ESANN 2010 proceedings, European Symposium on Artificial Neural Networks - Computational Intelligence and Machine Learning. Bruges (Belgium), 28-30 April 2010, d-side publi., ISBN 2-930307-10-2.

Extending reservoir computing with random static projections: a hybrid between extreme learning and RC

John Butcher[1], David Verstraeten[2], Benjamin Schrauwen[2], Charles Day[1] and Peter Haycock[1] *

1- Institute for the Environment, Physical Sciences and Applied Mathematics (EPSAM) Keele University, Staffordshire, ST5 5BG, United Kingdom

2- Department of Electronics and Information Systems Ghent University, Sint-Pietersnieuwstraat 41, 9000 Ghent, Belgium

Abstract. Reservoir Computing is a relatively new paradigm in the field of neural networks that has shown promise in applications where traditional recurrent neural networks have performed poorly. The main advantage of using reservoirs is that only the output weights are trained, reducing computational requirements significantly. There is a trade-off, however, between the amount of memory a reservoir can possess and its capability of mapping data into a highly non-linear transformation space. A new, hybrid architecture, combining a reservoir with an extreme learning machine, is presented which overcomes this trade-off, whose performance is demonstrated on a 4^{th} order polynomial modelling task and an isolated spoken digit recognition task.

1 Introduction

A recent addition to the field of Recurrent Artificial Neural Networks (RANNs) is Reservoir Computing (RC) [1], which is based on using a randomly, sparsely connected reservoir of neurons combined with a linear readout. RC is a unification of several techniques such as Echo State Networks (ESNs), Liquid State Machines (LSMs) and the back-propagation decorrelation neural network (see [1] for more details on RC techniques). Recent studies using RC for predicting and classifying complex time-series data have shown that RC techniques outperformed conventional RANNs considerably (see [2] for an overview). This paper reports on the use of RC for tasks that require both non-linearity and memory, and in which a novel customised RC architecture is presented to better fulfill both of these requirements.

Standard reservoir architectures consist of three layers of neurons, an input layer, a recurrently connected reservoir layer and an output layer. Each neuron in a layer is randomly connected to every neuron in the next layer. The reservoir weights are globally scaled by the spectral radius, which is the largest absolute eigenvalue of the weight matrix. Usually a spectral radius value of less than, but close to, unity is used to create reservoirs with suitable dynamics [3]. Increasing

*SciSite Ltd and the UK EPSRC are gratefully acknowledged for supporting this research. The Department of Electronics and Information Systems from Ghent University are also gratefully acknowledged for their very kind and useful feedback.

the spectral radius leads to the spreading of activation distributions and the eventual saturation of activations towards +1 and -1. This results in a chaotic reservoir whose generalisation characteristics disappear. See [4] for a thorough overview of the effects of different parameters, including the spectral radius, on the dynamics of the reservoir. The reservoirs presented in the remainder of this work comprise ESN type networks, which means that the reservoir consists of tanh-neurons. The activations of reservoir and output neurons at time t are given by Equations 1 and 2 respectively.

$$\mathbf{x}(t) = f(\mathbf{W}_{res}^{input}\mathbf{u}(t) + \mathbf{W}_{res}^{res}\mathbf{x}(t-1)) \tag{1}$$

$$\mathbf{y}(t) = \mathbf{W}_{out}^{res}(\mathbf{u}(t), \mathbf{x}(t)) \tag{2}$$

where t is the current time step, $t-1$ is the previous time step, $u(t)$ is the input activation at time t, $x(t-1)$ is the vector of activations of the reservoir neurons at time $t-1$, and $f(x)$ is the activation function, which is tanh here. W_b^a denotes a weight matrix from a to b.

The main advantage of using RC is that only the reservoir-to-output connections $\mathbf{W}_{out}^{res}$ are modified during training which can be done in a one-shot fashion using linear regression, whereas training traditional RANNs involves modification of all weights in the network. This, therefore, makes RC techniques much faster to train, which is useful for real-world applications where fast, and sometimes real-time training is required. Linear regression training also guarantees convergence, which is not the case with most other RANN learning rules that are based on gradient-descent error minimisation algorithms. These two advantages combined overcome the main stumbling blocks of using RANNs within real-world engineering domains.

2 The need for non-linearity and memory

Reservoirs can be created to have long term memory or to have highly non-linear mapping capabilities, depending on the task at hand. However, at present, there is no parameter that allows to tune these properties independently. For instance, tuning the spectral radius affects the memory of the reservoir, but also the non-linear mapping. In order to overcome this problem, a new architecture is outlined which combines the memory capabilities of a reservoir with the non-linear separation capabilities of traditional ANNs.

2.1 R^{2}SP: a Reservoir with Random Static Projections

The use of a feedforward network where only the output weights are trained has been previously presented in [5] as the *Extreme Learning Machine* (ELM) and was later extended as the *Optimally-Pruned Extreme Learning Machine* (OP-ELM) [6]. Both techniques involve two layers of feedforward networks. The first layer's weights are chosen at random at network creation, while the second layer's weights are trained using a simple linear regression. Therefore, as with

RC, only the readout weighted connections are trained. OP-ELM extends ELM further by pruning the least significant neurons from the hidden layer, which regularises the network, improving its generalisation capabilities.

The novel architecture proposed here, named *Reservoir with Random Static Projections* (R^2SP), contains a standard reservoir and two *static* hidden layers of non-recurrent neurons with no within-layer connections, combining the underlying principles of ELM with RC. The first static layer receives the same inputs as the reservoir and has output connections to the output layer, thus giving an instant non-linear transformation of the dataset at each time step. The second static layer receives input from the reservoir, giving a non-linear mapping of the reservoir at each time step. The outputs of the reservoir and the two static layers are combined, and, as in RC and with the ELM, only the weights connecting to the output nodes are trained, using linear regression. The activations of the newly added static layers are calculated as follows:

$$\mathbf{x}_{sl_1}(t) = f(\mathbf{W}^{input}_{sl_1}\mathbf{u}(t)) \tag{3}$$

$$\mathbf{x}_{sl_2}(t) = f(\mathbf{W}^{res}_{sl_2}\mathbf{x}(t)) \tag{4}$$

where $\mathbf{x}_{sl_1}(t)$, $\mathbf{x}_{sl_2}(t)$, $\mathbf{W}^{input}_{sl_1}$ and $\mathbf{W}^{res}_{sl_2}$ are the activations and input weight matrices of static layers 1 and 2 respectively. All other symbols are as in Equations 1 and 2. Figure 1 shows a schematic overview of the R^2SP.

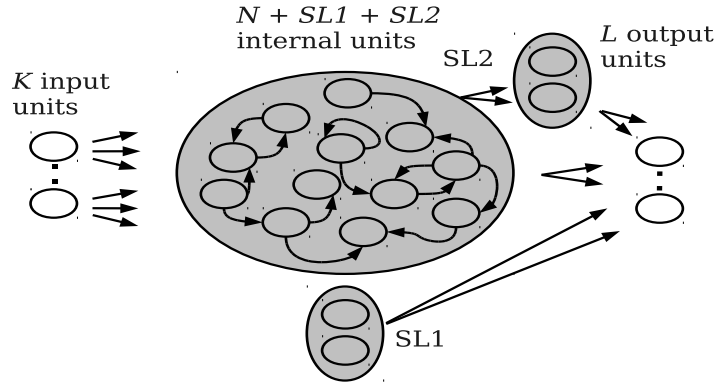

Fig. 1: Reservoir with Random Static Projections (R^2SP). SL1 and SL2 are the number of neurons in the respective static layers.

3 Experimental Setup and Datasets

The input scaling, spectral radius and reservoir size are known to influence the performance of a reservoir for a given task [1]. In order to find the optimal

settings for both reservoir architectures, separate parameter sweeps were carried out for both architectures for two different datasets. The leak rate (measure of how a neurons activation decreases over a period of time) of each neuron in the reservoir was also ranged over to investigate its effect on performance for both datasets, as it has been shown to improve performance by altering the dynamics of the reservoir to match the timescale of a given dataset. In order to overcome the randomness of reservoir weights at creation, 100 reservoirs of each type were trained[1].

3.1 Fourth order polynomial data

This artificial dataset contains random uniform inputs ranging from -1 to +1 and outputs determined by a 4^{th} order polynomial, with coefficients also chosen from a uniform random sample of -1 to +1. The dataset consisted of 20,000 datapoints and was split into 20 samples of 1,000 datapoints for training and testing. The output at time t was determined by the following equation:

$$\mathbf{y}_t = c_1 + c_2\mathbf{u}_t + c_3\mathbf{u}_{t-1} + ... + c_{15}\mathbf{u}_{t-1}^4 \tag{5}$$

where c denotes one of 15 random uniformly distributed coefficients. All other symbols are as described above. The output weights were calculated using ridge regression, with the performance of each reservoir calculated using the Normalised Root Mean Squared Error (NRMSE), using ten-fold cross-validation using a grid search for the optimal regularisation parameter.

3.2 Isolated spoken digits recognition task

The dataset is a subset of the TI46 isolated spoken digits dataset. It consists of 10 utterances of ten digits spoken by five different female speakers, resulting in 500 digit samples. Since this task is relatively easy to solve with standard reservoirs (in simulations not reported here, errors around 0 were obtained), it was made more challenging by adding babble noise[2] to the samples at a signal-to-noise ratio of 3dB. The samples were then preprocessed using the Lyon cochlear model with a subsampling factor of 128, resulting in a 77-dimensional time-varying feature vector. This 77-dimensional vector was used as input to both architectures. For this task, the error was again determined using ten-fold cross-validation but without optimization of the regularization parameter because the noise added to the dataset is enough to avoid overfitting.

4 Results

Both architectures contained 150 neurons (the R^2SP architecture consisted of 50 nodes in each layer). Table 1 shows the optimal parameters for both reservoir

[1] All training and testing was performed using the Opensource Matlab RCToolbox available from http://www.elis.ugent.be/rct.

[2] This is available from the NOISEX database from the Rice University: http://spib.rice.edu/spib/data/signals/noise/babble.html.

approaches and their corresponding error rates for both datasets. Varying the leak rate of the neurons for the polynomial task led to little improvement in performance for both architectures and therefore was not investigated further. Varying the leak rate for the digit recognition task did, however, affect the performance of both architectures. Figure 2 shows the test error plots of a standard reservoir and the R^{2}SP on the digit recognition task when varying over the spectral radius and the leak rate.

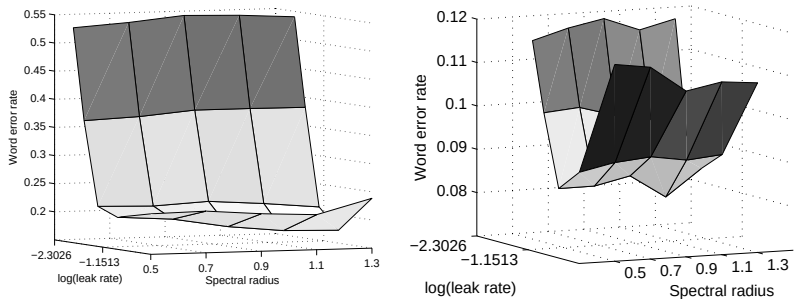

Fig. 2: Test error plots of the standard reservoir (left) and the R^{2}SP (right) when applied to the digit recognition task ranging over the leak rate and the spectral radius. Note the different scaling of the Z (error) axis.

Table 1: Errors obtained from a standard reservoir architecture (Std-Res) and the R^{2}SP on the polynomial (Poly) and the digit recognition (Speech) datasets.

Type	**Poly**		
	Error (NRMSE)	**Input Scale**	**Spec Rad**
Std Res	0.73	25.1	1.2
R^{2}SP	0.5	2.5	0.6
	Speech (Error in Word Error Rate)		
Std Res	0.17	1	1.3
R^{2}SP	0.08	1	1.1

5 Discussion

As the results show, the R^{2}SP outperforms a standard optimal reservoir quite significantly, particularly on the digit recognition task. Here the standard reservoir has a classification error of 17%, while a R^{2}SP achieves a smaller error of 8%. Analysis of Table 1 and Figure 2 shows that the optimal spectral radius of the R^{2}SP is lower than a standard reservoir. This indicates that the

static layers perform more of the non-linear mapping in the high dimensional state space, resulting in a reservoir with longer term memory as indicated by a lower spectral radius, therefore resulting in a system which has highly non-linear mapping capabilities and memory at the same time. Similar comments apply to the standard reservoir when applied to the polynomial dataset, as the input scaling to the reservoir and spectral radius are high, which results in a reservoir which has highly saturated states (activations at either +1 or -1 for the whole dataset). This creates a more non-linear reservoir which, therefore, has less memory capacity, giving a larger error rate.

6 Conclusion

A novel architecture, Reservoir with Random Static Projections (R^2SP), has been presented which combines the recurrent characteristics of a reservoir with the instantaneous non-linear separation capabilities of standard feedforward networks, enabling a reservoir to have both memory and non-linear mapping capabilities. Such characteristics are required for datasets with short lived non-linear features which also require memory, such as the polynomial and spoken digit recognition datasets presented. The ease of training of reservoirs is still present, as only the output weights of the newly added layers are trained alongside the reservoir outputs. As a result of the two combined characteristics, the new R^2SP architecture outperforms a standard reservoir approach significantly for the two tasks considered here.

Future work will include an investigation of applying further techniques used in the OP-ELM such as different activation functions and pruning of the least significant neurons to obtain an optimal network size for a given task. Further investigations will also be conducted on the application of the static reservoir architecture to data collected from the real world whose underlying characteristics are suspected to require highly non-linear mapping as well as short-term memory and should therefore benefit from the novel R^2SP architecture introduced here.

References

[1] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20:391–403, 2007.

[2] M. Lukosevicius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[3] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical report, German National Research Institute for Computer Science, 2001.

[4] X. Dutoit. *Reservoir Computing for Intelligent Mobile Systems*. PhD thesis, Department of Mechanical Engineering, Katholleke Universiteit Leuven, Belgium, 2009.

[5] G.B. Huang, Q.Y. Zhu, and C.H. Siew. Extreme learning machines: Theory and applications. *Neurocomputing*, 70:489–501, 2006.

[6] Y. Miche, A. Sorajamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse. OP-ELM: Optimally pruned extreme learning machine. *IEEE Transactions on Neural Networks*, 21(1):158–162, December 2009.