Towards Self-organized Online Extraction of Invariances Using a Hierarchy of Multiple-timescale Reservoirs

Joschka Boedecker[†] and Minoru Asada[*†]
[*]JST ERATO Asada Project
[†]Graduate School of Engineering, Osaka University
Suita 565-0871, Osaka, Japan
Email: {joschka.boedecker, asada}@ams.eng.osaka-u.ac.jp

Abstract

Real-world processes often generate data that have regularities on many different timescales. A robot operating in such an environment can benefit from extracting these regularities in order to build models for prediction of future events, and for decision making. A number of methods exist which enable this kind of feature extraction, but they are usually computationally very expensive, prohibiting their use in an online learning setting. We present an approach based on a hierarchy of reservoir networks with different timescales and argue that this setup is well suited to detect regularities and invariances in the sensory stream of a humanoid robot. In line with ideas from reservoir computing, many connections of the hierarchy are randomly initialized and remain fixed; only a few connections are trained by unsupervised, local learning rules. Training for the prediction of future sensory data is done by a recursive least squares method enabling efficient online learning. Preliminary results using synthetic data are presented, and future issues are outlined.

I. Introduction

Sensory data generated from real-world processes is often hierarchical and has structure on multiple timescales. The brain reflects this hierarchical organization in its anatomy, and also shows activity on a multitude of timescales [1]. Robots operating in real-world environments would benefit from a similar mechanism in order to extract useful patterns from their sensor stream for planning and decision making. Several methods exist to detect hierarchical spatiotemporal patterns in sensor data including Hidden Markov Models, Slow Feature Analysis (SFA) [2], and Recurrent Neural Networks (RNNs) (e.g. [3]). However, their suitability to continuous online learning is usually limited by their large computational cost. We propose an approach based on a hierarchy of multiple-timescale reservoir networks which are a special class of RNNs. Reservoir Computing (RC) approaches ([4], [5], [6] for a review, see [7]) have introduced important insights for simplified yet powerful algorithms. Briefly stated, the key concepts in RC are (a) to use a fixed random hidden and input layer connectivity and only train output connections, and (b) criteria for stability of the network. Similar to the ideas in RC, we fix a large number of connections in our three-layered network architecture, and only train some of the connections by local, unsupervised methods.

II. Background

A. Related Work

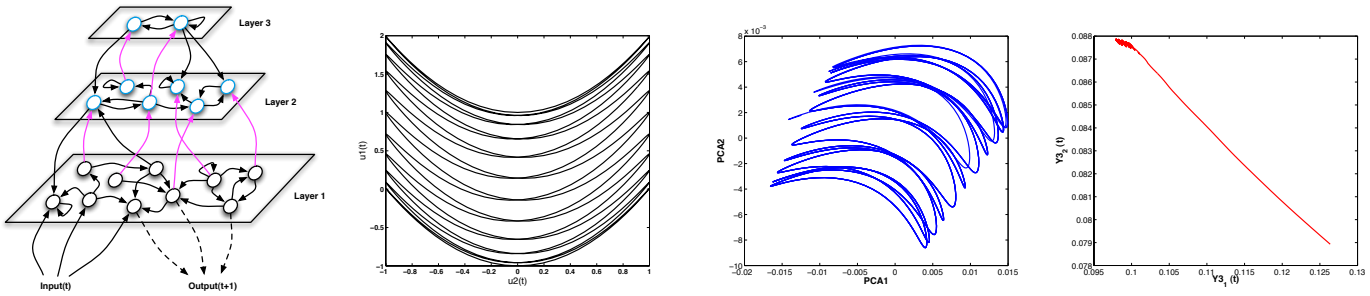
The work most closely related to our proposed architecture is Yamashita and Tani's hierarchical multiple-timescale recurrent neural network [3]. In fact, this architecture serves as the base for our investigations. An important difference in our work is that we do not employ any global learning rules based on back propagation of errors, but instead use local learning rules. These include homeostatic adaptation of the neurons' transfer function based on intrinsic plasticity (IP) [8], [9], an STDP-like mechanism, and synaptic normalization, as investigated in [10], [11] and [12].

III. Preliminary Experiments and Results

Following the work in [3], we use a 3-layered architecture in our network setup, as shown in Fig.1a. The number of neurons was 10 in the bottom layer, 6 in the middle, and 2 in the top layer. We used a leaky-integrator model for our neurons with time constant τ_i for neurons in layer i. The neuron activations $x^{(i)}$, and corresponding neuron outputs $y^{(i)}$ are updated according to

$$\mathbf{x}^{(i)}(t) = (1-\frac{1}{\tau_i})\mathbf{x}^{(i)}(t-1)+\frac{1}{\tau_i}(\mathbf{W}^{(i)}\mathbf{y}^{(i)}(t-1)+\mathbf{Win}^{(i)}\mathbf{y}^{(i-1)}(t)+\mathbf{Wout}^{(i+1)}\mathbf{y}^{(i+1)}(t)); \qquad \mathbf{y}^{(i)}(t) = \frac{1}{1+\exp(-\mathbf{x^{(i)}}(t))};$$

where $\mathbf{W}^{(i)}$ is the matrix of recurrent connection weights in layer i, $\mathbf{Win}^{(i)}$ is the matrix of input-to-hidden unit connections for layer i, and $\mathbf{Wout}^{(i)}$ is the output weight matrix of layer i. The input for layer i is the output of layer $i-1$, however in the first layer, this is the vector of inputs $\mathbf{u}$. In case of the last layer, there is no further layer from which to receive any output. The layers are successively fully connected, i.e. all neurons in layer 1 are connected to all neurons in layer 2, and all neurons in layer 2 are connected to all neurons in layer 3. Neurons of layers 1 and 3 are not connected to each other. All



(a) (b) (c) (d)

Fig. 1. (a) Illustration of the reservoir hierarchy: neurons in the three layers are connected randomly and remain fixed; dashed output connections of layer 1 are trained by Recursive Least Squares; connection in magenta going to higher layers are modified by an STDP-like learning rule; light blue neurons are modified by homeostatic mechanisms, namely IP and synaptic normalization. (b) The two-dimensional input signal. (c) Result of the principal component analysis of layer 2 hidden unit activations (PCA1 vs PCA2). A fast varying signal can be observed which strongly reflects the dynamics of the input signal. (d) Hidden unit activations of the layer 3 neurons. The signal varies much more slowly and many of the complex changes from the lower level activations are filtered out.

input and output connection weights are initialized uniformly random between $[-0.1, 0.1]$, and recurrent connections within a layer (reservoir connections) between $[-1.0, 1.0]$. Spectral radii for the reservoirs were initially scaled to 0.3, 0.6, and 0.9 for layers 1, 2, and 3, respectively. Output weights of layer 1 are trained with the Recursive Least Squares algorithm to predict the input at the next time step. Connections between layers 1 and 2 (matrix $\mathbf{Win}^{(2)}$) and between layers 2 and 3 (matrix $\mathbf{Win}^{(3)}$) were trained with a STDP-like learning rule in order to capture correlations in neural activity between those layers. Synaptic normalization and Intrinsic plasticity adaptation were also used for neurons in layers 2 and 3 to keep weights from growing too large and to keep the firing activity of neurons in a desired range, respectively.

The experiments we performed are very preliminary and the results mainly serve to illustrate the overall workings of the architecture. We trained the network hierarchy as described above to predict the input at the next time step for the signal (cf. [2]) $u_1(t) = \sin(t) + \cos^2(11t),\ u_2 = \cos(11t),\ t \in [0, 2\pi]$, exhibiting a slower and a faster timescale (see Fig. 1b). As shown in Figures 1c and 1d, the network dynamics after learning reflect these time scales in their activations. Prediction performance for this – very easy – task, achieved a normalized root mean squared error of 0.00329.

IV. Conclusion and Future Work

The presented work is just a very first step towards the investigation of these hierarchical multi-scale reservoirs and their adaptation by local, unsupervised learning rules. Many issues need to be studied, most importantly, how the different learning rules interact based on the timescale of the neurons in a given layer, and across different layers. It also needs to be clarified if the dynamical features extracted at the top level can indeed be used for actual robot tasks, for instance for automatic segmentation and sequencing of motor primitives as in [3], and what performance benefits can be achieved.

Acknowledgment

This study is partially supported by Grants-in-Aid for Scientific Research (Research Project Number: 22220002).

References

[1] S. J. Kiebel, J. Daunizeau, and K. J. Friston, "A hierarchy of time-scales and the brain," *PLoS Comput Biol*, vol. 4, no. 11, p. e1000209, 11 2008.
[2] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural Computation*, vol. 14, no. 4, pp. 715–770, 2002.
[3] Y. Yamashita and J. Tani, "Emergence of functional hierarchy in a multiple timescale neural network model: A humanoid robot experiment," *PLoS Comput Biol*, vol. 4, no. 11, p. e1000220, 11 2008.
[4] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks," GMD – German National Research Institute for Computer Science, Tech. Rep. 148, 2001.
[5] H. Jaeger and H. Haas, "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.
[6] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, 2002.
[7] M. Lukosevicius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.
[8] J. Triesch, "A gradient rule for the plasticity of a neuron's intrinsic excitability," in *Artificial Neural Networks: Biological Inspirations – ICANN 2005*. Springer, 2005, pp. 65–70.
[9] J. J. Steil, "Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning," *Neural Networks*, vol. 20, no. 3, pp. 353–364, Apr. 2007.
[10] J. Triesch, "Synergies between intrinsic and synaptic plasticity mechanisms." *Neural Computation*, vol. 19, no. 4, pp. 885–909, 2007.
[11] A. Lazar, G. Pipa, and J. Triesch, "Fading memory and time series prediction in recurrent networks with different forms of plasticity," *Neural Networks*, vol. 20, no. 3, pp. 312–322, 2007.
[12] ——, "Sorn: a self-organizing recurrent neural network," *Frontiers in Computational Neuroscience*, vol. 3, no. 23, 2009.