Online Spatio-Temporal Learning with Target Projection

Ortner, Thomas; Pes, Lorenzo; Gentinetta, Joris; Frenkel, Charlotte; Pantazi, Angeliki

DOI
10.1109/AICAS57966.2023.10168623

Publication date
2023

Document Version
Final published version

Published in
AICAS 2023 - IEEE International Conference on Artificial Intelligence Circuits and Systems, Proceeding

Online Spatio-Temporal Learning with Target Projection

Thomas Ortner[1†⋆], Lorenzo Pes[1,2†], Joris Gentinetta[1,3], Charlotte Frenkel[2] and Angeliki Pantazi[1]
[1] IBM Research – Zurich
[2] Microelectronics Department, Delft University of Technology, Delft, The Netherlands
[3] Department of Computer Science, ETH Zurich, Zurich, Switzerland
[†] *Equal contribution*
[⋆] *Correspondence to boh@zurich.ibm.com*

2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS) | 979-8-3503-3267-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/AICAS57966.2023.10168623

***Abstract*—Recurrent neural networks trained with the backpropagation through time (BPTT) algorithm have led to astounding successes in various temporal tasks. However, BPTT introduces severe limitations, such as the requirement to propagate information backwards through time, the weight symmetry requirement, as well as update-locking in space and time. These problems become roadblocks for AI systems where online training capabilities are vital. Recently, researchers have developed biologically-inspired training algorithms, addressing a subset of those problems. In this work, we propose a novel learning algorithm called online spatio-temporal learning with target projection (OSTTP) that resolves all aforementioned issues of BPTT. In particular, OSTTP equips a network with the capability to simultaneously process and learn from new incoming data, alleviating the weight symmetry and update-locking problems. We evaluate OSTTP on two temporal tasks, showcasing competitive performance compared to BPTT. Moreover, we present a proof-of-concept implementation of OSTTP on a memristive neuromorphic hardware system, demonstrating its versatility and applicability to resource-constrained AI devices.**

***Index Terms*—Online learning, bio-inspired training, neuromorphic hardware, update locking, phase-change memory**

I. Introduction

Recurrent neural networks (RNNs), a popular type of artificial neural networks (ANNs), have achieved great success in solving temporal tasks such as speech recognition [1]–[3], translation [4] and time series analysis [5], [6], thanks to powerful gradient-based training methods like backpropagation through time (BPTT). However, they still face severe limitations in scenarios where online training is required and cannot be efficiently realized on edge AI devices. In contrast, the human brain solves these tasks online, while consuming low power. When comparing conventional RNNs to biological systems, two striking differences become apparent: the neural dynamics of the units and the employed learning algorithms.

For instance, the neuronal dynamics used in RNNs, mainly based on long short-term memory units (LSTMs) or gated recurrent units (GRUs), are only remotely related to what is observed in the human brain. As an alternative, researchers developed spiking neural networks (SNNs) that aim at reproducing the key dynamics of biological neural cells to take advantage of their sparse and event-driven communication, which is key to the high energy efficiency of the human brain. Recently, SNNs have been incorporated into deep learning frameworks [7], [8], using pseudoderivatives that accommodate for the non-differentiability of the spiking activation function. This enabled SNNs to be trained with gradient-based learning algorithms, such as BPTT, and to achieve performance levels comparable to their conventional RNNs counterparts [9].

Furthermore, BPTT raises several concerns regarding its biological plausibility [10], [11]. Firstly, it requires to propagate *information backwards through time* by unrolling the network. Secondly, it uses non-local information to update the synaptic weights. Specifically, it relies on symmetric weights for the forward and the backward pass, leading to the *weight transport problem*, which constrains memory access patterns and leads to inefficient hardware implementations [12]. Additionally, it requires the update of synaptic weights to occur after all inputs have been processed (*update-locking in time*) and after all the layers of the network computed (*update-locking in space*).

To address these shortcomings, biologically-plausible training algorithms have been developed. Feedback alignment (FA [13]), direct feedback alignment (DFA [14]) solve the weight transport problem. In addition, direct random target projection (DRTP [15]) projects the target information directly to the individual layers allowing the weight updates to be performed as the forward pass progresses, thus solving the update-locking problem in space. However, these algorithms have not been designed for RNNs and thus cannot naturally be applied to temporal tasks. To solve the update-locking problem in time, eligibility propagation (e-prop [16]) and online spatio-temporal learning (OSTL [17]) have been introduced. These algorithms allow for online training on temporal tasks while simultaneously processing and learning from new incoming data.

To solve both update-locking problems in space and time, deep continuous local learning (DECOLLE [18]) uses local auxiliary loss functions per layer. Furthermore, parallel temporal neural coding network (P-TNCN [19]) combines the concept of local loss functions with predictive coding. However, the use of local loss functions reduces the performance in comparison to BPTT and predictive coding also requires top-down connections. In parallel to our work, the event-based three-factor local plasticity (ETLP [20]) algorithm has been developed, by combining e-prop and DRTP. However, because ETLP is based on e-prop, it cannot be extended to multi-layer RNNs.

In this work, we propose a novel online training algorithm, called online spatio-temporal learning with target projection (OSTTP), which computes the synaptic updates exclusively based on information that is locally available in space and time. OSTTP combines the eligibility traces of OSTL with the target projection signals from DRTP. Our approach is applicable to multi-layer networks and resolves the issues of BPTT by allowing learning while receiving new input data and propagating information only in a forward manner.

For this reason, our approach is a natural match for efficient implementations on edge AI devices. We demonstrate a proof-of-concept realization of OSTTP on memristive in-memory computing neuromorphic hardware (NMHW) [21]. In particular, our contributions are:

- a novel biologically-inspired training algorithm that allows for online training using exclusively locally available information;
- an application of OSTTP to spiking neural networks on two challenging temporal tasks showing competitive performance compared to BPTT and outperforming current local learning algorithms;
- a proof-of-concept implementation of OSTTP on a neuromorphic hardware-in-the-loop training system.

TABLE I
COMPARISON OF TRAINING ALGORITHMS FROM THE LITERATURE

Algorithm	No symmetric weights	No update-locking in time	No update-locking in space
BPTT	×	×	×
FA	√	-	×
DFA	√	-	×
DRTP	√	-	√
e-prop	√	√	×
OSTL	×	√	×
OSTL *rnd*	√	√	×
DECOLLE	√	√	√
ETLP	√	√	√
OSTTP	√	√	√

II. NEURAL DYNAMICS AND ONLINE SPATIO-TEMPORAL LEARNING WITH TARGET PROJECTION

Our biologically-inspired training algorithm is applicable to any neuron model, including RNNs and SNNs. However, we opted to use the spiking neural units (SNUs [8]) as they leverage the efficient spike-based communication and rich dynamics of biological neurons. The SNU, in its simplest form, implements the ubiquitous leaky integrate-and-fire (LIF) neuron model. Here, we outline the mathematical foundation of the SNU using two slightly different reset mechanisms: a full reset mechanism and a soft-reset mechanism, similar to [16]. Fig. 1a shows the SNU with the full reset mechanism, i.e., if the neuron emits a spike, the membrane potential is reset to 0. The governing equation for n SNUs with the full reset mechanism can be written as:

$$\boldsymbol{s}_l^t = \mathbf{g}\left(\boldsymbol{y}_0^t\mathbf{W} + \boldsymbol{y}_l^{t-1}\mathbf{H} + d\boldsymbol{s}_l^{t-1}\left(\mathbb{1} - \boldsymbol{y}_l^{t-1}\right)\right) \tag{1}$$

$$\boldsymbol{y}_l^t = \mathbf{h}\left(\boldsymbol{s}_l^t - \boldsymbol{b}_l\right), \tag{2}$$

where $\boldsymbol{s}_l^t \in \mathbb{R}^n$ is the membrane potential of layer l at time t, $\boldsymbol{y}_l^t \in \mathbb{R}^n$ is the output of layer l at time t, $\boldsymbol{y}_0^t \in \mathbb{R}^m$ signifies the input, $\mathbf{W} \in \mathbb{R}^{m\times n}$ and $\mathbf{H} \in \mathbb{R}^{n\times n}$ are the input and the recurrent weight matrices, while $d \in \mathbb{R}$ is the decay of the membrane potential. $\mathbf{g}$ and $\mathbf{h}$ represent the membrane and output activation functions, with $\mathbf{g}(x) = \mathbb{1}(x)$ and $\mathbf{h}(x) = \boldsymbol{\Theta}(x)$ for all our simulations and experiments. $\boldsymbol{\Theta}$ is the Heaviside step function, i.e. $\boldsymbol{\Theta}(\boldsymbol{x}) = 1$ if $\boldsymbol{x} > 0$ and 0 otherwise.

Fig. 1b shows the SNU with the soft-reset mechanism, i.e., if the neuron emits a spike, the current threshold is subtracted from the membrane potential, instead of resetting it to zero. The governing equation for n SNUs with the soft-reset mechanism can be written as:

$$\boldsymbol{s}_l^t = \boldsymbol{y}_0^t\mathbf{W} + \boldsymbol{y}_l^{t-1}\mathbf{H} + d\boldsymbol{s}_l^{t-1} - \boldsymbol{y}_l^{t-1}\boldsymbol{b}_l \tag{3}$$

$$\boldsymbol{y}_l^t = \boldsymbol{\Theta}(\boldsymbol{s}_l^t - \boldsymbol{b}_l). \tag{4}$$

While the full reset mechanism is more commonly observed in biological systems, the soft-reset mechanism maintains information in the membrane potential, even after a spiking event. Therefore, the soft-reset mechanism is particularly well-suited for temporal tasks that require long time horizons to be solved.

In gradient-based algorithms such as BPTT, illustrated in Fig. 2a, the update of any model parameter $\boldsymbol{\theta}_l$, for example the input weights $\mathbf{W}$, the recurrent weights $\mathbf{H}$, or the threshold $\boldsymbol{b}$, is performed according to

$$\Delta\boldsymbol{\theta}_l = -\eta\frac{\mathrm{d}E}{\mathrm{d}\boldsymbol{\theta}_l} = -\eta\sum_t\frac{\mathrm{d}E^t}{\mathrm{d}\boldsymbol{\theta}_l}, \tag{5}$$

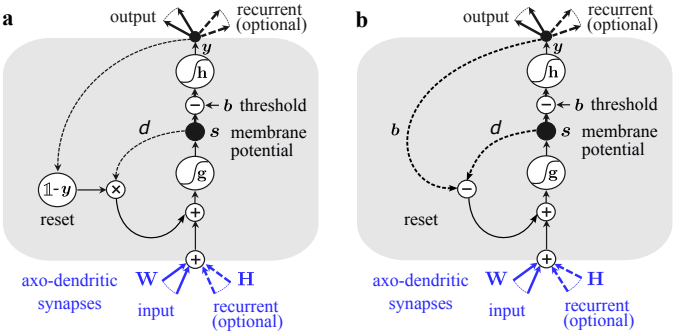

Fig. 1. **Spiking neural units with two different reset mechanisms. Drawings adapted from [8]. a** Full reset mechanism: in case the neuron emits a spike, the membrane potential is reset to 0. **b** Soft-reset mechanism: in case the neuron emits a spike, the threshold is subtracted from the membrane potential.

where η is the learning rate and $\frac{\mathrm{d}E}{\mathrm{d}\boldsymbol{\theta}_l}$ is the total derivative of the error with respect to the parameter θ to be updated. As one can see, BPTT requires to send information (i) through layers, starting from the last layer K, where the error E is computed, to the layer l, as well as (ii) through time, from timestep t until the first timestep $t = 0$. For the differentiation of the Heaviside step function, a pseudoderivative is employed, which is described in detail in Section III.

OSTL separates the gradient flow of BPTT into temporal components, traversing only through time within a single layer, and into spatial components, traversing the layers of the network architecture within a single timestep. It introduces eligibility traces $\boldsymbol{e}_l^{t,\theta_l}$ to capture the temporal components and learning signals $\mathbf{L}_l^t$ to capture the spatial components. The parameter update can then be formulated as

$$\Delta\boldsymbol{\theta}_l \approx \sum_t \mathbf{L_l^t}\mathbf{e_l^{t,\theta_l}}, \tag{6}$$

where the learning signals can be computed in a recurrent form using

$$\mathbf{L}_l^t = \mathbf{L}_{l+1}^t\left(\frac{\partial\boldsymbol{y}_{l+1}^t}{\partial\boldsymbol{s}_{l+1}^t}\frac{\partial\boldsymbol{s}_{l+1}^t}{\partial\boldsymbol{y}_l^t}\right) \text{ with } \mathbf{L}_K^t = \frac{\partial E^t}{\partial\boldsymbol{y}_K^t}, \tag{7}$$

and the eligibility traces as

$$\boldsymbol{\epsilon}_l^{t,\boldsymbol{\theta}_l} = \frac{\mathrm{d}\boldsymbol{s}_l^t}{\mathrm{d}\boldsymbol{\theta}_l} = \frac{\mathrm{d}\boldsymbol{s}_l^t}{\mathrm{d}\boldsymbol{s}_l^{t-1}}\boldsymbol{\epsilon}_l^{t-1,\boldsymbol{\theta}_l} + \left(\frac{\partial\boldsymbol{s}_l^t}{\partial\boldsymbol{\theta}_l} + \frac{\partial\boldsymbol{s}_l^t}{\partial\boldsymbol{y}_l^{t-1}}\frac{\partial\boldsymbol{y}_l^{t-1}}{\partial\boldsymbol{\theta}_l}\right) \tag{8}$$

$$\mathbf{e}_l^{t,\boldsymbol{\theta}_l} = \frac{\partial\boldsymbol{y}_l^t}{\partial\boldsymbol{s}_l^t}\boldsymbol{\epsilon}_l^{t,\boldsymbol{\theta}_l} + \frac{\partial\boldsymbol{y}_l^t}{\partial\boldsymbol{\theta}_l}. \tag{9}$$

Note that even though the generic form of the eligibility traces of e-prop appears similar to OSTL, the derivation of the latter maintains gradient equivalence to BPTT for a network with a single recurrent layer and is also applicable to multi-layer RNNs. More mathematical details and the proofs can be found in [17].

From the mathematical formulation of the learning signal expressed in (7), one can see that $\mathbf{L}_l^t$ can only be computed after the preceding layer has been computed, i.e., $\boldsymbol{y}_{l+1}^t$ is needed. Moreover, the learning signal traverses backwards through the layer, e.g., by using $\left(\frac{\partial\boldsymbol{y}_{l+1}^t}{\partial\boldsymbol{s}_{l+1}^t}\frac{\partial\boldsymbol{s}_{l+1}^t}{\partial\boldsymbol{y}_l^t}\right)$, and thus requiring symmetric weights W_{l+1} for the forward as well as for the backward passes. Therefore, it is evident that both the update-locking problem in space as well as the weight transport problem are present. To solve the latter, OSTL *rnd* has been introduced [17], pairing OSTL with DFA. In OSTL *rnd*, illustrated in

Fig. 2b, the learning signals for each layer are computed as follows, using a fixed random feedback matrix $\mathbf{B}_l$

$$\mathbf{L}_l^t = \begin{cases} \frac{\mathrm{d}E^t}{\mathrm{d}\boldsymbol{\theta}_l} & l = K \\ \mathbf{B}_l \frac{\mathrm{d}E^t}{\mathrm{d}\boldsymbol{\theta}_l} & l \leq K. \end{cases} \tag{10}$$

While OSTL *rnd* addresses the weight transport problem, it still leaves the update-locking problem in space unsolved. DRTP resolves this issue by directly propagating the target vector $\boldsymbol{y}^{t,*}$ to the loss function, as well as to the individual layers, also using a fixed random matrix $\mathbf{B}_l$, see Fig. 2c. In this work we combine OSTL and DRTP, resulting in our proposed training algorithm OSTTP, see Fig. 2d. While the eligibility traces of OSTL remain unchanged, the learning signal for this algorithm can be computed as

$$\mathbf{L}_l^t = \begin{cases} \frac{\mathrm{d}E^t}{\mathrm{d}\boldsymbol{\theta}_l} & l = K \\ \mathbf{B}_l \boldsymbol{y}^{t,*} & l \leq K. \end{cases} \tag{11}$$

As one can see, the learning signals for all layers of the network can be computed separately, without waiting for the preceding layer to be computed. Therefore, OSTTP resolves the update-locking problems, both in time and in space. Tab. I summarizes the major characteristics of the individual training algorithms from the literature.

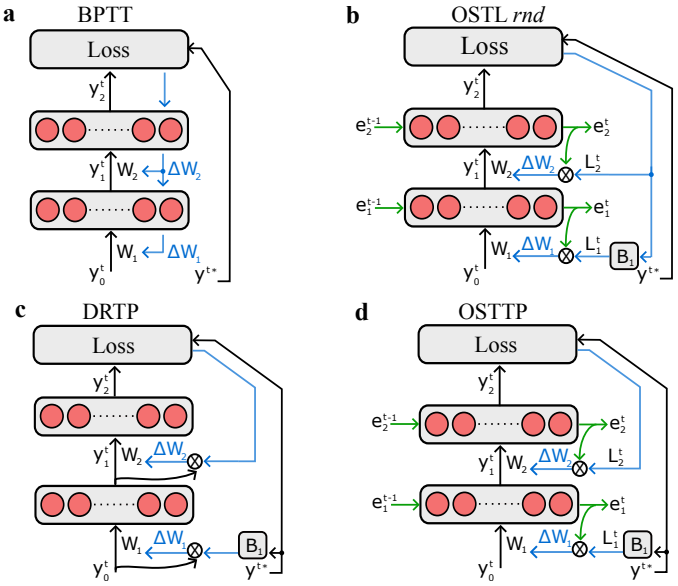

Fig. 2. **Comparison of training algorithms. a** In BPTT, the target vector $\boldsymbol{y}^{t,*}$ is only provided to the loss layer and the parameter updates are computed based on gradient signals traversing through the network architecture in space and time. **b** In OSTL, the gradient components of BPTT are cleanly separated into temporal gradients, captured by eligibility traces $\mathbf{e}_l^{t,\boldsymbol{\theta}_l}$, and into spatial gradients captured by the learning signals $\mathbf{L}_l^t$. **c** In DRTP, the target vector $\boldsymbol{y}^{t,*}$ is directly provided to the individual layers, allowing them to be updated without having to wait for the preceding layer to be computed. **d** OSTTP combines the eligibility traces from OSTL with the projected target signals from DRTP, making it a fully online training algorithm without any update-locking problems.

III. Results

A. Software simulations

The OSTTP performance was evaluated on two temporal datasets: The Johann Sebastian Bach (JSB) music chorales [22] and the Spiking Heidelberg Digits (SHD) [23].

For the first dataset, the objective at every timestep t is to predict the musical notes of a piano that will be played at timestep $t+1$, based on the already played notes until timestep t. To solve this task, we employed a single feedforward layer of 150 SNUs, using the full reset mechanism, and a dense readout layer with 88 units and a sigmoidal activation function. Fig. 3a illustrates the network architecture as well as its inputs and outputs. For evaluation, we use the negative log-likelihood metric, where a lower value indicates a better prediction of the network. The SNUs were configured with trainable parameters $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{H}, \boldsymbol{b}\}$, a decay of $d = 0.4$ for BPTT, $d = 0.5$ for OSTL, $d = 0.5$ for DRTP and $d = 0.6$ for OSTTP, as well as with the pseudoderivative $\frac{\mathrm{d}\boldsymbol{\Theta}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \mathbb{1} - \mathbf{tanh}^2(\boldsymbol{x})$. Furthermore, we employed a standard stochastic gradient descent optimizer (SGD) with learning rates of 0.001 for BPTT, 0.0005 for OSTL, 0.0005 for DRTP and 0.0005 for OSTTP. As one can see in Fig. 3b, OSTL maintains gradient equivalence to BPTT for this network architecture. However, both algorithms suffer from update-locking problems. When using DRTP, modified for a temporal task where the parameter updates are accumulated over time, the update-locking problem in space is resolved, but the performance of the network deteriorates. This is because, in contrast to the gradient signals in BPTT or in OSTL, the target signals used to compute the parameter updates at timestep t do not contain information from previous timesteps. On the contrary, OSTTP combines the benefits from the traces of OSTL, with the learning signals from DRTP, thereby resolving the update-locking problem of OSTL and at the same time lowering the performance gap between DRTP and BPTT.

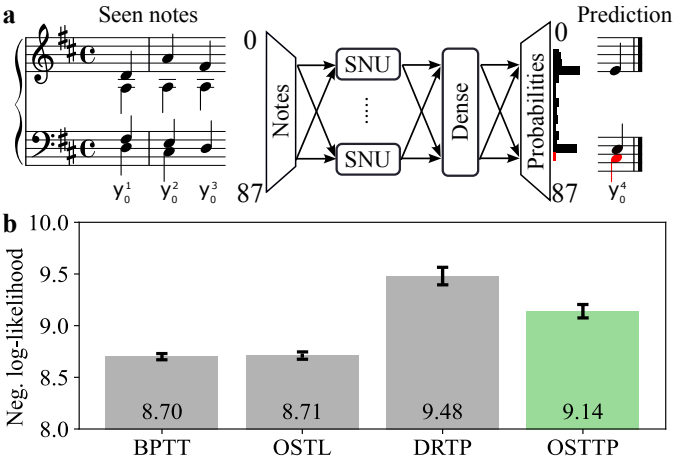

Fig. 3. **Music prediction task using the JSB dataset. a** Network architecture comprising an SNU layer with the full reset mechanism and a dense output layer with a sigmoidal activation function. The inputs to the network are time sequences of binary encoded vectors, representing the notes played on the 88 keys of a piano. The network takes these binary-encoded inputs for timesteps $t = 0$ to t and produces a probability distribution of the 88 keys that will be played at timestep $t+1$. We used the standard split for this dataset consisting of 229 music pieces for training and 77 pieces for testing. **b** Performance comparison of the different training algorithms.

For the SHD dataset, the objective is to classify spoken digits. We employed a single recurrent layer of 450 SNUs, using the soft-reset mechanism, and a leaky-integrating output layer of 20 units. Fig. 4a illustrates the network architecture as well as its inputs and outputs. The SNUs were configured with trainable parameters $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{H}\}$, a decay of $d = 0.83$ for BPTT, $d = 0.95$ for OSTTP, as well as with the pseudoderivative $\frac{\mathrm{d}\boldsymbol{\Theta}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \frac{1}{(100|\boldsymbol{x}|+1)^2}$. The leaky integrator was configured with a trainable parameter $\theta = \{\mathbf{W}\}$, with a decay of $\tau = 0.98$ for BPTT and of $\tau = 0.99$ for OSTTP. Furthermore, we employed the ADAM optimizer with learning rates

of 0.00035 for BPTT and 0.0002 for OSTTP. For evaluation we employed the common accuracy metric, which we computed as the maximum testing accuracy over 200 training epochs, averaged over 5 random seeds. The results are shown in Fig. 4b, where OSTTP achieves an accuracy of $(77 \pm 0.8)\%$ accuracy. Furthermore, its performance surpasses the one of ETLP, despite ETLP using threshold adaptive neurons that increase the computational complexity. Note that network performance degrades to $(75 \pm 0.8)\%$ when using the full reset mechanism, because the long-term temporal dependencies cannot be captured.

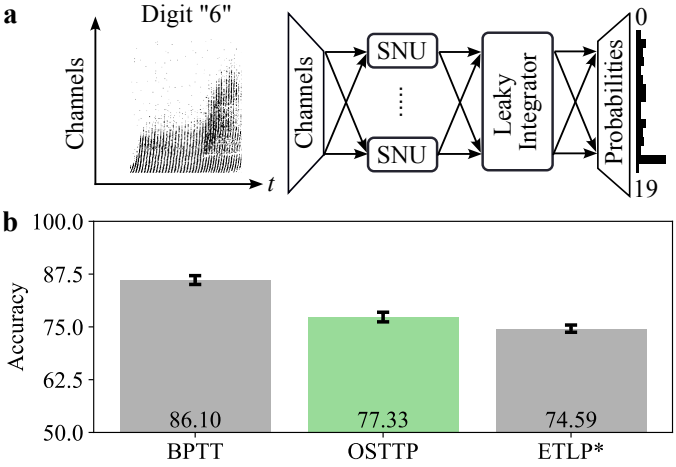

Fig. 4. **Temporal classification task using the SHD dataset. a** Network architecture comprising an SNU layer with the soft-reset mechanism and a leaky integrating output layer. The inputs to the network are spike trains, representing recordings from speakers uttering digits from zero to nine in both German and English. The original audio recordings were converted into spike trains, using an artificial cochlear model. The dataset includes 12 speakers and a total number of 10420 samples and we used the standard split of 8156 examples for training and 2264 for testing. **b** Performance comparison of the different training algorithms. *ETLP [20] developed in parallel to this work.

B. Demonstration on neuromorphic hardware

We also demonstrate the viability of OSTTP for a neuromorphic hardware system. To this end, we employ an in-memory computing platform [21], based on memristive phase-change memory (PCM) devices [24]. These devices belong to a special class of emerging materials that can represent information using their conductance values G and simultaneously perform computations within the memory elements. The particular neuromorphic hardware system used in this work is comprised of two chips, each hosting a crossbar arrangement of 256x256 unit cells, consisting of four PCM devices in a differential configuration [21]. We leveraged this setup to map the synaptic weights $\mathbf{W}_1$ of the SNU layer, configured with $d = 0.6$ and $\frac{\mathrm{d}\boldsymbol{\Theta}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \mathbb{1} - \mathbf{tanh}^2(\boldsymbol{x})$, as well as $\mathbf{W}_2$ of the dense output layer onto one chip. To parallelize the learning signal computation, the fixed random matrix $\mathbf{B}$ was mapped on the second chip. Each element of these matrices is represented by four PCM devices as $W_{\mathrm{ij}} = ((G^+_{\mathrm{A}} + G^+_{\mathrm{B}}) - (G^-_{\mathrm{A}} + G^-_{\mathrm{B}}))/2$. Fig. 5 shows an illustration of the realization of OSTTP on the NMHW. The weights are mapped onto the NMHW, which is used to perform the matrix multiplications, while the eligibility traces are calculated in software. In order to change the synaptic weights, electrical pulses are sent to the PCM devices, either increasing or decreasing their conductance levels. We used this hardware-in-the-loop setup, with SGD using a learning rate of 0.001 which we multiplied by 0.6 every 5 epochs, to train our network and achieved a negative log-likelihood of 11.0 ± 0.2 for the JSB task. Note that the network architecture used to solve the SHD dataset was beyond the size capabilities of the experimental NMWH.

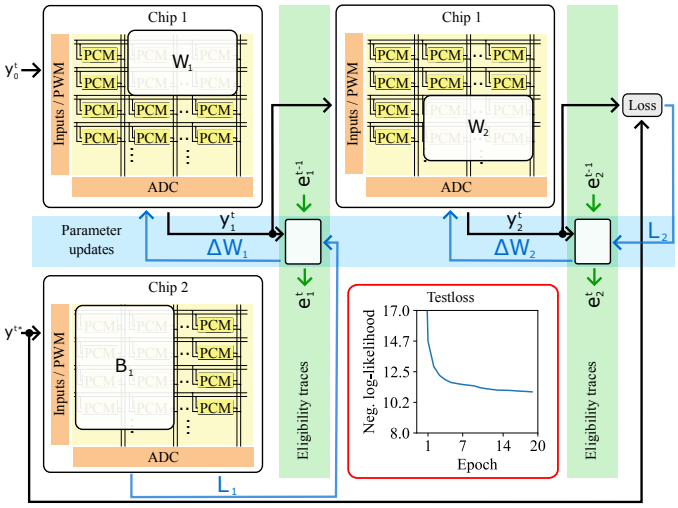

Fig. 5. **Proof-of-concept demonstration of OSTTP on NMHW.** Two chips with crossbar arrays implement the network architecture used for the JSB task. The weights of the SNU layer $\mathbf{W}_1$ and of the dense layer $\mathbf{W}_2$ are placed on chip 1, while the fixed random matrix $\mathbf{B}$ is placed on chip 2. The eligibility traces are kept in software and the computed parameter updates are applied to the NMHW in the form of individual electrical pulses. The red inset shows the test loss evolution over epochs.

IV. Discussion

In this paper, we introduced a novel biologically-inspired training algorithm called online spatio-temporal learning with target projection. This algorithm solves several shortcomings of BPTT, in particular the weight transport problem, the requirement to send information backwards through time and the update-locking problems, both in time and space. It enables a network to be trained fully online, exclusively with information that is locally available to the individual parameters. Although OSTTP has been formulated to accumulate weight updates in time before applying them to the parameters, which would make it non-local in time (see (6)), there are approaches to alleviate this by applying the parameter updates directly at every timestep [17], [25]. Additionally, we demonstrated the performance of OSTTP on two different temporal tasks and achieved performance levels close to the BPTT baseline. Moreover, we showcased the applicability of our algorithm on a PCM-based neuromorphic hardware system, using a hardware-in-the-loop setting. This demonstration highlights the main advantages of OSTTP: firstly, it does not require symmetric weights, which cannot be used with the specific NMHW, and secondly, the learning signals of OSTTP for all layers can be computed in parallel to the forward pass because only local information is used. This makes OSTTP a perfect match for low-latency massively-parallel NMHW systems operating in a fully forward manner.

Acknowledgment

We thank the In-Memory Computing team at IBM for their technical support with the PCM-based NMHW as well as the IBM Research AI Hardware Center. This project has received funding from the ERA-NET CHIST-ERA-18-ACAI-004 programme by SNSF under the project number 20CH21_186999 / 1.

REFERENCES

[1] A. Graves, "Sequence Transduction with Recurrent Neural Networks," *arXiv*, Nov. 2012. [Online]. Available: https://arxiv.org/abs/1211.3711v1

[2] W. Chan *et al.*, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Mar. 2016, pp. 4960–4964.

[3] J. Li, "Recent Advances in End-to-End Automatic Speech Recognition," *arXiv*, Nov. 2021.

[4] D. Bahdanau *et al.*, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv*, Sep. 2014.

[5] B. Lim *et al.*, "Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting," *arXiv*, Dec. 2019.

[6] ——, "Time-series forecasting with deep learning: a survey," *Phil. Trans. R. Soc. A.*, vol. 379, no. 2194, p. 20200209, Apr. 2021.

[7] E. O. Neftci *et al.*, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[8] S. Woźniak *et al.*, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nat. Mach. Intell.*, vol. 2, pp. 325–336, Jun. 2020.

[9] T. Bohnstingl *et al.*, "Speech Recognition Using Biologically-Inspired Neural Networks," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2022, pp. 6992–6996.

[10] Y. Bengio *et al.*, "Towards Biologically Plausible Deep Learning," *arXiv*, Feb. 2015.

[11] T. P. Lillicrap *et al.*, "Backpropagation and the brain," *Nat. Rev. Neurosci.*, vol. 21, pp. 335–346, Jun. 2020.

[12] C. Frenkel *et al.*, "Bottom-Up and Top-Down Neural Processing Systems Design: Neuromorphic Intelligence as the Convergence of Natural and Artificial Intelligence," *arXiv*, Jun. 2021.

[13] T. P. Lillicrap *et al.*, "Random synaptic feedback weights support error backpropagation for deep learning," *Nat. Commun.*, vol. 7, no. 13276, pp. 1–10, Nov. 2016.

[14] A. Nøkland, "Direct Feedback Alignment Provides Learning in Deep Neural Networks," *arXiv*, Sep. 2016.

[15] C. Frenkel *et al.*, "Learning Without Feedback: Fixed Random Learning Signals Allow for Feedforward Training of Deep Neural Networks," *Front. Neurosci.*, vol. 15, Feb. 2021.

[16] G. Bellec *et al.*, "A solution to the learning dilemma for recurrent networks of spiking neurons," *Nature Communications*, vol. 11, no. 1, p. 3625, Jul 2020. [Online]. Available: https://doi.org/10.1038/s41467-020-17236-y

[17] T. Bohnstingl *et al.*, "Online spatio-temporal learning in deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2022.

[18] J. Kaiser *et al.*, "Synaptic Plasticity Dynamics for Deep Continuous Local Learning (DECOLLE)," *Front. Neurosci.*, vol. 14, May 2020.

[19] A. Ororbia *et al.*, "Continual Learning of Recurrent Neural Networks by Locally Aligning Distributed Representations," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 10, pp. 4267–4278, Jan. 2020.

[20] F. M. Quintana *et al.*, "ETLP: Event-based Three-factor Local Plasticity for online learning with neuromorphic hardware," *arXiv*, Jan. 2023.

[21] R. Khaddam-Aljameh *et al.*, "HERMES Core – A 14nm CMOS and PCM-based In-Memory Compute Core using an array of 300ps/LSB Linearized CCO-based ADCs and local digital processing," in *2021 Symposium on VLSI Technology*. IEEE, Jun. 2021, pp. 1–2. [Online]. Available: https://ieeexplore.ieee.org/document/9508706

[22] N. Boulanger-Lewandowski *et al.*, "Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription," *arXiv*, Jun. 2012.

[23] B. Cramer *et al.*, "The Heidelberg Spiking Data Sets for the Systematic Evaluation of Spiking Neural Networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 7, pp. 2744–2757, Dec. 2020.

[24] A. Sebastian *et al.*, "Memory devices and applications for in-memory computing - Nature Nanotechnology," *Nat. Nanotechnol.*, vol. 15, no. 7, pp. 529–544, Jul 2020.

[25] C. Frenkel *et al.*, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," in *2022 IEEE International Solid- State Circuits Conference (ISSCC)*. IEEE, Feb. 2022, vol. 65, pp. 1–3.