Effects of STDP and Intrinsic Plasticity on Information Dynamics in a Self-Organising Recurrent Neural Network

Joschka Boedecker
Machine Learning Lab, University of Freiburg, 79110 Freiburg, Germany

Andreea Lazar
Ernst Strüngmann Institute (ESI) for Neuroscience in Cooperation with Max Planck Society, 60528 Frankfurt, Germany

Oliver Obst
CSIRO Computational Informatics, Epping, NSW 1710, Australia

Astrid Zeman
CSIRO Computational Informatics, Epping, NSW 1710, Australia and
Department of Cognitive Science, Macquarie University, Macquarie Park, NSW 2109, Australia

I. INTRODUCTION

Training recurrent neural networks (RNNs) with traditional training methods suffers from problems like slow convergence and vanishing gradients [1, 2]. The realisation of these fundamental issues led to alternative ways of using and training RNNs, some of which can be summarised in the field of Reservoir Computing methods (see Lukoševičius and Jaeger [3] for an overview), but specialised architectures like the Long Short Term Memory (LSTM) networks [4]) also exist. The Reservoir Computing field has been an active part of RNN research over the last decade, while there was less activity in gradient-descent-like methods which appear to generate renewed interest only recently [5], partially due to the development of more efficient training techniques as in [6].

Reservoir methods implement a fixed high-dimensional reservoir of neurons, using random connection weights between the hidden units, chosen small enough to guarantee dynamic stability. Input weights into this reservoir are also selected randomly, and reservoir learning procedures train only the output weights of the network to generate target outputs. A particular appeal of reservoir methods is their simplicity, and that the computation required for training is relatively low. On the other hand, methods like backpropagation through time (BPTT) [7] modify all weights of the network, but require a substantial amount of computation. Even though reservoir computing methods have been compared to cortical networks [8], reservoir computing methods in their basic form usually lack the variety of plasticity mechanisms that can be found in the neocortex. Experiments with individual plasticity mechanisms in RNNs, such as spike-timing-dependent plasticity (STDP) [9] or Intrinsic Plasticity (IP) [10] have proven difficult and led to mixed results at best, e.g., IP learning has shown success in some cases [11], but also has shown to be not effective in others [12].

SORN [13], a self-organising recurrent neural network with binary excitatory and inhibitory units, combines three distinct biologically plausible mechanisms of local plasticity. The synergistic effects of these mechanisms appear to help creating an efficient representation of spatio-temporal patterns presented to the network. In this work we investigate effects of the local plasticity mechanisms in SORN in terms of information storage and information transfer, to enable finding similar effective plasticity rules for other types of RNNs. We briefly present the network model, a definition of the measures we use, and some preliminary findings.

II. THE NETWORK MODEL

The network model distinguishes between excitatory and inhibitory units: with N^E excitatory units, the network uses $N^I = 0.2 \times N^E$ inhibitory units. The matrices W^{IE} and W^{EI} hold connections between inhibitory and excitatory units, and vice versa, both are fully connected. W^{EE} holds connections between excitatory units, these connections are random and sparse, with no self-recurrence. There are no direct connections between inhibitory units. All weights are drawn from the interval $[0, 1]$, and the three matrices W^{IE}, W^{EI}, and W^{EE} are normalised so that all "incoming" connections sum up to a constant, i.e., $\sum_j W_{ij} = 1$. For each discrete time step t, the network state is given by the two binary vectors $x(t) \in \{0, 1\}^{N^E}$, and $y(t) \in \{0, 1\}^{N^I}$, representing activity in the excitatory and the inhibitory layer, respectively. The network then evolves using the following update functions:

$$R_i(t+1) = \sum_{j=1}^{N^E} W_{ij}^{EE}(t)x_j(t) - \sum_{k=1}^{N^I} W_{ik}^{EI} y_k(t) - T_i^E(t) \tag{1}$$

$$x_i(t+1) = \Theta(R_i(t+1) + v_i^U(t)) \tag{2}$$

$$y_i(t+1) = \Theta(\sum_{j=1}^{N^E} W_{ij}^{IE}(t)x_j(t) - T_j^I) \tag{3}$$

T^E and T^I are threshold (bias) values, drawn randomly from positive intervals. Θ is the heaviside step function, and $v_i^U(t)$ the network input drive. Symbols are encoded so that a specific group of inputs are set ($v^U = 1$) when a symbol is active. Input is expanded to N^E dimensions, values in unused dimensions set to zero.

The plasticity mechanisms are STDP, synaptic scaling, and IP learning. STDP and synaptic scaling update excitatory connections, while IP changes the thresholds of excitatory units. The STDP rule for SORN is as follows, for some small learning constant η_{stdp}:

$$\Delta W_{ij}^{EE}(t) = \eta_{stdp}(x_i(t)x_j(t-1) - x_i(t-1)x_j(t)). \quad (4)$$

Synaptic scaling then normalises the values so that they sum up one:

$$\Delta W_{ij}^{EE}(t) = W_{ij}^{EE}(t) / \sum_j W_{ij}^{EE}(t). \quad (5)$$

Finally, IP learning is responsible for spreading activations more evenly and uses a learning rate $\eta_{ip} = 0.001$, and a target firing rate of $H_{IP} = 2 \times N^U / N^E$:

$$T_i^E(t+1) = T_i^E(t) + \eta_{ip}(x_i(t) - H_{IP} \quad (6)$$

III. INFORMATION DYNAMICS FOR RNNS

Active information storage has shown to be useful in general to analyse complex systems, and is based on Shannon entropy as the fundamental quantity. The concept is derived [14] as the information in an agent, process or variables past that can be used to predict its own future. Active information storage $A(X)$ expresses how much of the stored information is actually in use at the next time step when the next process value is computed. $A(X)$ is expressed as the mutual information between the semi-infinite past of the process X and its next state X', with $X^{(k)}$ denoting the last k states of that process:

$$A(X) = \lim_{k \to \infty} A^{(k)}(X) \quad (7)$$

$$A(X, k) = I(X^{(k)}; X') \quad (8)$$

Eq. (8) is also used to represent k-finite approximations of active information storage.

Transfer Entropy [15] is an information-theoretic measure for the information provided by a source about the next state of the destination which was not already contained in its own history. The transfer entropy from a source node Y to a destination node X is the mutual information between previous l states of the source $y_n^{(l)}$ and the next state of the destination x_{n+1},

$$T_{Y \to X} = \lim_{k,l \to \infty} \sum_{\mathbf{u}_n} p(\mathbf{u}_n) \log_2 \frac{p(x_{n+1} | x_n^{(k)}, y_n^{(l)})}{p(x_{n+1} | x_n^{(k)})}, \quad (9)$$

where $\mathbf{u}_n$ is the state transition tuple $(x_{n+1}, x^{(k)}, y_n^{(l)})$.

For our purposes, $T_{Y \to X}(k, l)$ represents finite k, l approximation.

To correctly estimate active information storage for input-driven systems, it has been proposed to condition out the input into the system [16]. The local input-corrected active information storage at time step $n+1$ for a process X with input U thus becomes:

$$a_X^U(n+1) = \lim_{k \to \infty} a_X^U(n+1, k) \quad (10)$$

$$a_X^U(n+1, k) = \log \frac{p(x_{n+1} | x_n^{(k)}, u_{n+1})}{p(x_{n+1} | u_{n+1})} \quad (11)$$

IV. RESULTS

Our preliminary results, computed using the information-dynamics toolkit JIDT [17], indicate that the plasticity mechanisms in SORN increase information storage and information transfer within the dynamical reservoir. We will further investigate how the individual mechanisms contribute to these effects. An interesting next step that we aim to address is also determination of possible stopping criteria for the pre-training by local plasticity mechanisms.

[1] Y. Bengio, P. Simard, and P. Frasconi, IEEE Transaction on Neural Networks **5**, 157 (1994).

[2] S. Hochreiter, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **6**, 107 (1998).

[3] M. Lukoševičius and H. Jaeger, Computer Science Review **3**, 127 (2009).

[4] S. Hochreiter and J. Schmidhuber, Neural computation **9**, 1735 (1997).

[5] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, *Advances in Optimizing Recurrent Networks*, arXiv preprint 1212.0901 (arXiv.org, 2012).

[6] J. Martens and I. Sutskever, in *Proceedings of the 28th International Conference on Machine Learning*, Vol. 46 (Omnipress Madison, WI, 2011) p. 68.

[7] P. J. Werbos, Proceedings of the IEEE **78**, 1550 (1990).

[8] A. Bernacchia, H. Seo, D. Lee, and X.-J. Wang, Nat Neurosci **14**, 366 (2011).

[9] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann, Science **275**, 213 (1997).

[10] J. Triesch, in *Proceedings of the International Conference on Artificial Neural Networks (ICANN 2005)*, Lecture Notes in Computer Science (Springer, 2005) pp. 65–70.

[11] B. Schrauwen, M. Wardermann, D. Verstraeten, J. J. Steil, and D. Stroobandt, Neurocomputing **71**, 1159 (2008).

[12] J. Boedecker, O. Obst, N. M. Mayer, and M. Asada, HFSP Journal **3**, 340 (2009).

[13] A. Lazar, G. Pipa, and J. Triesch, Frontiers in Computational Neuroscience **3** (2009), 10.3389/neuro.10.023.2009.

[14] J. T. Lizier, M. Prokopenko, and A. Y. Zomaya, Information Sciences **208**, 39 (2012).

[15] T. Schreiber, Physical Review Letters **85**, 461 (2000).

[16] O. Obst, J. Boedecker, B. Schmidt, and M. Asada, *On active information storage in input-driven systems*, arXiv preprint 1303.5526v1 (arXiv.org, 2013).

[17] J. T. Lizier, "JIDT: An information-theoretic toolkit for studying the dynamics of complex systems," `https://code.google.com/p/information-dynamics-toolkit/` (2013).