Slow processes of neurons enable a biologically plausible approximation to policy gradient

Anand Subramoney*
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Franz Scherr*
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Guillaume Bellec*
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Elias Hajek
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Darjan Salaj
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Robert Legenstein
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria

Wolfgang Maass
Institute of Theoretical Computer Science
Technische Universität Graz
A-8010 Graz, Austria
`maass@igi.tugraz.at`
* equal contributions

Abstract

Recurrent neural networks underlie the astounding information processing capabilities of the brain, and play a key role in many state-of-the-art algorithms in deep reinforcement learning. But it has remained an open question how such networks could learn from rewards in a biologically plausible manner, with synaptic plasticity that is both local and online. We describe such an algorithm that approximates actor-critic policy gradient in recurrent neural networks. Building on an approximation of backpropagation through time (*BPTT*): *e-prop*, and using the equivalence between forward and backward view in reinforcement learning (RL), we formulate a novel learning rule for RL that is both online and local, called *reward-based e-prop*. This learning rule uses neuroscience inspired slow processes and top-down signals, while still being rigorously derived as an approximation to actor-critic policy gradient. To empirically evaluate this algorithm, we consider a delayed reaching task, where an arm is controlled using a recurrent network of spiking neurons. In this task, we show that *reward-based e-prop* performs as well as an agent trained with actor-critic policy gradient with biologically implausible *BPTT*.

Introduction

Deep reinforcement learning (deep RL) has formed the basis for the staggeringly successful recent results in machine learning and AI [1, 2, 3]. A standard algorithm in deep RL is actor-critic policy gradient [4], where the network model outputs probabilities for each action (called the policy) and also predicts the sum of future rewards (called the value). For tasks that require working memory, recurrent neural networks (RNNs) are used, often in the form of LSTM units [5] in deep RL. These recurrent networks are trained using *backpropagation through time* (*BPTT*) to maximize the expected sum of future rewards while minimizing

But *BPTT* requires storing the intermediate states of all neurons during a network computation, and merging these in a subsequent offline process with gradients that are computed backwards in time. This makes it very unlikely that *BPTT* is used by the brain [6]. In addition, assigning credit to actions necessitates simulating the environment until the outcome becomes evident, i.e. until the end of an episode. This makes online implementation of actor-critic policy gradient difficult.

In this submission, we describe an algorithm called *reward-based e-prop* that solves both of these problems. The result is a local, online RL rule that can be used to train RNNs. It can also be shown to approximate the ideal learning rules based on gradient descent. This algorithm builds on an approximation to *BPTT* called *e-prop* [7] and incorporates the advantages of the actor-critic method into a learning rule that is both local and online.

This method derives its inspiration from experimental data in biology for how such a learning rule could be implemented in a biologically plausible way. The dynamics of neurons are known to have traces of past activity on a molecular level [8]. These traces record events that are known induce plasticity in the presence of top-down learning signals [9, 10, 11]. These type of local traces are referred to as eligibility traces in *e-prop* [7]. In addition, various kinds of top-down signals that are specific for target population of neurons are known to exist [12]. Among them are signals that predict upcoming rewards [13, 14] or movement errors in case of error-related negativity (ERN) [15]. These signals constitute the learning signals of *e-prop* and the reward prediction error that emerges in the context of *reward-based e-prop*.

Using such processes, and formulating online actor-critic policy gradient [16] using the mathematical framework of *e-prop* leads us to *reward-based e-prop*. In *reward-based e-prop*, the weight updates can be calculated at every time step without having to wait for all future rewards, without backpropagation of signals either through time (online) or synapses (local). We demonstrate this learning rule on a neuroscience inspired RL task (Fig. 1). We specifically consider the scenario where the agent needs to remember early parts of an episode in order to produce a successful policy in later parts of an episode. This requires the use of RNNs for working memory, and the ability to discover the relationship between early observations and later actions.

Results

Reward-based learning In the RL setting an agent interacts with an environment. In our case, the agent is implemented by a recurrent network, specifically a recurrent network of spiking neurons with working memory (see Appendix for details), although the theory is generally applicable to any model of recurrent network. This network receives observations $\boldsymbol{x}^t$ from the environment, and interacts with it by means of real-valued actions $\boldsymbol{a}^t$. These actions are distributed according to a Gaussian centered around the network output $\boldsymbol{y}^t$ with a fixed variance σ^2. The probability distribution of these actions $\pi(\boldsymbol{a}^t|\boldsymbol{y}^t)$ is the stochastic policy. In our setting, we restrict the actions to certain decision times $t_0, \ldots, t_n, \ldots$, and set $\pi \equiv 0$ at other times. The environment can provide a positive or negative reward r^t at any time t to inform the agent about favorable states.

The goal of reward-based learning is to maximize sum of discounted future rewards in expectation: $\max \mathbb{E}[R^0]$, where we define the return at time t as $R^t = \sum_{t' \geq t} \gamma^{t'-t} r^{t'}$. Here, $\gamma \leq 1$ is known as the discounting factor. This is achieved by the actor-critic variant of policy gradient, involving the policy π (the actor) and an additional output neuron V^t (with weights $W_j^{V,\mathrm{out}}$), predicting the value function $\mathbb{E}[R^t]$ (the critic). Both are learnt simultaneously by minimizing the loss function:

$$E = E_\pi + c_V E_V \ , \tag{1}$$

where $E_\pi = -\mathbb{E}[R^0]$ measures the performance of the stochastic policy π, $E_V = \mathbb{E}[\sum_t \frac{1}{2}(R^t - V^t)^2]$ measures the error in value prediction, and c_V is a hyper-parameter chosen appropriately.

To improve the agent's behavior, we minimize the loss function E using gradient descent on the network weights W_{ji}. For this purpose, we can derive an estimator $\widehat{\frac{dE}{dW_{ji}}}$ of the loss gradient with reduced variance (using the policy gradient theorem [16]):

$$\frac{dE}{dW_{ji}} \;=\; \mathbb{E}\Bigg[\underbrace{-\sum_t (R^t - V^t)\left(\frac{d}{dW_{ji}}\big(\log\pi(\boldsymbol{a}^t|\boldsymbol{y}^t) + c_V V^t\big)\right)}_{\overset{\text{def}}{=}\widehat{\frac{dE}{dW_{ji}}}}\Bigg]\,. \tag{2}$$

Equation (2) can be seen as the typical formulation of an actor-critic algorithm. However, in its current form, the value of this gradient cannot be computed in a biological plausible way:

A) the quantity R^t is not available at time t because its computation requires future rewards,

B) calculating the derivatives of the network output ($\log\pi$, V) with respect to the network input and recurrent weights requires biologically implausible *BPTT*.

Solution to A): Backward view We can alleviate the first problem involving the computation of R^t at time t by exploiting the equivalence between forward view and backward view in RL [16]. For this purpose, we introduce a temporal difference (TD) error $\delta^t = r^t + \gamma V^{t+1} - V^t$ that allows us to rewrite the error in value prediction by a sum over future TD errors: $R^t - V^t = \sum_{t'\geq t}\gamma^{t'-t}\delta^{t'}$. If we substitute this into the relevant part of equation (2), we obtain two sums over t and t'. Because the summation of t' starts from t, we can reorganize and collect terms such that only past terms are summed (commonly denoted backward view of RL [16]):

$$\sum_t (R^t - V^t)(\,\cdot\,) = \sum_t\sum_{t'\geq t}\gamma^{t'-t}\delta^{t'}(\,\cdot\,) = \sum_{t'}\delta^{t'}\sum_{t\leq t'}\gamma^{t'-t}(\,\cdot\,) = \sum_{t'}\delta^{t'}F_\gamma(\,\cdot\,)\,. \tag{3}$$

In order to simplify notation, we introduce a temporal filter $\mathcal{F}$ that is defined as $\mathcal{F}_\alpha(x^t) = \alpha\mathcal{F}_\alpha(x^{t-1}) + x^t$.

Solution to B): *E-prop* Computing the gradient $\frac{d}{dW_{ji}}\big(\log\pi(\boldsymbol{a}^t|\boldsymbol{y}^t) + c_V V^t\big)$ required in equation (2) with respect to input weights W^{in}_{ji} and recurrent weights W^{rec}_{ji} in RNNs with *BPTT* is biologically implausible. We approximate this gradient using *e-prop* [17] by ignoring both spatially and temporally non-local interactions. This gradient is approximated as a product of learning signals L^t_j, which are specific to the post-synaptic neuron j, and synapse-specific eligibility traces e^t_{ji}:

$$\widehat{\frac{d}{dW_{ji}}}\big(\log\pi(\boldsymbol{a}^t|\boldsymbol{y}^t) + c_V V^t\big) = L^t_j\bar{e}^t_{ji}\,, \tag{4}$$

Here, we define $\bar{e}^t_{ji} = \mathcal{F}_\kappa(e^t_{ji})$ with κ being the time constant associated with the output neuron, and the eligibility traces e^t_{ji} only depend on the activity that is local to the synapse. The eligibility traces are formally defined as $e^t_{ji} = \frac{\partial z^t_j}{\partial \boldsymbol{h}^t_j}\cdot\boldsymbol{\epsilon}^t_{ji}$, using a so-called eligibility vector that is propagated forward in time, according to the dynamics of neurons: $\boldsymbol{\epsilon}^{t+1}_{ji} = \frac{\partial \boldsymbol{h}^{t+1}_j}{\partial \boldsymbol{h}^t_j}\cdot\boldsymbol{\epsilon}^t_{ji} + \frac{\partial \boldsymbol{h}^{t+1}_j}{\partial W_{ji}}$, where $\boldsymbol{h}^t_j$ is the hidden state vector and z^t_j is the observable state of the neuron model. This definition precisely captures the dynamics of the hidden states. In particular, if the dynamics of the neurons contain slower processes as our network does (see appendix), long lived traces arise that can help with the credit assignment problem.

The learning signal $L^t_j = \frac{\partial\left(\log\pi(\boldsymbol{a}^t|\boldsymbol{y}^t) + c_V V^t\right)}{\partial z^t_j}$ is the impact of neuron spikes z^t_j on $\log\pi(\boldsymbol{a}^t|\boldsymbol{y}^t) + c_V V^t$. Considering our Gaussian policy π, we obtain:

$$L^t_j = -c_V W^{V,\text{out}}_j + \sum_k W^{\pi,\text{out}}_{jk}\frac{y^t_k - a^t_k}{\sigma^2}\,. \tag{5}$$

where $W^{\pi,\text{out}}_{jk}$ and $W^{V,\text{out}}_j$ are the output weights for the policy π and the value prediction V respectively.

Learning rule emerging from *reward-based e-prop* The learning rule emerging in equation (5) solves major issues of biological plausibility in *BPTT*. There is no need to backpropagate through time or store earlier network activity, and the algorithm can be implemented by online updates. Even the sum over t in (5) does not need store every element of the sum. Instead the sum can be accumulated, or every contribution can be applied directly.

With this definition of the learning signal (equation (5)), the deviation between the stochastic action a_k^t and its expected value y_k^t are fed back to neuron j with the same weights $W_{kj}^{\pi,\mathrm{out}}$ that define y_k^t, and this symmetry also arises for $W_j^{V,\mathrm{out}}$. To avoid such biologically implausible weight sharing between the feedforward and the feedback pathways, we use fixed random feedback weights B_{jk}^{π}, B_j^V as in [18, 19]. Using a simple local plasticity rule on B_{jk} that mirrors the plasticity rule applied to W_{kj} also leads to similar results. Putting together the results of equation (3), (4) and (5), we obtain the learning rule for *reward-based e-prop*:

$$\Delta W_{ji}^{\mathrm{rec}} = -\eta \sum_t \delta^t \mathcal{F}_\gamma\left(L_j^t\, \bar{e}_{ji}^t\right) \quad \text{for} \tag{6}$$

$$L_j^t = -c_V B_j^V + \sum_k B_{jk}^{\pi} \frac{y_k^t - a_k^t}{\sigma^2} \quad , \tag{7}$$

where η is a fixed learning rate.

Note that the filtering $\mathcal{F}_\gamma$ requires an additional eligibility trace per synapse. The term $\mathcal{F}_\gamma\left(L_j^t\, \bar{e}_{ji}^t\right)$ is identical to the eligibility traces commonly used in RL [16, 20], except that we use an approximation for the true gradient of the loss instead. Additionally, [20] uses a linear feedforward model, whereas we specifically address learning in recurrent neural networks. This term depends on the learning signal and does not have the same function as the eligibility trace e_{ji}^t that are employed by *e-prop*.

Empirical evaluation of *reward-based e-prop* We tested *reward-based e-prop* on a task that is representative of a common learning experiment paradigm in systems neuroscience: There an agent has to learn a delayed goal-directed movement, consisting of a sequence of many 2-dimensional continuous motor commands. The rewards are sparse and often arrive long after relevant actions have been taken. The agent first receives a spatial goal cue (Fig. 1C), which is followed by a delay period during which the agent has to remember the cue, and has to make sure the tip of the two-joint arm it controls remains within a center region (indicated by a dotted circle) in order to avoid punishment. The agent controls the arm movement by controlling the angular velocities of the two joints. After the delay period, the agent receives a go-cue, which instructs that the agent has to move the tip of the arm to the location that was indicated by the initial goal cue in order to receive a positive reward.

Note that the network was not given any forward- or inverse model of the arm but had to learn those implicitly. The agent also required working memory to remember the goal cue from the beginning of the episode until it received the go-cue. In addition, due to delayed rewards, credit assignment for actions which led to the reward was non-trivial.

We used a recurrent spiking network with an additional working memory mechanism called an LSNN (as in [17]) to control the arm. The overall architecture of the network, along with the components that contribute to the weight update rule, are shown in fig. 1A.

Three sample trials after learning are shown in Fig. 1D. Fig. 1B shows that *reward-based e-prop* is able to solve this demanding RL task about as well as policy gradient with biologically implausible *BPTT*. This is due to the fact that the eligibility traces that arise in *reward-based e-prop* are able to handle the long term credit assignment problem, while the slower dynamics of the neurons in the network provide the working memory. Note that the credit assignment here occurs successfully without any backpropagation of gradients through the entire history. We conjecture that variants of *reward-based e-prop* will be able to solve most RL tasks that can be solved by online actor-critic methods in machine learning.

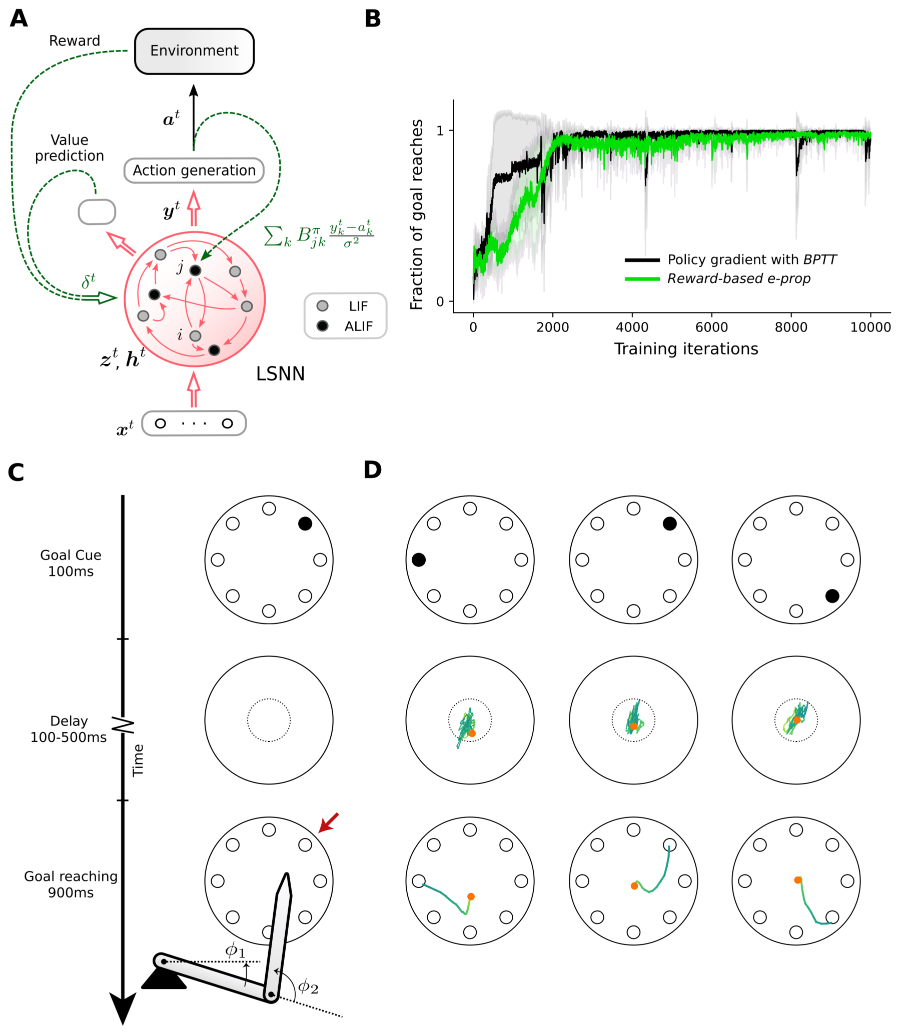

Figure 1: **Application of *e-prop* to RL. A**) Learning architecture for *reward-based e-prop*: The network input $\mathbf{x}^t$ consists of the current joint angles and input cues. The network produces output $\mathbf{y}^t$ which is used to stochastically generate the actions $\mathbf{a}^t$. In addition, the network produces the value prediction, which, along with the reward from the environment, is used to calculate the TD-error δ^t, An LSNN is the network model defined in the Appendix. The learning signals and the TD-errors are used to calculate the weight update, as denoted by the green dotted lines. **B**) Performance of *reward-based e-prop* and of a control where *e-prop* is replaced by *BPTT*, both for an LSNN consisting of 350 LIF and 150 adaptive LIF (ALIF) neurons. Solid curves show the mean over 5 different runs, and shaded area indicates 1 standard deviation. **C**) Scheme of the delayed arm movement task. The red arrow points to the formerly visible goal. The arm always starts moving from the center of the circle. **D**) Resulting arm movement in three sample trials after learning. The orange dot indicates the position of the tip of the arm at the end of the delay period.

Discussion

We have proposed a biologically plausible learning rule – *reward-based e-prop* – for RL that is both online and local. The combination of reward prediction error and neuron-specific learning signal has also been used in a plasticity rule for feedforward networks inspired by neuroscience [21]. But in the case of *reward-based e-prop*, it arises from the approximation of *BPTT* by *e-prop* in spiking RNNs tackling RL problems. Other previously proposed learning rules use the correlation of the noisy output of network neurons with rewards to estimate the gradients of the policy [22, 11]. But due to noisy gradient estimates, such learning rules are inefficient.

Since *reward-based e-prop* is based on a transparent mathematical principle, it provides a normative model for signals that are abundantly present in the brain, but whose precise role is not very well understood – eligibility traces, learning signals and reward prediction errors. Actor-critic policy gradient combined with BPTT for RNNs has been shown to be very powerful in deep RL. *Reward-based e-prop* approximates this, and so has the potential to be a very powerful online learning algorithm for RL for a wide variety of tasks. Here, we have demonstrated that it can be used to successfully train recurrent networks on a reinforcement learning task that requires complex arm movements, working memory and long term credit assignment. In addition, we have demonstrated this using a recurrent network model that is biologically realistic – using spiking neurons, and slower processes very similar to that found in biology.

Future work would explore scaling up this algorithm to tasks of greater complexity and memory demands. Furthermore, an online RL algorithm such as *reward-based e-prop* can also lead to a qualitative jump in on-chip learning capabilities of neuromorphic chips.

Acknowledgments

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[2] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojtek Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. `https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/`, 2019.

[3] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[4] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, pages 1928–1937, 2016.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[6] T. P. Lillicrap and A. Santoro. Backpropagation through time and the brain. *Current Opinion in Neurobiology*, 55:82–89, 2019.

[7] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *bioRxiv*, page 738385, 2019.

[8] Magdalena Sanhueza and John Lisman. The camkii/nmdar complex as a molecular memory. *Molecular brain*, 6(1):10, 2013.

[9] S. Cassenaer and G. Laurent. Conditional modulation of spike-timing-dependent plasticity for olfactory learning. *Nature*, 482(7383):47, 2012.

[10] Sho Yagishita, Akiko Hayashi-Takagi, Graham CR Ellis-Davies, Hidetoshi Urakubo, Shin Ishii, and Haruo Kasai. A critical time window for dopamine actions on the structural plasticity of dendritic spines. *Science*, 345(6204):1616–1620, 2014.

[11] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility Traces and Plasticity on Behavioral Time Scales: Experimental Support of NeoHebbian Three-Factor Learning Rules. *Frontiers in Neural Circuits*, 12, July 2018.

[12] Stephane J MacLean, Cameron D Hassall, Yoko Ishigami, Olav E Krigolson, and Gail A Eskes. Using brain potentials to understand prism adaptation: the error-related negativity and the p300. *Frontiers in human neuroscience*, 9:335, 2015.

[13] Ben Engelhard, Joel Finkelstein, Julia Cox, Weston Fleming, Hee Jae Jang, Sharon Ornelas, Sue Ann Koay, Stephan Y Thiberge, Nathaniel D Daw, David W Tank, et al. Specialized coding of sensory, motor and cognitive variables in vta dopamine neurons. *Nature*, page 1, 2019.

[14] J. Roeper. Dissecting the diversity of midbrain dopamine neurons. *Trends in neurosciences*, 36(6):336–342, 2013.

[15] Amirsaman Sajad, David C. Godlove, and Jeffrey D. Schall. Cortical microcircuitry of performance monitoring. *Nature Neuroscience*, 22(2):265, February 2019.

[16] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[17] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems*, pages 787–797, 2018.

[18] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:13276, 2016.

[19] Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *NIPS*, pages 1037–1045, 2016.

[20] Harm van Seijen and Rich Sutton. True Online TD(lambda). In *International Conference on Machine Learning*, pages 692–700, January 2014. 00063.

[21] Isabella Pozzi, Sander Bohté, and Pieter Roelfsema. A biologically plausible learning rule for deep learning in the brain. *arXiv preprint arXiv:1811.01768*, 2018.

[22] David Kappel, Robert Legenstein, Stefan Habenschuss, Michael Hsieh, and Wolfgang Maass. A dynamic connectome supports the emergence of stable computational function of neural circuits through reward-based learning. *eNeuro*, 2018.

Appendix: Network model

Our approach is applicable to a general class of RNN models, including LSTMs. Here, we consider recurrent spiking neural networks, where each neuron j is characterized by a membrane voltage v_j^t that integrates synaptic inputs $\boldsymbol{x}^t$, and processes these by propagating recurrent activity according to discrete time steps of 1 ms: $v_j^t = \alpha v_j^{t-1} + \sum_i W_{ji}^{\mathrm{in}} x_i^t + \sum_i W_{ji}^{\mathrm{rec}} z_i^{t-1}$, using input weights W_{ji}^{in} and recurrent weights W_{ji}^{rec}. Here, α relates to the decay of the membrane voltage with time constants of 20 ms. If v_j^t exceeds a threshold, the neuron emits a spike $z_j^t = 1$ (otherwise $z_j^t = 0$) and its membrane voltage is reset. We use an additional mechanism from [17], where some LIF neurons in the network are enriched by adaptive thresholds that increase whenever the corresponding neuron spikes, decaying back to a baseline with slower dynamics, having a time constant of 500 ms. Networks of LIF and adaptive LIF (ALIF) neurons are termed Long short-term memory Spiking Neural Networks (LSNNs). The output of such networks is denoted as $\boldsymbol{y}^t$ and its components are defined as leaky integrators: $y_k^t = \kappa y_k^{t-1} + \sum_j W_{kj}^{a,\mathrm{out}} z_j^t$, where κ relates to a decay factor with a time constant of 20 ms.