In: Proc. of the 25th Int. Conference on Artificial Neural Networks (ICANN), Barcelona, Spain (September, 2016)
The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-44778-0_16

Learning Multiple Timescales in Recurrent Neural Networks

Tayfun Alpay, Stefan Heinrich, and Stefan Wermter

University of Hamburg, Department of Informatics, Knowledge Technology
Vogt-Kölln-Straße 30, D - 22527 Hamburg, Germany
{alpay,heinrich,wermter}@informatik.uni-hamburg.de
http://www.informatik.uni-hamburg.de/WTM/

Abstract. Recurrent Neural Networks (RNNs) are powerful architectures for sequence learning. Recent advances on the vanishing gradient problem have led to improved results and an increased research interest. Among recent proposals are architectural innovations that allow the emergence of multiple timescales during training. This paper explores a number of architectures for sequence generation and prediction tasks with long-term relationships. We compare the Simple Recurrent Network (SRN) and Long Short-Term Memory (LSTM) with the recently proposed Clockwork RNN (CWRNN), Structurally Constrained Recurrent Network (SCRN), and Recurrent Plausibility Network (RPN) with regard to their capabilities of learning multiple timescales. Our results show that partitioning hidden layers under distinct temporal constraints enables the learning of multiple timescales, which contributes to the understanding of the fundamental conditions that allow RNNs to self-organize to accurate temporal abstractions.

Keywords: Recurrent Neural Networks, Sequence Learning, Multiple Timescales, Leaky Activation, Clocked Activation

1 Introduction

Until recently RNNs were mainly of theoretical interest as their initially perceived shortcomings proved too severe to be used in complex applications. One deficiency that has been reported early on is the vanishing gradient problem [1]. When RNNs are trained with backpropagation, error signals over time vanish exponentially in RNNs. This has led to multiple highly specialized architectures such as the Long Short-Term Memory (LSTM [2]). Their success has sparked a renewed research interest in RNNs, which has led to a number of recently proposed RNN architectures, including those that try to improve control over the self-organization of temporal dynamics by learning on multiple timescales. However, as these novel approaches have not yet been rigorously compared, the fundamental principles that allow the capturing of dynamics on different timescales are still unknown.

In this paper, we therefore aim at contributing to the following research question: what are key concepts that allow RNNs to build long-term memory and learn on multiple timescales? We approach this question by investigating the Clockwork RNN (CWRNN [3]), which has been shown to allow emergence of multiple timescales by restricting update frequencies to temporal constraints. A different method with the same effect is the use of leakage and hysteresis parameters that constrain the amount of change within a system between time steps. The concept of leakage is most popularly used in the Echo State Network (ESN [4]) but has also been shown to improve the Simple Recurrent Network (SRN [5]). A related concept can be found in the Recurrent Plausibility Network (RPN [6]) which introduces a related hysteresis parameter φ to perform time-averaging. It also has shortcut connections, which provide shorter error propagation paths for the temporal context layers. Shortcuts have been shown to allow better training in very deep networks [7]. Both shortcuts and leaky units are used in the Structurally Constrained Recurrent Network (SCRN [8]) that additionally partitions its layer into modules, similarly to the CWRNN.

As the RPN, SCRN, and CWRNN share similar architectural concepts such as leakage, shortcuts, and partitioning the hidden layer into modules, their investigation is of particular interest for studying the effect of these concepts on the self-organization of the temporal dynamics. We evaluate these architectures on sequence generation and prediction tasks, using the SRN and the LSTM as a baseline. Even though the LSTM has no specific time scaling mechanism, it is included in the experiments due to its reported ability to capture long-term dependencies.

2 Recurrent Neural Networks

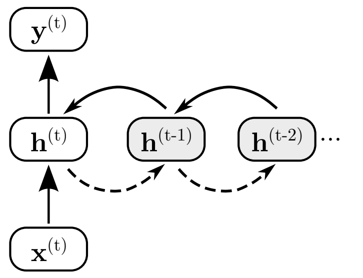
2.1 Recurrent Plausibility Network

The Recurrent Plausibility Network (RPN) was originally developed to learn and represent semantic relationships while disambiguating contextual relationships [9]. It is based on the state of an unfolded SRN during truncated BPTT (see Fig. 1(a)), i.e. each hidden layer $\mathbf{h}$ has its own set of m context layers $\mathbf{c}_k$ ($k \in \{1, ..., m\}$) which store past activations. The main difference to an unfolded SRN is the use of temporal shortcut connections for shorter context propagation paths, making vanishing or exploding gradients less likely (compare Fig. 1(b)). For time step t, the units of the hidden layer $\mathbf{h}$ are activated as follows:

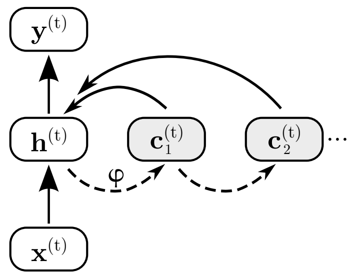
$$\mathbf{h}^{(t)} = f_h\left(\mathbf{x}^{(t)}\ \mathbf{W}_{xh} + \sum_{k=1}^{m} \mathbf{c}_m^{(t-1)}\ \mathbf{W}_{mh}\right), \tag{1}$$

where the vector $\mathbf{c}$ denotes the context layers, that are activated by shifting their contents with $\mathbf{c}_{m-1}^{(t)} = \mathbf{c}_m^{(t-1)}$. The respective context activation for units in $\mathbf{c}_m$ is further constrained under the hysteresis parameter φ [10]:

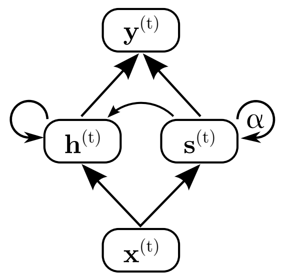
$$\mathbf{c}_k^{(t)} = \begin{cases} (1-\varphi_n)\cdot \mathbf{h}^{(t-1)} + \varphi \cdot \mathbf{c}_k^{(t-1)} & \text{iff } k=1, \\ \mathbf{c}_{k-1}^{(t-1)} & \text{otherwise} \end{cases} \tag{2}$$



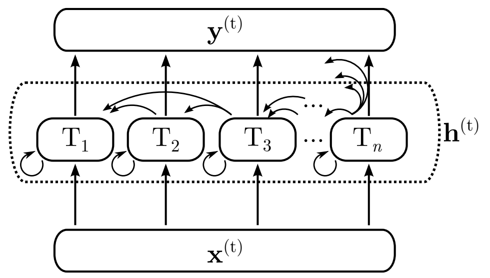
(a) SRN



(b) RPN



(c) SCRN



(d) CWRNN

Fig. 1. Comparison of investigated RNN architectures. Figure (a) shows an SRN unfolded in time. The RPN (b) extends the SRN with its temporal shortcuts and the hysteresis φ. In case of a deep RPN, each vertical layer $\mathbf{h}_n^{(t)}$ can have its own hysteresis value φ_n. The SCRN (c) has an additional layer $\mathbf{s}^{(t)}$ that learns slower than in $\mathbf{h}^{(t)}$ due to its high leakage $\alpha = 0.95$. The modules T_k of the CWRNN (d) are sorted by increasing numbers from left to right and are only updated for $t \bmod T_i = 0$.

The hysteresis mechanism allows for a finer adjustment of context memory than in the SRN. Rather than accumulating past activations in a single feedback loop, the network is able to specifically learn the contribution between specific time frames due to the temporal shortcuts.

2.2 Structurally Constrained Recurrent Network

The Structurally Constrained Recurrent Network (SCRN) was recently proposed by Mikolov et al. [8]. The motivation behind the architecture is to achieve specialization of hidden layers by partitioning them into parallel "modules" that operate independently and under distinct temporal constraints. This theoretically allows to train on multiple timescales. While the left path in the SCRN equals a SRN with a regular hidden layer $\mathbf{h}^{(t)}$, the additional module $\mathbf{s}^{(t)}$ has units with different temporal characteristics (compare Fig. 1(c)). It is initialized with the recurrent identity matrix and its updates constrained by a leakage parameter $\alpha \in [0, 1]$. The authors set this leakage to 0.95, causing the states to change on a much slower scale than in $\mathbf{h}^{(t)}$. Similarly to the RPN, this architecture makes use of shortcut connections ($\mathbf{W}_{sh}$) that allow $\mathbf{h}^{(t)}$ to access long-term context which is learned in $\mathbf{s}^{(t)}$. The update rules of the SCRN are as follows:

$$\mathbf{s}^{(t)} = (1-\alpha)\mathbf{W}_{xs}\ \mathbf{x}^{(t)} + \alpha\ \mathbf{s}^{(t-1)}, \tag{3}$$

$$\mathbf{h}^{(t)} = f_h(\mathbf{W}_{sh}\ \mathbf{s}^{(t)} + \mathbf{W}_{xh}\ \mathbf{x}^{(t)} + \mathbf{W}_{hh}\ \mathbf{h}^{(t-1)}), \tag{4}$$

$$\mathbf{y}^{(t)} = f_y(\mathbf{W}_{hy}\ \mathbf{h}^{(t)} + \mathbf{W}_{sy}\ \mathbf{s}^{(t)}), \tag{5}$$

where f_h and f_y are the respective activation functions for the hidden and output layers.

2.3 Clockwork Recurrent Neural Network

The discussed idea of partitioning the hidden layer into parallel modules with distinct temporal properties can also be found in the Clockwork Recurrent Neural Network (CWRNN). However, the main difference is that multiple timescales are not achieved by varying leakage but rather an external clock that determines *when* a module gets updated. This means that a module k is only updated if its clock period T_k satisfies the criterion $t \bmod T_k = 0$. Otherwise, the module is inactive in which case the previous activation $\mathbf{h}_k^{(t-1)}$ gets copied over:

$$\mathbf{h}_k^{(t)} = \begin{cases} f_h\left(\mathbf{x}^{(t)}\ \mathbf{W}_{xk} + \sum_{l=k}^{n} \mathbf{h}_l^{(t-1)}\ \mathbf{W}_{lk}\right) & \text{iff } t \bmod T_k = 0, \\ \mathbf{h}_k^{(t-1)} & \text{otherwise} \end{cases} \tag{6}$$

An additional constraint is that $T_l > T_k$ for $l < k$, i.e. the modules are ordered by increasing numbers from left to right (compare Fig. 1(d)). Therefore, modules on the left are updated more frequently than those on the right. Consequently, modules with greater periods (on the right) will self-organize slower and to long-term dependencies while those with small periods (on the left) change more often, focusing on short-term dependencies.

3 Experiments

All five architectures, the SRN, RPN, SCRN, CWRNN, and LSTM have been evaluated on two tasks; sequence generation of a sinusoid wave and sequence prediction of words created by embedded Reber grammar. They have been trained with RMSProp, which divides the current gradient by a sliding average of recent gradients [11]. Momentum was empirically set to 0.9 and the networks trained for a maximum number of 5000 epochs using early stopping. Weights were initialized using normalized initialization, sampling from $\mathcal{N}(0, 1/\sqrt{n+m})$ where n is the number of incoming and m the number of outgoing weights in the respective layer [12]. Linear and non-linear activation (tanh) were explored. The forget gate bias was initialized with a higher value of 2 to avoid initial forgetting [13]. All other hyperparameters were set empirically for each network and task. Each setup was run 100 times with different random initializations.

3.1 Sequence Generation

In the first task, the networks have to learn how to generate a target sequence. They receive no input while a single sequence is sequentially presented as the target. This sequence of length 256 is a composition of three different sine waves, normalized to $[-1, 1]$. A single output unit y_t encodes the respective sequence value at time step t. All networks were trained to minimize the mean squared error (MSE) with a learning rate of $\gamma = 10^{-4}$ and 64 hidden units. For the RPN, a context width $m = 5$ and $m = 15$ was explored with hysteresis values of $\varphi \in \{0.1, 0.2, 0.5\}$. Two variants of the SCRN were trained: i) a constant leakage of $\alpha = 0.95$ and ii) an adaptive leakage α_t that is trained as described in [8]. For the CWRNN, 8 equally sized modules with clock periods growing by the powers of 2 ($P_1 = \{1, 2, 4, 8, 16, 32, 64, 128\}$) are compared with a more coarse setup of 4 modules with the periods $P_2 = \{1, 4, 16, 64\}$.

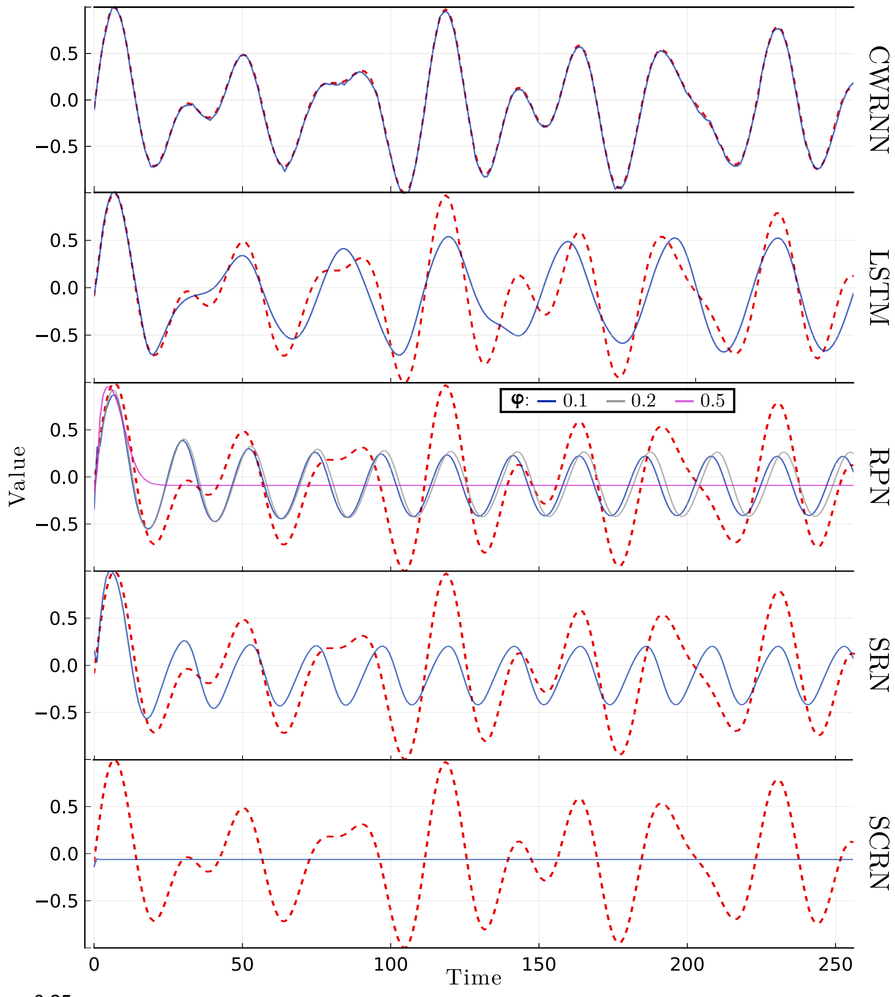
The results for the best networks are depicted in Fig. 2. The CWRNN generates the most accurate sequences, which indicates an ability to capture the underlying subfrequencies, learning multiple timescales. It was also found that the investigated clock-timings P_1 (8 modules) and P_2 (4 modules) perform equally well. The SRN on the other hand merely captures the most dominant subfrequency of the sequence while the LSTM gives a sliding average. The SCRN always converges to the mean, being the only network which seems to be completely unable to learn this task. Similar to the SRN, the RPN is able to capture only one subfrequency. For the tested φ values, only 0.1 and 0.2 lead to convergence that is not located around the mean. There is also a slight difference that can be observed between these values: increasing φ from 0.1 to 0.2 causes an increasing phase shift, i.e. the prediction gets increasingly delayed over time. This effect can be explained by the fact that the temporal context, which is time-averaged by the hysteresis, will span a larger time window with growing hysteresis values.

3.2 Embedded Reber Grammar

In the second task, the networks are trained to sequentially predict the next symbol produced by Embedded Reber Grammar (ERG). The ERG is a well-known test for RNNs, since a SRN cannot be trained with BPTT to learn the grammar due to the presence of long-term dependencies. It is defined as follows:

$$\begin{array}{lll} \mathtt{S} \rightarrow \mathtt{btRte} \mid \mathtt{bpRpe} & \mathtt{A} \rightarrow \mathtt{sA} \mid \mathtt{x} & \mathtt{C} \rightarrow \mathtt{xBD} \mid \mathtt{s} \\ \mathtt{R} \rightarrow \mathtt{btACe} \mid \mathtt{bpBDe} & \mathtt{B} \rightarrow \mathtt{tB} \mid \mathtt{v} & \mathtt{D} \rightarrow \mathtt{pC} \mid \mathtt{v} \end{array}$$

We randomly generate two different sets with respective sequence lengths of 20 and 30. Both data sets consist of 250 sequences and are further split into 60% training, 20% test, and 20% validation sets for cross validation. Each symbol is encoded with a feature vector of size 7 (1 unit per symbol), while softmax activation in the output layer yields the symbol probabilities. The minimized loss function is the Kullback-Leibler divergence [14]. For all networks, the number of hidden units was set to 15. For the SCRN, a learning rate of $\gamma = 0.01$ was



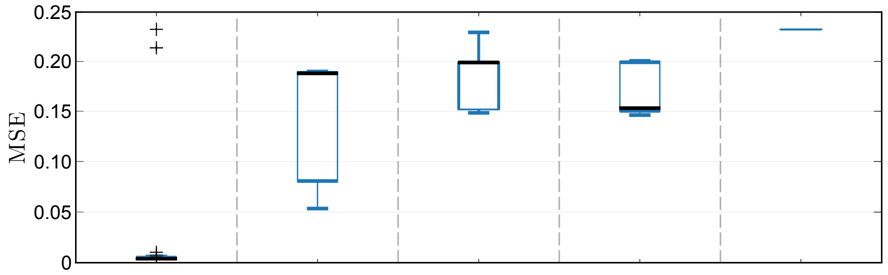


Fig. 2. Top: Sequences with the lowest MSE for the best trials. Generated sequences (solid lines) are plotted against the target sequence (dotted, red line). Bottom: MSE for each network. Boxes show 25% and 75% quartiles as well as the median (black line). The shown best trials were achieved with $\varphi = 0.2, m = 15$ for the RPN and $P_2 = \{1, 4, 16, 64\}$ for the CWRNN (P_1 produced nearly identical results).

found to be optimal, whereas $\gamma = 10^{-4}$ worked best for the other architectures. The CWRNN's hidden layer was partitioned into 5 modules with the periods $P = \{1, 2, 4, 8, 12\}$. All other hyperparameters are set as in the first task.

The results for the best trials are depicted in Fig. 3. When trained with sequences of length 20, the SCRN with $\alpha = 0.95$ emerges as the best performing architecture, whereas the CWRNN seems to have the most difficulties. The LSTM shows an average accuracy, while the RPN seems to be less prone to bad initialization than the SRN. Especially for longer sequences, a large number of SRNs yield considerably more prediction errors than all other networks, which in turn share a similar overall performance.

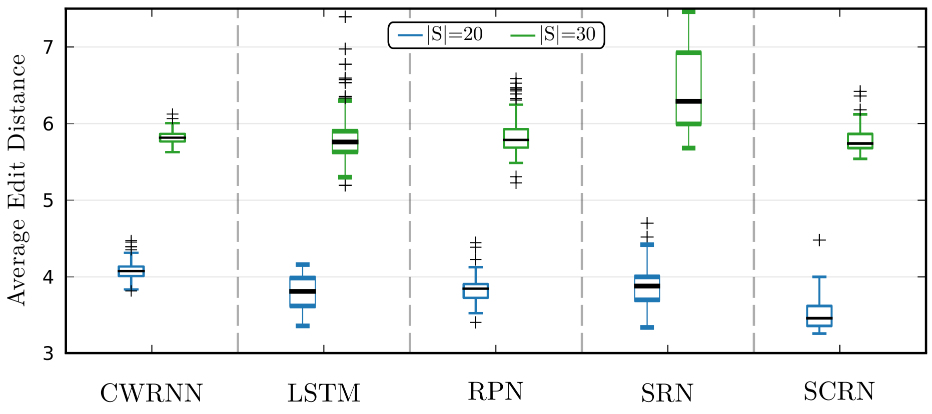


Fig. 3. Average edit distances (number of wrongly predicted symbols) for sequences of length $|S| = 20$ (left, blue) and $|S| = 30$ (right, green). Boxes show 25% and 75% quartiles as well as the median (black line). The best RPN trials were achieved with $\varphi = 0.2$.

4 Discussion

In this paper, we have explored various design concepts that allow emergence of multiple timescales and long-term memory in RNNs. Leaky and clocked activations have been investigated together with partitioning hidden layers into modules and using shortcut connections by comparing a number of architectures on the tasks of sequence generation and learning embedded Reber grammar.

Our results show that parallel hidden layers, which learn under different temporal constraints can lead to an emergence of multiple timescales in RNNs. Furthermore, shared weights in the form of shortcut connections (such as in the SCRN and CWRNN) allow units which self-organize to short-term context, to take long-term dependencies into account from specialized units that operating on a larger timescale. While the SCRN achieves this by means of leakage, the

CWRNN utilizes clocked module activations. For the sequence generation task, the CWRNN was the only architecture to learn the decomposition of the trained sinusoid wave into all its subfrequencies. All other networks converged to the mean or a single subfrequency. This suggests that the CWRNN is able to store the entire sequence in the memory of the clocked modules, although it has half as much parameters as the SRN [3]. For the second task, the complete opposite can be observed; the SCRN is able to outperform all other networks for sequence lengths of 20 while the CWRNN has difficulties.

Our findings suggest that the SCRN and RPN seem to work better for discrete, symbolic long-term decisions while the CWRNN is better at decomposing real-valued signals. Partitioning hidden layers with distinct temporal constraints has shown to be a viable method to capture different timescales. Future research should therefore concentrate on further exploring time scaling mechanisms on more challenging tasks such as sequence classification or language modeling.

References

1. Bengio, Y., Simard, P., Frasconi, P.: Learning Long-Term Dependencies with Gradient Descent is Difficult. In: IEEE Trans. Neural Networks. 5(2), pp. 157–166 (1994)
2. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation. 9(8), pp. 1735–1780 (1997)
3. Koutník, J., Greff, K., Gomez, F., Schmidhuber, J.: A Clockwork RNN. In: Proc. ICML-2014, pp. 1863–1871 (2014)
4. Jaeger, H., Lukoševičius, M., Popovici, D., Siewert, U.: Optimization and Applications of Echo State Networks with Leaky-Integrator Neurons. In: Neural Networks. 20(3), pp. 335–352 (2007).
5. Bengio, Y., Boulanger-Lewandowski, N., Pascanu, R.: Advances in Optimizing Recurrent Networks. In: Proc. ICASSP-2013, pp. 8624–8628 (2013)
6. Wermter, S., Panchev, C., Arevian, G.: Hybrid Neural Plausibility Networks for News Agents. In: Proc. AAAI-1999, pp. 93–98 (1999)
7. Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y.: How to Construct Deep Recurrent Neural Networks. ArXiv preprint arXiv:1312.6026v5 (2014)
8. Mikolov, T., Joulin, A., Chopra, S., Mathieu, M., Ranzato, M.: Learning Longer Memory in Recurrent Neural Networks. ArXiv preprint arXiv:1412.7753v2 (2015)
9. Wermter, S.: Hybrid Connectionist Natural Language Processing. Chapman and Hall, Thompson International, London, UK (1995)
10. Arevian G., Panchev, C.: Robust Text Classification Using a Hysteresis-Driven Extended SRN. In: Proc. ICANN-2007, pp. 425–434, Porto, Portugal (2007)
11. Graves, A.: Generating Sequences with Recurrent Neural Networks. Arxiv preprint arXiv:1308.0850 (2013)
12. Glorot, X., Bengio, Y.: Understanding the Difficulty of Training Deep Feedforward Neural Networks. In: Proc. AISTATS-2010, pp. 249–256 (2010)
13. Jozefowicz, R., Zaremba, W., Sutskever, I.: An Empirical Exploration of Recurrent Network Architectures. In: Proc. ICML-2015, pp. 2342–2350 (2015)
14. Kullback, S.: Information Theory and Statistics. New York: Wiley (1959)