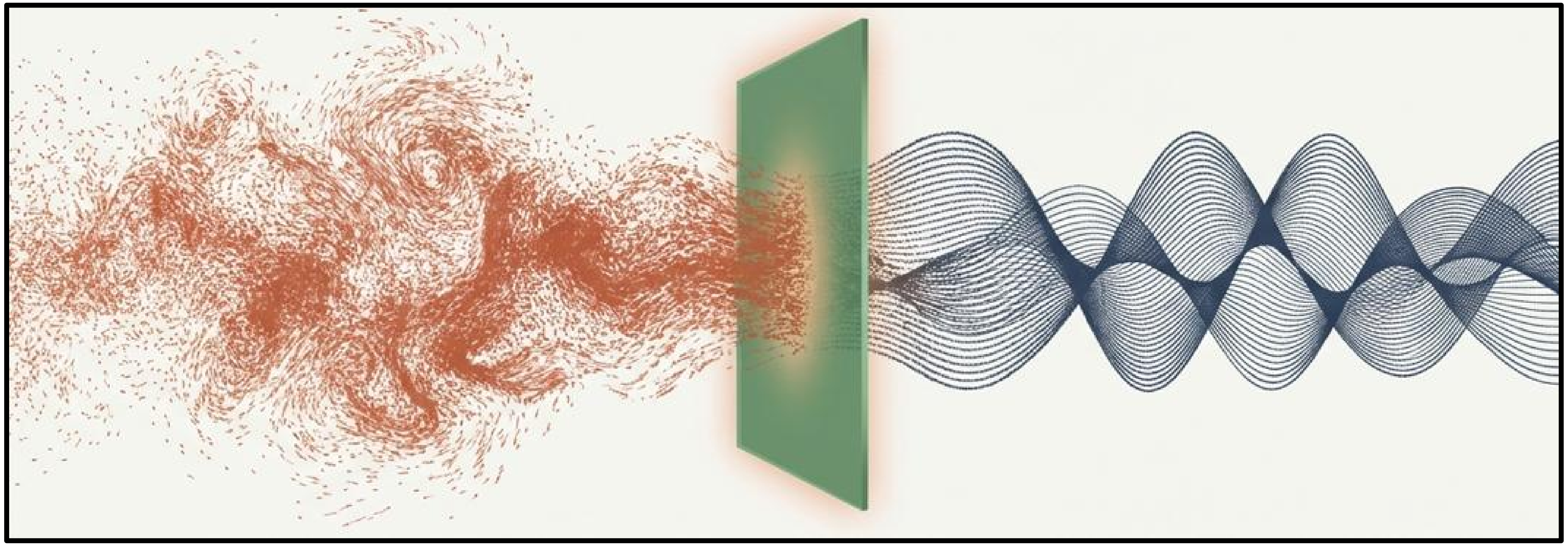
Global Credit Assignment via Dynamical Criticality



Notation

Notation Table: Credit Assignment via Dynamical Criticality

Symbol	Interpretation and Description
t, T	Discrete time step and total sequence length.
H, D, K	Dimensions of hidden state, input, and output, respectively.
$\mathbf{u}_t \in \mathbb{R}^D$	External input vector at time step t.
$\mathbf{x}_t, x_t^{(i)}$	Pre-activation (membrane potential) vector and its i-th component at time t.
$\mathbf{h}_t, h_t^{(i)}$	Hidden state (post-activation) vector and its i-th component, where $\mathbf{h}_t = \phi(\mathbf{x}_t)$.
$\mathbf{o}_t, \mathbf{p}_t$	Network output (logits) and Softmax probability distribution at time step t.
$\mathcal{L}_t, \mathcal{L}_{\text{total}}$	Instantaneous loss and total loss, defined as $\mathcal{L}_{\text{total}} = \sum_{t=1}^{T} \bar{w}_t \mathcal{L}_t$.
$\partial \mathcal{L}_t / \partial \mathbf{x}_t$	Sensitivity of the instantaneous loss to pre-activations (available within the current step).
$\boldsymbol{\delta}_t, \delta_t^{(i)}$	$\boldsymbol{\delta}_t \equiv \partial \mathcal{L}_{\text{total}} / \partial \mathbf{x}_t$: The true BPTT gradient (teaching signal) for pre-activations.
$\tilde{\boldsymbol{\delta}}_t, \tilde{\delta}_t^{(i)}$	Online, non-recursive local approximation of the teaching signal $\boldsymbol{\delta}_t$ (COLA).
ϕ, ϕ'	Activation function and its first derivative.
$\mathbf{U}_t$	Notation simplification: Diagonal matrix of derivatives $\mathbf{U}_t \equiv \text{diag}(\phi'(\mathbf{x}_t))$.
$\mathbf{W}_{xh}, \mathbf{W}_{hh}, \mathbf{W}_{hy}$	Input-to-hidden, recurrent, and hidden-to-output weight matrices.
$\mathbf{b}_h, \mathbf{b}_y$	Bias vectors for the hidden and output layers.
α_i	Temporal self-coupling gain (regression coefficient for $\delta_{t+1}^{(i)} \approx \alpha_i \delta_t^{(i)}$), estimated online.
μ_i	Equivalent loop gain for unit i (online closed-form estimate, no gradient required).
$z_t^{(i)}$	Aggregate slope-free feedback signal: $z_t^{(i)} \triangleq \sum_{j=1}^{H} W_{hh}^{ji} \alpha_j \delta_t^{(j)}$.
$\lambda_{\max}$	Maximum Lyapunov Exponent (MLE), used to quantify dynamical stability and criticality.
g, g_{crit}	Scalar gain for recurrent weights and the optimal gain for edge-of-chaos initialization.
ρ_α, ρ_μ	Exponential Moving Average (EMA) decay rates for tracking statistics α and μ.

Table 1: Summary of symbols and notation used in the COLA framework.

Introduction

Background：Temporal Credit Assignment is a central challenge for RNNs processing long sequences.

The Dilemma of Existing Methods:

- BPTT：Precise gradients but requires unrolling the computational graph; memory scales linearly with sequence length (O(T)), prone to vanishing/exploding gradients.

- Local/Online Learning：Computationally efficient (O(1) memory) but introduces significant bias, performing far worse than BPTT.

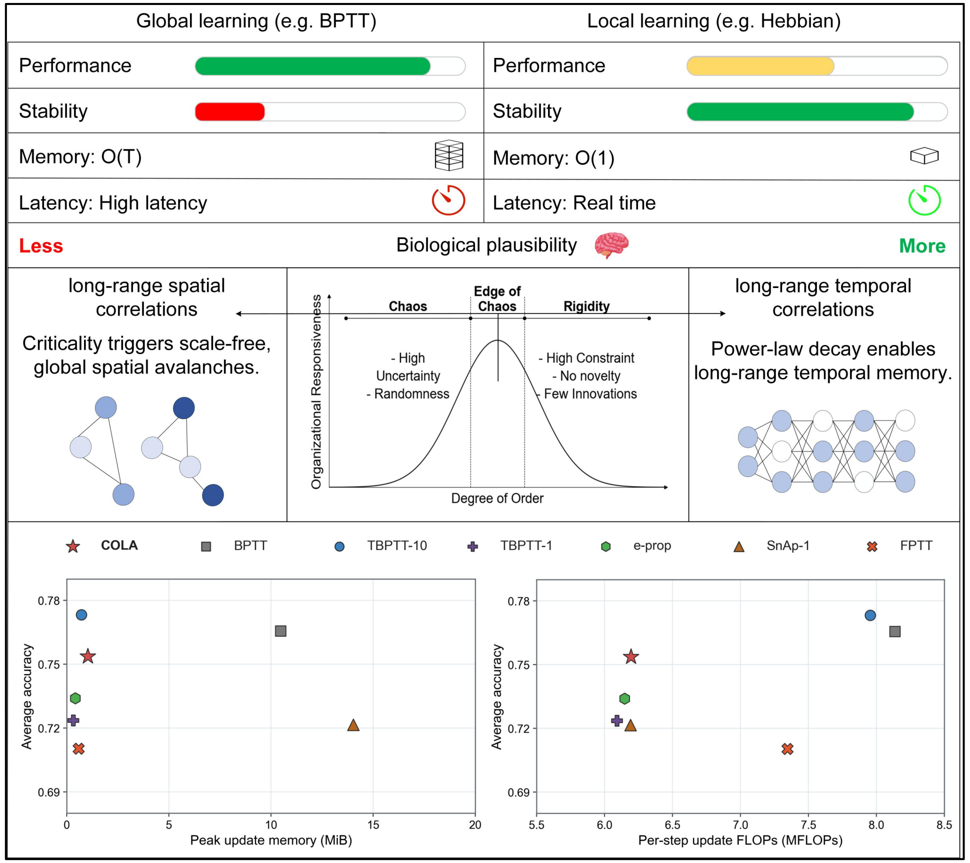
Method

COLA (Criticality-driven Online Local Alignment): A rigorous online local learning rule.

Mechanism: Uses Dynamical Criticality as an enabler to transform BPTT's temporal recursion into a local approximation at the same timestep.

Key Advantages:

- Constant Memory: Requires only O(H) auxiliary states; activation memory is O(1).
- High Performance: Matches BPTT in benchmarks and shows superior stability in long-period tasks.
- Linear Time Complexity: O(TP) temporal overhead.



Method

$$\mathbf{x}_t = \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{xh}\mathbf{u}_t + \mathbf{b}_h,$$
$$\mathbf{h}_t = \phi(\mathbf{x}_t),$$
$$\mathbf{o}_t = \mathbf{W}_{hy}\mathbf{h}_t + \mathbf{b}_y,$$
$$\mathbf{p}_t = \mathrm{softmax}(\mathbf{o}_t),$$

$$\frac{\partial \mathcal{L}_t}{\partial \mathbf{o}_t} = \bar{w}_t\,(\mathbf{p}_t - \mathbf{y}_t^{\star}).$$

$$\frac{\partial \mathcal{L}_t}{\partial \mathbf{x}_t} = \frac{\partial \mathbf{h}_t}{\partial \mathbf{x}_t} \odot \mathbf{W}_{hy}^{\top} \frac{\partial \mathcal{L}_t}{\partial \mathbf{o}_t}.$$

$$\frac{\partial \mathcal{L}}{\partial h_t^{(i)}} = \frac{\partial \mathcal{L}_t}{\partial h_t^{(i)}} + \frac{\partial \mathcal{L}_{>t}}{\partial h_t^{(i)}}, \qquad \mathcal{L}_{>t} \triangleq \sum_{k=t+1}^{T} \mathcal{L}_k.$$

$$\frac{\partial \mathcal{L}_{>t}}{\partial h_t^{(i)}} = \sum_{m=1}^{H} \frac{\partial \mathcal{L}}{\partial x_{t+1}^{(m)}} \frac{\partial x_{t+1}^{(m)}}{\partial h_t^{(i)}} = \sum_{m=1}^{H} \delta_{t+1}^{(m)} W_{hh}^{mi}.$$

$$\delta_t^{(i)} = \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}} \left(\frac{\partial \mathcal{L}_t}{\partial h_t^{(i)}} + \sum_{m=1}^{H} W_{hh}^{mi}\, \delta_{t+1}^{(m)} \right)$$

Method

$$\mathbf{J}_t \triangleq \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} = \mathrm{diag}(\frac{\partial \mathbf{h}_t}{\partial \mathbf{x}_t})\mathbf{W}_{hh}. \qquad \lambda_{\max} \approx \frac{1}{T}\log \sigma_{\max}\left(\prod_{t=1}^{T} \mathbf{J}_t\right) \qquad \mathbf{W}_{hh}(g) = g\,\widetilde{\mathbf{W}}_{hh}, \quad \widetilde{\mathbf{W}}_{hh,ij} \sim \mathcal{N}\left(0, \frac{1}{H}\right) \text{ (i.i.d.)}.$$

Two Core Hypotheses: Under the "edge-of-chaos" critical state, neural signals exhibit two key properties

Assumption 1: long-range temporal correlations: On short time scales in a stable near-critical regime, the BPTT pre-activation gradient $\delta_t^{(i)}$ continues approximately by a per-unit scalar

$$\delta_{t+1}^{(i)} \approx \alpha_i \delta_t^{(i)}$$

Assumption 2: long-range spatial correlations: In a stable near-critical regime, the within-step teaching signal is assumed to be spatially coherent. Specifically, for a fixed unit i, the error signal of any unit j is approximately proportional to that of unit i:

$$\delta_t^{(j)} \approx \beta_{ji} \delta_t^{(i)}$$

$\boldsymbol{\delta}_t, \delta_t^{(i)}$ $\qquad$ $\boldsymbol{\delta}_t \equiv \partial \mathcal{L}_{\text{total}} / \partial \mathbf{x}_t$: The true BPTT gradient

Method

Mathematical Derivation: From BPTT to COLA

ϕ, ϕ'	Activation function and its first derivative.
$\mathbf{W}_{xh}, \mathbf{W}_{hh}, \mathbf{W}_{hy}$	Input-to-hidden, recurrent, and hidden-to-output weight matrices.

1) The original BPTT form

$$\boldsymbol{\delta}_t = \frac{\partial \mathcal{L}_t}{\partial \mathbf{x}_t} + \mathbf{U}_t \mathbf{W}_{hh}^{\top} \boldsymbol{\delta}_{t+1}. \qquad \mathbf{U}_t \equiv \operatorname{diag}(\phi'(x_t^{(1)}), \ldots, \phi'(x_t^{(H)}))$$

2) By using assumption 1, we get

$$\boldsymbol{\delta}_{t+1} = \mathbf{D}_\alpha \boldsymbol{\delta}_t \qquad \mathbf{D}_\alpha \equiv \operatorname{diag}(\alpha_1, \ldots, \alpha_H)$$

3) Combine 1) and 2)

$$\boldsymbol{\delta}_t \approx \frac{\partial \mathcal{L}_t}{\partial \mathbf{x}_t} + \mathbf{A}_t \boldsymbol{\delta}_t, \qquad \mathbf{A}_t \equiv \mathbf{U}_t \mathbf{W}_{hh}^{\top} \mathbf{D}_\alpha.$$

4) For a fixed unit i, the ith component reads

$$\delta_t^{(i)} \approx \frac{\partial \mathcal{L}_t}{\partial x_t^{(i)}} + \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}} \sum_{j=1}^{H} W_{hh}^{ji} \alpha_j \delta_t^{(j)}.$$

Method

Mathematical Derivation: From BPTT to COLA

4) $$\delta_t^{(i)} \approx \frac{\partial \mathcal{L}_t}{\partial x_t^{(i)}} + \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}} \sum_{j=1}^{H} W_{hh}^{ji}\, \alpha_j\, \delta_t^{(j)}.$$

5) By using assumption 2, we get

$$\sum_{j=1}^{H} W_{hh}^{ji}\, \alpha_j\, \delta_t^{(j)} \approx \sum_{j=1}^{H} W_{hh}^{ji}\, \alpha_j \left(\beta_{ji}(t)\, \delta_t^{(i)}\right) = \left(\sum_{j=1}^{H} W_{hh}^{ji}\, \alpha_j\, \beta_{ji}(t)\right) \delta_t^{(i)}.$$

6) Define μ_i

$$\mu_t^{(i)} \approx \sum_{j=1}^{H} W_{hh}^{ji}\, \alpha_j\, \beta_{ji}(t).$$

7) From 4), 5) and 6)

$$\delta_t^{(i)} \approx \frac{\partial \mathcal{L}_t}{\partial x_t^{(i)}} + \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}}\, \mu_t^{(i)}\, \delta_t^{(i)}$$

8) By transposing terms in 7)

$$\tilde{\delta}_t^{(i)} \approx \frac{\partial \mathcal{L}_t / \partial x_t^{(i)}}{1 - \mu_i \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}}}$$

Method

Closed-form online estimation of α_i and μ_i and Online parameter updates $\widetilde{\delta}_t^{(i)} \approx \alpha_i \widetilde{\delta}_{t-1}^{(i)}$

To estimate α_i, the teaching signal is treated as a local autoregressive (AR(1)) model, $\widetilde{\delta}_t^{(i)} \approx \alpha_i \widetilde{\delta}_{t-1}^{(i)}$, and use the least-squares solution

$$\alpha_i^\star = \frac{\mathbb{E}[\widetilde{\delta}_t^{(i)} \widetilde{\delta}_{t-1}^{(i)}]}{\mathbb{E}[(\widetilde{\delta}_{t-1}^{(i)})^2]},$$

estimation of α_i

estimation of μ_i

$$\frac{\partial \mathcal{L}_t / \partial x_t^{(i)}}{1 - \mu_i \frac{\partial h_t^{(i)}}{\partial x_t^{(i)}}} \approx \alpha_i \frac{\partial \mathcal{L}_{t-1} / \partial x_{t-1}^{(i)}}{1 - \mu_i \phi'(x_{t-1}^{(i)})} \qquad \mu_i^{(t)} = \frac{\alpha_i \frac{\partial \mathcal{L}_{t-1}}{\partial x_{t-1}^{(i)}} - \frac{\partial \mathcal{L}_t}{\partial x_t^{(i)}}}{\alpha_i \phi'(x_t^{(i)}) \frac{\partial \mathcal{L}_{t-1}}{\partial x_{t-1}^{(i)}} - \phi'(x_{t-1}^{(i)}) \frac{\partial \mathcal{L}_t}{\partial x_t^{(i)}}}$$

How to update

Given $\widetilde{\delta}_t$, COLA updates recurrent parameters online with the same outer-product structure as BPTT:

$$\Delta \mathbf{W}_{hh} = -\eta \, \widetilde{\boldsymbol{\delta}}_t \mathbf{h}_{t-1}^\top,$$
$$\Delta \mathbf{W}_{xh} = -\eta \, \widetilde{\boldsymbol{\delta}}_t \mathbf{u}_t^\top,$$
$$\Delta \mathbf{b}_h = -\eta \, \widetilde{\boldsymbol{\delta}}_t.$$

while the readout parameters use the exact instantaneous gradient:

$$\Delta \mathbf{W}_{hy} = -\eta \left(\frac{\partial \mathcal{L}_t}{\partial \mathbf{o}_t} \right) \mathbf{h}_t^\top,$$
$$\Delta \mathbf{b}_y = -\eta \frac{\partial \mathcal{L}_t}{\partial \mathbf{o}_t}.$$

Method

How α_i *and* μ_i truly update

Step 1:

$$\alpha_i^\star = \frac{\mathbb{E}[\widetilde{\delta}_t^{(i)}\widetilde{\delta}_{t-1}^{(i)}]}{\mathbb{E}[(\widetilde{\delta}_{t-1}^{(i)})^2]},$$

Step 2:

$$\begin{aligned} S_i(t) &\leftarrow \rho_\alpha S_i(t-1) + (1-\rho_\alpha)\,\widetilde{\delta}_t^{(i)}\widetilde{\delta}_{t-1}^{(i)}, \\ Q_i(t) &\leftarrow \rho_\alpha Q_i(t-1) + (1-\rho_\alpha)\,(\widetilde{\delta}_{t-1}^{(i)})^2, \\ \alpha_i(t) &\leftarrow \mathrm{clip}\left(\frac{S_i(t)}{Q_i(t)+\varepsilon_\alpha}, \alpha_{\min}, \alpha_{\max}\right), \end{aligned}$$

Step 1:

$$\mu_i = \frac{\sum_{\tau=1}^t \omega_\tau A_i^\tau B_i^\tau}{\sum_{\tau=1}^t \omega_\tau (A_i^\tau)^2},$$

Step 2:

$$\begin{aligned} S_{A^2}^{(i)}(t) &\leftarrow \rho_\mu S_{A^2}^{(i)}(t-1) + (1-\rho_\mu)(A_t^{(i)})^2, \\ S_{AB}^{(i)}(t) &\leftarrow \rho_\mu S_{AB}^{(i)}(t-1) + (1-\rho_\mu)A_t^{(i)}B_t^{(i)}, \\ \mu_i(t) &\leftarrow \Pi\left[\frac{S_{AB}^{(i)}(t)}{S_{A^2}^{(i)}(t)+\varepsilon_\mu}\right], \end{aligned}$$

Method

I. ConvRNN Extension and Benchmarks

I.1. Convolutional extension: ConvRNN

The same construction extends to convolutional recurrent networks by treating each hidden channel as a unit and maintaining per-channel scalar statistics. A convolutional neural network (CNN) encoder maps an input image $\mathbf{I}$ to a feature map

$$\mathbf{f} = \mathrm{Enc}(\mathbf{I}). \tag{59}$$

ConvRNN maintains a hidden feature map $\mathbf{H}_t \in \mathbb{R}^{C_h \times H' \times W'}$; global average pooling (GAP) over spatial positions is used to obtain a channel vector readout:

$$\mathbf{X}_t = \mathrm{Conv}_{hh}(\mathbf{H}_{t-1}) + \mathrm{Conv}_{xh}(\mathbf{f}) + \mathbf{b}_h, \tag{60}$$

$$\mathbf{H}_t = \tanh(\mathbf{X}_t),$$

$$\mathbf{z}_t = \mathrm{GAP}(\mathbf{H}_t) \in \mathbb{R}^{C_h},$$

$$\mathbf{o}_t = \mathbf{W}_{hy}\mathbf{z}_t + \mathbf{b}_y. \tag{61}$$

Losses and step weights follow Eq. (3).

Let $\mathbf{U}_t = 1 - \mathbf{H}_t \odot \mathbf{H}_t$ denote elementwise slopes. Let $\mathbf{B}_t$ be the spatial broadcast of $\mathbf{W}_{hy}^\top \frac{\partial \mathcal{L}_t}{\partial \mathbf{o}_t} \in \mathbb{R}^{C_h}$, and maintain a per-channel loop gain $\boldsymbol{\mu} \in \mathbb{R}^{C_h}$ (broadcast spatially). The teaching signal is

$$\widetilde{\boldsymbol{\Delta}}_t = (\mathbf{U}_t \odot \mathbf{B}_t) \oslash (\mathbf{1} - \boldsymbol{\mu} \odot \mathbf{U}_t), \tag{62}$$

Method

Mathematical Derivation: From BPTT to COLA

$$i_{t,j} = \sum_{k=1}^{N_{l-1}} W_{\mathrm{in},jk}\, x_{t,k} + \sum_{k=1}^{N_l} W_{\mathrm{rec},jk}(g)\, M_{jk}\, s_{t-1,k},$$

$$v_{t,j} = \alpha_j v_{t-1,j} - \alpha_j s_{t-1,j} + (1-\alpha_j) i_{t,j}, \qquad \text{a)}$$

$$s_{t,j} = H(v_{t,j} - \vartheta), \qquad j = 1, \ldots, N_l.$$

1) The original BPTT form

$$\frac{\partial \mathcal{L}}{\partial v_{t,j}} = \psi_{t,j} \frac{\partial \mathcal{L}}{\partial s_{t,j}} + \alpha_j \frac{\partial \mathcal{L}}{\partial v_{t+1,j}} - \alpha_j \psi_{t,j} \frac{\partial \mathcal{L}}{\partial v_{t+1,j}} + \psi_{t,j} \sum_{k=1}^{N} \frac{\partial \mathcal{L}}{\partial i_{t+1,k}} \overline{W}_{\mathrm{rec},kj}(g),$$

2) By using assumption 1 and a), we get

$$\frac{\partial \mathcal{L}}{\partial v_{t+1,j}} \approx \hat{\alpha}_j \frac{\partial \mathcal{L}}{\partial v_{t,j}}. \qquad \frac{\partial \mathcal{L}}{\partial i_{t+1,j}} = (1-\alpha_j) \frac{\partial \mathcal{L}}{\partial v_{t+1,j}},$$

3) Combine 1) and 2)

$$\frac{\partial \mathcal{L}}{\partial v_{t,j}} \approx \psi_{t,j} \frac{\partial \mathcal{L}}{\partial s_{t,j}} + \alpha_j (1-\psi_{t,j}) \hat{\alpha}_j \frac{\partial \mathcal{L}}{\partial v_{t,j}} + \psi_{t,j} \sum_{k=1}^{N} (1-\alpha_k) \hat{\alpha}_k \frac{\partial \mathcal{L}}{\partial v_{t,k}} \overline{W}_{\mathrm{rec},kj}(g),$$

Method

Mathematical Derivation: From BPTT to COLA

3) $$\frac{\partial \mathcal{L}}{\partial v_{t,j}} \approx \psi_{t,j}\frac{\partial \mathcal{L}}{\partial s_{t,j}} + \alpha_j(1-\psi_{t,j})\hat{\alpha}_j\frac{\partial \mathcal{L}}{\partial v_{t,j}} + \psi_{t,j}\sum_{k=1}^{N}(1-\alpha_k)\hat{\alpha}_k\frac{\partial \mathcal{L}}{\partial v_{t,k}}\overline{W}_{\text{rec},kj}(g),$$

4) By using assumption 2, we get

$$z_{t,j} \approx \alpha_j(1-\psi_{t,j})\hat{\alpha}_j\frac{\partial \mathcal{L}}{\partial v_{t,j}} + \psi_{t,j}\sum_{k=1}^{N}(1-\alpha_k)\hat{\alpha}_k\beta_{kj}(t)\overline{W}_{\text{rec},kj}(g)\frac{\partial \mathcal{L}}{\partial v_{t,j}}$$

5) Define γ_i

$$\nu_{t,j} \triangleq \alpha_j(1-\psi_{t,j})\hat{\alpha}_j + \psi_{t,j}\sum_{k=1}^{N}(1-\alpha_k)\hat{\alpha}_k\beta_{kj}(t)\overline{W}_{\text{rec},kj}(g),$$

6) From 3), 4) and 5)

$$\frac{\partial \mathcal{L}}{\partial v_{t,j}} \approx \psi_{t,j}\frac{\partial \mathcal{L}}{\partial s_{t,j}} + \nu_{t,j}\frac{\partial \mathcal{L}}{\partial v_{t,j}},$$

7) By transposing terms in 6)

$$\frac{\partial \ell_t}{\partial v_{t,j}} \approx \frac{\psi_{t,j}\left(\frac{\partial \ell_t}{\partial s_{t,j}}\right)_{\text{local}}}{1-\mu_j\psi_{t,j}}.$$

Why $\nu_{t,j} \approx \mu_j\psi_{t,j}$,

When far from the threshold: $\psi_{t,j}$ is very small, and the local teaching signal should be virtually shut off.

When near the threshold: $\psi_{t,j}$ is active.

Cost analysis

Table 1. Asymptotic costs for a vanilla RNN. Here, T denotes the sequence length and H the number of hidden units. We report the recurrent-dominated scaling and suppress input/readout terms shared across methods. "Activation memory" refers to the storage required for hidden states that scales with the sequence length.

Method	Time (per sequence)	Activation memory	Extra state (excluding parameters)
Full BPTT (Werbos, 1990)	$\mathcal{O}(TH^2)$	$\mathcal{O}(TH)$	none
TBPTT (truncation τ)	$\mathcal{O}(TH^2)$	$\mathcal{O}(\tau H)$	none
RTRL (Williams & Zipser, 1989)	$\mathcal{O}(TH^4)$	$\mathcal{O}(1)$	$\mathcal{O}(H^3)$
UORO (Tallec & Ollivier, 2017)	$\mathcal{O}(TH^2)$	$\mathcal{O}(1)$	rank-one factors $\mathcal{O}(H^2)$
SnAp-1 (Menick et al., 2021)	$\mathcal{O}(TH^2)$	$\mathcal{O}(1)$	one-step sparse influences $\mathcal{O}(H^2)$
e-prop (Bellec et al., 2020)	$\mathcal{O}(TH^2)$	$\mathcal{O}(1)$	eligibility traces $\mathcal{O}(H^2)$
FPTT (Kag & Saligrama, 2021)	$\mathcal{O}(TH^2)$	$\mathcal{O}(1)$	shadow parameter copy $\mathcal{O}(H^2)$
Ours (COLA)	$\mathcal{O}(TH^2)$	$\mathcal{O}(1)$	α, μ stats $\mathcal{O}(H)$

Experiment

Task	BPTT	TBPTT-1	TBPTT-10	e-prop	FPTT	UORO	SnAp-1	**COLA**
Adding	0.1426	0.0863	0.0991	0.0813	0.1356	0.1444	0.0860	**0.0059**
MSE↓	± 0.0326	± 0.0031	± 0.0013	± 0.0031	± 0.0336	± 0.0150	± 0.0034	**± 0.0026**
Lorenz	0.6841	0.2019	0.6749	0.3602	0.8038	0.1395	0.2025	**0.0058**
MSE↓	± 0.0873	± 0.0845	± 0.0163	± 0.0201	± 0.0250	± 0.1072	± 0.0676	**± 0.0003**
PTB-Char	2.8695	1.8316	2.2272	2.0852	2.3184	2.2279	**1.8279**	1.8352
Loss↓	± 0.0203	± 0.0040	± 0.0029	± 0.0013	± 0.0057	± 0.0063	**± 0.0041**	± 0.0056
WikiText-2	3.3620	2.2833	2.8780	2.5851	3.0499	2.6225	**2.2793**	2.4174
Loss↓	± 0.0165	± 0.0058	± 0.0201	± 0.0073	± 0.0307	± 0.0653	**± 0.0054**	± 0.0035
Row-MNIST	0.9713	0.9343	**0.9734**	0.9565	0.9481	0.3060	0.9403	0.9544
Acc↑	± 0.0007	± 0.0011	**± 0.0020**	± 0.0005	± 0.0037	± 0.0193	± 0.0047	± 0.0036
Row-CIFAR10	0.4545	0.3894	**0.4759**	0.3836	0.4021	0.2185	0.3825	0.4418
Acc↑	± 0.0108	± 0.0162	**± 0.0130**	± 0.0026	± 0.0064	± 0.0057	± 0.0070	± 0.0083
UCI HAR	**0.8877**	0.8416	0.8799	0.8314	0.8367	0.5594	0.8420	0.8690
Acc↑	**± 0.0454**	± 0.0291	± 0.0390	± 0.0446	± 0.0171	± 0.0359	± 0.0134	± 0.0257

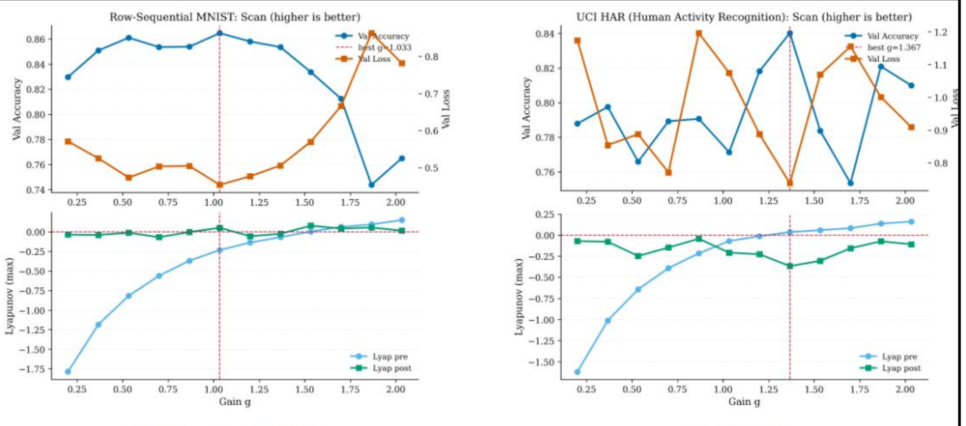
Task	BPTT	TBPTT-1	TBPTT-10	e-prop	FPTT	UORO	**COLA**
Fashion-MNIST	**0.8730**	0.8347	0.8685	0.8506	0.7941	0.5616	0.8467
Acc↑	**± 0.0031**	± 0.0066	± 0.0020	± 0.0084	± 0.0070	± 0.0124	± 0.0014
Permuted MNIST	0.9325	0.9114	**0.9340**	0.9174	0.8764	0.3918	0.9122
Acc↑	± 0.0030	± 0.0046	**± 0.0033**	± 0.0067	± 0.0071	± 0.0553	± 0.0027
DVS-Gesture	0.6678	0.6215	**0.6852**	0.6551	0.6377	0.1493	0.6597
Acc↑	± 0.0140	± 0.0309	**± 0.0145**	± 0.0244	± 0.0231	± 0.0874	± 0.0035
DVS-CIFAR10	0.4540	0.4300	**0.4680**	0.4263	0.3697	0.1360	0.4410
Acc↑	± 0.0056	± 0.0213	**± 0.0078**	± 0.0105	± 0.0093	± 0.0144	± 0.0141

Experiment

Dataset	Category	Training method	Network architecture	Testing accuracy
N-MNIST[26]	Ours	BPTT	RLIF	98.33 ± 0.04%
		pp-prop	RLIF	98.25 ± 0.03%
		CoLA	RLIF	**98.54%**
		BPTT	RadLIF	98.29 ± 0.02%
		pp-prop	RadLIF	98.40 ± 0.03%
		CoLA	RadLIF	**93.17%**
	Online	ETLP[31]	RLIF	94.30%
		e-prop[31]	RLIF	97.90%
		OSTL[9]	sSNU	96.80 ± 0.17%
SHD[25]	Ours	BPTT	RLIF	94.01 ± 0.12%
		pp-prop	RLIF	93.93 ± 0.28%
		CoLA	RLIF	**94.32%**
		BPTT	RadLIF	94.72 ± 1.06%
		pp-prop	RadLIF	95.33 ± 0.11%
		CoLA	RadLIF	**95.45%**
		ETLP[31]	RLIF	78.71 ± 1.49%
		e-prop[31]	RLIF	80.79 ± 0.39%

Dataset	Category	Training method	Network architecture	Testing accuracy
	Offline	OTPE[11]	LIF	76.70 ± 0.70%
		e-prop[60]	TC-RLIF	80.57%
		OSTTP[32]	sNU	77.33 ± 0.8%
		S-TLLR[33]	RLIF	78.24 ± 1.84%
		EventProp[61]	RLIF	93.50 ± 0.70%
		BPTT[35]	DCLS-Delays	95.07 ± 0.24%
		BPTT[29]	RadLIF	94.62%
Gesture[24]	Ours	BPTT	RLIF	93.97 ± 0.15%
		pp-prop	RLIF	94.26 ± 0.32%
		CoLA	RadLIF	**93.38%**
		BPTT	EGRU	97.45 ± 0.27%
		pp-prop	EGRU	97.29 ± 0.16%
		CoLA	EGRU	**97.29%**
	Online	FPTT[34]	RLIF	92.13 ± 0.87%
		FPTT[34]	CNN	97.22%
		OTTT[10]	VGG-11	96.88%
	Offline	BPTT[62]	STS-ResNet	96.7%
		BPTT[63]	STL-SNN	97.01 ± 0.23%
		BPTT[37]	PLIF	97.57%
		BPTT[36]	VGGSNN	97.57%

Hypothesis Validation



(c) Row-sequential MNIST (d) UCI HAR

$$R^2 = 1 - \frac{\sum_{i,t}\left(\bar{\delta}_t^{\text{true},(i)} - \widehat{\bar{\delta}}_t^{(i)}\right)^2}{\sum_{i,t}\left(\bar{\delta}_t^{\text{true},(i)} - \bar{\delta}_{\text{win}}^{(i)}\right)^2},$$

$$EVR_1(X) \triangleq \frac{\sigma_1^2}{\sum_k \sigma_k^2}.$$

$$r_{\text{eff}}(X) \triangleq \frac{\left(\sum_k \sigma_k^2\right)^2}{\sum_k \sigma_k^4},$$

Quantity	**Low-rank**	**IID**
Temporal AR(1)+bias one-step R^2	0.9944 ± 0.0094	0.8089 ± 0.0334
Teaching-signal EVR_1	0.9999 ± 0.0001	0.9836 ± 0.0112
Teaching-signal effective rank	1.0001 ± 0.0002	1.0339 ± 0.0237

Task	Gain band	$\lambda_{\max}$ band	Temporal AR(1) R^2	EVR_1	Effective rank
Adding Task	0.6–1.0	-0.3307 to -0.0231	0.5783–0.8097	0.86–0.97	1.05–1.32
Lorenz Image	0.6–1.4	-1.0657 to -0.6956	0.6248–0.8247	0.56–0.72	1.92–2.39
Row-MNIST	0.6–1.0	-0.1967 to -0.0447	0.5408–0.7654	0.73–0.88	1.29–1.81
Row-CIFAR10	0.6–1.0	-0.3707 to -0.0047	0.5890–0.5940	0.72–0.73	1.33–1.79
UCI HAR	0.6–1.4	-0.3480 to -0.1027	0.7079–0.8270	0.69–0.77	1.74–2.35

Thank you