Self-supervised consolidation of latent dynamics for training recurrent neural networks at scale

Editors: List of editors' names

Abstract

Recurrent neural networks are often used as model organisms to study how populations of neurons perform behavioral tasks. While robustness in biology is typically linked to large networks with millions of neurons, training very large recurrent neural networks is slow and inefficient. Most computational studies therefore rely on much smaller models, limiting insights into scalability of learned representations. Here, we discuss a self-supervised framework in which a small, fast-learning part of the recurrent network is trained with backpropagation-through-time. Once trained, its latent dynamics are then consolidated into a much larger portion of the network, and then, the latter acquires task-specific input–output mappings. This two-step process parallels biological consolidation, where rapid learning in small circuits guides large-scale network organization. Using this approach, large recurrent networks develop stable, robust representations from sparse training signals, providing a biologically inspired path toward training scalable models of neural computation.

Keywords: Computational neuroscience, RNNs, self-supervised learning

1. Introduction

Brains can rapidly acquire new tasks and perform them reliably, even though individual neurons are noisy and variable (Churchland et al., 2012; Inagaki et al., 2022). A common view is that such robustness arises from large populations of neurons (Dinc et al., 2025). RNNs have become important model organisms for probing these principles (Sussillo and Barak, 2013; Dubreuil et al., 2022), but training them at scale can become slow and unstable (Pascanu et al., 2013). Consequently, most studies focus on small networks with hundreds to thousand of neurons (Masse et al., 2019; Yang et al., 2019; Dubreuil et al., 2022), limiting insights into the scalability and robustness of the learned representations. Here, inspired by biological consolidation (Dudai et al., 2015) and knowledge transfer (Hinton et al., 2015), we propose a self-supervised framework.

2. Results

2.1. Training small rank-one RNNs to perform a 1-bit flip-flop task

We illustrate the proposed framework using a canonical example: recurrent neural networks (RNNs) trained on the 1-bit flip-flop task (Fig. 1**A**). In this task, the network receives brief ±1 input pulses at sparse times and must continuously output the identity of the most recently observed pulse. An RNN is defined by its input weights ($\mathbf{W}^{\mathrm{in}}$), recurrent weights and biases ($\mathbf{W}^{\mathrm{rec}}, \boldsymbol{b}$), and output weights and biases ($\mathbf{W}^{\mathrm{out}}, \boldsymbol{b}^{\mathrm{out}}$), with dynamics

$$\tau \dot{\boldsymbol{r}}(t) = -\boldsymbol{r}(t) + \tanh(\boldsymbol{W}^{\mathrm{rec}}\boldsymbol{r}(t) + \boldsymbol{W}^{\mathrm{in}}u(t) + \boldsymbol{b}), \tag{1}$$

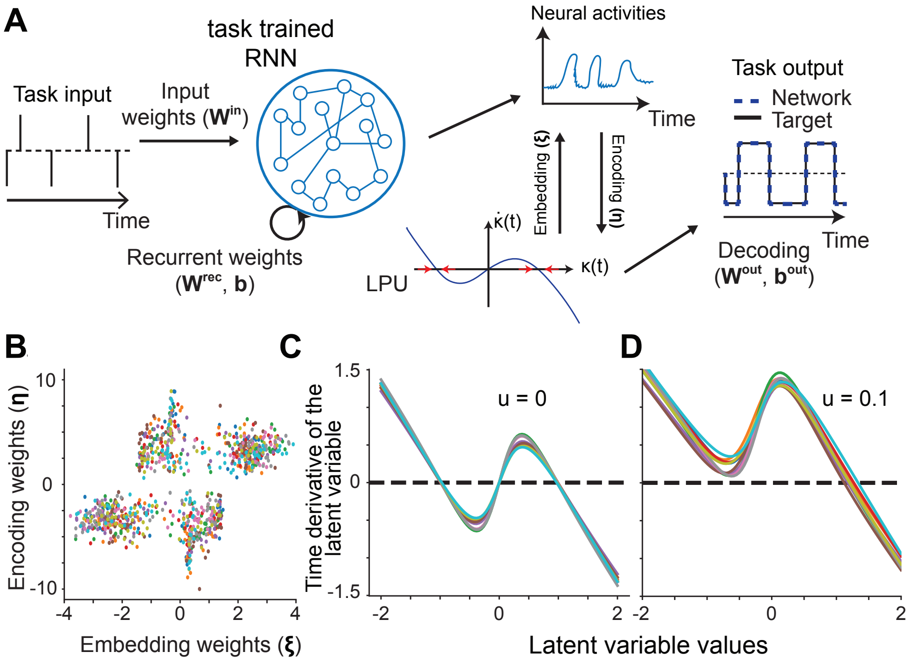

Figure 1: **Training a rank-one recurrent neural network is equivalent to training a low-dimensional dynamical system.** **A** Training process of a rank-one RNN performing the 1-bit flip-flop task. **B** Scatter plot of the learned encoding and embedding weights across ten random seeds. **C–D** Rank-one RNNs solve the task by creating bistable dynamical systems whose stable states can vanish depending on the input. **C** Learned dynamics of the LPU without input across all seeds. Each network converges to a bistable system with fixed points at $\kappa = \pm 1$, corresponding to the two flip-flop states. **D** Same as panel **C**, but with a small input $u = 0.1$. In all seeds, the LPU dynamics undergo a saddle-node bifurcation, leaving only the fixed point corresponding to the positive state. In **B–D**, results from different seeds are color-coded.

where $\tau > 0$ is the neuronal time scale, $\boldsymbol{r}(t) \in \mathbb{R}^N$ denotes activities of N neurons, and $u(t)$ is the input.

For many behavioral tasks, the recurrent connectivity can be constrained to have rank at most K, permitting a compact representation in terms of a latent processing unit (LPU as defined in Dinc et al., 2025). An LPU is parameterized by K encoding vectors ($\boldsymbol{\eta}^{(i)}$) and K embedding vectors ($\boldsymbol{\xi}^{(i)}$), such that

$$\mathbf{W}^{\text{rec}} = \sum_{i=1}^{K} \frac{1}{N} \boldsymbol{\xi}^{(i)} \boldsymbol{\eta}^{(i)}. \tag{2}$$

This factorization allows one to define a K-dimensional latent dynamical system (Dinc et al., 2025; Mastrogiuseppe and Ostojic, 2018; Beiran et al., 2021). In the rank-one case,

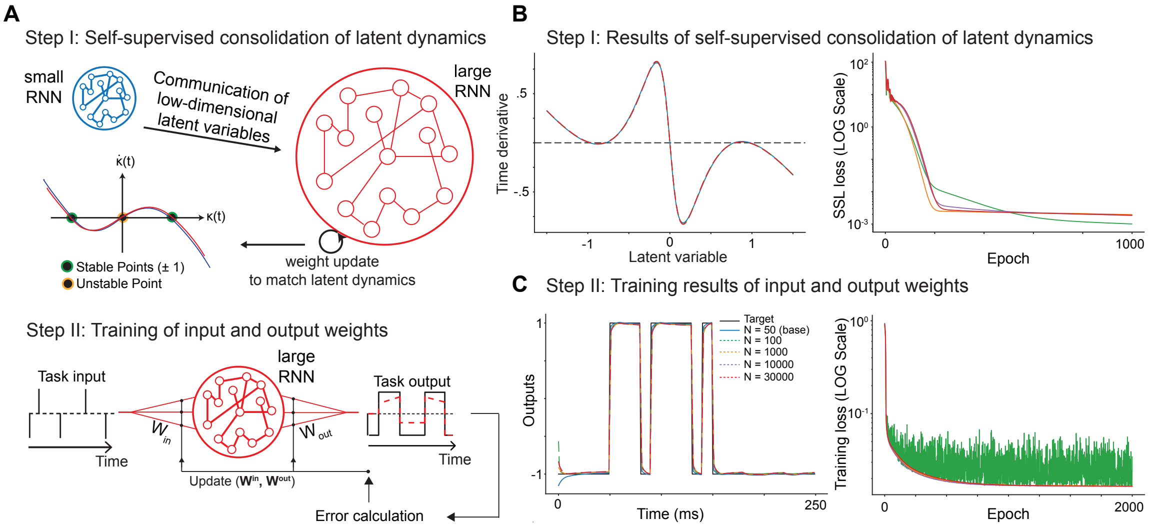

Figure 2: **Self-supervised consolidation enables training recurrent neural networks at scale. A** The overview of the consolidation process. Here, a smaller part of the network (referred to as small RNN) is trained with computationally expensive backpropagation through time to perform the task. Consolidation occurs in two steps: In stage one, the latent dynamics are learned in the absence of inputs or expectation of outputs. In stage two, the task is presented and the input-output mapping is learned rapidly. **B** The plot of learned latent representations for varying number of neurons in the large RNN (left) and the learning progress over epochs (right). **C** Learning the input-output mapping in stage two. For **B-C**, we used a laptop, each RNN took about few minutes to train despite their large size.

the latent variable

$$\kappa(t) = \frac{1}{N}\boldsymbol{\eta}^T \boldsymbol{r}(t) \tag{3}$$

obeys

$$\tau\dot{\kappa}(t) = -\kappa(t) + \frac{1}{N}\boldsymbol{\eta}^T \tanh(\boldsymbol{\xi}\kappa(t) + \boldsymbol{W}^{\text{in}}u(t) + \boldsymbol{b}). \tag{4}$$

Thus, training a rank-one RNN is equivalent to training a one-dimensional dynamical system. For simplicity, we omit biases and replace the output layer with an identity map in the following. Under these conditions, rank-one RNNs reliably learn the flip-flop task (Fig. 1**B–D**).

2.2. Self-supervised consolidation of latent variables for training large networks

A central challenge for both biological and artificial neural networks is how to train very large populations of neurons Lillicrap et al. (2020). From machine learning, it is well established that scaling up training can introduce instabilities and inefficiencies Pascanu et al.

(2013). Furthermore, it is often desirable to transfer or replicate representations learned in one network into another without loss of information Hinton et al. (2015). One strategy that addresses both issues is to train a small network with computationally expensive backpropagation-through-time (BPTT), extract the latent dynamical system it learns, and then consolidate this system into a much larger network through a self-supervised procedure.

To illustrate this idea, we considered networks with up to 30,000 neurons and trained them in two stages (Fig. 2**A**): In stage 1, *latent consolidation*, we trained the encoding ($\mathbf{n}$) and embedding ($\mathbf{m}$) weights so that the large network reproduces the latent dynamics. Specifically, the latent variable of the large network was defined as

$$z(t) = \frac{1}{N_{\text{large}}} \mathbf{n}^\top \mathbf{r}_{\text{large}}(t), \tag{5}$$

and matched to a set of randomly sampled latent trajectories κ_i in the absence of task-related inputs or outputs. In this stage, the loss function can be written as

$$\mathcal{L} = \frac{1}{N_{\text{samples}}} \sum_{j=1}^{N_{\text{samples}}} \left(\dot{\kappa}_i - \dot{z}_i\right)^2, \tag{6}$$

where $\dot{\kappa}_i$ and $\dot{z}_i$ follow Eq. (3), with $\kappa_i = z_i$ and $u = 0$. In stage 2, *task learning,* we fixed the output to $z(t)$ (for simplicity) and then trained the input layer so that the network could perform the behavioral task. Here, we minimize the mean-squared error between the predicted output $z(t)$ and the target output $o(t)$ across task trials.

To validate the approach, we performed a proof-of-principle experiment using gradient descent (Fig. 2**B**–**C**). The objectives are simple enough that they could likely also be optimized using local rules, such as the delta rule. As expected, large RNNs trained to consolidate the latent dynamics of smaller task-trained RNNs (Fig. 1) rapidly adjusted their parameters to reproduce the same dynamics (Fig. 2**B**). Once the latent system was in place, the input–output mapping was acquired in stage two (Fig. 2**C**). Notably, these experiments were performed on a standard laptop (Apple M2 Pro, 16 GB RAM), requiring only few minutes of training time per network for both stages combined.

3. Discussion and outlook

In this workshop presentation, we illustrated a simple yet powerful method for training large neural networks using latent representations learned by a much smaller network. In a biological context of consolidation (Dudai et al., 2015), this can be viewed as two interacting brain regions, where the representation acquired in one region is consolidated into another through sparse latent signals. We refer to this process as *self-supervised consolidation*, since the signals needed to learn the task are generated by a smaller part of the network rather than provided through external supervision. Once the latent dynamics were consolidated, the large network could learn input–output mappings rapidly. While we used gradient descent for illustration, the same framework can be combined with biologically plausible learning rules, with the delta rule being the most immediate candidate. Future extensions may incorporate additional biological constraints, such as minimizing overall neural activity after consolidation or enforcing weight sparsity.

References

Manuel Beiran, Alexis Dubreuil, Adrian Valente, Francesca Mastrogiuseppe, and Srdjan Ostojic. Shaping dynamics with multiple populations in low-rank recurrent networks. *Neural Computation*, 33(6):1572–1615, 2021.

Mark M Churchland, John P Cunningham, Matthew T Kaufman, Justin D Foster, Paul Nuyujukian, Stephen I Ryu, and Krishna V Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012.

Fatih Dinc, Marta Blanco-Pozo, David Klindt, Francisco Acosta, Yiqi Jiang, Sadegh Ebrahimi, Adam Shai, Hidenori Tanaka, Peng Yuan, Mark J Schnitzer, et al. Latent computing by biological neural networks: A dynamical systems framework. *arXiv preprint arXiv:2502.14337*, 2025.

Alexis Dubreuil, Adrian Valente, Manuel Beiran, Francesca Mastrogiuseppe, and Srdjan Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, pages 1–12, 2022.

Yadin Dudai, Avi Karni, and Jan Born. The consolidation and transformation of memory. *Neuron*, 88(1):20–32, 2015.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hidehiko K Inagaki, Susu Chen, Kayvon Daie, Arseny Finkelstein, Lorenzo Fontolan, Sandro Romani, and Karel Svoboda. Neural algorithms and circuits for motor planning. *Annual Review of Neuroscience*, 45:249–271, 2022.

Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

Nicolas Y Masse, Guangyu R Yang, H Francis Song, Xiao-Jing Wang, and David J Freedman. Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nature neuroscience*, 22(7):1159–1167, 2019.

Francesca Mastrogiuseppe and Srdjan Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99(3):609–623, 2018.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.

David Sussillo and Omri Barak. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural computation*, 25(3):626–649, 2013.

Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.