Studies on Reservoir Initialization and Dynamics Shaping in Echo State Networks

Joschka Boedecker[1], Oliver Obst[2], N. Michael Mayer[3], and Minoru Asada[1,4]

1- Dep. of Adaptive Machine Systems, Osaka University, Suita, Osaka, Japan

2- CSIRO ICT Centre, Autonomous Systems Laboratory
Locked Bag 17, North Ryde, NSW 1670 - Australia

3- Department of Electrical Engineering, National Chung Cheng University
Chia-Yi, Taiwan, R.O.C.

4- JST ERATO Asada Synergistic Intelligence Project, Suita, Osaka, Japan

Abstract. The fixed random connectivity of networks in reservoir computing leads to significant variation in performance. Only few problem specific optimization procedures are known to date. We study a general initialization method using permutation matrices and derive a new unsupervised learning rule based on intrinsic plasticity (IP) for echo state networks. Using three different benchmarks, we show that networks with permutation matrices for the reservoir connectivity have much longer memory than the other methods, but are also able to perform highly non-linear mappings. We also show that IP based on sigmoid transfer functions is limited concerning the output distributions that can be achieved.

1 Introduction

In reservoir computing, a network with fixed, recurrent connections is used to solve the problem of slow convergence of traditional recurrent neural network learning approaches. Only connections to the output units are trained. Echo state networks (ESN) are one particular instance of reservoir computing. The reservoir of an ESN is typically created using random connection weights, where one condition has to be met: Connections have to be weak enough so that the states of two equally constructed ESN converge to each other for long enough input independent of their starting states. ESN learning is efficient, because only output weights are trained using a linear regression, but its performance is also dependent on the fixed random connections of the reservoir. Thus, the quality varies between different random initializations, and only few problem specific optimization methods have been presented yet (e.g. [1, 2, 3]).

We study the effect of approaches aiming to reduce this dependency by either initializing the reservoir not entirely at random [4, 5], or by adapting it online using *intrinsic plasticity* (IP) [6]. These methods have an effect on prediction quality and short-term memory capacity of ESN. The reservoir initialization method is based on the idea to optimally exploit the high dimensionality of the reservoir, while methods based on IP aim to adapt the reservoir for a high entropy of codes (Sect. 2). Furthermore, we investigate an IP adaptation for high sparsity of codes. All methods are experimentally evaluated (Sect. 3) and the results are discussed in Sect. 4. Finally, Sect. 5 concludes.

2 Reservoir initialization and adaptation

We briefly review a reservoir initialization method and revisit IP optimization methods, which have been used to maximize the entropy of reservoir neuron outputs [1, 2]. Subsequently, we develop an IP rule with the goal to achieve sparser codes, as these have been shown to improve information processing [7].

2.1 Reservoirs based on permutation matrices

Orthogonal networks [4] have an orthogonal reservoir matrix $\mathbf{W}$ (i.e. $\mathbf{W}\mathbf{W}^{\mathbf{T}} = 1$) and linear activation functions. These networks are inspired by a distributed version of a delay line, where input values are embedded in distinct orthogonal directions, leading to high memory capacity [4]. Permutation matrices, as used in [5], consist of randomly permuted diagonal matrices and are a special case of orthogonal networks. Here, and in [5], the hyperbolic tangent (tanh) activation function was used, in order to facilitate non-linear tasks beyond memorization.

2.2 IP learning and a rule for a Laplace output distribution

IP learning is based on the idea of adapting gain and bias of the transfer function in order to change the output distribution of a neuron. It has been used with exponential [1] and Gaussian [2] distributions. In both cases, IP led to an entropy maximization for the reservoir neuron outputs and improved the quality of the information encoding in the tested reservoirs. A Laplace distribution would lead to sparser codes than the Gaussian, and our hypothesis is that enough entropy would be preserved for a good input signal approximation. Researching Laplace output distributions was also suggested in [2] for similar reasons. Here, we derive an IP learning rule for this distribution to test our hypothesis:

Our neurons' transfer function are defined as $y = f(x) = \tanh(x)$. The reservoir node activation vector x here is given as $x(t) = diag(a)\mathbf{W}^{res}_{res}x(t-1) + diag(a)\mathbf{W}^{res}_{inp}u(t) + c$ with a being the gain vector, and c being the bias vector. The Laplace distribution is defined as $f(x \mid \mu, b) = \frac{1}{2b}\exp(-\frac{|x-\mu|}{b}), \quad b \neq 0$. Let $\tilde{p}(y)$ denote the sampled output distribution of a reservoir neuron and let the desired output distribution be $p(y)$, thus $p(y) = f(y \mid \mu, b)$. In the learning process, we try to minimize the difference between $\tilde{p}(y)$ and $p(y)$ which can be measured with the Kullback-Leibler divergence D_{KL}. Thus, we try to minimize:

$$D_{KL} = \int \tilde{p}(y) \log\left(\frac{\tilde{p}(y)}{\frac{1}{2b}\exp(-\frac{|y-\mu|}{b})}\right) dy$$

Analogous to the calculations in [6, 2], we derive the following learning rules for stochastic gradient descent with learning rate η:

$$\Delta c = -\eta\left(2y + \frac{y(1-y^2+\mu y)-\mu}{b|y-\mu|}\right). \quad y \neq \mu$$

$$\Delta a = -\eta(-\frac{1}{a}) - \eta\left(2xy + \frac{yx(1-y^2+\mu y)-\mu x}{b|y-\mu|}\right) = \frac{\eta}{a} + \Delta c x. \quad y \neq \mu$$

3 Experimental setup and results

In order to compare the different reservoir shaping and initialization methods, we tested them on three different standard benchmarks. The first set of experiments evaluated the short-term memory capacity (MC). In addition, we looked at one-step prediction performance on the 30th order NARMA and Mackey-Glass time-series. These tasks cover a reasonably wide spectrum of tests for different useful properties of reservoirs and are widely used in the literature (e.g. [8, 2, 3, 9, 1]).

For all of the experiments, we used ESN with 1 input and 100 reservoir nodes. The number of output nodes was 1 for the NARMA and Mackey-Glass tasks, and 200 for the MC evaluation. In the latter, the 200 output nodes were trained on the input signal delayed by k steps ($k = 1 \ldots 200$). The input weights were always initialized with values from a uniform random distribution in the range $[-0.1; 0.1]$. The output weights for each output node were computed offline using the pseudoinverse of a matrix $\mathbf{X}$ composed of the reservoir node activations over the last 1000 of a total of 2000 steps as columns, and the input signal. In the case of the MC task, the delayed input was used as follows: $\mathbf{W}_{res}^{out,k} = (\mathbf{u}_{1001-k \ldots 2000-k} * \mathbf{X}^{\dagger})^T$ with $\mathbf{X}^{\dagger}$ denoting the pseudoinverse, and $k = 1 \ldots 200$.

We tested ESN with four different conditions for the connectivity matrix of the reservoir. In condition **RND**, the reservoir matrix was initialized with uniform random values between $[-1; 1]$. Condition **PMT** tested a permutation matrix for the reservoir connectivity. Finally, we used IP optimization with a Gaussian distribution (cf. [2]) in **IPGAUSS** and a Laplace distribution (as described in Sect. 2.2) in **IPLAP**. For **IPGAUSS**, parameters μ and σ of the Gaussian were set to 0.0 and 0.09, respectively. For **IPLAP**, parameters μ and b were set to 0.0 and 0.08 (values empirically determined in both cases). For both IP methods the reservoir was pre-trained for 100000 steps with a learning rate of 0.0005. In all conditions, the spectral radius of the reservoir connectivity matrix was scaled to 0.95 (prior to pre-training in case of IP).

The input for the MC task was random values sampled from a uniform random distribution in the range $[-0.8; 0.8]$. For the NARMA task, the input time series was calculated as:

$$y(t+1) = 0.2y(t) + 0.004y(t) \sum_{i=0}^{29} y(t-i) + 1.5x(t-29)x(t) + 0.001,$$

with $x(t)$ being values sampled from a uniform random distribution between $[0, 0.5]$. The Mackey-Glass time-series for the last experiment was computed by integrating the system

$$\dot{y} = 0.2y(t-\tau)/(1 + y(t-\tau)^{10}) - 0.1y(t),$$

from time step t to $t+1$. The τ parameter was set to 17 to get a mildly chaotic behavior. Different input time-series were used for training the output weights and for testing in all cases. The input length was always 2000 steps. The first

Table 1: Average memory capacity and NRMSE for the NARMA and Mackey-Glass prediction tasks in the four different conditions (averaged over 30 simulation runs), standard dev. in parenthesis. Note that the errors for NARMA are scaled by a factor for 10^{-2} and the ones for Mackey-Glass by a factor of 10^{-5}.

	PMT	**RND**	**IPGAUSS**	**IPLAP**
Memory Capacity	**62.39** (4.22)	31.11 (2.47)	32.64 (3.77)	32.70 (2.35)
$\mathrm{NRMSE}_{\mathrm{NARMA}}$	**45.42** (5.47)	72.62 (4.3)	67.22 (3.53)	67.69 (4.18)
$\mathrm{NRMSE}_{\mathrm{Mackey-Glass}}$	33.18 (3.81)	**24.30** (2.38)	44.36 (3.81)	40.67 (15.84)

1000 steps of the reservoir node activations were discarded to get rid of transient states to due random initialization before calculating the output weights and the test error.

To evaluate the short-term memory capacity of the different networks, we computed the k-delay memory capacity (MC_k) defined by Jaeger in [8] as

$$MC_k = cov^2(\mathbf{u}_{t-k}, \mathbf{o}_t)/(\sigma^2(\mathbf{u}_{t-k})\sigma^2(\mathbf{o}_t))$$

The actual short-term memory capacity of the network is defined as $MC = \sum_{k=1}^{\infty} MC_k$, but since we can only use a finite number of output nodes, we limited their number to 200 which is sufficient to see a significant drop-off in performance for the networks in all of the tested conditions.

The evaluation for the NARMA and Mackey-Glass prediction tasks was done using the normalized root mean squared error measure, defined as:

$$NRMSE = \sqrt{\langle(\tilde{y}(t) - y(t))^2\rangle_t / \langle(y(t) - \langle y(t)\rangle_t)^2\rangle_t},$$

where $\tilde{y}(t)$ is the sampled output and $y(t)$ is the desired output.

The results of the experiments are given in Table 1, averaged over 30 simulation runs for each of the four conditions. The networks in the **PMT** condition show a memory capacity which is essentially double that of the other conditions, while networks pre-trained with **IPGAUSS** and **IPLAP** have very similar values and show a slight increase compared to condition **RND**. The results for the NARMA one-step prediction look similar in that the **PMT** networks perform significantly better than the other tested conditions. The NRMSE for **IPLAP** and **IPGAUSS** is very similar again, and slightly lower than the **RND** one. For the Mackey-Glass one-step prediction, the performance of the **RND** networks is better than the other ones. The **PMT** networks perform slightly better on this task than the pre-trained ones in **IPGAUSS** and **IPLAP**.

4 Discussion

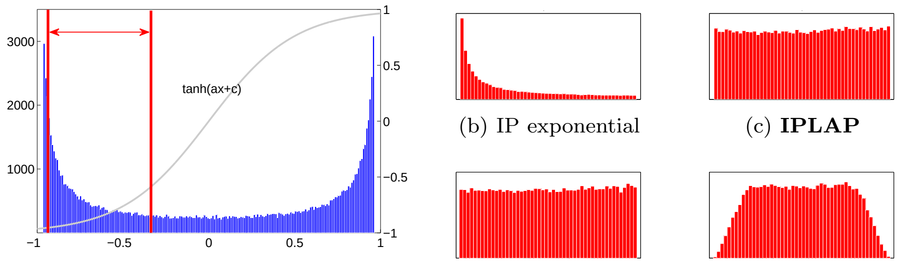
For the short-term memory capacity task, the networks in **PMT** perform much better than the others. This could be expected from the motivation for these networks presented in Section 2.1. They also outperform the other methods on the highly non-linear NARMA task which is less obvious. The NARMA task needs long memory which the orthogonal reservoirs in **PMT** are able to provide, but one might suspect that the specific connectivity would not be able to perform the kind of non-linear mappings that the task requires. The results show that this is not the case. The Mackey-Glass prediction task requires shorter time constants and less memory than the other two tasks. In this case, the **RND** networks perform best. The **PMT** networks have the same spectral radius as the ones in **RND**, however, the longer memory in the **PMT** case seems to interfere with influence of the input. The IP adaptation has the tendency to increase the spectral radius of the reservoir connectivity matrix, resulting in even longer time constants which hurts performance for the prediction.

4.1 IP revisited

A closer investigation of the almost identical performance of both IP methods revealed that **IPLAP** also generated normally distributed output, very similar to **IPGAUSS**. To better understand the effect of the different IP rules, we used IP to approximate the Laplace, the Gaussian (both with a tanh activation function), and the exponential distribution (fermi activation function), respectively, with a single feedforward unit and uniformly distributed input. As expected, the IP learning rule can successfully generate exponentially distributed output values. IP fails, however, to generate output distributions that resemble the Gaussian or the Laplace (Fig. 1, b–d). This seems surprising in particular for the Gaussian, as IP has successfully been used to shape the output distribution of a reservoir [2]. From Fig. 1 a, it becomes clear why an approximation of some distributions is more difficult than others: given a uniform input distribution and a sigmoid transfer function, IP learning selects a slice from an output distribution that peaks towards either end of the input range, but never in the center. The output of an IP trained self-recurrent unit gives an intuition why it is possible to achieve a Gaussian output distribution in a reservoir (Fig. 1, e). From the central limit theorem it follows that the sum of many i.i.d. random variables approximates a Gaussian. Even though in case of a recurrent reservoir not all inputs to a unit will be i.i.d., IP has to make input distributions only similar to each other to approximate a normal distribution in the output. For uniform input and a single unit without recurrence, the best IP can do is to choose the linear part of the activation function, so that the output is also uniformly distributed. With self-recurrent connections, this leads to initially uniform distributions added up. The resulting output, and eventually the whole reservoir become more and more Gaussian. A consequence of our result is that IP with sigmoid transfer functions cannot be generalized to arbitrary distributions.

5 Conclusion

We studied different initialization and dynamics shaping methods for reservoirs of ESN. One of our findings is that networks with a reservoir connectivity based



(a) (b) IP exponential (c) **IPLAP** (d) **IPGAUSS** (e)

Fig. 1: (a) Uniform input and a tanh transfer function lead to the output distribution in the histogram. IP selects a slice of this distribution, as illustrated by the vertical lines. Adapting gain and bias changes width and position of the slice. (b–d): Effect of IP learning on a single feedforward neuron. (e) shows the effect of **IPGAUSS** for a single self-recurrent unit.

on permutation matrices are superior in tasks requiring long short-term memory capacity, but are also able to perform complex non-linear mappings. Furthermore, we derived a new learning rule for IP based reservoir adaptation, and found a limitation of this approach that prevents generalization to arbitrary distributions if sigmoid transfer functions are used.

Acknowledgements This work was partially supported by a JSPS Fellowship for Young Researchers, and by the JST ERATO Synergistic Intelligence Project. We thank the anonymous reviewers for their very constructive feedback.

References

[1] J. J. Steil. Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning. *Neural Networks*, 20(3):353–364, April 2007.

[2] B. Schrauwen, M. Wardermann, D. Verstraeten, J. J. Steil, and D. Stroobandt. Improving reservoirs using intrinsic plasticity. *Neurocomputing*, 71(7-9):1159–1171, 2008.

[3] A. A. Rad, M. Jalili, and M. Hasler. Reservoir optimization in recurrent neural networks using kronecker kernels. *IEEE Int. Symp. on Circuits and Systems*, pages 868–871, 2008.

[4] O. L. White, D. D. Lee, and H. Sompolinsky. Short-term memory in orthogonal neural networks. *Physical Review Letters*, 92(14):148102.1–148102.4, 2004.

[5] M. Hajnal and A. Lőrincz. Critical echo state networks. In *Artificial Neural Networks – ICANN 2006*, pages 658–667, 2006.

[6] J. Triesch. A gradient rule for the plasticity of a neuron's intrinsic excitability. In *Artificial Neural Networks: Biological Inspirations – ICANN 2005*, pages 65–70. Springer, 2005.

[7] D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6(4):559–601, 1994.

[8] H. Jaeger. Short term memory in echo state networks. GMD Report 152, German National Research Institute for Computer Science, 2001.

[9] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, German National Research Institute for Computer Science, 2001.