Working memory facilitates reward-modulated Hebbian learning in recurrent neural networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reservoir computing is a powerful tool to explain how the brain learns temporal sequences, such as movements, but existing learning schemes are either biologically implausible or too inefficient to explain animal performance. We show that a network can learn complicated sequences with a reward-modulated Hebbian learning rule if the network of reservoir neurons is combined with a second network that serves as a dynamic working memory and provides a spatio-temporal backbone signal to the reservoir. In combination with the working memory, reward-modulated Hebbian learning of the readout neurons performs as well as FORCE learning, but with the advantage of a biologically plausible interpretation of both learning rule and learning paradigm.

1 Introduction

Learning complex temporal sequences that extend over a few seconds – such as a movement to grab a bottle or to write a number on the blackboard – looks easy to us but is challenging for computational brain models. A common framework for learning temporal sequences is reservoir computing (alternatively called liquid computing or echo-state networks) [1, 2, 3]. It combines a reservoir, a chaotic recurrent network of rate units [4], with a linear readout that also feeds back to the reservoir. Training of the readout weights with FORCE, a recursive least-squares estimator [1], leads to near-optimal performance on many tasks such as motor movements.

The FORCE rule is, however, biologically implausible: update steps of synapses are rapid and large, and require an immediate and precisely timed feedback signal. A more realistic alternative to FORCE is the family of reward-modulated Hebbian learning rules [5, 6, 7], but plausibility comes at a price: when the feedback (reward minus expected reward) is given only after a long delay, reward-modulated Hebbian learning cannot overcome the chaotic nature of the network in complex tasks.

We propose stabilizing the reservoir activity by combining it with a continuous attractor: a working memory model that can store and update a continuous variable [8]. In particular, we use a two-dimensional ("bump") attractor, studied in [9] as a model of time elapsed since a start signal. Our proposition is related to that of feeding an abstract oscillatory input [10] or a "temporal backbone signal" [11] to the reservoir in order to overcome structural instabilities of the FORCE rule that occur if large time spans need to be covered.

In computational experiments, we show that a dynamic working memory that serves as an input to a reservoir network facilitates reward-modulated Hebbian learning in multiple ways: it makes a biologically plausible three-factor rule as efficient as FORCE; it admits a delay in the feedback signal; and it allows a single reservoir network to learn and perform multiple tasks.

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

2 Model

Our architecture is simple: the attractor network (the "memory") receives some task-specific input and produces a robust two-dimensional neural trajectory; the reservoir network (the "motor cortex") shapes its dynamics with this trajectory, and produces a potentially high-dimensional output (fig. 1).

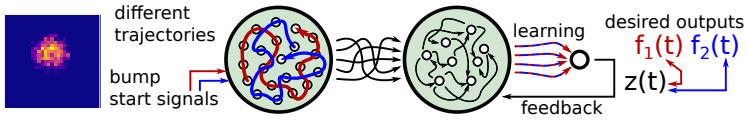

Figure 1: Model architecture: a moving 2D bump (left: activity bump (red) surrounded by inactive neurons (blue)) in an attractor network (left circle with two bump trajectories) projects to a reservoir (right circle); the output $\mathbf{z}(t)$ is read out from the reservoir and approximates the target function.

Attractor network. Following [9], the bump attractor consists of 2500 neurons evolving as

$$\begin{aligned} \tau_{\mathrm{m}}\dot{\mathbf{x}} &= -\mathbf{x} + [\mathbf{J}\mathbf{x} + \mathbf{e} - \mathbf{h}]_+ \,, \\ \tau_{\mathrm{a}}\dot{\mathbf{h}} &= -\mathbf{h} + s\,\mathbf{x}\,, \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the vector of firing rates evolving with time constant τ_{m}, $\mathbf{e}$ is the task-specific external input, $\mathbf{h}$ is an adaptation variable with time constant τ_{a} and s is the strength of adaptation. The weight matrix $\mathbf{J} = \mathbf{J}_{\mathrm{s}} + \mathbf{J}_{\mathrm{h}}$ has two parts. The symmetric part $\mathbf{J}_{\mathrm{s}}$ creates a two-dimensional translation-invariant structure resulting in bump-like stable activity patterns, whereas $\mathbf{J}_{\mathrm{h}}$ represents structural noise. Due to the adaptation $\mathbf{h}$, the bump moves across a path defined by the initial conditions and structural noise, creating long–lasting reliable activity patterns which also depend on the input $\mathbf{e}$.

Reservoir network. The reservoir learns to approximate a target function $\mathbf{f}(t)$ with the output $\mathbf{z}(t)$ by linearly combining the firing rate $\mathbf{r}$ with readout weights $\mathbf{W}_{\mathrm{ro}}$: $\mathbf{z} = \mathbf{W}_{\mathrm{ro}}\mathbf{r} + \boldsymbol{\eta} \equiv \hat{\mathbf{z}} + \boldsymbol{\eta}$ with readout noise $\boldsymbol{\eta}$. We use the same number of neurons (1000) and parameters as [1, 6],

$$\tau\dot{\mathbf{u}} = -\mathbf{u} + \lambda\mathbf{W}_{\mathrm{rec}}\mathbf{r} + \mathbf{W}_{\mathrm{fb}}\mathbf{z} + c\,\mathbf{W}_{\mathrm{attr}}\mathbf{x}\,, \quad \mathbf{r} = \tanh(\mathbf{u}) + \boldsymbol{\xi}\,, \tag{2}$$

where $\mathbf{u}$ is the membrane potential, $\boldsymbol{\xi}$ is the firing rate noise, $\mathbf{W}_{\mathrm{attr}}$ centers and scales attractor input with coupling c, $\mathbf{W}_{\mathrm{rec}}$ and λ regulate chaotic activity [4], and $\mathbf{W}_{\mathrm{fb}}$ implements the feedback loop.

Learning rule. We use the reward-modulated Hebbian rule of [6] for the readout weights $\mathbf{W}_{\mathrm{ro}}$,

$$\Delta\mathbf{W}_{\mathrm{ro}}(t) = \eta(t)M(t)(\mathbf{z}(t) - \bar{\mathbf{z}}(t))\mathbf{r}^{\top}(t)\,, \quad \eta(t) = \eta_0/(1 + t/\tau_\eta)\,, \tag{3}$$

where $\bar{x}$ denotes low-pass filtering of x, such that $\mathbf{z}(t) - \bar{\mathbf{z}}(t) \approx \boldsymbol{\eta}(t)$. The reward modulation $M(t)$ tracks performance $P(t)$ as

$$P(t) = -\|\mathbf{f}(t) - \mathbf{z}(t)\|^2\,, \quad M(t) = \begin{cases} 1, & P(t) > \bar{P}(t)\,, \\ 0, & P(t) \le \bar{P}(t)\,. \end{cases} \tag{4}$$

The update rule is an example of a NeoHebbian three-factor learning rule [12] and mimics gradient descent if we ignore the feedback loop [5].

3 Experiments

Performance evaluation. We measure performance (ranging from -1 to 1, where 1 implies a perfect match) with normalized cross-correlation between the output and the target on a single trial with frozen weights. In fig. 2, reward-modulated Hebbian and FORCE learning are compared on 50 target functions sampled from a Gaussian Process (GP) with exponential squared kernel ($\sigma^2 = 10^4$ to match the complexity of hand-picked functions from [1, 6]). The **code** is available at https://github.com/neuroai-workshop-anon-1224113/working-memory-facilitating-reservoir-learning.

3.1 Reward-modulated Hebbian learning with attractor input reaches FORCE performance

When tested on one-second signals similar to those of [1, 6] (two insets in fig. 2), the full network with attractor input and reward-modulated Hebbian learning learns faster and more reliably than

reward-modulated Hebbian learning without the input from the attractor network. After about 90 training trials, the full network achieves the performance of the FORCE rule (for which train error approaches one in the first trial, [1], while test error does so after 30-50 trials, [6]; fig. 2A).

For target signals that extend over 10 seconds (same smoothness of the target functions, two insets in fig. 2B), the reward-modulated Hebbian rule achieves a performance of 1 after 200 trials if combined with input from the attractor network (fig. 2B) but fails completely without the attractor network (tuning of the hyperparameters on the logarithmic scale did not help; data not shown). Thus a three-factor learning rule succeeds to learn complex tasks if combined with a temporally structured input from the attractor network.

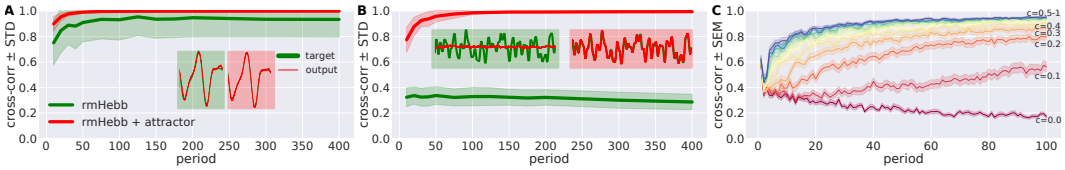

Figure 2: **A.** Mean test performance as a function of trials with a reward-modulated Hebbian (rmHebb) learning rule combined with (red) or without (green) input from attractor. Inset: target function over 1s and examples of the output after training. **B**. Same, but for a target signal that spans over 10s. Shaded area shows standard deviation. The FORCE rule achieves a performance of 1 in all trials. **C**. Learning performance for different couplings c of the attractor output to reservoir input ($c = 1$ normal coupling, $c = 0$ no coupling) when updates are delayed to the end of every one-second trial.

3.2 Attractor input allows realistic frequency of weight updates

FORCE learning needs a feedback signal in every time step. Standard reward-modulated Hebbian learning can support very small delays, but fails if updates are less frequent that every few ms [6]. We show that input from our dynamic working memory module overcomes this problem. In our approach proposed updates are summed up in the background, but applied only at the end of a one-second trial. We find that even with such a temporally sparse update, learning is still possible (fig. 2C)

The input from the dynamic working was necessary to achieve this task: when the strength of the attractor input gradually decreases, performance drops; only in the total absence of attractor input ($c = 0.0$; note that the reservoir still received weak input noise) learning completely failed. Strikingly, delayed updates did not hurt performance, and the system achieved high (> 0.9) cross-correlation in fewer than 100 training trials if the input from the attractor network was strong enough. The transient drop in performance shortly after the start in fig. 2C is likely due to $\mathbf{W}_{\mathrm{ro}} = 0$ in the beginning, meaning that the output is uncorrelated with the firing rates, and therefore the cumulative weight update does not approximate gradient decent.

3.3 Working memory translates to efficient reservoir learning of multiple signals

It is well known that reservoir networks can learn multiple tasks given different initial conditions with both FORCE [1] and the reward-modulated Hebbian rule [6]. We want to check whether this also holds for our approach. We conjecture that different inputs to the attractor network generate unique neural trajectories [9] that can be exploited by the reservoir network.

To test this hypothesis, we train the network to produce hand-written digits. The **static** input to the attractor comes from the pre-processed MNIST dataset (network inputs are taken from one of the last layers of a deep network trained to classify MNIST) in order to provide a realistic input to the attractor network which transforms the static input into dynamic trajectories (noiseless, fig. 3B, and noisy, fig. 3D). We recorded 50 attractor trajectories for training (used 4 times, resulting in total 2000 training trials) and 50 for testing of each digit (1 second each), where each trajectory corresponded to a distinct input pattern. The reservoir learned a single drawing for each class. The variance of the structural noise in the attractor network was 3 times larger compared to the previous experiments in order to produce more robust bump trajectories (fig. 3D).

The reward-modulated Hebbian rule mastered 10 out of 10 digits when driven by a noiseless input from the attractor network (fig. 3A). In the presence of noise in the attractor network (fig. 3D), the

performance is imperfect for “five” and “six” (fig. 3C). We checked that FORCE learning with the same noisy input did not improve the performance (data not shown). Note that a linear readout of the attractor (without the reservoir) would be insufficient: first, sometimes single digit trajectories are too dissimilar (e.g. the zero’s and one’s in fig. 3D); second, frequent overlaps in trajectories mean that a linear readout would produce the same output at those points, no matter what the digit is.

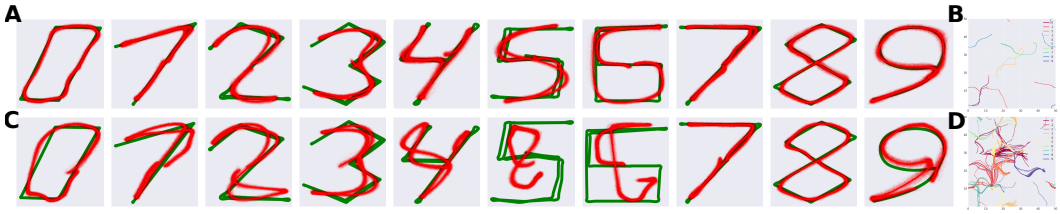

Figure 3: **A.** Targets (green) and test outputs (red) for reward-modulated Hebbian rule with noiseless attractor input. **B.** Noiseless attractor trajectories for each digit. **C.** Same as **A** but with noisy attractor trajectories. **D.** Same as **B** but static input’s noise resulted in noisy trajectories (individual lines).

4 Discussion

We showed that a dynamic working memory can facilitate learning of complex tasks with biologically plausible three-factor learning rules. Our results indicate that, when combined with a bump attractor, reservoir computing with reward-modulated learning can be as efficient as FORCE [1], a widely used but biologically unrealistic rule. The proposed network relies on a limited number of trajectories in the attractor network. To increase its capacity, a possible future direction would be to mix periodic attractor input with another, also input-specific, but transient input that would bring the reservoir into a different initial state. In this case the attractor network would work as a time variable (as in [9]), and the other input as the control signal (as in [1]).

Apart from the biological relevance, the proposed method might be used for real-world applications of reservoir computing (e.g. wind forecasting [13]) as it is computationally less expensive than FORCE. It might also be an interesting alternative for learning in neuromorphic devices.

References

[1] D. Sussillo and L.F. Abbott. *Neuron*, 63:544–557, 2009.

[2] W. Maass, T. Natschläger, and H. Markram. *Neural Computation*, 14:2531–2560, 2002.

[3] H. Jaeger and H. Haas. *Science*, 304:78–80, 2004.

[4] H. Sompolinsky, A. Crisanti, and H. J. Sommers. *Physical Review Letters*, 61:259–262, 1988.

[5] R. Legenstein, S. M. Chase, A. B. Schwartz, and W. Maass. *Journal of Neuroscience*, 30:8400–8410, 2010.

[6] G.M. Hoerzer, R. Legenstein, and W. Maass. *Cerebral cortex*, 24:677–90, 2014.

[7] N. Frémaux and W. Gerstner. *Frontiers in Neural Circuits*, 9:85, 2016.

[8] S. Wu, K. Hamaguchi, and S. Amari. *Neural computation*, 20:994–1025, 2008.

[9] V. Itskov, C. Curto, E. Pastalkova, and G. Buzsaki. *Journal of Neuroscience*, 31:2828–2834, 2011.

[10] P. Vincent-Lamarre, G. Lajoie, and J.P. Thivierge. *J. Comput. Neurosci.*, 41:305–322, 2016.

[11] W. Nicola and C. Clopath. *Nat. Comm.*, 8:2208, 2017.

[12] W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, and J. Brea. *Front. Neural Circ.*, 12:53, 2018.

[13] E. López, C. Valle, H. Allende, E. Gil, and H. Madsen. *Energies*, 11:526, 2018.