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/.

Short-term Memory of Deep RNN

Claudio Gallicchio

Department of Computer Science, University of Pisa
Largo Bruno Pontecorvo 3 - 56127 Pisa, Italy

Abstract. The extension of deep learning towards temporal data processing is gaining an increasing research interest. In this paper we investigate the properties of state dynamics developed in successive levels of deep recurrent neural networks (RNNs) in terms of short-term memory abilities. Our results reveal interesting insights that shed light on the nature of layering as a factor of RNN design. Noticeably, higher layers in a hierarchically organized RNN architecture results to be inherently biased towards longer memory spans even prior to training of the recurrent connections. Moreover, in the context of Reservoir Computing framework, our analysis also points out the benefit of a layered recurrent organization as an efficient approach to improve the memory skills of reservoir models.

1 Introduction

Deep learning is an attractive area of research in constant growth [1]. In particular, in the neuro-computing field, the study of deep neural networks composed by multiple non-linear layers has proved able to learn feature representations at progressively higher levels of abstraction, leading to eminent performance e.g. in vision tasks. Extending the benefits of depth to recurrent neural networks (RNNs) is an intriguing research direction that is recently gaining an increasing attention [2]. In this context, the study of deep RNNs has pointed out that hierarchically organized recurrent models have the potentiality of developing multiple time-scales representations of the input history in their internal states, which can be of great help, e.g., when dealing with text processing tasks [3]. More recently, studies in the area of Reservoir Computing (RC) [4, 5] have shown that the ability of developing such a structured state space organization is indeed an intrinsic property of layered RNN architectures [6, 7]. The study of deep RC networks on the one hand allowed the development of efficiently trained deep models for learning in the temporal domain, and on the other hand it paved the way to further studies on the properties of deep RNNs dynamics even in the absence of (or prior to) learning of the recurrent connections.

An aspect of prominent relevance in the study of dynamical models is represented by the analysis of their memory abilities. In this paper, exploiting the ground provided by the deep RC framework, we explicitly address the problem of analyzing the short-term memory capacity of individual (progressively higher) layers in deep recurrent architectures. Contributing to highlight the intrinsic diversification of transient state dynamics in hierarchically constructed recurrent networks, our investigation aims at shedding more light on the bias of layering in the RNN architectural design. Framed in the RC area, our analysis is also intended to provide insights on the process of reservoir network construction.

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 Deep Stacked RNN

We consider deep RNNs [3] whose recurrent architecture is obtained by a stacked composition of multiple non-linear recurrent hidden layers, as illustrated in Fig. 1. The state computation proceeds by following the hierarchical network organization, from the lowest layer to the highest one. Specifically, at each time step t the first recurrent layer in the network is fed by the external input while each successive layer is fed by the activation of the previous one.

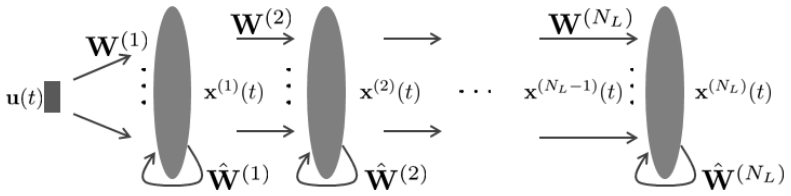

Fig. 1: Hierarchical organization of hidden layers in a deep RNN.

Under a dynamical system perspective, a deep RNN implements an input-driven discrete-time non-linear dynamical system, in which the state evolution in each layer i is ruled by a state transition function $F^{(i)}$. Here we denote the input dimension by N_U and assume, for the sake of simplicity, that each hidden layer contains N_R recurrent units. In the following, we use $\mathbf{u}(t)$ and $\mathbf{x}^{(i)}(t)$, respectively, to indicate the external input and state of i-th hidden layer at time step t. Based on this notation, the state in the first layer is updated according to the following equation:

$$\mathbf{x}^{(1)}(t) = F^{(1)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1)) = \tanh(\mathbf{W}^{(1)}\mathbf{u}(t) + \hat{\mathbf{W}}^{(1)}\mathbf{x}^{(1)}(t-1)), \quad (1)$$

where $\mathbf{W}^{(1)} \in \mathbb{R}^{N_R \times N_U}$ is the input weight matrix and $\hat{\mathbf{W}}^{(1)} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent weight matrix for the first layer. For each successive layer $i > 1$, the state is updated according to:

$$\mathbf{x}^{(i)}(t) = F^{(i)}(\mathbf{x}^{(i-1)}(t), \mathbf{x}^{(i)}(t-1)) = \tanh(\mathbf{W}^{(i)}\mathbf{x}^{(i-1)}(t) + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1)), \quad (2)$$

where $\mathbf{W}^{(i)} \in \mathbb{R}^{N_R \times N_R}$ collects the weights for the inter-layer connections from layer $i-1$ to layer i and $\hat{\mathbf{W}}^{(i)} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent weight matrix for layer i. Note that in both above eq. 1 and 2, a tanh non-linearity is used as element-wise applied activation function of recurrent units and bias terms are omitted for the ease of notation. Here it is also worth observing that, although the deep recurrent dynamics globally evolve as a whole, locally to each layer i the state information coming from the previous level $i-1$ actually acts as an independent input information that encodes the history of the external input up to the present time step.

Taking aside the aspects related to learning of the recurrent connections (and the specific aspects involved by different training strategies), here we focus our analysis on the case of untrained deep recurrent dynamics. To do so, we resort to

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 recently introduced deep RC framework [6], according to which the recurrent part of the deep RNN architecture is left untrained after initialization subject to stability constraints [7]. Specifically, the network is initialized with weights from a uniform distribution in $[-1, 1]$ and then re-scaled to control, for each layer i, the values of $\|\mathbf{W}^{(i)}\|_2$ and $\rho(\hat{\mathbf{W}}^{(i)})$, where $\rho(\cdot)$ denotes the spectral radius of its matrix argument (i.e. the maximum among the eigenvalues magnitudes). These quantities are hyper-parameters of the model that influence its state dynamics and that are typically set to small values in order to guarantee a stable regime, a standard initialization approach also for trained networks. Notice that this framework allows us on the one hand to investigate the fixed characterization of state dynamics in successive levels of a deep RC network, and on the other hand to study of the bias due to layering in deep recurrent architectures.

Output computation is implemented by using an output layer of size N_Y. Though different choices are possible for the state-output connection settings (see e.g. [3, 8]), following from our analysis aims here we consider output modules that are individually applied to each layer of the recurrent network. This enables us to study separately the characteristics of the state behavior emerging at the different levels in the architecture. We use linear output modules, such that for each layer i the output is computed as $\mathbf{y}^{(i)}(t) = \mathbf{W}_{out}^{(i)}\mathbf{x}^{(i)}(t)$, where matrices $\mathbf{W}_{out}^{(i)} \in \mathbb{R}^{N_Y \times N_R}$ are trained for each layer individually, using a direct method such as pseudo-inversion. In the RC framework this setting also ensures the same training cost for every layer.

3 Experiments

We investigate the short-term memory abilities of deep RNN architectures by resorting to the Memory Capacity (MC) task [9]. This aims at measuring the extent to which past input events can be recalled from present state activations. Specifically, the recurrent system is tested in its ability to reconstruct delayed versions of a stationary uni-variate driving input signal ($N_U = N_Y = 1$), with the MC at layer i computed as a squared correlation coefficient, as follows:

$$MC^{(i)} = \sum_{k=1}^{\infty} MC_k^{(i)} = \sum_{k=1}^{\infty} \frac{Cov^2(u(t-k), y_k^{(i)}(t))}{Var(u(t))\, Var(y_k^{(i)}(t))}, \tag{3}$$

where $y_k^{(i)}(t)$ is the activation of the output unit trained to reconstruct the $u(t-k)$ signal from the state of layer i, while Cov and Var respectively denote the covariance and variance operators. In order to maximally exercise the memory capability of the systems, we used i.i.d. input signals from a uniform distribution in $[-0.8, 0.8]$, i.e. an unstructured temporal stream in which $u(t)$ does not carry information on previous inputs $\ldots, u(t-2), u(t-1)$. For this task, we considered a 6000 time-step long sequence, where the first 5000 time steps were used for training[1] and the remaining 1000 for MC assessment.

[1]The first 1000 time steps were considered as transient to washout the initial conditions.

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/.

We considered deep RNNs with $N_L = 10$ recurrent layers, each of which containing $N_R = 100$ recurrent units. Input and inter-layer weights were re-scaled such that $\|\mathbf{W}^{(i)}\|_2 = 1$ for $i = 1, \ldots, N_L$. Weights of recurrent connections were re-scaled to the same spectral radius in all the layers, i.e. $\rho = \rho(\hat{\mathbf{W}}^{(i)})$ for $i = 1, \ldots, N_L$, with ρ values ranging in $[0.1, 1.5]$. Note that, under the considered experimental settings, for higher values of $\rho > 1$ the network dynamics tend to exhibit a chaotic behavior, as shown in previous works in terms of local Lyapunov exponents [10, 11]. Although recurrent dynamics in chaotic regimes are generally not interesting under a practical point of view, in this paper we consider also these cases ($\rho > 1$) for the scope of analysis. For each choice of ρ we independently generated 50 networks realizations (with different seeds for random generation), and averaged the achieved results over such realizations. By referring to an RC-based experimental setup, we trained only the output connections, using pseudo-inversion. Input, inter-layer and recurrent connections were left untrained after initialization.

For practical assessment of the MC values, it is useful to recall a basic theoretical result provided in [9], which states that the MC of an N_R-dimensional recurrent system driven by an i.i.d. uni-variate input signal is upper bounded by N_R. Accordingly, we considered a maximum value for the delay k in eq. 3 equal to twice the size of the state space, i.e. 200, which is sufficient to account for the correlations that are practically involved in our experimental settings.

Fig. 2 shows the MC values achieved in correspondence of progressively higher layers in the architecture, and for the different cases of ρ considered for network initialization. Results clearly point out that for recurrent networks in the ordered regime (ρ not exceeding 1) higher layers in the deep architecture are

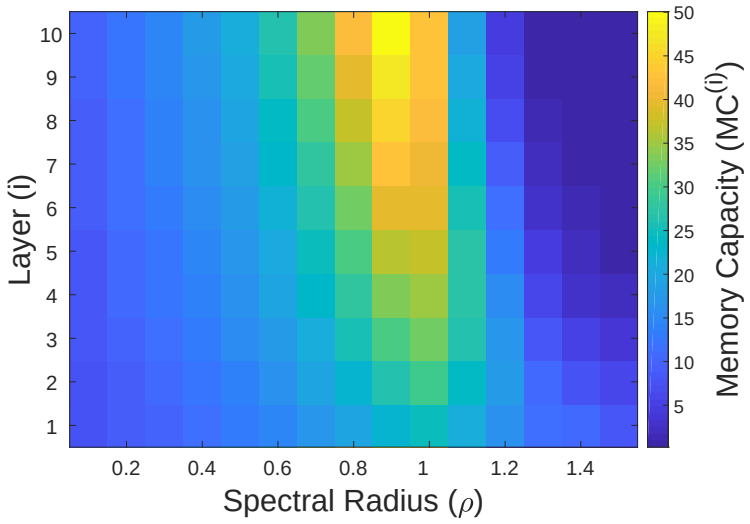

Fig. 2: MC of different layers in deep RNNs for increasing values of ρ.

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/.

naturally biased toward progressively longer short-term memory abilities. For networks in a chaotic regime (ρ above 1) higher layers tend to show a poorer MC. The MC performance shown in Fig. 2 has a peak in correspondence of $\rho = 0.9$, in which case the score improves from 22.2 in the 1st layer, up to 50.1 in the 10th layer. Interestingly, our results also highlight the effectiveness of layering and its striking advantage as a convenient process for RC networks architectural design. The memory of an N_R-dimensional reservoir can be indeed easily improved by using an underlying stack of recurrent layers to filter the external input signal. Note that such improvement comes at the only price of a modestly increased cost for the state computation (that increases linearly with the number of layers), while the cost for output training remains the same.

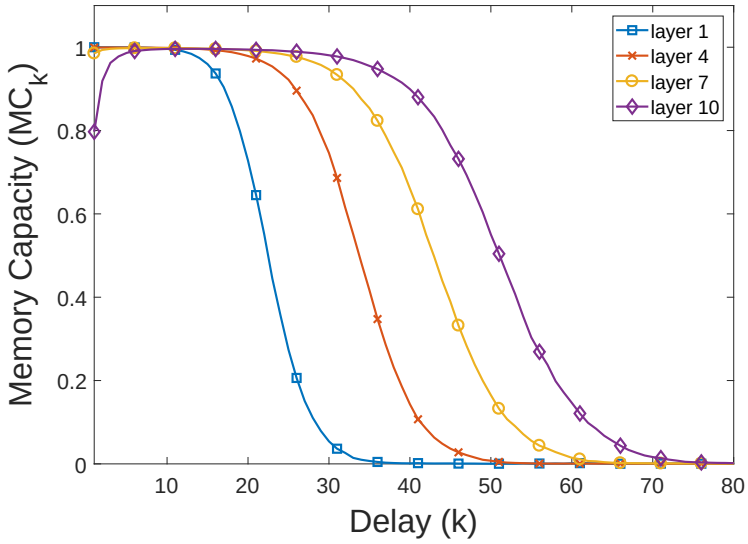

Fig. 3: k-delay memory capacity of deep RNN layers at increasing height.

We further inquire into the memory structure developed by the layers of deep RNNs by analyzing the MC values for increasing delays. Fig. 3 shows the forgetting curves of individual (progressively higher) layers, i.e. the values of $MC_k^{(i)}$ as a function of k, obtained in the case of $\rho = 0.9$. The plot in Fig. 3 clearly reveals the diversification of memory spans in the components of the deep recurrent architecture: higher layers are able to store information about past inputs for longer times. While for the 1st layer the memory recall is almost null after delay 30, the dynamics developed in the 10th layer lead to a value that is above zero even for a delay of 60. We can also see that input signals with smaller delays are better reconstructed in the lower layers, while higher layers are characterized by a peak that tends to shift to the right (more evident for layer 10 in Fig. 3), and by a slope of the forgetting curve that tends to be increasingly smoother. Besides, the highlighted diversification of short-term memory spans

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/.

among the successive layers in the deep RNN architecture is also interesting as a way of characterizing (in a quantitative way) the intrinsic richness of state representations globally developed by the deep recurrent system.

4 Conclusions

In this paper we have provided a computational analysis of short-term memory in deep RNNs. To do so, we have resorted to the MC task as a mean to quantify the memory of state dynamics in successive levels of a deep recurrent system.

Our results clearly showed that higher layers in a hierarchically organized RNN architecture are inherently featured, even prior to learning of recurrent connections, by an improved ability to latch input information for longer time spans. The analysis provided in this paper also revealed interesting insights on the diversification of the memory structure developed within deep stacked RNN dynamics, showing that higher layers tend to forget the past input history more slowly and more smoothly compared lower ones. Furthermore, framed within the deep RC framework, our results provided evidence that support the practical benefit of the layered recurrent organization as a way to improve the memory skills of reservoir networks in a cost-effective fashion.

Overall, though further studies in this research direction are certainly demanded (e.g. on the theoretical side), we believe that the outcomes provided in this paper can contribute to better understand and characterize the bias due to layering in deep recurrent neural models.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[2] P. Angelov and A. Sperduti. Challenges in deep learning. In *Proc. of the 24th European Symposium on Artificial Neural Networks (ESANN)*, pages 489–495. i6doc.com, 2016.

[3] M. Hermans and B. Schrauwen. Training and analysing deep recurrent neural networks. In *NIPS*, pages 190–198, 2013.

[4] D. Verstraeten, B. Schrauwen, M. d'Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural networks*, 20(3):391–403, 2007.

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

[6] C. Gallicchio, A. Micheli, and L. Pedrelli. Deep reservoir computing: A critical experimental analysis. *Neurocomputing*, 268:87–99, 2017.

[7] C. Gallicchio and A. Micheli. Echo state property of deep reservoir computing networks. *Cognitive Computation*, 9(3):337–350, 2017.

[8] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026v5*, 2014.

[9] H. Jaeger. Short term memory in echo state networks. Technical report, German National Research Center for Information Technology, 2001.

[10] C. Gallicchio, A. Micheli, and L. Silvestri. Local lyapunov exponents of deep rnn. In *Proc. of the 25th European Symposium on Artificial Neural Networks (ESANN)*, pages 559–564. i6doc.com, 2017.

[11] C. Gallicchio, A. Micheli, and L. Silvestri. Local lyapunov exponents of deep echo state networks. *Neurocomputing*, 2018.