ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

Controlling Biological Neural Networks with Deep Reinforcement Learning

Jan Wülfing*[1], Sreedhar S Kumar*[2,3], Joschka Boedecker[1],
Martin Riedmiller[4], Ulrich Egert[2,3] †

1- Univ. of Freiburg - Dept. of Computer Science, Germany
2- Univ. of Freiburg - Dept. of Microsystems Engineering, Germany
3- Univ. of Freiburg - Bernstein Center Freiburg, Germany
4- DeepMind - London, UK

Abstract. Targeted interaction with networks in the brain is of immense therapeutic relevance. The highly dynamic nature of neuronal networks and changes with progressive diseases create an urgent need for closed-loop control. Without adequate mathematical models of such complex networks, however, it remains unclear how tractable control problems can be formulated for neurobiological systems. Reinforcement learning (RL) could be a promising tool to address such challenges. Nevertheless, RL methods have rarely been applied to live, plastic neural networks. This study demonstrates that RL methods could help control response properties of biological neural networks with little prior knowledge of their complex dynamics.

1 Introduction

Electrical stimulation of the brain is a promising strategy to manage the symptoms of neurological disorders like epilepsy and Parkinson's disease. Networks in the brain are highly dynamic with activity patterns evolving under the influence of multiple dynamic processes including disease progression. Hence, for more effective outcomes, there is an urgent need for closed-loop interaction schemes where stimuli adapt appropriately to activity in the network. Without adequate mathematical models of the complex activity dynamics and their interaction with stimuli, it is unclear how suitable control laws can be found. To address this challenge, we propose to use model-free RL to learn stimulation policies by trial-and-error. We used a living system – generic biological neural networks (BNN) grown on microelectrode arrays – to develop and validate our approach. Unlike typical RL benchmarks, our model system is impossible to simulate in real-time due to its biophysical complexity. Its observation space is high-dimensional and each instantiation is unique. Since its neurons undergo plastic changes, activity is highly variable and fluctuates over a range of time scales. Together, this makes designing and tuning RL algorithms a difficult problem.

*These authors contributed equally

†This study was supported by the BrainLinks-BrainTools Cluster of Excellence (DFG, grant number EXC 1086), BMBF(FKZ 01GQ0830) and the EU (NAMASEN #264872)

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

2 Biological Neural Networks

The BNNs in this study consisted of approximately 250.000 primary dissociated rat cortical neurons cultured on substrate integrated microelectrode arrays (MEA). MEAs had 60 electrodes (rectangular 6x10 grid) and allowed us to record extracellular potentials and to deliver electrical stimuli. Interaction with such BNNs preserves many of the challenges RL algorithms will face in a neurotechnological context such as high-dimensional state spaces, continuous action spaces and non-stationary activity dynamics. Since they retain the richness of cellular level processes and interacting neurophysiological mechanisms underlying neuronal network dynamics, concepts and challenges emerging from co-adaptive interactions between BNNs and RL are expected to be generalizable. The production and culturing process of BNNs was as described in [4].

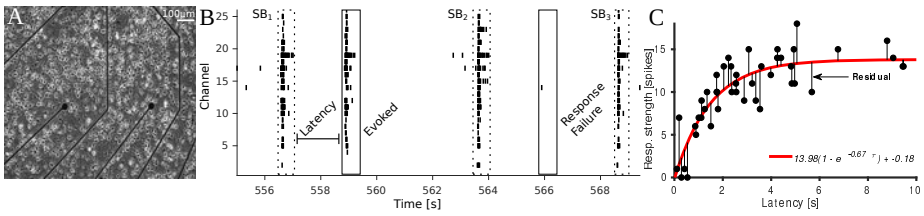

Fig. 1: (A) Phase contrast micrograph of the network on a MEA (DIV 21). (B) Raster plot of ≈ 15 s of activity recorded from a BNN. Spikes detected at 25 electrodes, network-wide SBs (dotted box) and responses to electrical stimulation (solid box) at two different sites are shown. (C) Response strengths at an electrode increased exponentially with stimulus latencies from preceding SBs.

Recordings were performed at 19–35 days *in vitro* (DIV). Networks exhibited an ongoing activity component, characterized by intermittent network-wide spontaneous bursts (SBs) separated by periods of reduced activity. Stimulating the network also evoked bursts of action potentials (responses, Fig. 1B). Ongoing SB activity is known to influence the network's interaction with external stimuli [8]. Response strength (RS) – the count of spikes detected in a 500 m sec post-stimulus interval – depended on the stimulus latency relative to the previous SB, and can be described by a saturating exponential model [8, 3](Fig. 1C). However, we found that this dependency was non-stationary when observed over long time scales. The evolution of residuals with respect to the model revealed a systematic structure (drifts and fluctuations; solid line in Fig. 2), the origin and nature of which remains unclear.

3 Task: Controlling Network Responses

Can an RL agent learn to achieve control of network responses to preset RSs and sustain it over long durations? This is a challenging question since (1) it is unknown a-priori if this can be achieved at all (since the model in Fig. 1C

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

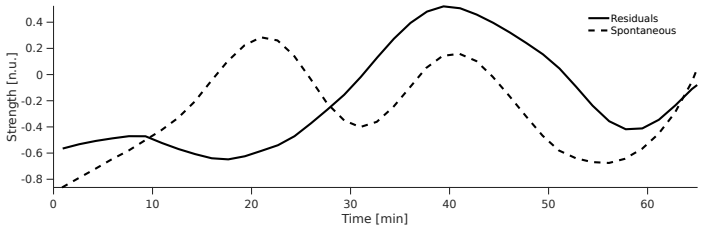

Fig. 2: Smoothed traces of normalized residuals (solid line) and SB strengths (dashed line).

applies exactly only on short time scales and its parameters drift over time) and (2) to solve it, the agent needs to track and adapt to the input-output dynamics of the vastly complex BNNs. It is, however, important to address these questions since adapting to background processes is a central challenge in the development of smart neurostimulation devices. We devised the following experiment: after each SB, the agent had to stimulate the network (trial) at a chosen stimulation electrode (SE), with an appropriate latency and amplitude (in some cases) such that the target number of spikes was evoked at a chosen recording electrode (RE). The agent forfeited the opportunity to stimulate when the network triggered SBs before stimulation. The cycle repeated until a preset number of trials was reached. For each network, we selected a feedback RE, a set of 3–5 possible SEs, an achievable target and a suitable range of latencies based on ongoing and evoked activity patterns. Site selection procedure was similar to that in [3]. Only networks not exhibiting superbursts (stereotyped trains of repetitive short bursts) during spontaneous activity were selected.

4 Reinforcement Learning

With RL we aim to infer the optimal sequence of actions – the action policy π – for an agent acting in a dynamic environment formalized as a Markov Decision Processes (MDP). An MDP is a 5-tuple $(\mathcal{S}, \mathcal{A}, P, R, \mathcal{T})$ consisting of a set of states $\mathcal{S} = \{\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(|\mathcal{S}|)}\}$, a set of actions $\mathcal{A} = \{\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(|\mathcal{A}|)}\}$, a stochastic transition model P, an immediate reward function R and a set of time points $\mathcal{T}$. Since P is unknown for BNNs, we employ model-free Q-learning [7] where the action-value function $Q^*(\mathbf{s}, \mathbf{a}) = \max_\pi \mathbb{E}_\pi \left[\sum_{t \in \mathcal{T}} \gamma^t R(\mathbf{s}_t, \pi(\mathbf{s}_t)) \mid \mathbf{s}_0 = \mathbf{s}\right]$ is learned as described below.

4.1 Deep Reinforcement Learning

Due to the curse of dimensionality and to facilitate generalization, the Q-function is usually approximated. In Deep RL the Q-function is approximated with an artificial neural network (ANN) that is parameterized by weights θ: $Q(\mathbf{s}, \mathbf{a}; \theta) \approx Q^*(\mathbf{s}, \mathbf{a})$. ANNs are powerful function approximators that allow to approximate non-linear Q-functions (linear function approximation failed in our experiments).

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

The ANN was trained by minimizing a sequence of loss functions $L_i(\theta_i) = \mathbb{E}_{\mathbf{s},\mathbf{a}}\left[(y_i - Q(\mathbf{s},\mathbf{a};\theta_i))^2\right]$, where $y_i = \mathbb{E}_{\mathbf{s}'}\left[r(\mathbf{s}) + \gamma \max_{\mathbf{a}'} Q(\mathbf{s}',\mathbf{a}';\theta_{i-1})|\mathbf{s},\mathbf{a}\right]$, here $\mathbf{s}'$ denotes the next state. We used a variant of NFQ [5] (see Algorithm 1) due to its sample efficiency and optimized it with stability in mind (important for envisioned therapeutic applications): The ANN was not reset after each episode and trained from scratch, but rather continuously optimized. A mini-batch training scheme and SMORMS3 [2] were used for optimization. The network consisted of 100×100 hidden rectified linear units. To avoid over-fitting, training was regularized with Dropout ([6], $p = 0.1$). In our experience, these changes improved learning stability on standard benchmark problems such as the cart-pole swing-up task. To accommodate dynamic changes of BNNs within the time scale of the learning experiments (see Fig. 2), the batch size was limited to 300 samples, which enabled the agent to forget old, possibly conflicting experiences.

Algorithm 1 NFQ with mini-batches and continuous optimization

$\mathcal{B}$ = batch of samples with limited capacity C
Setup ANN with random weights θ
for episode i = 0, K **do**
  Set current target network parameters: $\theta_i \leftarrow \theta$
  Sample trajectory according to current $\pi_i(\mathbf{s}) = \text{argmax}_{\mathbf{a}}\, Q(\mathbf{s},\mathbf{a};\theta_i)$ with epsilon-greedy exploration and add to batch $\mathcal{B}$
  for epoch e = 0, E **do**
    Shuffle $\mathcal{B}$
    for all mini-batches $b \in B$ **do**
      for each sample $(\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, r_j) \in$ b **do**

$$\text{Calc. targets } y_j = \begin{cases} r_j & \text{if } \mathbf{s}'_j \text{ is terminal state} \\ r_j + \gamma \max_{\mathbf{a}'} Q(\mathbf{s}'_j, \mathbf{a}'_j; \theta_i) & \text{else} \end{cases}$$

      Update current parameters: $\theta = \theta + \nabla_\theta (Q(\mathbf{s}_j, \mathbf{a}_j; \theta) - y_j)^2$

4.2 State Space, Action Space and Reward Function

We used the history of two evoked and spontaneously generated event strengths preceding a stimulus as state features for the learning problem. This was motivated as follows: Since observed fluctuations in response residuals (see Section 3) were slow and in the range of tens of minutes, the history of response strengths could be a useful indicator of the phase of the dynamic process. Moreover, SB events interspersed between stimulus trials also exhibited slow temporal trends in their strengths, though the coupling between SB strengths and residuals was found to evolve over time (Fig. 2). Since both fluctuations were likely mediated by the same background process, they may be also useful to predict subsequent responses. Additionally, since closely spaced stimuli are known to influence responses to successive trials, the last inter-stimulus interval was also included as a state feature [1, 8]. In total, the state was five dimensional. The agent could choose between actions that determined where and when a stimulation occurred.

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

Latencies were discretized in steps of 0.5 s. Since the chosen latency range and number of available SEs varied across networks, the cardinality of the action set also varied between 15-35. Stimuli were 400 µs wide and between -0.5 and -1 V in amplitude. The agent was rewarded with the negative absolute difference between target and current RS. In case the trial was interrupted by ongoing activity, a constant penalty was given.

5 Experiments

Only networks where the feedback site retained activity throughout the closed-loop session were analyzed (n=29 BNNs). We ran NFQ for each BNN from scratch in experiments that typically lasted 10-17 h (≈ 3500 trials). For validation purposes, each experiment started with a random exploration phase (these samples were also used for learning). This allowed us to compare performances of random stimulation and the learned policy. Furthermore, we added up to two SEs that consistently failed to evoke responses, to the action set (dud-sites). These were expected to be avoided by a reasonable learned policy. Improvement in goal-directed behavior was observed as the learned controller was deployed after random action exploration (dashed line in Fig. 3A). Thereafter, RSs stayed closed to target, albeit with small oscillations. The policy adapted over time – lower latencies were preferred during later stages of the session – with no noticeable changes in RSs (Fig. 3B). Further, duds were consistently the least preferred SEs after learning and multiple SEs were selected, with a systematic trend over time (SE 53 was preferred after trial 3000, Fig. 3C). Similar observations held for almost all networks studied. Improved target reachability and

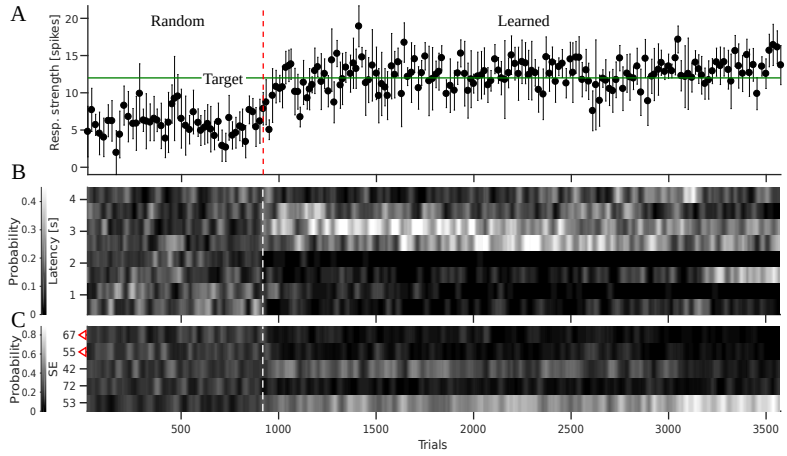

Fig. 3: (A) RSs ($\mu \pm \sigma$) binned every 5 min after smoothing (5-sample Gaussian window). (B, C) Latencies (B) and SEs (C) chosen by the controller during the session. Probabilities (in grayscale) were computed over a 10 trial sliding window. Triangles indicate the designated dud-sites.

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

reduced response failures were achieved in 27 out of 29 networks we studied, compared to a random policy (Fig. 4). Feedback instabilities were observed in two networks.

6 Conclusion

Overall, we demonstrated experimentally that Deep Reinforcement Learning is a promising tool to autonomously control response properties of a system as complex as a BNN, with only little knowledge about its activity dynamics.

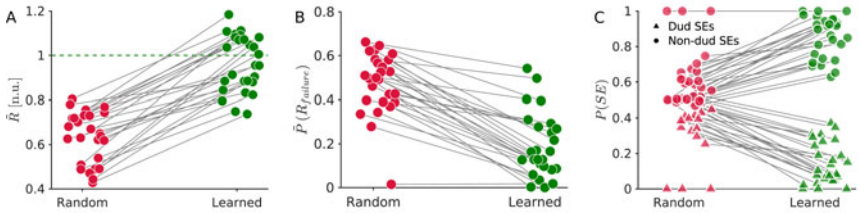

Fig. 4: (A) RSs after learning moved closer to target (R=1) in 27 out of 29 networks studied. (B) Response failures were also less likely after learning in each network. Circles correspond to the median value of each distribution. (C) Dud SEs were consistently the least preferred sites after learning. Circles and triangles correspond to the assigned non-dud and dud SEs respectively, for each network.

References

[1] D. Eytan, N. Brenner, and S. Marom. Selective adaptation in networks of cortical neurons. *J. Neurosci.*, 23(28):9349–9356, 2003.

[2] S. Funk. SMORMS3 - blog entry: RMSprop loses to SMORMS3 - beware the epsilon! `http://sifter.org/simon/journal/20150420.html`, 2015. Accessed: 2017-11-25.

[3] S. S. Kumar, J. Wülfing, S. Okujeni, J. Boedecker, M. Riedmiller, and U. Egert. Autonomous optimization of targeted stimulation of neuronal networks. *PLOS Comput. Biol.*, 12(8), 2016.

[4] S. Okujeni, S. Kandler, and U. Egert. Mesoscale architecture shapes initiation and richness of spontaneous network activity. *J. Neurosci.*, 37:2552–16, 2017.

[5] M. Riedmiller. Neural fitted Q iteration - First experiences with a data efficient neural Reinforcement Learning method. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3720 LNAI, pages 317–328, 2005.

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958, 2014.

[7] C. J. C. H. Watkins and P. Dayan. Technical Note: Q-Learning. *Mach. Learn.*, 8(3):279–292, 1992.

[8] O. Weihberger, S. Okujeni, J. E. Mikkonen, and U. Egert. Quantitative examination of stimulus-response relations in cortical networks *in vitro*. *J. Neurophysiol.*, 109(7):1764–1774, 2013.