Online learning of long-range dependencies

Nicolas Zucchet*, Robert Meier*, Simon Schug*, Asier Mujika, João Sacramento

ETH zürich

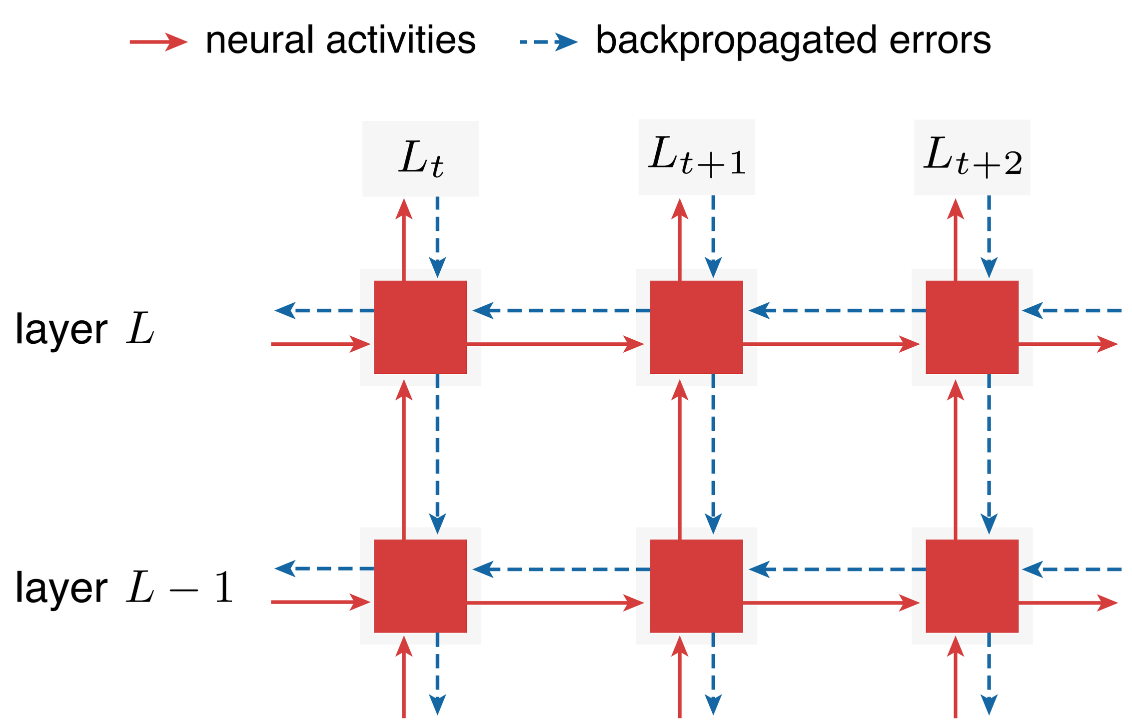
Paper | Code

Summary

We propose a new rule for learning RNNs **online** that

1. Leverages element-wise recurrence for accurate gradient estimation
2. Backpropagate instantaneous error signals across the network hierarchy
3. Scales to challenging tasks that require modeling long-range dependencies
4. Has same time complexity as backpropagation-through-time while only doubling memory in the forward pass
5. Helps understanding learning in the brain

The problem with backpropagation-through-time



Limitations:

- store entire trajectory of neural activities (O(T))
- non-causal
- process entire input before computing gradient

They become a problem when:

- moving to low memory hardware (brain / neuromorphic computing)
- we cannot wait the end of a sequence to update the parameters (e.g. RL)

Real-time recurrent learning

RTRL is forward-mode differentiation applied to RNNs

- activity update: $h_{t+1} = f(h_t, \theta)$
- sensitivity update: $\frac{\mathrm{d}h_{t+1}}{\mathrm{d}\theta} = \frac{\mathrm{d}h_{t+1}}{\mathrm{d}h_t}\frac{\mathrm{d}h_t}{\mathrm{d}\theta}$
- gradient calculation: $\frac{\mathrm{d}L}{\mathrm{d}\theta} = \sum_t \frac{\partial L_t}{\partial h_t}\frac{\mathrm{d}h_t}{\mathrm{d}\theta}$

Gradients can be calculated online + constant memory (w.r.t T)!
But O(n³) memory complexity and O(n⁴) operations
Our work: improved complexity by leveraging modularity

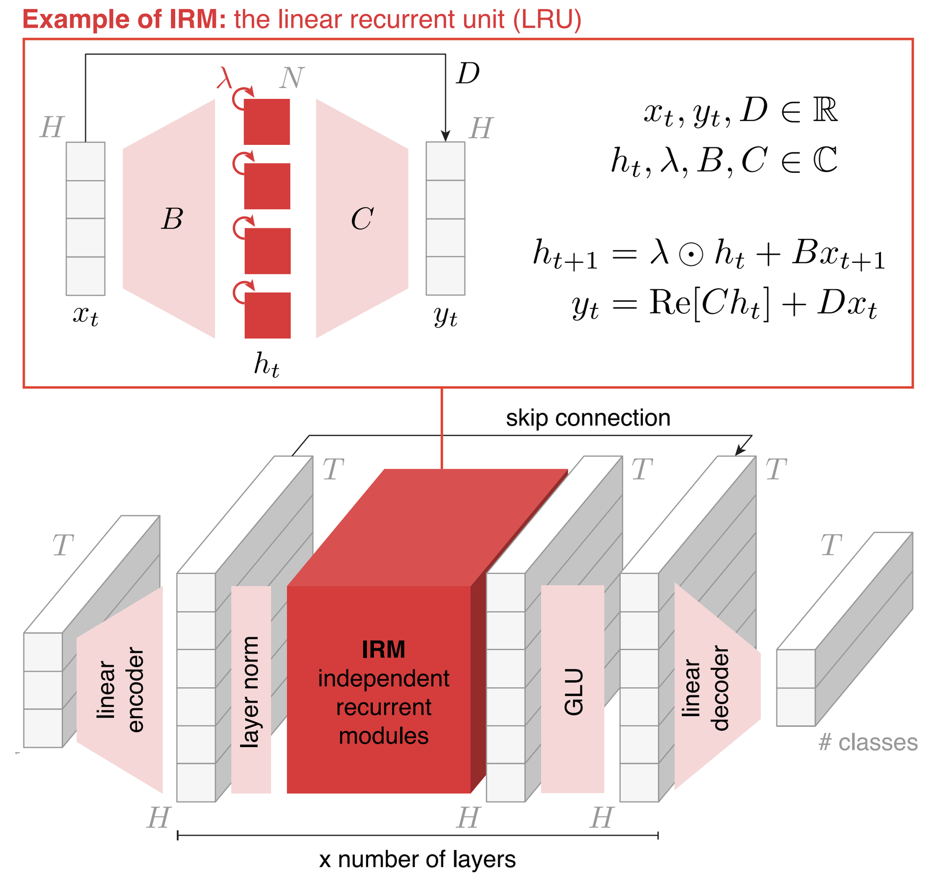
Wiliams and Zipser, *Neur. Comp.* 1989

Independent recurrent mechanisms

Previous work either:

1. approximates the sensitivity update to make RTRL tractable
2. remarks that element-wise recurrence makes RTRL tractable

Point 2 is not as limiting as it may seem! (deep SSM, LRU)



Sensitivities are now of the size of the parameters (O(n²)):

$$e^{\lambda}_{t+1} = \lambda \odot e^{\lambda}_t$$

$$\Delta\lambda \propto \sum_{t=1}^{T} \delta_t \odot e^{\lambda}_t$$

$$e^{B}_{t+1} = \mathrm{diag}(\lambda)e^{B}_t + 1x^{\top}_{t+1}$$

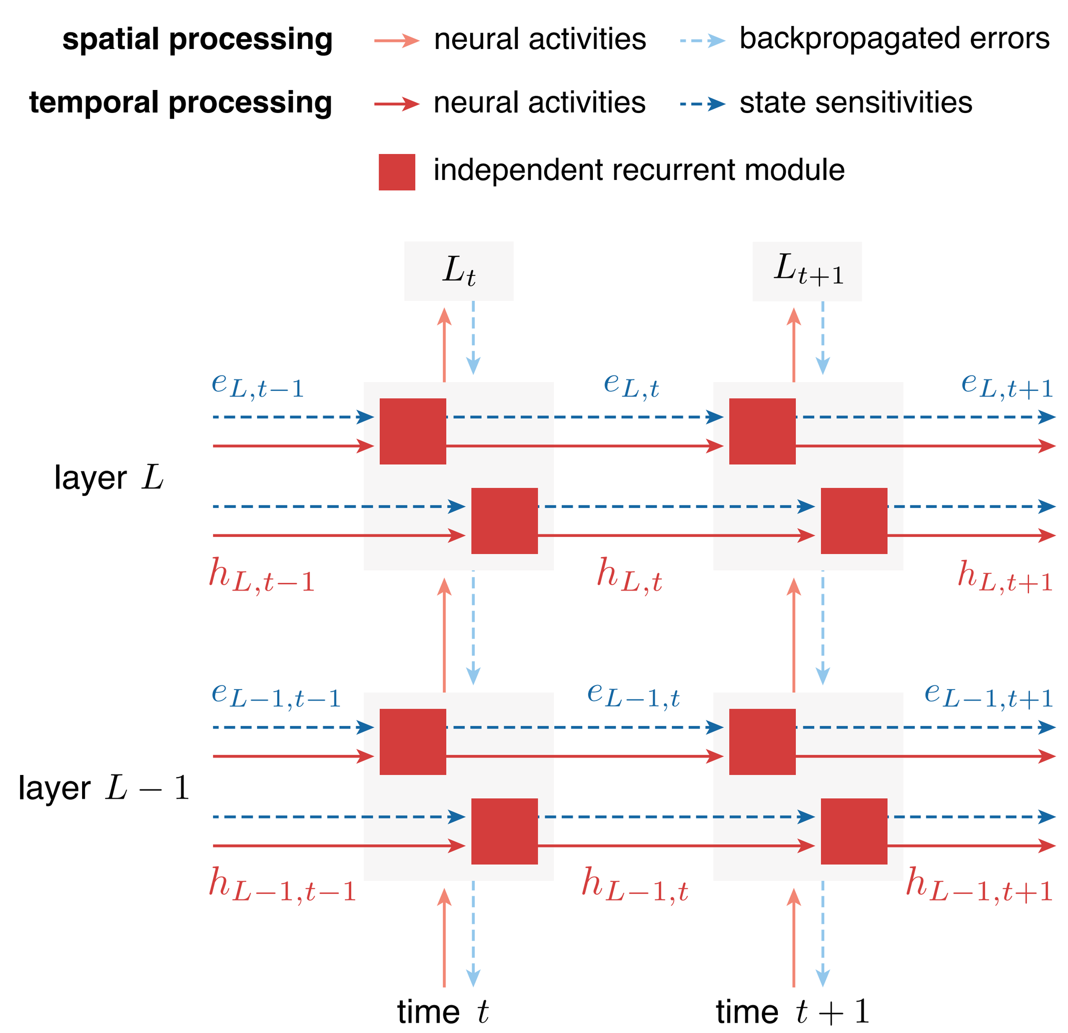
$$\Delta B \propto \sum_{t=1}^{T} \mathrm{diag}(\delta_t) \odot e^{B}_t$$

Independent recurrent modules may be a useful inductive bias for the brain to be able to learn online

Tallec and Ollivier, *ICLR* 2018; Murray, *eLife* 2019; Bellec, *Nat. Comm.* 2020; Menick et al., *ICLR* 2021; Mozer, *Comp. Sys.* 1989; Orvieto et al., *ICML* 2023

Spatial backpropagation across layers

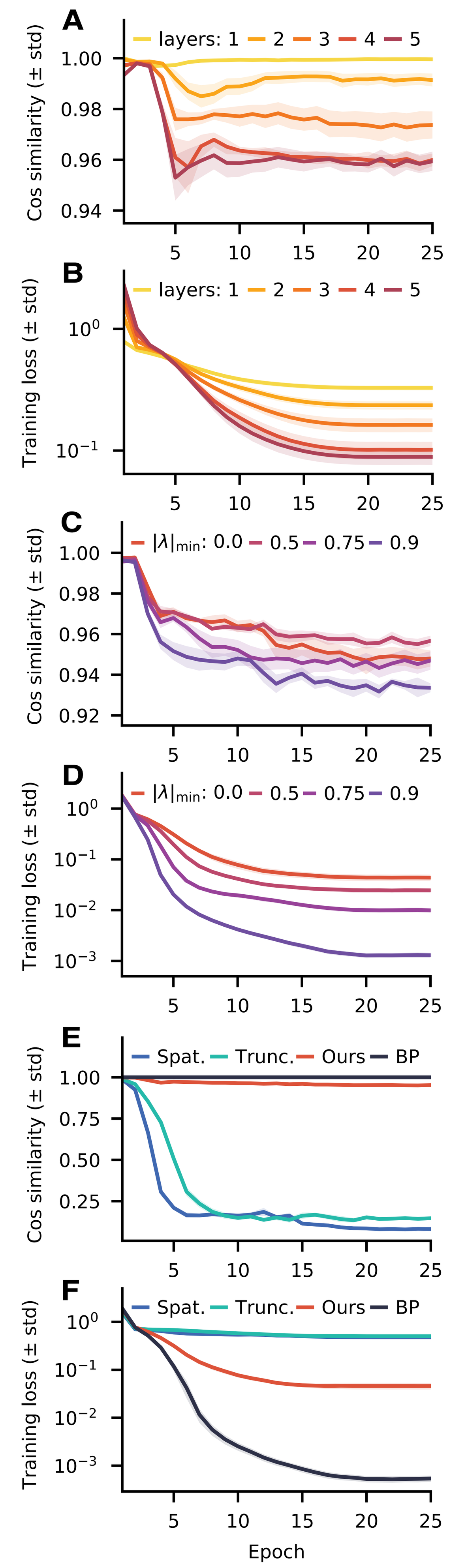
Stacking multiple layers is key to achieving good performance
We enable it by leveraging spatially backpropagated errors, i.e. approximate $\mathrm{d}_{h_t} L \approx \partial_{h_t} L_t$



Exact gradient for the parameters of the last layer

Approximate gradients for the rest

Understanding the bias in practice



Synthetic memory task (7-bits pattern of length 20 to remember)

Report training loss and cosine similarity with true gradient (averaged over layers)

A, B: vary the depth of the network
Bias increase with depth as approximation we make becomes cruder but still enough to benefit from depth

C, D: vary the initial recurrence eigenvalues
Bias increase as eigenvalues get closer to 1 but still benefit from it (for this task)

E, F: compare against other algorithms

- spatial backpropagation (online)
- 1-step truncated backprop. (online)
- exact gradient (BPTT, offline)

Additional experiments in the paper showing that IRMs improve online learning performance:

- On a linear RNN: approximate RTRL is lagging behind BPTT (more than ours)
- On a GRU: approximate RTRL perform competitively for 1 layer but does not benefit from depth

Results

Long-range arena benchmark, test accuracies reported below
Adjustments compared to traditional setting: no batch norm, loss at every timestep (averaging all the digits so far)

	sCIFAR	IMDB	ListOPS	sCIFAR (lin. RNN)
Spatial BP	58.20 ± 0.70	83.50 ± 0.20	32.02 ± 0.20	50.63 ± 0.23
1-step TBPTT	60.01 ± 1.26	84.04 ± 0.47	31.88 ± 0.59	50.53 ± 0.43
Ours / SnAp-1	79.59 ± 1.01	86.48 ± 0.41	37.62 ± 0.68	63.71 ± 0.33
BPTT	83.40 ± 1.54	87.69 ± 0.39	39.75 ± 0.17	65.23 ± 0.56