Effective Online SNN Training with One-Step Backpropagation

Saya Higuchi
Adaptive AI Lab
Institute for Robotics and Cognitive Systems
University of Lübeck
Ratzeburger Allee 160
23562 Lübeck, Germany
sa.higuchi@uni-luebeck.de

Federico Corradi
Neuromorphic Edge Computing Systems Lab
Department of Electrical Engineering
Eindhoven University of Technology
Groene Loper 19, Flux room 4.130
5612 AP Eindhoven, The Netherlands
f.corradi@tue.nl

Sander M. Bohté
Machine Learning Group
Centrum Wiskunde & Informatica (CWI)
Science Park 123
1098 XG Amsterdam, The Netherlands
S.M.Bohte@cwi.nl

Sebastian Otte
Adaptive AI Lab
Institute for Robotics and Cognitive Systems
University of Lübeck
Ratzeburger Allee 160
23562 Lübeck, Germany
sebastian.otte@uni-luebeck.de

Abstract

Backpropagation through time (BPTT) remains the gold-standard for training recurrent spiking neural networks, but its need to store long temporal computation graphs makes it memory-intensive and incompatible with strict online updates. This has motivated a range of alternative online learning rules, such as e-prop, further trace-based methods, and forward-only approximations, which reduce sequence-length-dependent overhead but typically require custom implementations and often sacrifice task performance. In this work, we revisit the simplest possible alternative: truncated BPTT with truncation length $k = 1$ (tBPTT$_1$). Although this setting is usually regarded as an overly limited-horizon baseline with poor temporal credit assignment, we show that it is a widely underestimated learning strategy. In a standard surrogate gradient learning setup, tBPTT$_1$ achieves performance competitive with or better than more sophisticated online learning rules. Our experiments identify two key ingredients for this result: a substantially smaller learning rate than commonly used and an optimizer with slow temporal averaging through its momentum statistics. These findings suggest that, for many practical spiking network settings, elaborate online credit-assignment rules may not be necessary: plain one-step backprop, when paired with appropriate optimization, appears as an overlooked training strategy provides effective, memory-efficient, and implementation-friendly learning.

1 Introduction

Spiking neural networks (SNNs) are well suited for event-driven computation because they process information as sparse spike trains and maintain internal states that evolve over time, making them attractive for neuromorphic sensors and low-power hardware. While backpropagation through time (BPTT) remains a strong training baseline for SNNs, it requires storing every intermediate step and updating the model in an offline manner, which conflict with both biological plausibility and memory-constrained deployment [11, 16].

The Third Austrian Symposium on AI and Vision (AIROV26).

This has motivated a range of online learning rules, the most prominent being E-prop, which approximates BPTT using eligibility traces combined with local learning signals [2]. Other approaches pursue temporally local plasticity through biologically inspired three-factor rules, such as ETLP and S-TLLR [17, 1]. In contrast, DECOLLE enables local learning through layer-wise auxiliary losses [9]. More recent methods further reduce the need for backward-through-time computation by relying on forward or target-based learning signals, as in OSTTP and traces propagation (TP) [13, 16]. These methods clearly show that online learning is feasible, but they often require custom update rules, additional auxiliary pathways, or model-specific derivations. Another form of online learning is the forward propagation through time (FPTT) [8], which introduced forward-propagated learning signals that regularize training under truncated temporal credit assignment and help stabilize recurrent spiking networks [20]. Recent works have implemented and explored truncated backpropagation through time (tBPTT), which is memory efficient but restricted in its temporal credit assignment [3, 7].

From a biological perspective, such temporally truncated gradient-based methods are better understood as local approximations than as literal models of neural learning. BPTT and related gradient-based approaches remain biologically problematic because they rely on backward credit assignment through stored trajectories, whereas biological accounts of temporal credit assignment more commonly appeal to eligibility traces, local synaptic dynamics, and modulatory learning signals [11, 6, 2]. At the same time, adaptive update rules can maintain slow hidden state across learning steps, which can be loosely interpreted through metaplastic or synaptic-dynamic views of learning [5, 18]. This perspective does not make truncated gradient methods biologically exact, but it suggests that temporally local training combined with stateful adaptive optimization may provide a useful bridge between engineered learning algorithms and biologically motivated locality constraints.

In this work, we study a simple online learning approach based on standard automatic differentiation. We apply tBPTT with fixed truncation length $k = 1$ (tBPTT$_1$) in combination with the Adam optimizer [10]. While this setup removes explicit temporal credit assignment, Adam implicitly aggregates information over time through its first- and second-moment estimates and stabilizes per-time-step updates via second-moment normalization. Our experiments identify two key ingredients for making this setting work in practice: the use of a substantially smaller learning rate than commonly employed, and an optimizer with slow temporal averaging through its momentum statistics. Although these statistics do not recover exact long-range gradients, they preserve useful directional and scale information across successive updates. The resulting method is straightforward to implement with standard deep-learning tools and achieves competitive performance on N-MNIST and SHD without requiring explicit eligibility traces or custom learning rules.

2 Methods

We consider feedforward (FF) and recurrent SNNs on two benchmark datasets for online learning, SHD [4] and N-MNIST [12], with input dimensions of 700 and 2,312, and time windows of 10 ms and 1 ms, respectively. For the leaky integrate-and-fire (LIF) neurons, the membrane potential is updated as $u_t = \alpha(u^{t-1} - \vartheta z^{t-1}) + I^t$, where u^t is the membrane potential, $\alpha = exp(-dt/\tau_m) \in (0,1)$ is the fixed leakage factor, I^t denotes the total synaptic input, ϑ is the threshold. The spike output is given by $z^t = H(u^t - \vartheta)$ with $H(\cdot)$ the Heaviside step function. The double-Gaussian surrogate function [19] with FGI [14] is used. For recurrent networks, the synaptic input can be written as $I^t = W^t_{\text{in}} x^t + W^t_{\text{rec}} z^{t-1} + b$, where x^t is the external input, W^t_{in} and W^t_{rec} are input and recurrent weights, and b the optional bias.

The output layer is modeled as a leaky integrator, $o^t = \kappa o^{t-1} + W^t_{\text{out}} z_t + b_{\text{out}}$, where o^t denotes the output state, $\kappa = exp(-dt/\tau_m) \in (0,1)$ is the fixed output leak factor, and W^t_{out} the output weights. The logits derived from o^t are used to compute a per-time-step cross-entropy loss $\ell^t = \ell(\hat{y}^t, y)$, where $\hat{y}^t$ is the prediction at time step t and y is the target label. Note that the weights have increment t due to the online update via tBPTT$_1$. We update the parameters and truncate the computation graph at every time step.

2.1 The optimizer as a temporal gradient accumulator and stabilizer

The parameters W are updated with the PyTorch implementation of the Adam optimizer [15, 10]. Given the instantaneous gradient $g^t = {}^{\partial \ell^t}/_{\partial W^t}$ at time step t, Adam maintains the first and second moments $m_t = \beta_1 m_{t-1} + (1-\beta_1)g^t$ and $v_t = \beta_2 v_{t-1} + (1-\beta_2)\cdot(g^t)^2$ with the bias-corrected estimates $\hat{m}_t = {}^{m_t}/_{1-\beta_1^t}$ and $\hat{v}_t = {}^{v_t}/_{1-\beta_2^t}$. The parameters are updated as: $W^t = W^{t-1} - {}^{\eta\hat{m}_t}/_{\sqrt{\hat{v}_t}+\epsilon}$.

3 Results

The vanilla stochastic gradient descent (SGD) was first tested to check the effect of the optimizer as a temporal gradient accumulator and stabilizer. Figure 1 shows the result of increasing momentum rate without changing any other hyperparameters. SGD without momentum only achieved $19.58 \pm 1.00\%$ whereas increasing momentum strength increased the performance of the model.

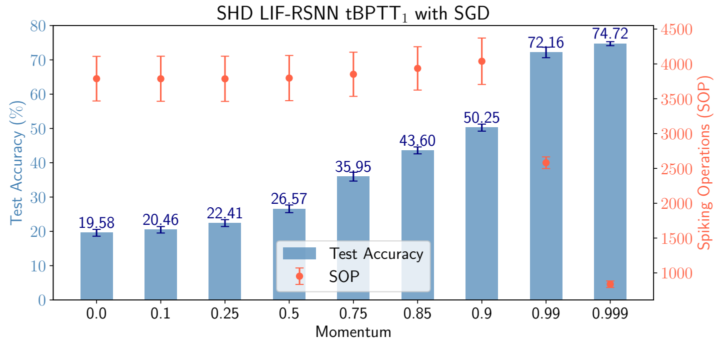

Figure 1: SHD LIF-RSNN trained with tBPTT$_1$ using the stochastic gradient descent with varying momentum term. Best test accuracy over 5 runs.

Furthermore, a substantial reduction in the average spiking operation (SOP) can be observed, indicating that the models learn sparser activity patterns. This reduction accompanies the performance gains obtained with stronger temporal accumulation in the optimizer.

Table 1 places the proposed method in the context of existing online and local-learning approaches. On N-MNIST, tBPTT$_1$ reaches $97.43 \pm 0.15\%$, outperforming DECOLLE and TP while remaining below full BPTT and e-prop. On feedforward SHD, it achieves $68.19 \pm 1.12\%$, slightly exceeding TP ($67.06 \pm 0.96\%$) and clearly outperforming e-prop and ETLP, although still trailing offline BPTT. The strongest result appears on recurrent SHD, where tBPTT$_1$ reaches $84.29 \pm 0.98\%$, outperforming all compared online methods and even slightly surpassing full BPTT ($83.23 \pm 1.00\%$). Overall, these results show that even under the extreme truncation condition $k = 1$, a carefully tuned one-step training setup, combining a small learning rate with slow temporal gradient filtering, remains highly competitive, especially in recurrent settings where temporal structure is most relevant.

4 Discussion

Our results highlight an apparent contradiction: learning remains strong even when temporal credit assignment is reduced to a single step. Because the graph is truncated at every step, the method does not explicitly recover long-range temporal Jacobian chains as in BPTT or cell-to-cell temporal dependencies as in e-prop. The only information available for learning is the stream of adapting local gradients $\{g_t\}_{t=1}^{T}$. A naive expectation would therefore be that performance collapses when training becomes fully online. Our results show that this need not be the case when the optimizer is allowed to accumulate and rescale these local gradients across time.

The central idea of this paper is to interpret (m_t, v_t) as a compact memory of the recent gradient history, as well as a stabilizer for per step updates. In other online learning methods, temporal information is stored explicitly in eligibility traces or other neuron-specific state variables [2, 17]. Here, by contrast, part of that temporal accumulation is carried by a slowly evolving optimizer state while learning remains stable. The first-moment term m^t smooths the direction of successive local gradients, while the second-moment term v^t normalizes their scale and reduces sensitivity to sharp fluctuations.

Table 1: Comparison to SoTA results on N-MNIST and SHD. Our results for N-MNIST and SHD report average best test accuracy over 5 runs and average highest test accuracy over 5 runs, respectively.

Model	Architecture Type	Neuron Type	Number of Neurons	Local Learning	Time Steps	Test Accuracy
			N-MNIST			
BPTT	FF	LIF	200	✗	100	98.45 ± 0.04[1]
eProp [2]	FF	LIF	200	Partial (time)	100	**97.90**[2]
DECOLLE [9]	FF	LIF	200	✓	100	96.27[2]
TP [16]	FF	LIF	200	✓	100	97.33 ± 0.06
tBPTT_1 (ours)	FF	LIF	200	Partial (time)	100	97.43 ± 0.15
			SHD			
BPTT	FF	LIF	450	✗	100	75.85 ± 0.48[1]
eProp [2]	FF	LIF	450	Partial (time)	100	63.04[2]
ETLP [17]	FF	ALIF	450	✓	100	59.19
TP [16]	FF	LIF	400	✓	100	67.06 ± 0.96
tBPTT_1 (ours)	FF	LIF	450	Partial (time)	100	$\mathbf{68.19 \pm 1.12}$
BPTT	Recurrent	LIF	450	✗	100	83.23 ± 1.00[1]
S-TLLR [1]	Recurrent	LIF	450	Partial (time)	100	78.24 ± 1.84
eProp [2]	Recurrent	LIF	450	Partial (time)	100	80.79[2]
ETLP [17]	Recurrent	ALIF	450	✓	100	74.59
TP [16]	Recurrent	LIF	450	✓	100	81.80 ± 0.51
tBPTT_1 (ours)	Recurrent	LIF	450	Partial (time)	100	$\mathbf{84.29 \pm 0.98}$

[1] Results from [16]. [2] Results from [17].

This does not reconstruct exact long-range credit assignment, since dependencies through earlier hidden states remain absent, but it does provide a simple mechanism through which past gradients continue to influence future updates. In this sense, our findings also clarify the relation to FPTT [8, 20]: rather than enforcing temporal consistency through an explicit regularizer across successive updates, much of the same stabilizing effect appears to arise here from a small learning rate together with slow temporal filtering in the optimizer itself, without auxiliary regularizers or extra buffer variables.

From an implementation perspective, this is appealing because it requires no custom backward rule beyond standard surrogate-gradient differentiation. The method can be realized with ordinary autograd, one forward pass and one backward pass per time step, and a standard optimizer call. As a consequence, it offers a lightweight baseline for online SNN learning and a useful reference point for understanding how much temporal structure can already be captured by optimizer dynamics alone.

5 Conclusion

We revisited one-step truncated BPTT as a minimal online training strategy for SNNs and found that, despite its extreme temporal truncation, it can remain highly effective for standard surrogate-gradient training when combined with careful gradient step scaling and slow temporal filtering. Across N-MNIST and SHD, tBPTT_1 achieved competitive performance against more specialized online learning methods, and on recurrent SHD it even slightly surpassed full BPTT. Our results show that this behavior depends critically on two ingredients: a substantially smaller learning rate than commonly used and optimizer dynamics that accumulate and rescale local gradients over time through slow momentum statistics. Taken together, these findings suggest that effective online SNN learning does not necessarily require elaborate custom credit-assignment rules, and that plain one-step backpropagation, when combined with appropriate optimization, provides a simple, memory-efficient, and strong baseline for future work.

References

[1] Marco Paul E Apolinario and Kaushik Roy. S-tllr: Stdp-inspired temporal local learning rule for spiking neural networks. *Transactions on Machine Learning Research*.

[2] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature communications*, 11(1):3625, 2020.

[3] Hojae Choi, Jaewook Kim, Jongkil Park, Seongsik Park, Hyun Jae Jang, Seung Hwan Lee, Byeong-Kwon Ju, and YeonJoo Jeong. Star-snn: A spatio-temporal adaptive recurrent spiking neural network with separated propagation surrogate gradient for hardware efficient real-time learning. *Neurocomputing*, page 132968, 2026.

[4] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2744–2757, 2020.

[5] Shiva Farashahi, Christopher H Donahue, Peyman Khorsand, Hyojung Seo, Daeyeol Lee, and Alireza Soltani. Metaplasticity as a neural substrate for adaptive learning and choice under uncertainty. *Neuron*, 94(2):401–414, 2017.

[6] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. *Frontiers in neural circuits*, 12:53, 2018.

[7] Stijn Groenen, Marzieh Hassanshahi Varposhti, and Mahyar Shahsavari. Gazescrnn: Event-based near-eye gaze tracking using a spiking neural network. *arXiv preprint arXiv:2503.16012*, 2025.

[8] Anil Kag and Venkatesh Saligrama. Training recurrent neural networks via forward propagation through time. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5189–5200. PMLR, 18–24 Jul 2021.

[9] Jacques Kaiser, Hesham Mostafa, and Emre Neftci. Synaptic plasticity dynamics for deep continuous local learning (decolle). *Frontiers in Neuroscience*, 14:424, 2020.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Timothy P Lillicrap and Adam Santoro. Backpropagation through time and the brain. *Current opinion in neurobiology*, 55:82–89, 2019.

[12] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9:437, 2015.

[13] Thomas Ortner, Lorenzo Pes, Joris Gentinetta, Charlotte Frenkel, and Angeliki Pantazi. Online spatio-temporal learning with target projection. In *5th IEEE International Conference on Artificial Intelligence Circuits and Systems, AICAS 2023*. IEEE, 2023.

[14] Sebastian Otte. Flexible and efficient surrogate gradient modeling with forward gradient injection. In *Proceedings of Austrian Symposium on AI, Robotics, and Vision 2024*, pages 451–459. University of Innsbruck, 2024.

[15] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[16] Lorenzo Pes, Bojian Yin, Sander Stuijk, and Federico Corradi. Traces propagation: memory-efficient and scalable forward-only learning in spiking neural networks. *Neuromorphic Computing and Engineering*, 6(1):014002, 2026.

[17] Fernando M Quintana, Fernando Perez-Peña, Pedro L Galindo, Emre O Neftci, Elisabetta Chicca, and Lyes Khacef. Etlp: event-based three-factor local plasticity for online learning with neuromorphic hardware. *Neuromorphic Computing and Engineering*, 4(3):034006, 2024.

[18] Yukun Yang and Peng Li. Synaptic dynamics realize first-order adaptive learning and weight symmetry. *arXiv preprint arXiv:2212.09440*, 2022.

[19] Bojian Yin, Federico Corradi, and Sander M Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10):905–913, 2021.

[20] Bojian Yin, Federico Corradi, and Sander M Bohté. Accurate online training of dynamical spiking neural networks through forward propagation through time. *Nature Machine Intelligence*, 5(5):518–527, 2023.