Gated Echo State Networks: a preliminary study

Daniele Di Sarli
Department of Computer Science
University of Pisa
Pisa, Italy
daniele.disarli@phd.unipi.it

Claudio Gallicchio
Department of Computer Science
University of Pisa
Pisa, Italy
gallicch@di.unipi.it

Alessio Micheli
Department of Computer Science
University of Pisa
Pisa, Italy
micheli@di.unipi.it

***Abstract*—Gating mechanisms are widely used in the context of Recurrent Neural Networks (RNNs) to improve the network's ability to deal with long-term dependencies within the data. The typical approach for training such networks involves the expensive algorithm of gradient descent and backpropagation. On the other hand, Reservoir Computing (RC) approaches like Echo State Networks (ESNs) are extremely efficient in terms of training time and resources thanks to their use of randomly initialized parameters that do not need to be trained. Unfortunately, basic ESNs are also unable to effectively deal with complex long-term dependencies. In this work, we start investigating the problem of equipping ESNs with gating mechanisms. Under rigorous experimental settings, we compare the behaviour of an ESN with randomized gate parameters (initialized with RC techniques) against several other models, among which a leaky ESN and a fully trained gated RNN. We observe that the use of randomized gates by itself can increase the predictive accuracy of a ESN, but this increase is not meaningful when compared with other techniques. Given these results, we propose a research direction for successfully designing ESN models with gating mechanisms.**

***Index Terms*—Echo State Networks, Gated Recurrent Neural Networks**

I. Introduction

The paradigm of Reservoir Computing (RC) [1], [2] has proven to be an extremely efficient approach for designing and training Recurrent Neural Networks (RNNs). The widely known Echo State Network (ESN) [3], [4] is a RC model that inherits the same architecture of a vanilla RNN, but thanks to a proper initialization of the parameters in the state transition function it allows to completely avoid any kind of training for the recurrent neurons in the network.

On the other hand, the traditional approach of employing gradient descent training for RNNs has allowed the emergence of several architectural evolutions that, while in theory remaining computationally equivalent to a vanilla RNN, can make training easier in the presence of long-term dependencies. Examples of such evolutions are LSTM [5] and GRU [6], which introduce gating mechanisms that help the network to selectively remember and forget relevant information from the input and consequently boost the predictive accuracy. Unfortunately, these kind of models still require gradient descent for training, which is often much more computationally expensive than the RC approaches.

Recently, there has been an increasing research interest regarding solutions for maintaining information over long time spans in recurrent models. An example is the application of the Learning-to-Learn paradigm to spiking neural networks in the context of RC methodologies [7], [8]. In this paper, we focus on this direct issue: what happens if the dynamics of the gating mechanisms are treated as reservoir systems?

In ESNs, which are based on the exploitation of the discrimination capabilities of the underlying dynamical system, it is not immediate to extend the architecture with gate-like mechanisms. In this work we describe the first steps in investigating whether it is possible to introduce efficient gating mechanisms within ESNs, so that their predictive performance can be improved without giving up on their exceptionally fast training process. We test our research questions on a Natural Language Processing task which has been chosen for its potential to highlight the effect of gating mechanisms. Note that the application of ESNs to Natural Language Processing has been quite limited: to the best of our knowledge there are only a few of such works [9]–[13].

In section II we briefly describe the ESN and the GRU, which are the models that our work is based on. In section III we introduce the novel models that we have used for our experiments. The experiments that have been performed, and the methodology, are described in section IV. Finally, in section V we discuss the implications of the results, and we propose directions for the development of efficient gated recurrent neural networks.

II. Background

In this section we briefly describe the models that serve as the basis of our study.

A. Echo State Networks

ESNs [3], [4] are a very efficient machine learning approach for modeling sequences. In ESNs, an untrained dynamical system (the *reservoir*) is responsible for embedding the input into a high dimensional state space. Then, the states are used as input to a linear layer (the *readout*) that is trained to perform classification or regression. The key characteristic of ESNs is that the parameters of the dynamical system are not trained;

This work has been partially supported by the European Union's Horizon 2020 Research and Innovation program, under project TEACHING (Grant agreement ID: 871385). https://www.teaching-h2020.eu

instead, they are randomly initialized and then rescaled to meet mathematical conditions for stability, but are then kept unchanged. In all cases, since the only parameters that are subject to training are those in the linear readout, a closed-form solution can be obtained by extremely fast algorithms such as ridge regression.

Consider an ESN with N_U input units, N_R reservoir units and N_Y output units. Let $\mathbf{u}(1), \ldots, \mathbf{u}(T) \in \mathbb{R}^{N_U}$ be an input sequence of length T. In its simplest form, the state $\mathbf{x}(t) \in \mathbb{R}^{N_R}$ of the reservoir at time step t is computed as

$$\begin{aligned} &\mathbf{x}(0) = \mathbf{0}, \\ &\mathbf{x}(t) = \tanh\left(\mathbf{W}_{in}\mathbf{u}(t) + \hat{\mathbf{W}}\mathbf{x}(t-1)\right), \end{aligned} \tag{1}$$

where $\mathbf{W}_{in} \in \mathbb{R}^{N_R \times N_U}$ is the input-to-reservoir weight matrix, and $\hat{\mathbf{W}} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent reservoir-to-reservoir weight matrix.

The values within these matrices are not trained. Instead, after being randomly initialized, the values in $\hat{\mathbf{W}}$ are rescaled to control the value of the spectral radius $\rho(\hat{\mathbf{W}})$ (the largest eigenvalue in absolute value) in order to meet the conditions for stability [4]. Similarly, also the values in $\mathbf{W}_{in}$ are randomly initialized and then rescaled based on the value of a hyperparameter.

Leaky ESN: A notable variant of a basic ESN is denoted as *leaky ESN*. The reservoir of a leaky ESN uses leaky-integrator neurons [14], which act as a lowpass filter whose leaking rate is determined and fixed at model selection time. In this case, Equation 1 is modified as

$$\begin{aligned} \mathbf{x}(0) &= \mathbf{0}, \\ \mathbf{x}(t) &= (1-a)\,\mathbf{x}(t-1) \\ &\quad + a\tanh\left(\mathbf{W}_{in}\mathbf{u}(t) + \hat{\mathbf{W}}\mathbf{x}(t-1)\right), \end{aligned} \tag{2}$$

where $a \in \mathbb{R}$ is the leaking rate, under the constraint that $0 < a \leq 1$.

After the states for the input sequence have been collected, the output is computed as

$$\mathbf{y}(t) = \mathbf{W}_{out}\mathbf{x}(t), \tag{3}$$

with $\mathbf{W}_{out} \in \mathbb{R}^{N_Y \times N_R}$.

Training: In ESNs, the only parameters that are subject to training are those in $\mathbf{W}_{out}$ and thus a closed-form solution can be obtained by extremely fast algorithms such as ridge regression. More in detail, after the input data has been fed to the reservoir and the N_{train} states that need to be classified are collected column-wise into a matrix $\mathbf{X} \in \mathbb{R}^{N_R \times N_{train}}$, the readout can be trained by finding a solution to the following least squares problem:

$$\min_{\mathbf{W}_{out}} \|\mathbf{W}_{out}\mathbf{X} - \mathbf{Y}_{tg}\|_2^2. \tag{4}$$

In Eq. 4, $\mathbf{Y}_{tg} \in \mathbb{R}^{N_Y \times N_{train}}$ indicates the column-wise concatenation of the target vectors. A solution to Eq. 4 can be computed in closed-form as follows:

$$\mathbf{W}_{out} = \mathbf{Y}_{tg}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda_r\mathbf{I})^{-1}, \tag{5}$$

where $\mathbf{I}$ is the identity matrix, and $\lambda_r \in \mathbb{R}^+$ is the regularization parameter which can be chosen by model selection.

B. Gated Recurrent Units

Gated RNN models such as LSTM [5] and GRU [6] were developed to alleviate the issues associated with gradient descent training over long input sequences. In the case of GRU, the fundamental characteristic is the dependency of the state transition function at time t on the values of the two gates $\mathbf{r}(t)$ (reset gate) and $\mathbf{z}(t)$ (update gate). Intuitively, the purpose of the gates is to open and close to regulate the flow of information within the state: the reset gate can zero out information from the previous state $\mathbf{x}(t-1)$, while the update gate can merge information from the previous state and the current candidate state $\mathbf{h}(t)$ into the new state $\mathbf{x}(t)$. More in detail, the recurrent state $\mathbf{x}(t)$ of a GRU at each time step t is computed as:

$$\begin{aligned} \mathbf{x}(0) &= \mathbf{0}, \\ \mathbf{r}(t) &= \sigma(\mathbf{W}_{in}^r\mathbf{u}(t) + \hat{\mathbf{W}}^r\mathbf{x}(t-1)) \\ \mathbf{z}(t) &= \sigma(\mathbf{W}_{in}^z\mathbf{u}(t) + \hat{\mathbf{W}}^z\mathbf{x}(t-1)) \\ \mathbf{h}(t) &= \tanh(\mathbf{W}_{in}\mathbf{u}(t) + \hat{\mathbf{W}}(\mathbf{r}(t) \odot \mathbf{x}(t-1))) \\ \mathbf{x}(t) &= \mathbf{z}(t) \odot \mathbf{x}(t-1) \ + \ (1-\mathbf{z}(t)) \odot \mathbf{h}(t). \end{aligned} \tag{6}$$

Here, we have $\mathbf{r}(t), \mathbf{z}(t), \mathbf{h}(t), \mathbf{x}(t) \in \mathbb{R}^{N_R}$, and in particular $\mathbf{r}(t), \mathbf{z}(t) \in [0,1]$.

Any kind of differentiable output layer can be used to transform the states into the final network prediction: the whole model can be trained end-to-end by backpropagation through time.

III. Gated ESN

The untrained discrimination capabilities of the ESNs reservoir depend on the guarantees given by the so-called Echo State Property [4]: namely, a properly initialized reservoir (obtained by controlling the value of $\rho(\hat{\mathbf{W}})$) will asymptotically wash out any information from the initial conditions. However, this also means that in the case of long input sequences, the fundamental Markovian bias of ESNs [15] can prevent the readout to easily perform predictions when key information is located within the distant, initial part of a given input sequence (more in detail, information is memorized according to a suffix-based organization of the state space). In other words, ESNs have no way of dynamically and selectively remember or forget parts of a sequence while preserving generalization capabilities to sequences of different lengths. To understand whether it is possible to use gating mechanisms to improve the predictive performance of an ESN, we define here two alternative RC models. Both models add gating mechanisms to an ESN by borrowing the state transition function of the GRU, thus introducing an update gate and a reset gate to steer the trajectories of the reservoir. The architecture of both models is illustrated in Fig. 1.

While the name may bear similarity with the Gating ESN model [16], we point out that our approach is radically different. Instead of using a combination of many parallel instances

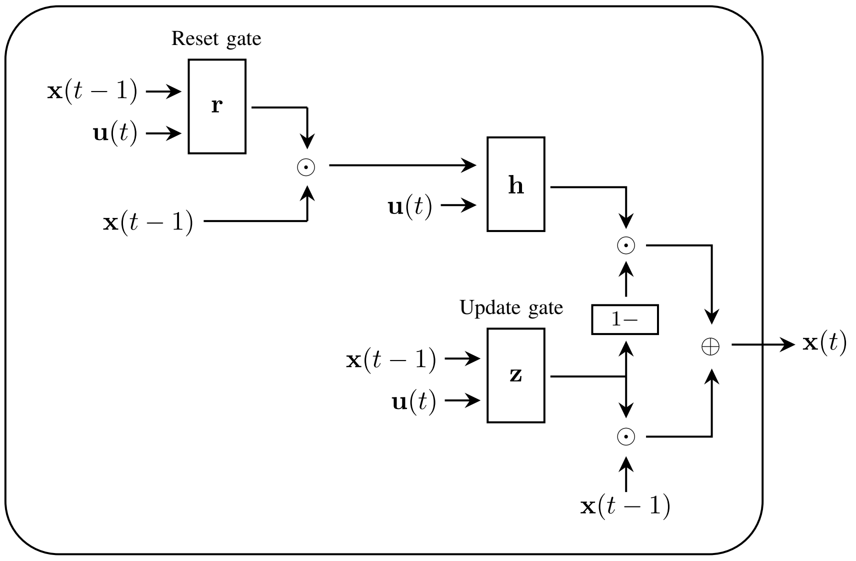


Fig. 1. Graphical representation of the recurrent cell of *Gated ESN* for a generic time step t. Symbols $\odot$ and $\oplus$ respectively denote the elementwise product and the elementwise sum of two vectors. The architecture of the cell is identical to the one of a GRU, however in *Gated ESN* the parameters controlling the activations of $\mathbf{r}(t), \mathbf{z}(t)$, and $\mathbf{h}(t)$ are not trained. In our other variant, *Gated ESN RZ*, the architecture is still unchanged, but the parameters controlling $\mathbf{r}(t)$ and $\mathbf{z}(t)$ are trained while the dynamics of $\mathbf{h}(t)$ remain untrained.

of ESNs, each initialized with different hyperparameters, we aim to enrich the dynamics of the state of a single ESN by explicitly introducing gated units (cells) in the spirit of architectures such as GRU [6] and LSTM [5].

A. Gated ESN

For the first variation, which we simply denote as *Gated ESN*, we design a gated model which is trained like an ESN, in a purely RC fashion. In particular, the reservoir uses the same state transition function of a GRU to compute the states of the network without any previous training of the parameters. In fact, except for the readout, all matrices (including those in the gates) are randomly initialized and then rescaled according to the value of the appropriate hyperparameters, just like what happens in a standard ESN. After the states have been collected, the linear readout is trained by ridge regression.

B. Gated ESN RZ

In the second variation, denoted as *Gated ESN RZ*, we maintain the same state transition function of the Gated ESN but we train some of the parameters by backpropagation through time. In particular, the two matrices that are used to compute a candidate state are still randomly initialized, rescaled and then kept fixed as it happens in a standard ESN, however here the parameters in the reset and update gates are trained by backpropagating the gradient of the error at the readout. To implement this, the readout and the gates are jointly trained by backpropagation.

IV. Experiments

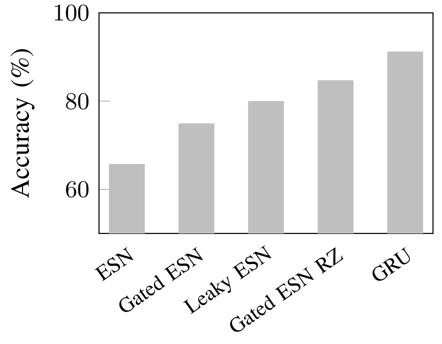
We compared the predictive performance of several ESN variants over a natural language classification task: the Question Clasification task from the TREC dataset [17]. In this task, input sentences are questions that need to be classified into one of six categories based on whether the question is asking about a location, a person, a number, a human being, a description, or an entity. In our case, each input question that is fed to the models is represented as a sequence of pretrained word embeddings, and the output is a vector indicating which one of the six categories the input falls into. We have selected this dataset since the particular characteristics of this task (key words are often located at the beginning of the input sentences) make it ideal to be tackled by gated models.

The dataset, which is composed by a total of 5952 sentences, has been split into training, validation and test. In particular, for the test data we have used the split provided by the authors of the dataset, which is composed by 500 questions. From the remaining data, after shuffling, 80% (4362 questions) have been used for training and 20% (1090 questions) have been used for validation. With this split, we performed model selection on the validation set and reported the results on the test set.

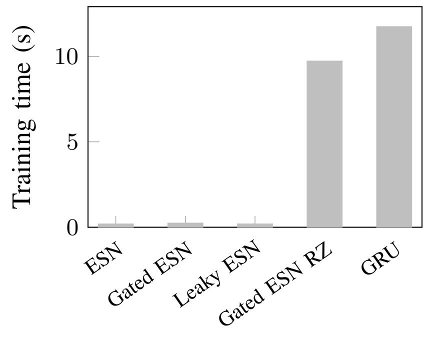
To provide a fair and rigorous comparison, we made sure to keep the total number of trainable parameters uniform between all models by controlling the number of recurrent units. For reference, the number of trainable parameters has been chosen to be the one used in the best performing GRU. We point out that several architectural modifications can be introduced to significantly boost the predictive performance of an ESN on this task [13]. However, here we deliberately consider only the simplest architectures in order to focus on the improvements brought by the gates.

A. Results

In Fig. 2 we have reported the predictive accuracy and training time for the investigated variants of ESNs and for



(a) Predictive performance. The highest accuracy is reached by the fully trained model, while the basic ESN is the baseline over which the improvements are built.



(b) Training time. As soon as backpropagation is used (last two bars), even if only partially, the very fast training times of the pure RC models grow significantly.

Fig. 2. Results of the experiments. *Top:* Test set accuracy. *Bottom:* Training times. The gates are able to improve the predictive performance of the ESN, but training their parameters seems necessary. Unfortunately, using backpropagation for this training process drastically increases the training time.

a fully trained GRU. The great efficiency advantage of RC models is confirmed by the training times in Fig. 2b: training the ESN is about 63 times faster than training the GRU.

However, as expected, from Fig. 2a it can be observed that the best performing model is the GRU, in which all the parameters are trained by gradient descent. Also as expected, the basic ESN displays the lowest level of accuracy, which is likely due to the fact that the most important words to be considered for prediction are often located at the beginning of the sentences, thus their contribution has a very low influence on the final states of the network which are used by the classifier [13].

Interestingly, the introduction of randomized gates (*Gated ESN*) produces a significant increase in the predictive accuracy of an ESN. However, the results show that removing the gates and simply using leaky-integrator neurons produces better results. A possible explanation is that the matrices in the update gate $\mathbf{z}(t)$ of the *Gated ESN* may get rescaled by model selection to approximate on average the behavior of a leaky ESN. In fact, a preliminary analysis on the standard deviations σ_r and σ_z of the activations of the reset and update gates suggests that they may tend to roughly behave like constants ($\sigma_r = 0.002, \sigma_z = 0.097$).

On the other hand, when the gates in the ESN follow a behavior that has been learned by training (*Gated ESN RZ*), then a large improvement in accuracy with respect to all pure RC models occurs. However, as it can be observed from Fig. 2b, the introduction of the backpropagation of the gradient in the training process causes a severe increase in the training time, which is definitely not comparable to the relative increase in accuracy. This makes the approach of training the gates via backpropagation unappealing in practice, as the advantages coming from the RC approach are vanishing.

V. Discussion and Perspectives

The extremely efficient approach to sequence modeling offered by ESNs is made possible thanks to the Markovian bias that is at the foundation of the RC paradigm. Unfortunately, this also means that, in tasks where there are information dependencies spanning large distances, ESNs can have difficulty with generating meaningful dynamics. The results of our study show that gating mechanisms have the potential of significantly improve the predictive performance of an ESN, but in practice we have observed meaningful improvements only in the case where the gates are trained by backpropagation, which is not desirable because of the re-increasing training times.

It seems reasonable to assume that, in order to obtain an effective gating mechanism, it is crucial to ensure that the activation patterns of the gates are influenced by information that is inherently target-dependent. In fact, it is trivial to construct examples in which the optimal activation pattern of the gates cannot be determined without some information about the target output for the data. For example, the optimal behaviour of the gates for a classification model which is trained over long random strings of characters should drastically change depending on whether the target output is equal to the first or to the last character of each input string. To make the gates useful in RC contexts, it is then necessary to adopt semi-supervised, supervised, or reward-based learning mechanisms that can adapt the parameters in the gates on the basis of their contribution to the task at hand. At the same time, these techniques should be implemented in such a way as to be more computationally efficient than simply using the expensive backpropagation through time as in *Gated ESN RZ*.

A potential alternative to backpropagation is the biologically inspired Hebbian learning. By using local rules for adapting the synapses within the gates, there is no need to compute the whole backpropagation chain of matrix multiplications. This would allow for a fast and highly parallelizable training of the gates. In its simplest form, Hebbian learning is unsupervised. In order to guide the learning process by the characteristics of the target of the task at hand, it is necessary to modulate the Hebbian changes at each time step based on whether the last configuration of the synapses has proven beneficial for solving the task. Unfortunately, in the typical classification tasks the information about whether the task has been solved correctly only comes at the end of the trial. In order to keep track of how much each synapse has contributed to the result of the trial in case of long delays (credit assignment problem [18]), it is possible to use the biologically motivated mechanism of *eligibility traces* [19], which is compatible with approximations of the gradients computed by backpropagation [20], [21]. There exist recent works that use reward-modulated Hebbian learning for training all parameters of a recurrent network [22], but we propose to still exploit the architectural characteristics of RNNs via RC techniques (i.e. without training) and just steer the trajectories through the use of gates trained by the above-mentioned approach. This should allow for an efficient and effective method that would make ESNs strong on tasks in which long-term dependencies cannot currently be easily captured.

References

[1] M. Lukosevicius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.

[2] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007.

[3] 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.

[4] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks – with an erratum note'," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 2001.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, 2014, pp. 1724–1734.

[7] A. Subramoney, F. Scherr, and W. Maass, "Reservoirs learn to learn," *CoRR*, vol. abs/1909.07486, 2019.

[8] G. Bellec, D. Salaj, A. Subramoney, R. A. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *NeurIPS*, 2018, pp. 795–805.

[9] R. Ramamurthy, R. Stenzel, R. Sifa, A. Ladi, and C. Bauckhage, "Echo state networks for named entity recognition," in *ICANN (Workshop)*, ser. Lecture Notes in Computer Science, vol. 11731. Springer, 2019, pp. 110–120.

[10] K. Simov, P. Koprinkova-Hristova, A. Popov, and P. Osenova, "Word embeddings improvement via echo state networks," in *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE, 2019, pp. 1–6.

[11] A. Popov, P. Koprinkova-Hristova, K. Simov, and P. Osenova, "Echo state vs. lstm networks for word sense disambiguation," in *International Conference on Artificial Neural Networks*. Springer, 2019, pp. 94–109.

[12] K. I. Simov, P. D. Koprinkova-Hristova, A. Popov, and P. Osenova, "Word embeddings improvement via echo state networks," in *INISTA*. IEEE, 2019, pp. 1–6.

[13] D. Di Sarli, C. Gallicchio, and A. Micheli, "Text classification by untrained sentence embeddings," *Intelligenza Artificiale*, 2020, accepted.

[14] H. Jaeger, M. Lukosevicius, D. Popovici, and U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Networks*, vol. 20, no. 3, pp. 335–352, 2007.

[15] C. Gallicchio and A. Micheli, "Architectural and markovian factors of echo state networks," *Neural Networks*, vol. 24, no. 5, pp. 440–456, 2011.

[16] S. Babinec and J. Pospichal, "Gating echo state neural networks for time series forecasting," in *ICONIP (1)*, ser. Lecture Notes in Computer Science, vol. 5506. Springer, 2008, pp. 200–207.

[17] X. Li and D. Roth, "Learning question classifiers," in *19th International Conference on Computational Linguistics, COLING 2002*, 2002.

[18] M. Minsky, "Steps toward artificial intelligence," *Proceedings of the IRE*, vol. 49, no. 1, pp. 8–30, 1961.

[19] E. M. Izhikevich, "Solving the distal reward problem through linkage of stdp and dopamine signaling," *Cerebral cortex*, vol. 17, no. 10, pp. 2443–2452, 2007.

[20] G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass, "A solution to the learning dilemma for recurrent networks of spiking neurons," *bioRxiv*, p. 738385, 2019.

[21] G. Bellec, F. Scherr, E. Hajek, D. Salaj, A. Subramoney, R. A. Legenstein, and W. Maass, "Eligibility traces provide a data-inspired alternative to backpropagation through time," in *Neuro AI Workshop, NeurIPS*, 2019.

[22] T. Miconi, "Biologically plausible learning in recurrent neural networks reproduces neural dynamics observed during cognitive tasks," *Elife*, vol. 6, p. e20899, 2017.