Learning to Map Input-Output Spike Patterns by Reward-Modulated STDP

Brian Gardner, Ioana Sporea and André Grüning

Department of Computing, University of Surrey,
Guildford, Surrey, GU2 7XH, United Kingdom
`{b.gardner,i.sporea,a.gruning}@surrey.ac.uk`

Abstract. Reward-modulated learning rules for spiking neural networks have emerged, that have been demonstrated to solve a wide range of reinforcement learning tasks. Despite this, few attempts have been made in teaching a spiking network to learn target spike trains. Here, we apply a reward-maximising learning rule to teach a spiking neural network to map between multiple input patterns and single-spike target trains. Furthermore, we compare the performance of two escape rate functions that drive output spiking activity: the Arrhenius & Current (A&C) model and Exponential (EXP) model. We find A&C consistently outperforms EXP: both in terms of the accuracy of responses and the time taken to converge in learning. We also show that jittering input patterns with a low noise amplitude leads to an improvement in learning, especially by reducing fluctuations in the network responses.

Keywords: Neuronal Plasticity, Stochastic Neuron, Spiking Neural Network, Latency Encoding

1 Introduction

There is increasing interest in Reward-modulated Spike-Timing-Dependent Plasticity (R-STDP) as a biologically plausible rule for modelling learning in the brain, where candidate weight changes triggered by STDP only become effective by the presence of an external reward signal [4, 11, 12]. However, whilst there has been significant progress in applying R-STDP to solving many reinforcement learning problems [19, 6, 7], there still remains a lack of work that has aimed to apply R-STDP to learning target spike trains.

In the literature, two main forms of reward-modulated learning for spiking networks have been identified: an empirically formulated R-STDP rule, and a theoretically derived Reward-maximising (R-max) rule [11, 13, 5]. Both approaches rely on three factors for learning to succeed: 1. presynaptic activity as the stimulus, 2. postsynaptic activity as the response, and 3. the feedback of a success signal to allow synaptic changes. The approaches differ by the lack of an unsupervised learning bias with R-max, as opposed to R-STDP. In general, an unsupervised component hinders reward-modulated learning, as R-STDP was

2 Learning to Map Input-Output Spike Patterns

found to have decreased performance in comparison with R-max when learning several target spike trains [5].

Therefore, we aim to further investigate the efficacy of the R-max rule in learning to map between multiple input patterns and target spike trains, where we also compare the performance of two escape rate functions that drive output spiking activity. This paper is an extension of our previous work [8], where only a single mapping between a fixed input pattern and target spike train was considered.

2 Method

2.1 Single Neuron Model

A single readout neuron receives input from m presynaptic neurons, where the spike train due to the j^{th} input, $1 \leq j \leq m$, is given as a sum of δ -functions: $X_j(t) = \sum_f \delta(t - t_j^f)$. In response to an input pattern $\mathbf{X} = \{X_j\}$, the membrane potential of the readout neuron at time t , given a last spike time $\hat{t}$, is defined by the Spike Response Model [9]:

$$u(t|\mathbf{X}, \hat{t}) := U_{rest} + \sum_j w_j \int_{\hat{t}}^t \epsilon(t - t') X_j(t') dt' + \kappa(t - \hat{t}), \quad (1)$$

where $U_{rest} = -70$ mV is the resting membrane potential and w_j the j^{th} afferent synaptic weight. The Postsynaptic Potential (PSP) kernel is taken as $\epsilon(s) = \frac{\epsilon_0}{\tau_m - \tau_s} (e^{-s/\tau_m} - e^{-s/\tau_s})$, where $\epsilon_0 = 20$ mV·ms is a scaling constant, $\tau_m = 10$ ms the membrane time constant and $\tau_s = 2$ ms the synaptic rise time. The reset kernel is $\kappa(s) = \kappa_0 e^{-s/\tau_m}$, with $\kappa_0 = -5$ mV. Both kernels are set to 0 for $s < 0$. Only PSP's evoked by presynaptic spikes after $\hat{t}$ are summed over, ensuring a hard-reset at $\hat{t}$ to prevent excessive postsynaptic spiking [5]. Refractory effects do not accumulate, where only the most recent postsynaptic spike at time $\hat{t}$ contributes.

Output spike times are distributed according to an instantaneous firing rate or escape rate $\rho(t) = \rho(u(t))$ [9], where the probability for a spike being generated at time t over a small interval δt is given as $\rho(t) \delta t$. Throughout our simulations we take $\delta t = 1$ ms, being the simulation time step.

In our simulations we consider two different functional forms for the escape rate, the first being the Arrhenius & Current (A&C) model [9]:

$$\rho^{\text{A\&C}}(u, \dot{u}) = 2 \left(\frac{c_1}{\tau_m} + \frac{c_2}{\sigma} [\dot{u}]_+ \right) \frac{\exp \left\{ -\frac{[u - \vartheta]^2}{\sigma^2} \right\}}{1 + \operatorname{erf} \left\{ -\frac{u - \vartheta}{\sigma} \right\}}, \quad (2)$$

where u is the time-dependent, noise-free trajectory of the neuron's membrane potential and $\dot{u} = du/dt$ its time derivative. We set the firing threshold as $\vartheta = -55$ mV. The noise amplitude is set as $\sigma = 5$ mV, that corresponds to the variation in u due to background noise. The parameters c_1 and c_2 are set

to 0.72 and $\frac{1}{\sqrt{\pi}}$ respectively [14]. The term $[\dot{u}]_+$ indicates that only positive gradients in the membrane potential contribute to the firing intensity, defined as $[\dot{u}]_+ = (|\dot{u}| + \dot{u})/2$. The error function erf is included to ensure a linear increase in the firing rate for $u > \vartheta$ [10].

The second simpler escape rate, the Exponential (EXP) model, is more commonly used [9]:

$$\rho^{\text{EXP}}(u) = \rho_0 \exp\left(\frac{u - \vartheta}{\Delta u}\right), \quad (3)$$

where $\rho_0 = 0.1 \text{ ms}^{-1}$ is the firing rate at threshold and $\Delta u = 3 \text{ mV}$ determines the degree of noise about the threshold $\vartheta = -55 \text{ mV}$. For both the A&C and EXP escape rate models, the stochasticity parameters selected were found to give the best performance on preliminary simulations.

2.2 Learning Algorithm

Learning takes place on an episodic basis, where each episode corresponds to the presentation of an input pattern lasting duration $T = 500 \text{ ms}$. Feedback to the network is given in the form of a success signal $S(R)$, where S is a monotonic function of the reward R . The delivery of reward to the network is delayed until the end of each episode, therefore it is necessary to introduce an eligibility trace $\tilde{e}_j$ that determines candidate changes in the weights w_j [1]. More specifically, weights are updated at the end of each episode according to [5, 11]:

$$\tau_e \frac{d\tilde{e}_j}{dt} = -\tilde{e}_j + \eta e_j(t) \quad (4)$$

$$\Delta w_j = S(R) \tilde{e}_j(T), \quad (5)$$

where the time constant $\tau_e = 500 \text{ ms}$ is matched to the duration of each episode, η is the learning rate and $e_j(t)$ the synaptic eligibility. We use the R-max learning rule in our simulations, where the eligibility is given as [4, 13]:

$$e_j(t) = \frac{\rho'(u)}{\rho(u)} [\mathcal{Y}(t) - \rho(u)] \int_{\hat{t}}^t \epsilon(t - t') X_j(t') dt', \quad (6)$$

where $\rho'(u) = \frac{d\rho(u)}{du}$ and $\mathcal{Y}(t) = \sum_f \delta(t - t^f)$ is the output spike train.

In [8], we originally determined the eligibility for the A&C model, given here as:

$$e_j^{\text{A\&C}}(t) = \mathcal{A}(u) [\mathcal{Y}(t) - \rho^{\text{A\&C}}(u, \dot{u})] \int_{\hat{t}}^t \epsilon(t - t') X_j(t') dt' \quad \text{with} \quad (7)$$

$$\mathcal{A}(u) = \frac{2}{\sigma} \left(\frac{1}{\sqrt{\pi}} \frac{\exp\left\{-\frac{[u - \vartheta]^2}{\sigma^2}\right\}}{1 + \text{erf}\left\{-\frac{u - \vartheta}{\sigma}\right\}} - \frac{u - \vartheta}{\sigma} \right). \quad (8)$$

For the EXP model, the eligibility is:

$$e_j^{\text{EXP}}(t) = \frac{1}{\Delta u} [\mathcal{Y}(t) - \rho^{\text{EXP}}(u)] \int_{\hat{t}}^t \epsilon(t - t') X_j(t') dt'. \quad (9)$$

2.3 Learning a Target Spike Train

To teach the network to map an input pattern to a target spike train, we follow the same approach as [8]. Specifically, at the end of each learning episode, the van Rossum Distance (vRD) [15] is used to measure the dissimilarity between the output/target spike trains, giving the metric $\mathcal{D}$, with the coincidence time constant set as $\tau_c = 15$ ms. $\mathcal{D} \in [0, \infty)$ is then mapped to a reward value $R \in (0, 1]$ as $R = \exp(-\alpha\mathcal{D})$, where we set $\alpha = 2$ such that reward becomes negligible for large $\mathcal{D}$. Furthermore, we set $\mathcal{R} = 0$ for no output spikes, since reinforcing a lack of firing activity would negatively impact on learning. Finally, we define the success signal as $S(R) = R - \bar{R}$, with $\bar{R}$ being a moving average of the reward: $\bar{R} \leftarrow \bar{R} + (\bar{R} - R)/\tau_R$ with each episode, where the time constant is set to $\tau_R = 5p$ for p input patterns. Taking a moving average has been shown to reduce the impact of noise on learning [5], that allows for a larger learning rate to enable more rapid learning.

2.4 Plasticity Rules

During learning, weights are 'hard bound' to the range $[w_{\min}, w_{\max}]$, where we set $w_{\min} = -10$ and $w_{\max} = 10$ as the minimum and maximum allowable values. Weights are additionally subject to synaptic scaling [17], where weights are additively modified by the following scaling rule:

$$\Delta w_j = \gamma |w_j| [N^{\text{ref}} - N^{\text{out}}] , \quad (10)$$

where $\gamma = 1 \times 10^{-3}$ is the scaling strength, $N^{\text{ref}} = 1$ the number of target spikes and N^{out} the number of output spikes over each episode. As found in [16, 2], synaptic scaling decreases the sensitivity of the network to its initial state, and is essential to prevent extremes in the firing rate of the readout neuron whilst learning target spike trains.

2.5 Network Setup and Pattern Statistics

We implemented a two-layer fully-connected feedforward network, with 500 input neurons and a single readout neuron. Either the A&C or the EXP escape rate defined the spiking activity of the readout neuron. Input patterns consisted of Poisson-distributed spike trains, independently generated at each input neuron, with a mean firing rate of 6 Hz. Synaptic weights were initialized by independently selecting each value from a Gaussian distribution, with the mean and standard deviation both equal to 0.09 and 0.06 for A&C and EXP respectively. This gave an initial distribution of weights where $\sim 16\%$ had negative values, and drove the initial firing rate of the readout neuron to ~ 2 Hz. Depending on the learning task, jitter was added to input pattern presentations for increased realism, where each input spike was randomly selected from a normalised Gaussian distribution centered around its reference time with an amplitude ς .

3 Results

The network was tasked with learning to map between $p = 10$ input patterns and 10 target output spike trains, where each input pattern was assigned a unique target. Each target spike train contained a single spike, with the timing of spikes distributed as $k T/(p + 1)$ for each target $k \in \{1, \dots, p\}$ [3]. In all cases, we set the number of learning episodes to $5000p$, where input patterns were randomly selected to be presented to the network on each episode. For the learning rate, we set $\eta = 20$ for both A&C and EXP to ensure convergence in learning by the end of each simulation run, and to give the best performance for each escape rate.

The vRD measure $\mathcal{D}$ was used to quantify the accuracy of network output spikes with respect to its target spike times. To give a perspective of this measure, for a single actual output spike, values for $\mathcal{D} < 1/2$ correspond to an output timing that lies within 10 ms its target. A value $\mathcal{D} = 0$ corresponds to an exact match. Since we implemented a stochastic neuron model in our simulations, it was necessary to take a moving average of $\mathcal{D}$ with each episode: $\tilde{\mathcal{D}} \leftarrow (1 - \lambda)\tilde{\mathcal{D}} + \lambda\mathcal{D}$, with the timing parameter $\lambda = 2/(1 + 100p)$.

Figure 1 shows the progression in learning of the network, where either the A&C or EXP escape rate defined the readout neuron. In our simulations, we also considered the effect of input noise on learning by jittering input spike times with an amplitude $\varsigma = 5$ ms, corresponding to a mean timing displacement of $\Delta t \approx 4$ ms. A&C was found to be somewhat more accurate than EXP, and converged much more rapidly during learning. The increased accuracy of A&C over EXP held for both noiseless and jittered input patterns. In the case of no input noise, the final distances were found to be $\tilde{\mathcal{D}} = 0.31 \pm 0.04$ and $\tilde{\mathcal{D}} = 0.37 \pm 0.04$ for A&C and EXP respectively, and for jittered inputs the final distances were $\tilde{\mathcal{D}} = 0.235 \pm 0.009$ and $\tilde{\mathcal{D}} = 0.31 \pm 0.02$ for A&C and EXP respectively (errors given as standard error of mean). It was also found that adding input jitter led to a dramatic reduction in the variation of the network responses, and especially for A&C, where the variation was reduced by a factor of at least 4. This improvement in the performance can likely be attributed to a reduction in the overlearning of target responses, since the timing of individual input spike times could no longer be relied upon to always facilitate accurate output spikes.

To measure the performance of the network in terms of its ability to classify input patterns based on the timing of output spikes, we introduced an episodic performance measure P that takes the value $P = 100\%$ given a single output spike that lies within 10 ms of its target, and $P = 0$ otherwise [8]. Similarly as before, it was necessary to take a moving average of the performance with each episode: $\tilde{P} \leftarrow (1 - \lambda)\tilde{P} + \lambda P$.

Figure 2 shows the performance of the network with each episode, that can be interpreted as the classification rate. As before, results were obtained for both noiseless and jittered input patterns, with the same input noise amplitude $\varsigma = 5$ ms. It was found that A&C converged more rapidly than EXP, with a greater convergent performance, for both noiseless and jittered inputs. The increased performance of A&C over EXP was most pronounced for noiseless

6 Learning to Map Input-Output Spike Patterns

Fig. 1. Distance $\tilde{D}$ as a function of the episode number, when learning to map between 10 input-output spike patterns, with single-spike targets. (Left) No input jitter. (Right) Gaussian jitter is added to the presentation of each input pattern. (Top) The A&C escape rate. (Bottom) The EXP escape rate. Each data point was averaged over 24 different realizations, each with a random network initialization. The shaded region around each curve shows the standard deviation.

inputs, with a convergent performance $\tilde{P} = 81 \pm 3\%$ and $\tilde{P} = 67 \pm 6\%$ for A&C and EXP respectively (errors given as standard error of mean). Both escape rate models benefited from input jitter, in terms of having a reduced variation in the performance. However, only EXP found an improvement in the convergent performance, with final values $\tilde{P} = 81 \pm 1\%$ and $\tilde{P} = 70 \pm 3\%$ for A&C and EXP respectively.

4 Discussion

This paper applied a reward-maximising learning rule to teach a spiking neural network to generate temporally precise target spike times in response to multiple input patterns. We also compared the performance of two escape rate functions that define neuronal spike-timing: A&C and EXP, both for noiseless and jittered input patterns.

We found A&C to outperform EXP when learning to map between 10 input patterns and 10 single-spike target trains: both in terms of the convergent performance and the time taken to converge in learning. A&C also demonstrated greater reliability over EXP during learning, with a smaller variance for both

Fig. 2. Performance $\tilde{P}$ as a function of the episode number when learning to classify 10 input patterns by single-spike outputs, with an accuracy of at least 10 ms between an actual output spike time and target spike time. (Left) No input jitter. (Right) Gaussian jitter is added to the presentation of each input pattern. (Top) The A&C escape rate. (Bottom) The EXP escape rate. These results correspond to those shown in figure 1, with 24 averages per data point and the shaded region showing the standard deviation.

noiseless and jittered inputs. These results further support the advantages of implementing A&C over EXP when learning target spike trains, that was first indicated in [8]. We assume this arises from the the gradient term in the A&C model $\dot{u}(t)$, that provides the model with an additional degree of freedom over EXP during learning. Additionally, both escape rates benefited from jittered input patterns, which we attributed to a reduction in the overlearning of target spike trains.

With our setup we only considered single-spike target trains. Teaching the network to map between multiple input patterns and multiple-spike target trains would increase the complexity of the task significantly, since reward delivery is delayed and the network would have to stochastically explore a wider range of possible output spike times for learning to succeed. Single-spike targets was not an unreasonable choice: especially given the evidence that the brain has been found to utilize a first-spike latency coding scheme [18]. However, future work could aim to further quantify the performance of the network when learning to classify more than 10 input patterns.

Acknowledgement. BG was fully supported by EPSRC grant EP/J500562/1.

References

1. Barto, A., Sutton, R.: Reinforcement learning: An introduction. MIT press Cambridge, MA (1998)
2. Farries, M., Fairhall, A.: Reinforcement learning with modulated spike timing dependent synaptic plasticity. *Journal of Neurophysiology* 98(6), 3648–3665 (2007)
3. Florian, R.V.: The chronotron: A neuron that learns to fire temporally precise spike patterns. *PloS ONE* 7(8), e40233 (2012)
4. Florian, R.: Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation* 19(6), 1468–1502 (2007)
5. Frémaux, N., Sprekeler, H., Gerstner, W.: Functional requirements for reward-modulated spike-timing-dependent plasticity. *The Journal of Neuroscience* 30(40), 13326–13337 (2010)
6. Frémaux, N., Sprekeler, H., Gerstner, W.: Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Computational Biology* 9(4), e1003024 (2013)
7. Friedrich, J., Urbanczik, R., Senn, W.: Spatio-temporal credit assignment in neuronal population learning. *PLoS Computational Biology* 7(6), e1002092 (2011)
8. Gardner, B., Grüning, A.: Learning temporally precise spiking patterns through reward modulated spike-timing-dependent plasticity. In: *Artificial Neural Networks and Machine Learning–ICANN 2013*, pp. 256–263. Springer (2013)
9. Gerstner, W., Kistler, W.: Spiking neuron models: Single neurons, populations, plasticity. Cambridge University Press, Cambridge (2002)
10. Herrmann, A., Gerstner, W.: Noise and the psth response to current transients: I. general theory and application to the integrate-and-fire neuron. *Journal of Computational Neuroscience* 11(2), 135–151 (2001)
11. Izhikevich, E.: Solving the distal reward problem through linkage of stdp and dopamine signaling. *Cerebral Cortex* 17(10), 2443–2452 (2007)
12. Legenstein, R., Pecevski, D., Maass, W.: A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Computational Biology* 4(10), e1000180 (2008)
13. Pfister, J., Toyozumi, T., Barber, D., Gerstner, W.: Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural Computation* 18(6), 1318–1348 (2006)
14. Plesser, H., Gerstner, W.: Noise in integrate-and-fire neurons: from stochastic input to escape rates. *Neural Computation* 12(2), 367–384 (2000)
15. Rossum, M.: A novel spike distance. *Neural Computation* 13(4), 751–763 (2001)
16. Sporea, I., Grüning, A.: Supervised learning in multilayer spiking neural networks. *Neural Computation* 25(2), 473–509 (2013)
17. Van Rossum, M., Bi, G., Turrigiano, G.: Stable hebbian learning from spike timing-dependent plasticity. *The Journal of Neuroscience* 20(23), 8812–8821 (2000)
18. VanRullen, R., Guyonneau, R., Thorpe, S.: Spike times make sense. *Trends in Neurosciences* 28(1), 1–4 (2005)
19. Vasilaki, E., Frémaux, N., Urbanczik, R., Senn, W., Gerstner, W.: Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail. *PLoS Computational Biology* 5(12), e1000586 (2009)