{\mathrm{local}}(\lambda^l)\Delta \boldsymbol{W}_{\mathbf{local}}^{\boldsymbol{l}} \tag{8}$$

where η is the learning rate.

Fifth, the decoding of spike train σ_l^{dec} was calculated by the mean firing rate in a time window T_w at the output layer L, shown as follows

$$\sigma_L^{\mathrm{dec}}(\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}}) = \frac{1}{T_w}\sum_{t=1}^{T_w}\boldsymbol{S}_{\boldsymbol{t}}^{\boldsymbol{L}} \tag{9}$$

where σ_L^{dec} is a decoder function to obtain the firing rate of spike trains during the time window T_w.

Furthermore, an input-based masking $\boldsymbol{M}^{\boldsymbol{l}}$ is applied in the feedforward process during CL, corresponding to the neuromodulation on selected neurons. This part of NACA at layer $l \in [2, L-1]$ is calculated as follows

$$\begin{cases} \boldsymbol{M}^{\boldsymbol{l}} = G(f_{\mathrm{rsp}}^l(\overline{\boldsymbol{S}^{\mathbf{1}}}) - b^l) \\ \boldsymbol{S}_{\boldsymbol{m}}^{\boldsymbol{l}} = \boldsymbol{M}^{\boldsymbol{l}}\sigma_l(\boldsymbol{W}^{\boldsymbol{l}}\boldsymbol{S}_{\boldsymbol{m}}^{\boldsymbol{l-1}}) \end{cases} \tag{10}$$

where $\overline{\boldsymbol{S}^{\mathbf{1}}}$ is the average of input signals in a training batch, f_{rsp}^l is a dimension-reshaping interpolate function, $G(\cdot)$ is the gate function that gives one if $x > 0$ or zero otherwise, b^l is a predefined bias, σ_l is the hidden layer activation function which is implemented as $\sigma_l^{\mathrm{lif}}(\cdot)$ in SNNs or $\sigma_l^{\mathrm{tanh}}(\cdot)$ in ANNs, and $\boldsymbol{S}_{\boldsymbol{m}}^{\boldsymbol{l}}$ is the modulated neuron outputs transformed from the original activities $\boldsymbol{S}^{\boldsymbol{l}}$.

NACA for ANNs

When dealing with synapses in ANNs, the local plasticity is replaced by the local gradient. We take an example of an ANN with full connections between each neighborhood layer: The activity states at hidden layer $\boldsymbol{l} \in [\mathbf{2}, \boldsymbol{L}-\mathbf{1}]$ are represented as $\boldsymbol{u}^{\boldsymbol{l}}$. The information processing in the ANN using NACA is depicted as follows

$$\underbrace{\boldsymbol{u}^{\mathbf{1}}}_{\text{Input}} \rightarrow \underbrace{\lambda^l = \boldsymbol{N}_{\mathbf{in}}^{\boldsymbol{l}} \times \boldsymbol{E}}_{\text{Global neuromodulation}} \rightarrow \underbrace{\boldsymbol{u}^{\boldsymbol{l}}}_{\text{Propagation}}$$
$$\rightarrow \underbrace{\Delta \boldsymbol{W}_{\mathbf{NACA}}^{\mathbf{ANN},\boldsymbol{l}} = -\eta f_{\mathrm{local}}(\lambda^l)\Delta \boldsymbol{W}_{\mathbf{local}}^{\mathbf{ANN},\boldsymbol{l}}}_{\text{Modulation of local gradient}} \tag{11}$$

The input signal $\boldsymbol{u}^{\mathbf{1}}$ is propagated sequentially from input to output layers, calculated as follows

$$\begin{cases} \boldsymbol{u}^{\boldsymbol{l}} = \sigma_l^{\mathrm{tanh}}(\boldsymbol{W}^{\boldsymbol{l}}\boldsymbol{u}^{\boldsymbol{l-1}}) \\ \boldsymbol{u}^{\boldsymbol{L}} = \sigma_L^{\mathrm{sig}}(\boldsymbol{W}^{\boldsymbol{L}}\boldsymbol{u}^{\boldsymbol{L-1}}) \end{cases} \tag{12}$$

where $\sigma_l^{\mathrm{tanh}}(\cdot)$ and $\sigma_l^{\mathrm{sig}}(\cdot)$ are the tanh and sigmoid activation functions in ANNs, respectively. The whole procedure of the NACA in

ANNs is similar to that in SNNs, obtained as follows

$$\begin{cases} \Delta W_{\mathbf{NACA}}^{\mathbf{ANN},l} = -\eta f_{\mathrm{local}}(\lambda^l)\Delta W_{\mathbf{local}}^{\mathbf{ANN},l} = -\eta f_{\mathrm{local}}(\boldsymbol{N}_{\mathbf{in}}^{l} \times \boldsymbol{E})(\sigma_l^{\tanh})' \boldsymbol{u}^{l-1} \\ \Delta W_{\mathbf{NACA}}^{\mathbf{ANN},L} = -\eta f_{\mathrm{local}}(\lambda^L)\Delta W_{\mathbf{local}}^{\mathbf{ANN},L} = -\eta f_{\mathrm{local}}(\boldsymbol{N}_{\mathbf{out}}^{L} \times (\boldsymbol{u}^{L} - \boldsymbol{E}))(\sigma_L^{\mathrm{sig}})' \boldsymbol{u}^{L-1} \end{cases} \tag{13}$$

where η is the learning rate, $\Delta W_{\mathbf{local}}^{\mathbf{ANN},l} = \frac{\partial \boldsymbol{u}^l}{\partial \boldsymbol{W}^l} = (\sigma_l^{\tanh})f' \boldsymbol{u}^{l-1}$ is the local gradient calculated automatically by PyTorch (*47*).

Recognition accuracy and computational cost of NACA learning

The accuracy (Acc) of the network in performing two recognition tasks is defined as the number of correctly classified samples $\boldsymbol{Y}^+$ divided by the total number of samples (the sum of $\boldsymbol{Y}^+$ and wrongly classified samples $\boldsymbol{Y}^-$)

$$\mathrm{Acc} = \frac{Y^+}{Y^+ + Y^-} \tag{14}$$

The computational cost (Cost_i) of the ith training trial is defined by the product of the mean epoch number to achieve five defined accuracy levels (Fig. 3A). The value $O(n)_i$ represents the algorithmic complexity per epoch (Fig. 3B), described by the number of FLOPs in the neuromorphic hardware. For comparison of three trials ($i = 1$, 2, 3), the computational cost is calculated as follows

$$\mathrm{Cost}_i = \frac{1}{N}\sum_{l=1}^{N} \mathrm{Argmin}(f_i(x) = \mathrm{Acc}_l) \times O(n)_i \tag{15}$$

where Argmin(·) is an argument of the minimum of iterations, $f_i(x)$ is an accuracy rate curve with input epoch x, $O(n)_i$ is the algorithmic complexity with n depicting the number of network parameters, and N is the number of predefined accuracy levels (N= 5). Acc_l is selected out from a range of accuracy levels, with a lower bound of Max[Min (f_1), Min (f_2), Min (f_3), f_0], defined as the relatively higher minimal accuracy of $f_1(x)$, $f_2(x)$, and $f_3(x)$ and an additional predefined f_0 (the minimally acceptable accuracy), which has an upper bound of Min[Max (f_1), Max (f_2), Max (f_3)], defined as the relatively lower maximal accuracy among $f_1(x)$, $f_2(x)$, and $f_3(x)$. The computational cost is calculated by averaging the cost at five different accuracy levels. We have calculated computational costs for different benchmark algorithms (see tables S3 and S4).

Classification accuracy and computational cost for class-CL tasks

We introduce the accuracy in continuous learning as follows

$$\mathrm{acc}^{\leq c} = \frac{1}{c}\sum_{\tau=1}^{c} \mathrm{Acc}^{\tau} \tag{16}$$

where Acc^{τ} is the test accuracy for the τ-th class, and c represents the total number of classes for the class-CL task. Similarly, the computational cost ($\mathrm{cost}^{\leq c}$) for continuous learning of c classes is defined as follows:

$$\mathrm{cost}^{\leq c} = \frac{1}{c}\sum_{\tau=1}^{c} \mathrm{Cost}^{\tau} \tag{17}$$

Datasets

The spatial MNIST (*34*) and temporal TIDigits (*35*) are selected as two benchmark classification datasets. The MNIST dataset contains 10-class handwritten digit numbers, while the TIDigits dataset contains 10-class spoken digits. The MNIST (*34*), Alphabet (*40*), MathGreek (*41*), Cifar10 (*42*), and DvsGesture (*43*) are selected as five benchmark class-CL datasets. The MNIST class-CL contains 10-class handwritten digit numbers, the Alphabet class-CL contains 26-class A-to-Z handwritten alphabet characters, the MathGreek class-CL contains 46-class handwritten Greek characters and some mathematical symbols, the Cifar10 class-CL contains 10-class natural images, and the DvsGesture class-CL contains 11-class event-based gesture records. The detailed download links for these publicly available datasets are listed in the Supplementary Materials.

Supplementary Materials

This PDF file includes:
Figs. S1 to S3
Tables S1 to S4
References

REFERENCES AND NOTES

1. T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, G. Hinton, Backpropagation and the brain. *Nat. Rev. Neurosci.* **21**, 335–346 (2020).
2. R. S. Zucker, Short-term synaptic plasticity. *Annu. Rev. Neurosci.* **12**, 13–31 (1989).
3. D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory* (Psychology Press, 2005).
4. Y. Dan, M. M. Poo, Spike timing-dependent plasticity of neural circuits. *Neuron* **44**, 23–30 (2004).
5. G. Bi, M. Poo, Synaptic modification by correlated activity: Hebb's postulate revisited. *Annu. Rev. Neurosci.* **24**, 139–166 (2001).
6. S. B. Flagel, J. J. Clark, T. E. Robinson, L. Mayo, A. Czuj, I. Willuhn, C. A. Akers, S. M. Clinton, P. E. M. Phillips, H. Akil, A selective role for dopamine in stimulus-reward learning. *Nature* **469**, 53–57 (2011).
7. S. J. Sara, The locus coeruleus and noradrenergic modulation of cognition. *Nat. Rev. Neurosci.* **10**, 211–223 (2009).
8. K. W. Miyazaki, K. Miyazaki, K. Doya, Activation of dorsal raphe serotonin neurons is necessary for waiting for delayed rewards. *J. Neurosci.* **32**, 10451–10457 (2012).
9. A. C. Kruse, B. K. Kobilka, D. Gautam, P. M. Sexton, A. Christopoulos, J. Wess, Muscarinic acetylcholine receptors: Novel opportunities for drug development. *Nat. Rev. Drug Discov.* **13**, 549–560 (2014).
10. W. C. Abraham, M. F. Bear, Metaplasticity: The plasticity of synaptic plasticity. *Trends Neurosci.* **19**, 126–130 (1996).
11. Z. Brzosko, S. B. Mierau, O. Paulsen, Neuromodulation of spike-timing-dependent plasticity: Past, present, and future. *Neuron* **103**, 563–581 (2019).
12. J.-C. Zhang, P.-K. Lau, G.-Q. Bi, Gain in sensitivity and loss in temporal contrast of STDP by dopaminergic modulation at hippocampal synapses. *Proc. Natl. Acad. Sci. U.S.A.* **106**, 13028–13033 (2009).
13. R. C. Froemke, D. Debanne, G.-Q. Bi, Temporal modulation of spike-timing-dependent plasticity. *Front. Synaptic Neurosci.* **2**, 19 (2010).
14. N. Caporale, Y. Dan, Spike timing-dependent plasticity: A Hebbian learning rule. *Annu. Rev. Neurosci.* **31**, 25–46 (2008).
15. T. Zhang, X. Cheng, S. Jia, M. M. Poo, Y. Zeng, B. Xu, Self-backpropagation of synaptic modifications elevates the efficiency of spiking and artificial neural networks. *Sci. Adv.* **7**, eabh0146 (2021).
16. M. Graupner, N. Brunel, Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location. *Proc. Natl. Acad. Sci. U.S.A.* **109**, 3991–3996 (2012).
17. Y. Inglebert, J. Aljadeff, N. Brunel, D. Debanne, Synaptic plasticity rules with physiological calcium levels. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 33639–33648 (2020).
18. C. Li, Y. Li, Y. Zhao, P. Peng, X. Geng, SLER: Self-generated long-term experience replay for continual reinforcement learning. *Appl. Intell.* **51**, 185–201 (2021).

19. G. M. van de Ven, H. T. Siegelmann, A. S. Tolias, Brain-inspired replay for continual learning with artificial neural networks. *Nat. Commun.* **11**, 4069 (2020).
20. G. Zeng, Y. Chen, B. Cui, S. Yu, Continual learning of context-dependent processing in neural networks. *Nat. Mach. Intell.* **1**, 364–372 (2019).
21. J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U.S.A.* **114**, 3521–3526 (2017).
22. H. Li, C. Ma, X. Chen, X. Liu, Dynamic consolidation for continual learning. *Neural Comput.* **35**, 228–248 (2023).
23. B. Tsuda, K. M. Tye, H. T. Siegelmann, T. J. Sejnowski, A modeling framework for adaptive lifelong learning with transfer and savings through gating in the prefrontal cortex. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 29872–29882 (2020).
24. N. Y. Masse, G. D. Grant, D. J. Freedman, Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proc. Natl. Acad. Sci. U.S.A.* **115**, E10467–E10475 (2018).
25. W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, J. Brea, Eligibility traces and plasticity on behavioral time scales: Experimental support of neohebbian three-factor learning rules. *Front Neural Circuits* **12**, 53 (2018).
26. E. M. Izhikevich, Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cereb. Cortex* **17**, 2443–2452 (2007).
27. G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, W. Maass, A solution to the learning dilemma for recurrent networks of spiking neurons. *Nat. Commun.* **11**, 3625 (2020).
28. J. Mei, E. Muller, S. Ramaswamy, Informing deep neural networks by multiscale principles of neuromodulatory systems. *Trends Neurosci.* **45**, 237–250 (2022).
29. J. N. J. Reynolds, J. R. Wickens, Dopamine-dependent plasticity of corticostriatal synapses. *Neural Netw.* **15**, 507–521 (2002).
30. T. Nakano, T. Doi, J. Yoshimoto, K. Doya, A kinetic model of dopamine- and calcium-dependent striatal synaptic plasticity. *PLOS Comput. Biol.* **6**, e1000670 (2010).
31. Z. Brzosko, W. Schultz, O. Paulsen, Retroactive modulation of spike timing-dependent plasticity by dopamine. *eLife* **4**, e09685 (2015).
32. K. Yang, J. A. Dani, Dopamine D1 and D5 receptors modulate spike timing-dependent plasticity at medial perforant path to dentate granule cell synapses. *J. Neurosci.* **34**, 15888–15897 (2014).
33. J. Tian, R. Huang, J. Y. Cohen, F. Osakada, D. Kobak, C. K. Machens, E. M. Callaway, N. Uchida, M. Watabe-Uchida, Distributed and mixed information in monosynaptic inputs to dopamine neurons. *Neuron* **91**, 1374–1389 (2016).
34. Y. LeCun, The MNIST database of handwritten digits (1998); http://yann.lecun.com/exdb/mnist/.
35. R. G. Leonard, G. R. Doddington, TIDIGITS LDC93S10. Web Download. Philadelphia: Linguistic Data Consortium (1993).
36. T. P. Lillicrap, D. Cownden, D. B. Tweed, C. J. Akerman, Random synaptic feedback weights support error backpropagation for deep learning. *Nat. Commun.* **7**, 13276 (2016).
37. T. Zhang, S. Jia, X. Cheng, B. Xu, Tuning convolutional spiking neural network with biologically plausible reward propagation. *IEEE Trans. Neural Netw. Learn. Syst.* **33**, 7621–7631 (2022).
38. A. Meulemans, F. S. Carzaniga, J. Suykens, J. Sacramento, B. F. Grewe, A theoretical framework for target propagation. *Adv. Neu. Info. Process. Sys.* **33**, 20024–20036 (2020).
39. D.-H. Lee, S. Zhang, A. Fischer, Y. Bengio, Difference target propagation. *Mach. Learn. Knowl. Discov. Databases* **9284**, 10.1007/978-3-319-23528-8_31, (2015).
40. P. Grother, K. Hanaoka, NIST handprinted forms and characters database. Web Download. NIST (2016).
41. H. Mouchère, R. Zanibbi, U. Garain, Handwritten math symbols dataset. CROHME: Competition on Recognition of Online Handwritten Mathematical Expressions (2014).
42. A. Krizhevsky, "Learning multiple layers of features from tiny images" (2009); www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.
43. A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, D. Modha, A low power, fully event-based gesture recognition system. *2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 7243–7252 (2017).
44. J. Yosinski, H. Lipson, Visually debugging restricted boltzmann machine training with a 3D example, in *Representation Learning Workshop, 29th International Conference on Machine Learning* (Edinburgh, Scotland, 26 June 2012).
45. K. Volzhenin, J. P. Changeux, G. Dumas, Multilevel development of cognitive abilities in an artificial neural network. *Proc. Natl. Acad. Sci. U.S.A.* **119**, e2201304119 (2022).
46. J. Mei, R. Meshkinnejad, Y. Mohsenzadeh, Effects of neuromodulation-inspired mechanisms on the performance of deep neural networks in a spatial learning task. *iScience* **26**, 106026 (2023).
47. S. Rudy, A. Alla, S. L. Brunton, J. N. Kutz, Data-driven identification of parametric partial differential equations. *SIAM J. Appl. Dyn. Syst.* **18**, 643–660 (2019).
48. P. U. Diehl, M. Cook, Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.* **9**, 99 (2015).
49. H. Hazan, D. Saunders, D. T. Sanghavi, H. Siegelmann, R. Kozma, Unsupervised learning with self-organizing spiking neural networks, in *2018 International Joint Conference on Neural Networks (IJCNN)*, 8 to 13 July 2018, Rio de Janeiro (IEEE, 2018), pp. 1–6.
50. Y. Bengio, N. Leonard, A. C. Courville, Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv:1305.2982 (2013).
51. D. Huh, T. J. Sejnowski, Gradient descent for spiking neural networks. *Adv. Neural Info. Process. System.* **31**, (2018).
52. Y. Bengio, T. Mesnard, A. Fischer, S. Zhang, Y. Wu, STDP-compatible approximation of backpropagation in an energy-based model. *Neural Comput.* **29**, 555–577 (2017).

Acknowledgments: We thank H. Liu for refining Fig. 5 and X. Song for the discussion. **Funding:** This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences (grant no. XDA27010404), the Shanghai Municipal Science and Technology Major Project (grant no. 2021SHZDZX), the Lingang Laboratory Fund (grant no. LG202105-01-07), and the Youth Innovation Promotion Association of CAS. **Author contributions:** B.X., M.P., and T.Z. designed the study. T.Z., S.J., and X.C. performed the experiments. T.Z., B.X., C.L., and M.P. performed the analyses and wrote the paper together. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials. All source codes can be downloaded from https://doi.org/10.5281/zenodo.8037309.

Submitted 16 April 2023
Accepted 27 July 2023
Published 25 August 2023
10.1126/sciadv.adi2947

A brain-inspired algorithm that mitigates catastrophic forgetting of artificial and spiking neural networks with low computational cost

Tielin Zhang, Xiang Cheng, Shuncheng Jia, Chengyu T Li, Mu-ming Poo, and Bo Xu

Sci. Adv. **9** (34), eadi2947. DOI: 10.1126/sciadv.adi2947