Balanced Resonate-and-Fire Neurons

Saya Higuchi [1] **Sebastian Kairat** [1] **Sander M. Bohté** [2] **Sebastian Otte** [1]

Abstract

The resonate-and-fire (RF) neuron, introduced over two decades ago, is a simple, efficient, yet biologically plausible spiking neuron model, which can extract frequency patterns within the time domain due to its resonating membrane dynamics. However, previous RF formulations suffer from intrinsic shortcomings that limit effective learning and prevent exploiting the principled advantage of RF neurons. Here, we introduce the balanced RF (BRF) neuron, which alleviates some of the intrinsic limitations of vanilla RF neurons and demonstrates its effectiveness within recurrent spiking neural networks (RSNNs) on various sequence learning tasks. We show that networks of BRF neurons achieve overall higher task performance, produce only a fraction of the spikes, and require significantly fewer parameters as compared to modern RSNNs. Moreover, BRF-RSNN consistently provide much faster and more stable training convergence, even when bridging many hundreds of time steps during backpropagation through time (BPTT). These results underscore that our BRF-RSNN is a strong candidate for future large-scale RSNN architectures, further lines of research in SNN methodology, and more efficient hardware implementations.

1. Introduction

Artificial neural networks (ANNs) have become the main method for solving machine learning problems in recent years (Goodfellow et al., 2016; Wu & Feng, 2018; Abiodun et al., 2019). However, they require massive computation and energy, making them inefficient for large-scale real-world applications—particularly in the realm of edge computing— due to the deep non-linear continuous estimators to represent features of the data (Pfeiffer & Pfeil, 2018).

[1]Adaptive AI Lab, Institute of Robotics and Cognitive Systems University of Lübeck, Germany [2]Machine Learning Group, Centrum Wiskunde & Informatica (CWI)
Amsterdam, The Netherlands. Correspondence to: Saya Higuchi <saya.higuchi@student.uni-luebeck.de>, Sebastian Otte <sebastian.otte@uni-luebeck.de>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

Spiking neural networks (SNNs) circumvent this drawback by processing information through the precise timing of action potentials, or *spikes*. SNNs can potentially be more efficient compared to ANNs due to their event-driven property, for which computation is only required when spikes are propagated in the system (Paugam-Moisy & Bohté, 2012). Furthermore, SNNs are self-recurrent, as the neurons have a dynamic internal state that modulates their activity over time, and are more biologically realistic than conventional ANNs. Some examples of spiking neurons with ascending biological plausibility include the leaky integrate-and-fire (LIF) neuron, the adaptive leaky integrate-and-fire (ALIF) neuron (Bellec et al., 2018), the Izhikevich neuron (Izhikevich, 2003), and the Hodgkin-Huxley (HH) neuron (Hodgkin & Huxley, 1952). While the HH model is too computationally expensive for practical application, it exhibits rich and complex membrane dynamics that simpler models lack.

One mode of dynamics gaining interest in ML applications in particular is the resonating behavior seen in oscillatory neurons. Subthreshold oscillations of the membrane potential have been observed in various mammalian nervous systems, including the neurons in the frontal cortex (Llinas et al., 1991), in the thalamus (Pedroarena & Llinás, 1997), and in layer II of the medial entorhinal cortex (Alonso & Llinás, 1989), which is crucial in spatial information processing (Doeller et al., 2010). The resonate-and-fire (RF) neuron proposed by Izhikevich (2001) models such damped or sustained subthreshold oscillations of biological neurons: the RF neuron fires when the frequency of the incoming spikes matches that of the neuron's damped oscillation. Incoming signals with higher frequency leads to more firing for a LIF neuron but less firing for an RF neuron with slow oscillation. As the RF neuron is similarly computationally efficient as the LIF neuron, it is a spiking neuron model potentially suitable for large-scale SNNs (Izhikevich, 2001).

Previous works showed RF neurons implemented in the Intel Loihi 2 (Davies et al., 2018), a neuromorphic processor, can be used to compute the short-time Fourier transform (STFT) of signals (Frady et al., 2022; Shrestha et al., 2023). Moreover, the RF neurons successfully converted raw data signals to spike trains (Shaaban et al., 2024), and further input to SNNs with LIF neurons for detection and classification tasks (Hille et al., 2022; Lehmann et al., 2023). RF neurons have also been implemented in the framework of (feedforward) SNNs as harmonic oscillators for image classification tasks (AlKhamissi et al., 2021), as well as in optical flow estimation and audio classification tasks (Frady et al., 2022). Still, the performance of the RF models did not significantly exceed that of deep LIF SNNs (Frady et al., 2022) or LSTM cells (AlKhamissi et al., 2021), and these studies did not compare with state-of-art spiking neuron models, such as ALIF. Considering the strengths of the RF neuron, these results suggest that the full potential of the RF neuron has yet to be explored.

Recent methodological advances in training recurrent SNNs (RSNNs) have demonstrated their potential for effective time series learning, particularly in combination with BPTT (Bellec et al., 2018; Yin et al., 2021). Nonetheless, they do face a "convergence dilemma"—it requires usually up to many hundreds of epochs for RSNNs to converge properly (Yin et al., 2021; Fang et al., 2021; Zhang et al., 2023).

Here we propose a novel spiking neuron model—the balanced RF neuron—which overcomes both, the intrinsic limitations of vanilla RF neurons as well as shortcomings that arise during training of ALIF-RSNNs. As a result, our RF variants achieve not only comparable and higher task performance than the ALIF networks, but also remarkably fast and stable convergence, reaching 95% of the mean final accuracy within the first five epochs with up to seven times less spikes than ALIF networks.

2. Resonate-and-Fire Neurons

The oscillatory behavior of the membrane potential in an RF neuron is formulated with two linear differential equations:

$$\dot{x} = b\,x - \omega\,y + I \tag{1}$$
$$\dot{y} = \omega\,x + b\,y \tag{2}$$

which is expressed as a single complex equation:

$$\dot{u} = (b + i\,\omega)\,u + I \tag{3}$$

with $u = (x + i\,y) \in \mathbb{C}$ and I the injected current (Izhikevich, 2001). $\omega > 0$ is the angular frequency of the neuron, which describes how many radians the neuron progresses per second. $b < 0$ is the dampening factor that exponentially decays the oscillation. The smaller b is, the faster the oscillation dampens to the resting state.

Izhikevich RF Neuron. We applied the Euler method (Atkinson, 1989) to Equation 3 with step size (time scale) δ:

$$u(t) = u(t-\delta) + \delta\left((b + i\,\omega)\,u(t-\delta) + I(t)\right) \tag{4}$$

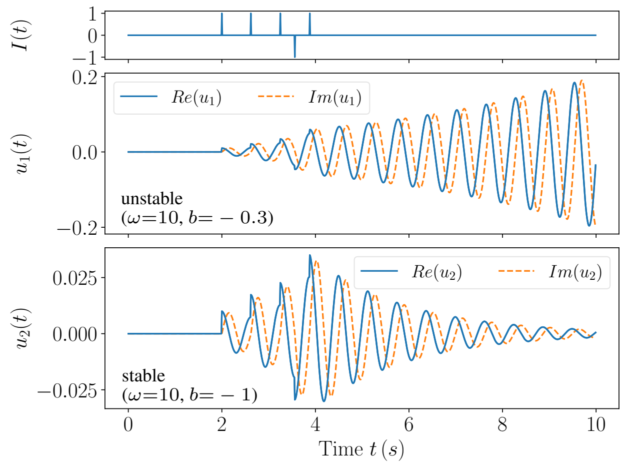

Figure 1. Membrane dynamics of two RF neurons. The four excitatory input spikes are in phase with the neuron's angular frequency $\omega = 10$. The inhibitory spike at half-phase enhances the sensitivity of the neuron to forthcoming excitatory input. Depending on its parameterization, the neuron exhibits divergence behavior (top) with $b = -0.3$ and $\delta = 0.01$ or convergence behavior (bottom) with $b = -1$. $I(t) \in \mathbb{R}$ and $u_1(t), u_2(t) \in \mathbb{C}$ refer to the injected current and the membrane potential of the neuron, respectively.

which is used to simulate the RF neuron in Figure 1. It shows the oscillatory behavior when similar frequency-timed input is injected into an RF neuron with an angular frequency of 10 rad/s and a dampening factor of -1.

Harmonic RF Neuron. In the harmonic RF (HRF) neuron, the membrane potential changes based on the dynamics of dampened harmonic oscillation (AlKhamissi et al., 2021). Instead of using a complex representation, the membrane potential is split into two states as follows:

$$\dot{u} = -2b\,u - \omega^2\,v + I \tag{5}$$
$$\dot{v} = u \tag{6}$$

By applying Euler integration we get the equations for discrete time steps:

$$\begin{aligned} u(t) = u(t-\delta) + \delta(&-2b\,u(t-\delta) \\ &- \omega^2 v(t-\delta) + I(t)) \end{aligned} \tag{7}$$
$$v(t) = v(t-\delta) + \delta u(t-\delta). \tag{8}$$

Here, $b > 0$ is the dampening factor, and $\omega > 0$ is the angular frequency.

3. Balanced RF Models

Initial exploration of the RF neurons in RSNN with a random subset of 32 samples from the sequential MNIST dataset (Deng, 2012) showed excessive spiking, divergent behavior, and the traditional hard and soft reset (Equation 23) hindering immediate resonance behavior.

Divergence is due to the approximation of the dynamic system in discrete steps and is dependent on the combination of ω, b, and the discrete-time scale δ, as shown in Figure 1 (top). The membrane potential diverges and continuously produces spikes unrelated to the input signal, causing noise and artificial signals that disrupt the original frequencies, which may have hindered effective learning of the model.

Balanced Izhikevich RF Neuron. Considering the intrinsic limitations of the basic RF neuron, we introduce the balanced RF *(BRF)* neuron with a variation in the threshold, reset mechanism, and divergence boundary. To reduce continuous spiking of the neuron and induce spiking sparsity, a **refractory period**, referred to as $q(t)$, is implemented in the threshold, increasing it after a neuron fires:

$$\vartheta(t) = \vartheta_c + q(t) \tag{9}$$

$$z(t) = \Theta\left(\mathrm{Re}(u(t)) - \vartheta(t)\right) \tag{10}$$

with ϑ_c the constant threshold and $z(t)$ the output spiking. The real part of the Equation 3 was considered for the threshold mechanism, as it induces an immediate response. The refractory period decays exponentially with time:

$$q(t) = \gamma q(t-\delta) + z(t-\delta) \tag{11}$$

The default refractory period constant is $\gamma = 0.9$.

Another limitation of the basic RF model was the traditional reset mechanism, which reduces the amplitude but disrupts the oscillation. Hence, we propose the **smooth reset** as an alternative that temporarily increases the dampening of the amplitude to decay faster after the neuron fires by means of integrating the refractory period into the dampening term:

$$b(t) = b_c - q(t) \tag{12}$$

with b_c the constant dampening factor.

The influence of implementing the refractory period and the smooth reset can be seen in Figure 2. The single RF neuron was simulated with $\omega = 10$, $b = -1$, and the input signal was frequency-timed. The combination of refractory period and smooth reset significantly reduced the number of spikes, while the output spikes effectively reflected the period of the angular frequency of the neuron. The effect of implementing both are highlighted in Figure 10 in Section A.8.

To alleviate the divergence problem, we propose to restrict the parameter space to a subspace below the **divergence boundary**—an analytically derived relation between δ, b_c, and ω that ensures convergence. For an RF neuron to converge or show sustained oscillation, the magnitude of the membrane potential decreases or stays constant with time given an incoming signal:

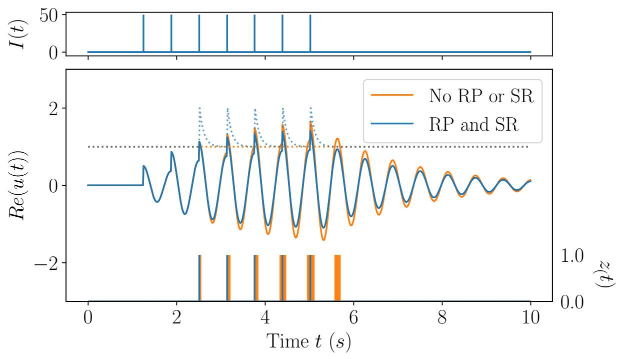

Figure 2. Membrane potential $u(t)$ and spiking response $z(t)$ of an RF neuron without refractory period (RP) or smooth reset (SR) (orange) compared to both (blue) for the given input signal $I(t)$.

$$|u(t)| \leq |u(t-\delta)| \tag{13}$$

After spike onset the magnitude converges—but does not precisely become 0, $|u(t-\delta)| \neq 0$, thus both side is divisible by $|u(t-\delta)|$

$$\frac{|u(t)|}{|u(t-\delta)|} \leq 1 \tag{14}$$

Furthermore, the explicit form of the subthreshold oscillatory behavior of the membrane potential after an incoming spike is derived in Section A.3 and given as:

$$u(t) = \delta\left(1 + \delta(b_c + i\,\omega)\right)^{\frac{t}{\delta}-1} \tag{15}$$

The refractory period and the reset are not considered in the derivation, as post-spiking behavior is irrelevant for modeling subthreshold behavior. The explicit form is further implemented into the inequality above and simplified:

$$|1 + \delta(b_c + i\,\omega)| \leq 1$$
$$\iff \sqrt{(1+\delta b_c)^2 + (\delta\omega)^2} \leq 1$$

Since the radicand is positive, both sides of the inequality can be squared:

$$\implies (1+\delta b_c)^2 + (\delta\omega)^2 - 1 \leq 0 \tag{16}$$

The condition for the neuron to show damped oscillation is found by solving the quadratic inequality for b_c, where b_c is in the following range of the solutions given a constant ω:

$$\frac{-1-\sqrt{1-(\delta\,\omega)^2}}{\delta} < b_c < \frac{-1+\sqrt{1-(\delta\,\omega)^2}}{\delta} \tag{17}$$

Furthermore, the neuron is a sustained oscillator if:

$$b_c = \frac{-1 \pm \sqrt{1-(\delta\,\omega)^2}}{\delta} \tag{18}$$

which leads us to another condition of the neuron, since $b_c \in \mathbb{R}_{<0}$ and $\omega \in \mathbb{R}_{>0}$:

$$\sqrt{1-(\delta\,\omega)^2} > 0 \Rightarrow \omega \leq \frac{1}{\delta} \tag{19}$$

With a default $\delta = 0.01$, the highest frequency that the neuron can resonate at is the frequency corresponding to an angular frequency of $100\,rad/s$. We define this angular frequency as the upper boundary ω_{ub}. The upper bound of b_c that leads to sustained oscillation is implemented in this paper as $p(\omega)$, also considered the divergence boundary:

$$p(\omega) = \frac{-1+\sqrt{1-(\delta\,\omega)^2}}{\delta} \tag{20}$$

combined with a trainable b-offset $b' > 0$ to ensure flexibility and convergence: $b_c = p(\omega) - b'$, which is constant throughout one sequence length. Figure 9 shows exemplary divergence boundaries for various δ values. By combining the derived dampening factor with the smooth reset, we present the final equation of $b(t)$ applied for optimization:

$$b(t) = p(\omega) - b' - q(t) \tag{21}$$

Implementing the refractory period, smooth reset, and the divergence boundary leads to efficient learning and sparse spiking for all four datasets.

Balanced Harmonic RF Neuron. Similarly, we propose the balanced HRF (BHRF) neuron with the refractory period, smooth reset, and a tailored divergence boundary for b depending on ω:

$$p(\omega) = \frac{\omega^2}{200} \tag{22}$$

For $\delta = 0.01$ and $b_c = p(\omega)$, the BHRF neuron shows sustained oscillation.

Frequency Response of the BRF Neuron. The frequency response plots shown in Figure 3 assess how well the RF neuron responds to specific frequencies. A peak on a frequency response plot indicates a high sensitivity of the neuron to the respective input signal frequency (details in Section A.4). The figure shows alignment of the response peak and ω of the RF neuron, demonstrating the resonance property to also be present in the discrete case. The slight offset of the response peak and ω of the RF neuron is an artifact of the numerical integration of the membrane dynamics' differential equation and disappears with decreasing the temporal step size δ.

The RF neuron can also be considered a narrow band pass filter, for which only a specific range of frequencies are filtered depending on its angular frequency ω. Combined with the discrete time scale δ and dampening parameter b', they determine how narrow and sensitive the band pass filter

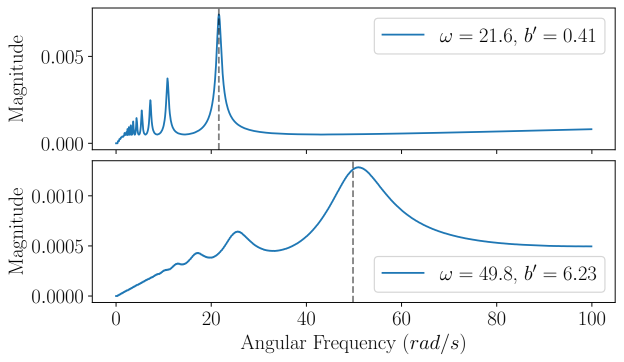

Figure 3. RF neuron frequency response plots for exemplary omega ω and b-offset b' combinations with $\delta = 0.001$.

is. It should be noted that RF neurons are naturally sensitive to the lower order subharmonics of their angular frequency, that is (with decaying sensitivity) $\frac{1}{2}\omega$, $\frac{1}{3}\omega$, $\frac{1}{4}\omega$ and so forth.

We focused on the BRF neuron and the BHRF remained preliminary in its exploration, as the BRF responses were more sensitive to the input frequencies than BHRF responses shown in Figure 8.

4. Network Implementation

We implement the RF, BRF, and BHRF neurons within RSNNs and applied them to simulations with several benchmark datasets[1]. The essential formulations of the BRF and the BHRF neuron are summarized in Algorithm 1 and Algorithm 2, respectively. Note that for the algorithmic formulation we change the notation to time-discrete tensor operations and superscript the time index, whereas the transition from t to $t+1$ is considered as a time delay of δ.

We trained the networks with BPTT with the double-Gaussian function (Equation 39, Yin et al., 2021) as the surrogate gradient (Bellec et al., 2018; Neftci et al., 2019). Further details of the networks are described in Section A.6.

Algorithm 1 BRF Forward Pass

```
$\mathbf{b}^t = p(\boldsymbol{\omega}) - \mathbf{b}' - \mathbf{q}^{t-1}$
$\mathbf{u}^t = \mathbf{u}^{t-1} + \delta((\mathbf{b}^t + i\boldsymbol{\omega})\,\mathbf{u}^{t-1} + \mathbf{x}^t)$
$\boldsymbol{\vartheta}^t = \vartheta_c + \mathbf{q}^{t-1}$
$\mathbf{z}^t = \Theta(\mathrm{Re}(\mathbf{u}^t) - \boldsymbol{\vartheta}^t)$
$\mathbf{q}^t = \gamma\mathbf{q}^{t-1} + \mathbf{z}^t$
```

$\vartheta_c = 1$, $\gamma = 0.9$, and $p(\boldsymbol{\omega}) = \frac{-1+\sqrt{1-(\delta\,\omega)^2}}{\delta}$
(Re, Θ, p are applied component-wise.)

[1]Source code avaliable at https://github.com/AdaptiveAILab/brf-neurons

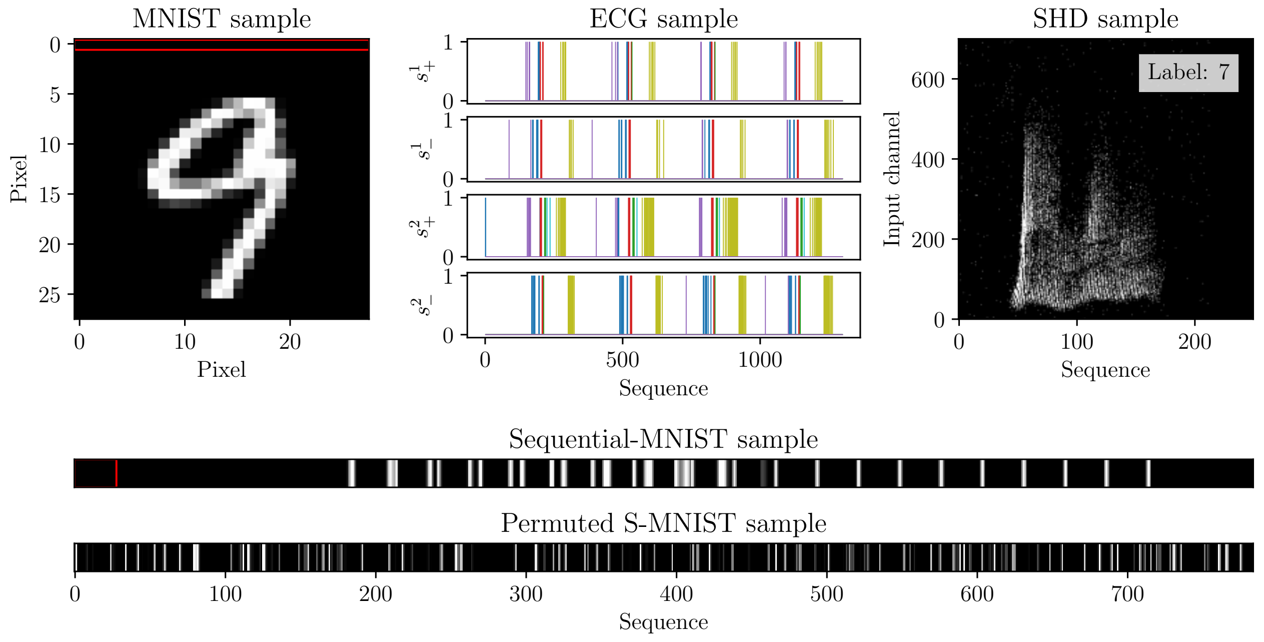

Figure 4. Overview of the datasets. Examplary MNIST image and its corresponding sequential and permuted representations. Common pixel row outlined in red on MNIST and S-MNIST sample. ECG sample after level cross encoding. SHD sample after preprocessing.

5. Experiments

The MNIST dataset consists of grayscale 28×28 pixel hand-written digit images for classification. The sequential-MNIST (**S-MNIST**), which converts the image to a sequence of 1×784, is a prominent benchmark dataset that enable comparison between sequential models with 54,000 images for training, 6,000 for validation, and 10,000 for testing. In the permuted S-MNIST variant (**PS-MNIST**), the pixel positions were first permuted randomly and then input to the network sequentially.

Electrocardiogram (**ECG**) recordings of the heart are represented in voltage over time and consist of cyclic activity with six characteristic waveforms: P, PQ, QR, RS, ST, and TP. The QT database consisted of ECG recordings with per-time step labels classified by experts in the field (Laguna et al., 1997). There were 105 recordings, each recorded for 15 minutes with two electrodes. We left out 24 signals, as they were sudden death recordings without annotation. For the preprocessing shown in Figure 4, refer to Section A.7.

Algorithm 2 BHRF Forward Pass

```
$\mathbf{b}^t = p(\boldsymbol{\omega}) - \mathbf{b}' - \mathbf{q}^{t-1}$
$\mathbf{u}^t = \mathbf{u}^{t-1} + \delta(-2\mathbf{b}^t\mathbf{u}^{t-1} - \boldsymbol{\omega}^2\mathbf{v}^{t-1} + \mathbf{x}^t)$
$\mathbf{v}^t = \mathbf{v}^{t-1} + \delta\mathbf{u}^{t-1}$
$\boldsymbol{\vartheta}^t = \vartheta_c + \mathbf{q}^{t-1}$
$\mathbf{z}^t = \Theta(\mathbf{u}^t - \boldsymbol{\vartheta}^t)$
$\mathbf{q}^t = \gamma\mathbf{q}^{t-1} + \mathbf{z}^t$

$\vartheta_c = 1$, $\gamma = 0.9$, and $p(\boldsymbol{\omega}) = \frac{\boldsymbol{\omega}^2}{200}$
(Re, $\Theta$, $p$ are applied component-wise.)
```

The Spiking Heidelberg dataset (**SHD**) is a benchmark audio-to-spike dataset specifically generated for SNNs (Cramer et al., 2020). It contains 10,420 recordings of spoken digits from 0 to 9 in English and German. The recordings were made with high fidelity and then converted into spike trains for 700 channels through a precise artificial inner ear model (Cramer et al., 2020). The spike trains were further processed with a discrete time scale of 4e-3 s to obtain sequence length of 250 with zero-padding. This preprocessing was conducted to make the dataset comparable to Yin et al. (2021). We used 7,341 sequences for training, 815 for validation, and 2,264 for inference.

6. Results

The results, the network architecture, the number of parameters, and the number of produced spikes between the best-performing BRF-, RF- and BHRF-RSNN and the baseline ALIF-RSNNs (see Section A.2) or other state-of-the-art (SoTa) models, are compared for each classification task in Table 1. The BRF- and BHRF-RSNN hyperparameters for each dataset are shown in Table 3 in Section A.12. The BHRF and BRF models outperformed the RSNN SoTa for four and three tasks respectively. Complementing accuracy, we also find much enhanced sparseness in the BRF and

Table 1. Results between vanilla RF, BRF, BHRF, ALIF (Yin et al., 2021), and other *SoTa* models: DCLS-Delays (DCLS-D) (Hammouamri et al., 2023) and RadLIF (Bittar & Garner, 2022) on various sequential classification tasks. Architecture represents the number of neurons in each layer. The average accuracy over five runs shown for the RF, BRF and BHRF models, excluding PS-MNIST BHRF. SOPs: average spike operations. **RSNN-SoTa* ***SNN-SoTa*

Task	Model	Architecture	No. params. (↓)	Test Acc. (↑)	SOPs (↓)	SOPs/step (↓)	SOP Ratio (↓)
S-MNIST	ALIF*	4,64,$(256)^2$,10	156,126	98.7 %	70,810.8	90.32	1.00×
	RF	1,256,10	**68,874**	98.0±0.4 %	29,034.8	37.03	0.41×
	BRF	1,256,10	**68,874**	99.0±0.1 %	**15,462.6**	**19.72**	0.22×
	BHRF	1,256,10	**68,874**	**99.1**±0.1 %	21,565.7	25.51	0.30×
PS-MNIST	ALIF*	4,64,$(256)^2$,10	156,126	94.3 %	59,772.1	76.24	1.00×
	RF	1,256,10	**68,874**	9.9±0.8 %	66,474.2	84.79	1.11×
	BRF	1,256,10	**68,874**	95.0±0.2 %	27,839.7	35.51	0.46×
	BHRF	1,256,10	**68,874**	**95.2 %**	**24,564.2**	**33.33**	0.41×
ECG	ALIF*	4,36,6	1,776	85.9 %	35,011.2	26.93	1.00×
	RF	4,36,6	**1,734**	85.5±0.7 %	11981.9	9.22	0.34×
	BRF	4,36,6	**1,734**	85.8±0.7 %	6,307.7	4.85	0.18×
	BHRF	4,36,6	**1,734**	**87.0**±0.4 %	**6,233.8**	**4.80**	0.18×
SHD	DCLS-D**	700,$(256)^2$,20	≈200,000	**95.1**±0.2 %	-	-	-
	RadLIF*	700,$(1024)^3$,20	3,893,288	94.62%	-	-	-
	ALIF	700,$(128)^2$,20	142,120	90.4 %	24,690.0	98.76	1.00×
	RF	700,128,20	**108,820**	89.2±0.6 %	4750.2	19.00	0.19×
	BRF	700,128,20	**108,820**	91.7±0.8 %	**3,502.6**	**14.01**	0.14×
	BHRF	700,128,20	**108,820**	92.7±0.7 %	4,139.5	16.56	0.17×

BHRF networks while using substantially fewer parameters compared to the ALIF networks. Note that RF model shown in Table 1 did not include a reset. Adding either a hard or soft reset to the vanilla RF neuron dramatically affects its task performance as shown in Figure 11 in Section A.9.

We also compare the theoretical energy efficiency of the RF and ALIF models by computing their respective total spiking operations (SOPs) (see Equation 41) and SOPs per sequence step. As both measures were considerably smaller for the BRF and BHRF models, this implies they require less computation to achieve better or comparable performance. This may be due to the resonating spiking behavior of the neuron, needing fewer spikes to represent its frequency, as well as the refractory period and smooth reset that hinders continuous spiking. A particularly large difference was seen in the SHD task, where the BRF model spiked only 14% of the ALIF model.

Compared to the standard RF model without reset, our balanced variants achieved significantly sparser activity while obtaining higher performance. Especially for the PS-MNIST, the standard RF model stayed at chance, while our balanced RF variations outperformed RSNN-SoTA. This is due to the damping factor and angular frequency combination leading to a highly diverging behavior. It especially affects the PS-MNIST dataset, as it requires a wider range of resonant frequencies to counter the increased randomness in the signal. For example, neurons with $\omega = 70$ rad/s and $b_c = -1$ leads to higher magnitude towards the end of the sequence. Having many of such neurons causes the whole system to diverge and become unstable. On a different note, the performance of the BRF model depends on the initialization of the parameters and it generally under-performs when the angular frequencies are too far from the frequencies underlying the dataset.

Overall, we find that RSNNs comprised of either BRF and BHRF neurons exceeded performance for all classification tasks compared to the baseline ALIF-RSNN. Compared to standard RF neurons, the smooth reset, refractory period, and divergence boundary significantly improved the stability and efficiency of the RF parameters, possibly exploiting the model's resonant properties. We study this in detail below.

Convergence. The RF, BRF, and ALIF model learning curves are presented in Figure 5. We reduced the ALIF model to the same size and number of trainable parameters as the BRF model. For the S-MNIST and PS-MNIST tasks, the ALIF networks only learn effectively when the truncated BPTT (TBPTT) with truncation step 50 is applied. Nonetheless, the fully back-propagated BRF networks converge significantly faster than the TBPTT ALIF networks, also evident from the quantitative convergence results (Table 2).

A similar pattern of fast and stable convergence is observed for the ECG dataset and SHD. The convergence between RF and BRF for SHD and ECG are similar because the vanilla RF neurons resonate with lower frequencies, affected less

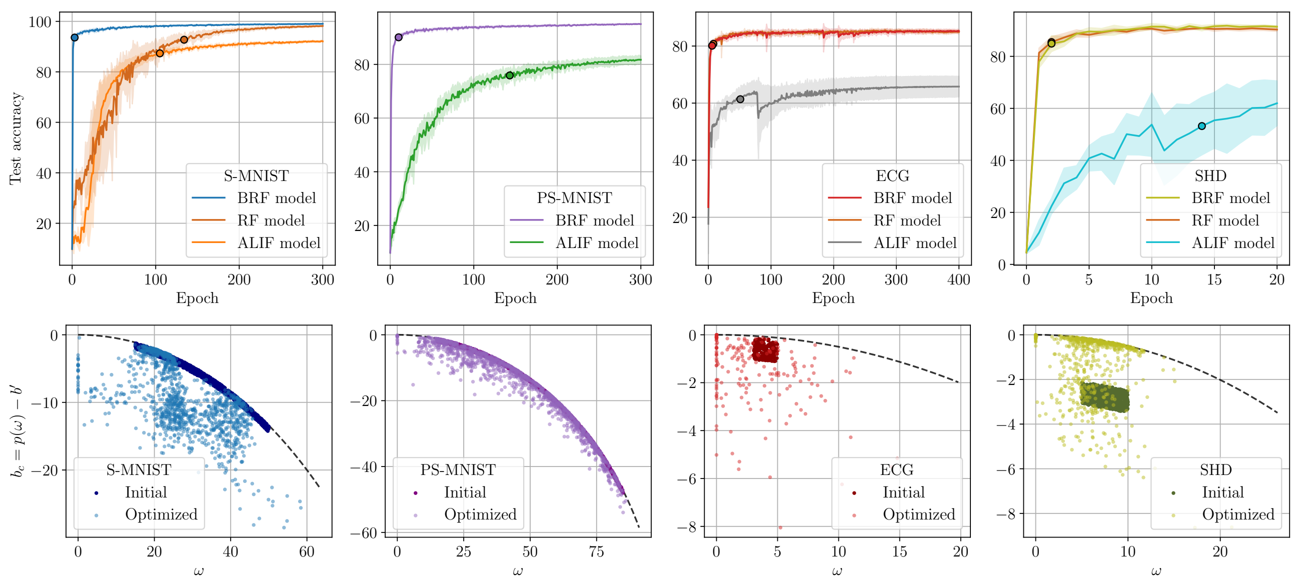

Figure 5. Top row: S-MNIST, PS-MNIST, ECG, and SHD learning curve between BRF, RF and ALIF model. Each curve averaged per epoch (solid line) with standard deviation (shaded area) over five runs. The dot on the accuracy curves depict the point at which 95 % of the final accuracy was reached. Bottom row: S-MNIST, PS-MNIST, ECG, and SHD initial and optimized BRF parameter combinations: angular frequency ω and b-offset b' for all runs. Dashed line is the divergence boundary. RF models are simulated without reset. For convergence with traditional reset see Section A.9.

Table 2. Quantitative convergence results. The average number of epochs after which 95 %, 98 %, and 100 % of the final test accuracy on the learning curve was reached, respectively.

Task	Model	95 % (↓)	98 % (↓)	100 % (↓)
S-MNIST	ALIF	105	162	276
	RF	134	172	263
	BRF	**3**	29	246
	BHRF	5	**12**	**119**
PS-MNIST	ALIF	143	200	265
	BRF	10	39	282
	BHRF	**6**	**19**	**123**
ECG	ALIF	51	157	282
	RF	8	**29**	112
	BRF	**6**	32	**75**
	BHRF	15	38	109
SHD	ALIF	14	15	16
	RF	**2**	4	**5**
	BRF	**2**	**3**	7
	BHRF	3	5	8

by diverging behavior. Note that the RF model shown in Figure 5 did not include a reset. Adding either soft or hard reset leads to slower and more unstable learning, as shown in Figure 11 in Section A.9. Figure 5 highlights the variation in training of RF networks depending on the dataset, which is alleviated by the BRF network.

While numerically large, the learning rates of the (B)RF-RSNNs led to stable and fast convergence for most datasets. This is in contrast to standard SNNs where large learning rates lead to poor performance. Here, the double-Gaussian surrogate gradient function (Yin et al., 2021) we used may have contributed by ensuring gradient flow even for neurons far from firing. Combined with the high learning rate, this may have enabled non-spiking neurons to effectively shift their angular frequencies.

The ECG task was learned most effectively for the BRF and BHRF networks with a small batch size and high learning rate. A smaller batch size can better capture detailed variations within the dataset, notably the timing of the input spikes, which may have been important for per-step wave classification.

Ongoing investigations (Higuchi et al., 2024) suggest that the shape of the error landscape may be a major contributor to the fast and stable convergence of the BRF model (see Figure 13 in Section A.11).

Parameter analysis. We also compared the trainable BRF parameters ω and b' between the initial and trained networks to obtain an intuition of the functioning of the neuron, as shown in Figure 5. We see that optimization substantially shifted the parameters, demonstrating that the gradients in the B(H)RF networks were effectively propagated.

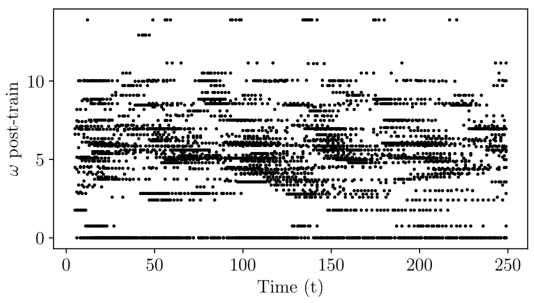

Figure 6. Raster plot of network activity for an SHD sample with label 10 after training. The BRF neurons are sorted by their angular frequency. Black dots denote output spikes.

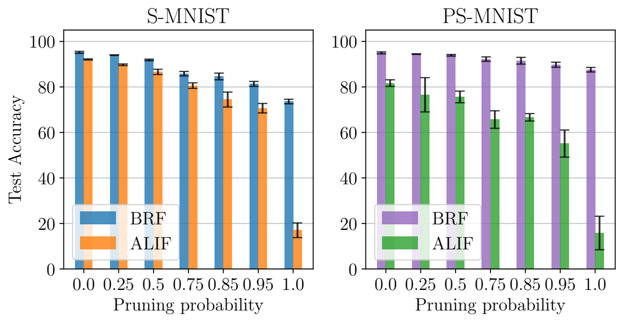

Figure 7. Test accuracy over pruning probability for (a) S-MNIST and (b) PS-MNIST. 1.0 pruning indicate no recurrent weights.

For the S-MNIST task, plotting ω-b_c (Figure 5) shows clusters around ω values of 18 to 28 rad/s and around ω of 40 rad/s with a wide range of b_c. The dampening factor close to zero indicates near-sustained oscillation behavior, preserving resonance amplitude for an extended duration. The clustered variety in b_c values suggests that short- and long-term memory are essential for effective learning of the S-MNIST task in B(H)RF networks. Additionally, the distribution of ω for S-MNIST task showed a bimodal distribution with the highest peak at 23.3 to 25.3 rad/s and the second peak at 40.8 to 42.8 rad/s.

The MNIST digits all contain some form of diagonal line with a width of 3 to 6 pixels, most prominently observed in digit 1. When converted into a sequence row-by-row, the lines are distributed among the sequence with a periodic signal of about 25 pixels. Considering the pixels as discrete time steps of $\delta = 0.01$, the theoretical period T frequently observed in the signal is T = 0.25 s. Thus, the most frequent theoretical angular frequency ω' underlying the signal is around $\omega' = 2\pi/0.25\,s = 25.13\,rad/s$ which is close to the most frequently learned angular frequencies of the BRF neuron. This simple calculation thus suggests that the BRF neurons were able to learn meaningful frequencies that underlie the S-MNIST dataset. For the PS-MNIST, we see more spread of ω and b_c parameters, as the specific characteristic frequency structure from S-MNIST is essentially obscured by the permutation.

Inspecting spike train ECG samples (Figure 4), we observe distinguishable periodic spikes with 300 time step intervals, corresponding to a theoretical angular frequency of $\omega' = 2.09\,rad/s$, and indeed a cluster at about 2 rad/s is present in the ω-b_c plot. The theoretical and optimized angular frequencies align, demonstrating learning of the dynamical signal by the network.

Weight Sparsity. We explored the effect of pruning recurrent weights in an ablation study for the S-MNIST and PS-MNIST tasks between comparable BRF and ALIF networks, as detailed in Table 3 and results are shown in Figure 7. While the BRF neuron provides consistent performance throughout pruning, the ALIF model performs poorly when lacking recurrent weights. This demonstrates that explicit recurrencies are not necessary for the BRF networks with resonating properties to learn reasonably well, while these connections are crucial for the ALIF networks. Moreover, the BRF networks' accuracy has a smaller standard deviation throughout compared to the ALIF networks, also showing that the BRF network performance is not affected by the specific pruned weights as much as the ALIF networks.

7. Discussion

The results from our BRF-RSNN implementation lead to considerable insight into the workings of the RF neurons. Importantly, the learning curve and parameter analyses of the datasets show that the BRF networks learn meaningful parameters by favorably tuning the resonating behavior, modeling the complex dynamics not yet observed in large-scale simulations. We observe favorable oscillatory behavior of the membrane potential and increasing frequency of periodic spikes with higher angular frequency in the raster plot for all datasets. For SHD, output spikes are present towards the end of the sequence, even with the zero padding of the input signal. As most signals are still periodic, it suggests controlled oscillation and spiking that maintains relevant frequencies until the end of the sequence.

Note that we conducted the experiments without biases for all RF models to investigate how many spikes are needed to keep the network dynamics alive and to solve the tasks. As a result, a few of the BRF neurons in the network learned to fire consistently throughout the sequence as seen in Figure 6 and thus effectively "imitated" a bias injecting constant current into the network. In contrast, ALIF networks need explicit biases to perform well. The BRF networks are thus more flexible in modeling behavior that cannot be learned by the integrator neurons alone.

The neurons learn their own favored frequencies and, consequently, focus on individual time scales. This can be loosely seen as a homeostatic regulatory process maximizing local information gain in the realms of distributed, collective task solving within the recurrent networks.

Further exploration of the post-trained models against various noise types and numerical quantization (Stromatias et al., 2015; Park et al., 2021), shows consistently high robustness of the BRF networks compared to the ALIF networks, detailed in Section A.10, Figure 12. The BRF networks are slightly more robust than the RF networks for some noise variants, despite the constant low SOP of the BRF networks. Hardware implementations are prone to such noises (Stromatias et al., 2015), which suggests the BRF neurons to be suitable for hardware applications. We further aim to realize inference with these neurons on Loihi 2, which provides support for RF neuron variations (Davies et al., 2018; Shrestha et al., 2023).

Concerning the network dimension, we found that adding additional resonator neurons into the single hidden layer beyond a certain capacity did not significantly improve the performance further.

One drawback of our BRF- and BHRF-RSNNs is the difficulty in finding an optimal initial parameterization of the angular frequencies and the dampening factor offset, since both determine the models' performance. The analyses of the parameters suggest the possibility of applying the Fourier transform on the datasets a-priori to approximate the range of angular frequencies the BRF neurons require to optimally learn the data.

It should be noted that the computational complexity of BRF-RSNNs is similar to single-layered ALIF models. However, due to the faster convergence the training can be stopped much earlier, which effectively results in a drastically reduced training time. Implementation-wise, another speed-up can be achieved by using forward gradient injection (Otte, 2024) for modeling the surrogate gradient functions in combination with automatic model optimization routines such as TorchScript in PyTorch (some results are presented in Section A.13).

8. Conclusion

We introduced BRF- and BHRF-RSNNs as a principled resonating spiking neuron model that solves the training difficulties encountered for networks comprised of standard RF neurons while demonstrating the effectiveness of internal resonating state as a form of long-term memory. Our BRF- and BHRF-RSNNs with smaller network architectures and sparser spiking outperformed deep baseline models. The learning curves showed that the BRF networks converged much faster than the ALIF networks. Furthermore, the stable convergence hints at good reproducibility of the BRF model's performance. The BRF parameter analyses indicate the network to learn frequencies underlying the input signals and showed resilience against sparser recurrent connectivity compared to the ALIF network.

Possible future research may be to scale up the simulations to much larger and more complex problems, such as the CIFAR (Krizhevsky & Hinton, 2009) or the Google Speech Command dataset (Warden, 2018). It is of particular interest to apply the BRF model to raw audio processing tasks, because of its ability to extract periodic pattern within the time domain.

Another line of research could be to combine BRF neurons with online learning approaches, such as e-prop (Bellec et al., 2020), which would broaden its range of applications due to memory efficiency and faster processing of longer sequences. Additionally, trainable refractory period decays for flexible adaptive thresholding could be introduced, as well as trainable simulated variable time constants to foster multi-temporal resolution processing.

This study marks an initial exploration to investigate the working of stable resonating neurons in the context of recurrent spiking neural network. Our models show comparable results and/or exceeds the state-of-the-art for RSNNs. We see many possible approaches to extending our work, as we kept the implementation to its simplest form. The presented work thus provides a foundation for further research of possible BRF-RSNN variants.

Acknowledgements

A part of the experiments were conducted using the ML Cloud infrastructure of the Machine Learning Cluster of Excellence (EXC number 2064/1 – Project number 390727645) at the University of Tübingen.

Sebastian Otte was supported by a Feodor Lynen fellowship of the Alexander von Humboldt Foundation.

Impact Statement

The aim of this work is to advance the field of machine learning, with a specific focus on spiking neural networks (SNNs) to augment the effectiveness and energy efficiency of AI systems, making them more sustainable. It is worth noting that, like most developments in machine learning research, our work may have various societal consequences, but none of which we think needs special consideration here.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., Arshad, H., Kazaure, A. A., Gana, U., and Kiru, M. U. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE access*, 7:158820–158846, 2019.

AlKhamissi, B., ElNokrashy, M., and Bernal-Casas, D. Deep spiking neural networks with resonate-and-fire neurons. *arXiv preprint arXiv:2109.08234*, 2021.

Alonso, A. and Llinás, R. R. Subthreshold na+-dependent theta-like rhythmicity in stellate cells of entorhinal cortex layer ii. *Nature*, 342(6246):175–177, 1989.

Atkinson, K. E. An introduction to numerical analysis, john wiley and sons. *New York*, 19781, 1989.

Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., and Maass, W. Long short-term memory and learning-to-learn in networks of spiking neurons. *Advances in neural information processing systems*, 31, 2018.

Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., and Maass, W. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature communications*, 11(1):3625, 2020.

Bittar, A. and Garner, P. N. A surrogate gradient spiking baseline for speech command recognition. *Frontiers in Neuroscience*, 16:865897, 2022.

Cramer, B., Stradmann, Y., Schemmel, J., and Zenke, F. 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.

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., Dimou, G., Joshi, P., Imam, N., Jain, S., et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Doeller, C. F., Barry, C., and Burgess, N. Evidence for grid cells in a human memory network. *Nature*, 463(7281): 657–661, 2010.

Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., and Tian, Y. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2641–2651, 2021. doi: 10.1109/ICCV48922.2021.00266.

Frady, E. P., Sanborn, S., Shrestha, S. B., Rubin, D. B. D., Orchard, G., Sommer, F. T., and Davies, M. Efficient neuromorphic signal processing with resonator neurons. *Journal of Signal Processing Systems*, 94(10):917–927, 2022.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Hammouamri, I., Khalfaoui-Hassani, I., and Masquelier, T. Learning delays in spiking neural networks using dilated convolutions with learnable spacings. *arXiv preprint arXiv:2306.17670*, 2023.

Higuchi, S., Bohte, S. M., and Otte, S. Understanding the convergence in balanced resonate-and-fire neurons, 2024. arXiv preprint arXiv:2406.00389.

Hille, J., Auge, D., Grassmann, C., and Knoll, A. Resonate-and-fire neurons for radar interference detection. In *Proceedings of the International Conference on Neuromorphic Systems 2022*, pp. 1–4, 2022.

Hinton, G., Srivastava, N., and Swersky, K. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

Hodgkin, A. L. and Huxley, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117 (4):500, 1952.

Izhikevich, E. M. Resonate-and-fire neurons. *Neural networks*, 14(6-7):883–894, 2001.

Izhikevich, E. M. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images, 2009.

Laguna, P., Mark, R., Goldberger, A., and Moody, G. A database for evaluation of algorithms for measurement of qt and other waveform intervals in the ecg. *Computers in Cardiology*, 24(1):673–676, 1997.

Lehmann, H. M., Hille, J., Grassmann, C., and Issakov, V. Direct signal encoding with analog resonate-and-fire neurons. *IEEE Access*, 2023.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. On the variance of the adaptive learning rate and beyond. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*, April 2020.

Llinas, R. R., Grace, A. A., and Yarom, Y. In vitro neurons in mammalian cortical layer 4 exhibit intrinsic oscillatory activity in the 10-to 50-hz frequency range. *Proceedings of the National Academy of Sciences*, 88(3):897–901, 1991.

Lu, L., Shin, Y., Su, Y., and Karniadakis, G. E. Dying relu and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*, 2019.

Neftci, E. O., Mostafa, H., and Zenke, F. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

Otte, S. Flexible and efficient surrogate gradient modeling with forward gradient injection, 2024. arXiv preprint arXiv:2406.00177.

Park, S., Lee, D., and Yoon, S. Noise-robust deep spiking neural networks with temporal information. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 373–378. IEEE, 2021.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Paugam-Moisy, H. and Bohté, S. M. Computing with spiking neuron networks. *Handbook of natural computing*, 1: 1–47, 2012.

Pedroarena, C. and Llinás, R. Dendritic calcium conductances generate high-frequency oscillation in thalamocortical neurons. *Proceedings of the National Academy of Sciences*, 94(2):724–728, 1997.

Pfeiffer, M. and Pfeil, T. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in neuroscience*, 12:774, 2018.

Shaaban, A., Chaabouni, Z., Strobel, M., Furtner, W., Weigel, R., and Lurz, F. Resonate-and-fire spiking neurons for hand gesture label refinement. In *2024 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNeT)*, pp. 53–56. IEEE, 2024.

Shrestha, S. B., Timcheck, J., Frady, P., Campos-Macias, L., and Davies, M. Efficient video and audio processing with loihi 2. *arXiv preprint arXiv:2310.03251*, 2023.

Stromatias, E., Neil, D., Pfeiffer, M., Galluppi, F., Furber, S. B., and Liu, S.-C. Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms. *Frontiers in neuroscience*, 9:141542, 2015.

Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Wu, Y.-c. and Feng, J.-w. Development and application of artificial neural network. *Wireless Personal Communications*, 102:1645–1656, 2018.

Yin, B., Corradi, F., and Bohté, S. M. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10): 905–913, 2021.

Zhang, S., Yang, Q., Ma, C., Wu, J., Li, H., and Tan, K. C. Long short-term memory with two-compartment spiking neuron. *arXiv preprint arXiv:2307.07231*, 2023.

A. Appendix

A.1. Traditional soft and hard reset

The traditional soft and hard reset mechanisms were formulated respectively:

$$u(t) = u'(t) - (1 + i)\, z(t)\, \vartheta \tag{23}$$

$$u(t) = [1 - (1 + i)\, z(t)]\, u'(t) \tag{24}$$

for which $u'(t)$ represents the membrane potential before reset.

A.2. Baseline ALIF neuron

The formulation of the baseline ALIF neuron (Yin et al., 2021) is as follows with $\vartheta = 0.01$ and $\beta = 1.8$:

$$\vartheta^t = \vartheta + \beta\, a(t) \tag{25}$$

$$a(t) = \rho\, a(t - \delta) + (1 - \rho)\, z(t - \delta) \tag{26}$$

$$u'(t) = \alpha\, u(t - \delta) + (1 - \alpha)\, I(t) \tag{27}$$

$$z(t) = \Theta\left(u'(t) - \vartheta^t\right) \tag{28}$$

$$u(t) = u'(t) - z(t)\, \vartheta^t \tag{29}$$

where $\rho = e^{-\frac{\delta}{\tau_a}} \in (0, 1)$ is the adaptive threshold decay constant, τ_a the time constant and $\alpha = e^{-\frac{\delta}{\tau_m}} \in (0, 1)$ the membrane potential decay constant. $a(t)$ is the accumulative activity of the spiking behavior of the neuron. The membrane potential $u(t)$ is soft reset with the adaptive threshold ϑ^t when the neuron fires ($z(t) = 1$).

A.3. Explicit form of the Izhikevich RF neuron

We can reform the membrane equation to $u(t) = (1 + \delta\, (b + i\, \omega))\, u(t - \delta) + \delta\, I(t)$. Consider discrete time step δ with t = 0, δ, 2δ, 3δ, $, T\delta$. Assume a spike injected only at $t = \delta$ with $I(\delta) = 1$ and initial membrane potential $u(0) = 0$, then:

$$u(\delta) = \delta \tag{30}$$

$$u(2\delta) = \delta\, (1 + \delta\, (b + i\, \omega)) \tag{31}$$

$$u(3\delta) = (1 + \delta\, (b + i\, \omega))(\delta\, (1 + \delta\, (b + i\, \omega))) \tag{32}$$

$$= \delta\, (1 + \delta\, (b + i\, \omega))^2 \tag{33}$$

$$u(4\delta) = (1 + \delta\, (b + i\, \omega))(\delta\, (1 + \delta\, (b + i\, \omega))^2) \tag{34}$$

$$= \delta\, (1 + \delta\, (b + i\, \omega))^3 \tag{35}$$

$$\cdots \tag{36}$$

$$u(t) = \delta\, (1 + \delta\, (b + i\, \omega))^{\frac{t}{\delta} - 1} \tag{37}$$

A.4. Frequency response plot generation

The subthreshold responses were explored with randomly chosen angular frequency $\omega \in [0, 100)$ and $b' \in (0, 10)$ for the RF neurons. Spiking input signals with frequencies relative to the angular frequencies of $\{0.1, 0.2 \cdots, 100\}$ were input to the neurons over 20 seconds with a discrete-time scale of $\delta = 0.001$. The positive spikes were input in-phase of the period corresponding to the angular frequency:

$$T = \frac{2\pi}{\text{angular frequency}} \tag{38}$$

The mean absolute magnitude of the membrane potential over the whole sequence was calculated and plotted to get the response of the neuron per tested frequency signal. Similarly conducted for the HRF neurons.

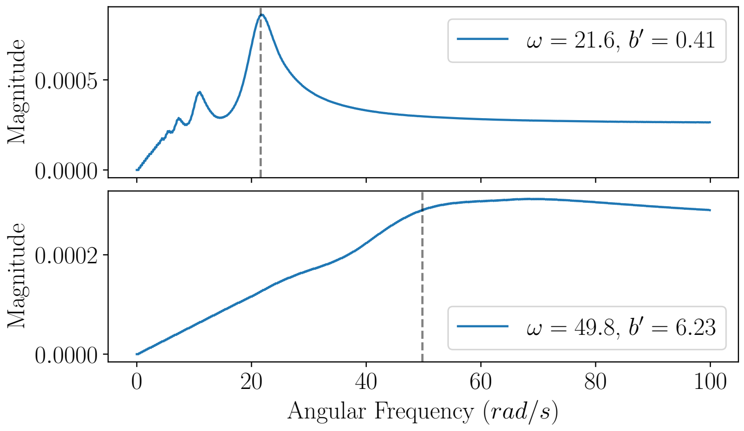

Figure 8. HRF neuron frequency response plots for exemplary omega ω and b-offset b' combinations with $\delta = 0.001$.

A.5. Divergence Boundary

The discrete-time scale δ has a large impact on the divergence behavior. Figure below shows the curve of divergence boundary for time scales above and below the default $\delta = 0.01$. The membrane potential converges if the combination of ω and b_c is below the line. When considering the ω range of 0 to 100 rad/s, the slope of the sustained oscillation curve becomes flatter with smaller δ. The more precise the oscillation is modeled, the less likely the system will diverge. Indeed, the discrete approximation of the continuous differential equation becomes more precise by computing smaller time scales.

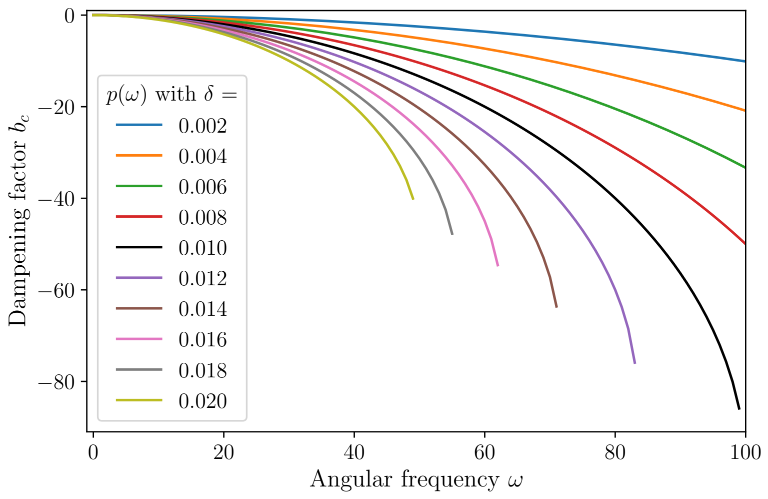

Figure 9. Divergence boundary depending on $\delta \in \{0.002, \cdots, 0.02\}$

A.6. Experimental Setup

The fully recurrent RF-RSNN with m input, h hidden BRF or BHRF, and C leaky integrator (LI) output neurons was computed using PyTorch (Paszke et al., 2017). The backpropagation through time (BPTT) algorithm with minibatch size B was used with the Adam (Kingma & Ba, 2014), RAdam (Liu et al., 2020), or the RMSprop (Hinton et al., 2012) depending on the network and task. The input from the current sequence step $\mathbf{x}^t \in \mathbb{M}_{B\times m}$ and the output spiking of the recurrent hidden neurons from the previous time step $\mathbf{z}^{t-1} \in \mathbb{M}_{B\times h}$ were combined by implementing a fully connected linear layer. This combined signal $\mathbf{x}^t \in \mathbb{M}^{B\times h}$ corresponded to the familiar injected current $I(t)$ but was written in matrix form to represent all hidden neurons in the minibatch. Although not explicit in the Algorithms, $\mathbf{w}_{\mathbf{in,rec}} \in \mathbb{M}_{h\times(m+h)}$ and $\mathbf{w}_{\mathbf{out}} \in \mathbb{M}_{C\times h}$ represent the strength of the connections between the input-to-hidden, hidden-to-hidden, and hidden-to-output neurons, which were optimized. No biases were trained or used in the network. The B(H)RF neurons with $\boldsymbol{\omega}$ and $\mathbf{b}' \in \mathbb{R}^h$ were updated over the time sequence of B data samples with $\mathbf{u}^t, (\mathbf{v}^t), \mathbf{b}^t, \mathbf{q}^t, \boldsymbol{\vartheta}^t \in \mathbb{M}_{B\times h}$. The LI membrane potential time decay constant $\boldsymbol{\tau_{m,out}} \in \mathbb{R}^C_{>0}$ was also learned. The negative log-likelihood (NLL) or the cross-entropy (CE) loss was implemented as our criterion depending on the simulations.

For the ALIF implementation, the adaptive threshold time decay constant $\boldsymbol{\tau_a} \in \mathbb{R}^h_{>0}$ is additionally learned with the membrane potential time decay constant $\boldsymbol{\tau_m} \in \mathbb{R}^h_{>0}$.

In the case of average sequence loss with NLL, the logarithmic softmax function was computed for all output neurons per time step $\hat{\mathbf{y}}^t = log(\text{softmax}(\mathbf{u}^t_{out}))$, which was further propagated into the loss function: $\mathcal{L}^t = \text{criterion}(\hat{\mathbf{y}}^t, \mathbf{y}^t) = \frac{1}{B}\sum_{c=1}^{C} -y^t_c\, \hat{y}^t_c$ where $\mathbf{y}^t \in \mathbb{M}_{B\times C}$ is the target label at time step t represented as one-hot vectors. The mean loss was computed as $\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}^t$ with sequence length T. In the case of label-last loss, only the loss from the last time step was propagated backwards.

The backward pass of the BPTT algorithm was internally computed with the automatic differentiation engine in PyTorch (Paszke et al., 2017). For the non-differentiable Heaviside function in the B(H)RF and ALIF neurons, the multi-Gaussian function (Yin et al., 2021) was manually implemented as the surrogate gradient (Neftci et al., 2019):

$$\frac{\partial \Theta}{\partial u} =_{def} (1+h)g(u,\vartheta,\sigma) - 2hg(u,\vartheta,s\sigma) \tag{39}$$

with

$$g(x,\mu,\sigma) = e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{40}$$

with h=0.15, s=6, σ=0.5 similar to Yin et al. (2021). The negative gradient values were motivated by the leaky rectified and exponential linear unit that prevented the "dying-ReLU" problem (Lu et al., 2019). With these values, the multi-Gaussian surrogate gradient function promoted the neurons to spike despite an initial low membrane potential, which contributed to effective learning.

The best model was the model saved with the highest average test loss over the first five runs by early stopping on the validation loss. The loss and accuracy of the train, validation, and test set were logged into Tensorboard (Abadi et al., 2016) for analysis of the learning pattern. After obtaining the best model, the spiking operations (SOPs) were computed based on the five saved models by taking the total number of output spikes over the whole dataset z_{sum} with respect to the number of data samples N:

$$\text{SOPs} = \frac{z_{sum}}{N} \tag{41}$$

The results were compared with the baseline ALIF-RSNN.

For simulating the models and performing the experiments, we used multiple systems with different deep learning accelerators including NVIDIA GeForce RTX 2060, NVIDIA GeForce RTX 2080 Ti, NVIDIA GeForce RTX 3090, and NVIDIA A100, with PyTorch 2.0.1 on Python 3.10.4 and CUDA 11.7.

A.7. ECG-QT database preprocessing

The preprocessed QT data were extracted from Yin et al. (2021) to gain comparable results. The original sequences were segmented into smaller intervals, each containing 1300 ms of the recordings. Then, the two ECG signals were normalized and encoded into two separate spike trains via level-cross encoding. The positive gradients above and negative gradients below the threshold led to spikes with the threshold $L = 0.3$:

$$s_+ = \begin{cases} 1 & \text{if } x_t - x_{t-1} \geq L \\ 0 & \text{otherwise} \end{cases} \tag{42}$$

$$s_- = \begin{cases} 1 & \text{if } x_t - x_{t-1} \leq -L \\ 0 & \text{otherwise} \end{cases} \tag{43}$$

The target labels were segmented correspondingly and predicted for each time step. Figure 4 color: blue, red, green, cyan, olive and purple corresponds to the label: P, PQ, QR, RS, ST. Five hundred fifty-seven segments were used for training, 61 for validation, and 141 for testing.

A.8. Effect of refractory period and smooth reset

Refractory period (RP) and smooth reset (SmR) both result in reduced spiking operations, with their effects varying depending on the dataset. The SOP of the BRF neuron consistently yields the lowest SOP. It suggests the flexibility of the BRF neuron to learn various types of data efficiently, compared to the RF neuron. ECG optimized with RP spikes less than with SmR, whereas SHD spikes more with SmR; however even more efficient in both cases is the BRF with the combination of RP and SmR. PS-MNIST diverges with: no reset, RP, SmR, and combination of RP and SmR. Thus, it shows significantly high SOP for these variants. It also highlights the stability brought out by the divergence boundary (DB). S-MNIST trained with RP results in unstable performance and SOP.

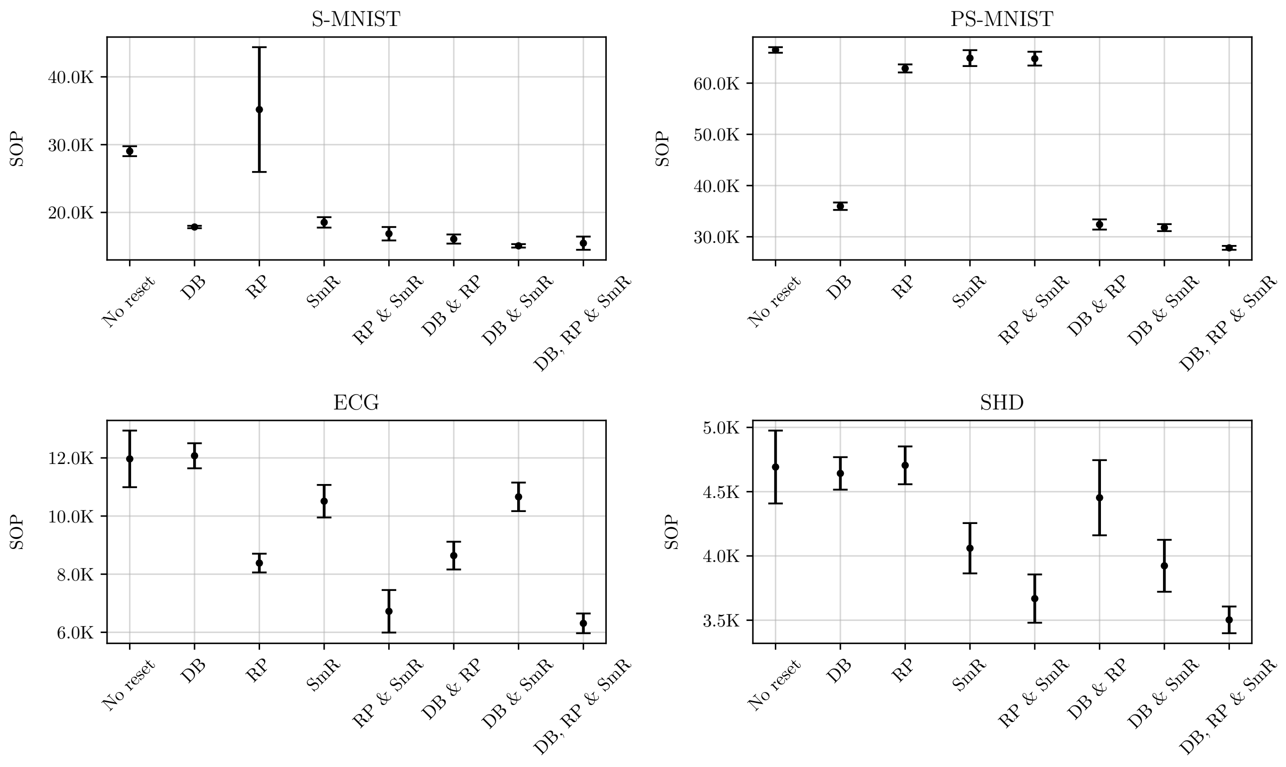

Figure 10. SOP result in RF networks with divergence boundary (DB), refractory period (RP), and smooth reset (SmR). Mean SOP over 5 runs plotted with standard deviation. DB, RP & SmR denote the complete BRF neuron.

A.9. Reset variation and performance

To explore the impact of the standard reset mechanisms (mentioned in Section A.1) on the resonator network, RF models with smooth, soft, and hard reset are optimized for ECG and SHD, as shown in Figure 11. Note that the traditional reset mechanisms applied to S-MNIST and PS-MNIST failed to converge. The figures show performance drop and slower convergence with the soft and hard reset, presumably due to the altered phase of the oscillation and the difficulty in continuous resonant spiking, as spikes may occur with twice the period of the original signal. Further facilitated by the result without reset, in which the convergences are on-par with the BRF neurons where the phase itself is preserved. As mentioned in the main text, the system did not diverge for the RF without reset for ECG and SHD in particular, due to the small angular frequencies leading to smaller error accumulation over time.

A.10. Noise robustness

Noise robustness of post-trained models are further investigated with the methods applied by Stromatias et al. (2015) and Park et al. (2021). Hardware implementations of SNNs pose limitations to the performance due to restricted precision or noise in the signal. Here, four different type of restrictions and noise are explored. Quantization reduces the bit precision

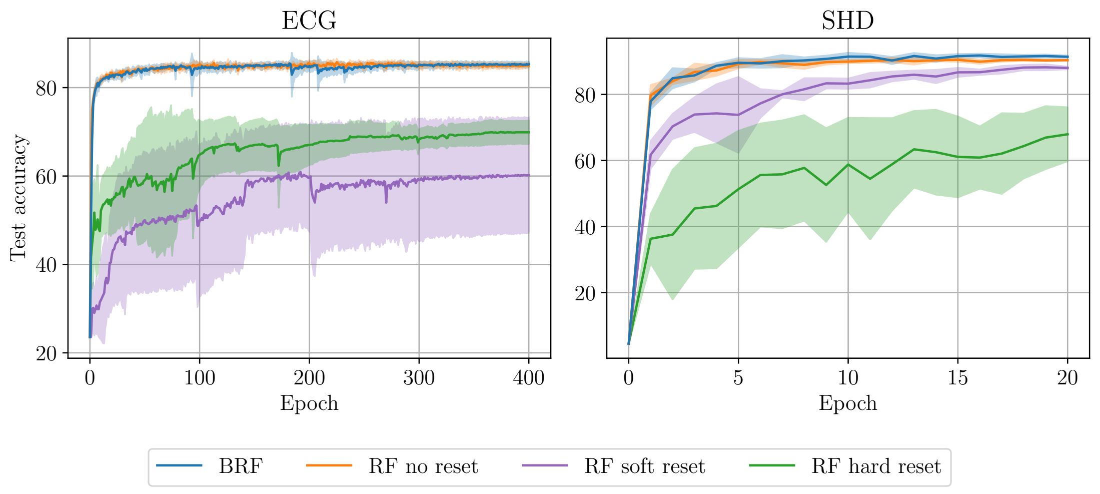

Figure 11. Convergence comparison of ECG and SHD over RF reset mechanisms.

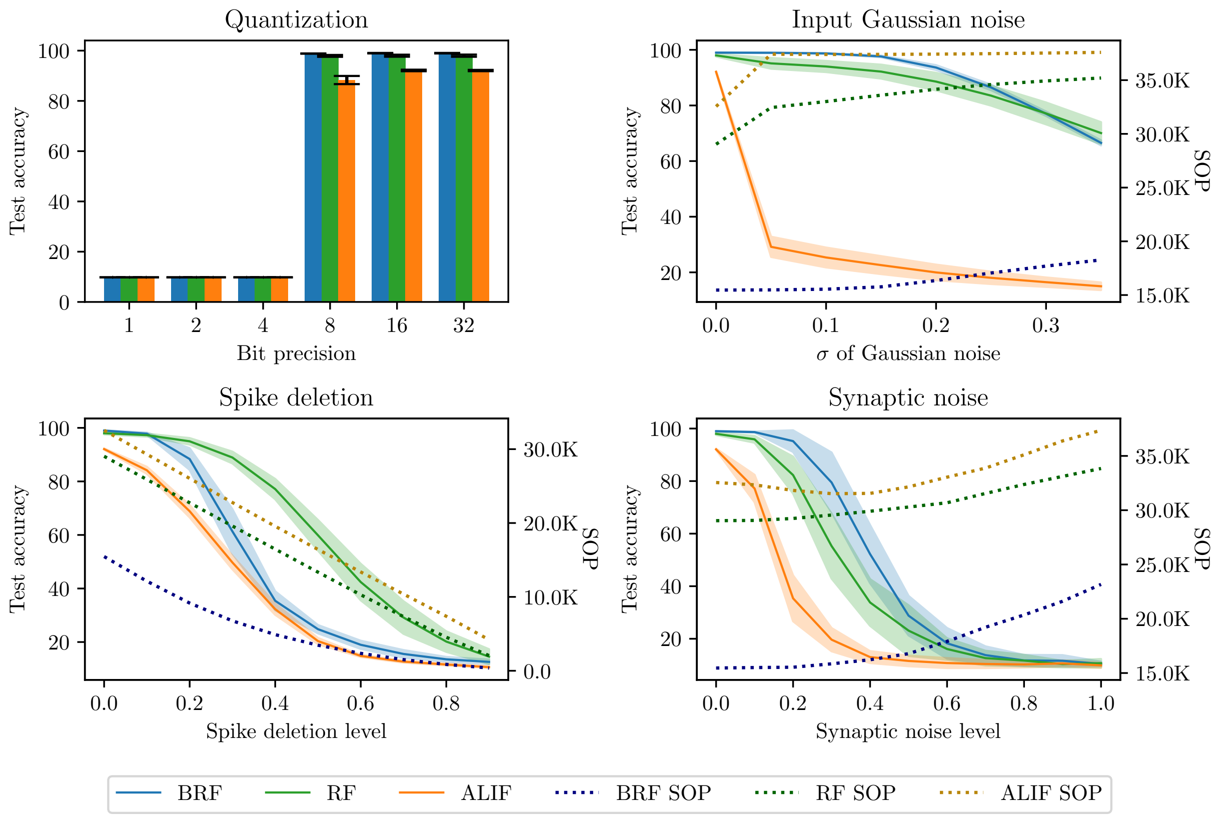

Figure 12. S-MNIST noise robustness studies conducted on saved best performing models. Solid line denote performance and dotted line the spiking operations (SOP) for each network.

representation (Figure 12 top left). Noise in the input signal from the dataset are simulated with Gaussian noise with increasing standard deviation (Figure 12 top right). For spike deletion (Park et al., 2021) level, a percentage of the spikes are removed from the network at each sequence step (Figure 12 bottom left). Synaptic noise added Gaussian noise to the weights.

Overall the vanilla RF network without reset and the BRF network performed better when faced with various noise than the ALIF network. Especially for the noise introduced in the input, the RF and BRF network maintained their performance, while the ALIF network failed already with a small standard deviation in noise. For the BRF network, this is achieved despite the high spiking sparsity of the BRF network compared to the RF or ALIF network. The RF network is more robust against spike deletion than the BRF or ALIF network. This may be an effect of redundant spiking for the RF networks, which highlights a trade-off of spike deletion robustness and high spiking sparsity.

A.11. Error landscape

BRF-RSNN landscape has smooth and convex-like structure, whereas landscape of RF- and especially, ALIF-RSNN, has a rough surface with a narrow valley. Such smooth landscape represent high generalization and straightforward optimization, accounting for the fast convergence.

We further found that the spectral radius of the membrane state transition mapping computed over one discrete time step results in unity or below by implementing the divergence boundary, thus effectively stabilizing the gradient and preserving its magnitude (Higuchi et al., 2024). This indicates that the divergence boundary contributes to fast and smooth learning.

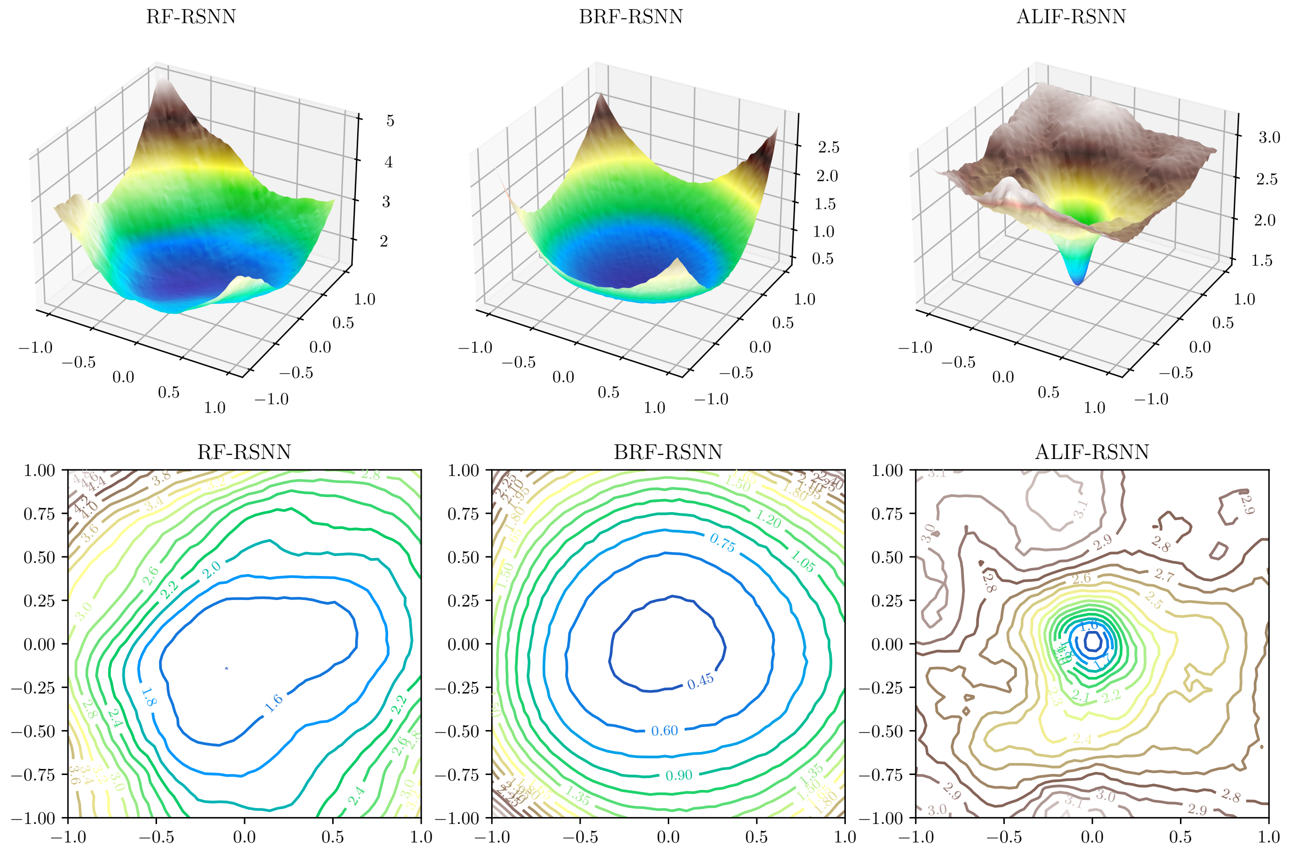

Figure 13. Error landscape plots for RF, BRF, and ALIF network on the S-MNIST dataset adopted from Higuchi et al. (2024). Top: error surface plots. x and y axis correspond to α and β, i.e. parameter deviations, and the z axis to $f(\alpha, \beta)$, i.e. error (Li et al., 2018). Bottom: error contour plots. Note that the value range and corresponding coloring differ between the diagrams to enhance visualization.

A.12. Hyperparameters

Table 3. Hyperparameters applied for the best-performing BRF, HBRF, and ALIF model with label-last loss (S-MNIST) and average sequence loss (PS-MNIST, ECG, SHD) and truncation step of 50 for TBPTT (in pruning). Same hyperparameters applied for BRF- and RF-BPTT models, where $b = b'$ for RF-RSNN.

S-MNIST	BRF-RSNN		BHRF-RSNN	ALIF-RSNN	
	BPTT	TBPTT	BPTT	BPTT	TBPTT
Network	1 + 256 (fully recurrent) + 10		1 + 256 (fully recurrent) + 10	1 + 256 (fully recurrent) + 10	
Learning rate (Lr)	0.1	0.006	0.1	0.001	
Loss function	NLL		CE	NLL	
Minibatch size	256		256	256	
Lr scheduling	LinearLR		LinearLR	LinearLR	
Optimizer	Adam		RAdam	Adam	
Epochs	300		400	300	
Parameter initialization	$\omega : \mathcal{U}(15, 50)$, $b' : \mathcal{U}(0.1, 1)$, $\tau_{m,out} : \mathcal{N}(20, 5)$		$\omega : \mathcal{U}(15, 35)$, $b' : \mathcal{U}(0.1, 1)$, $\alpha : Sigmoid(\mathcal{N}(0, 0.1))$	$\tau_m : \mathcal{N}(20, 5)$, $\tau_a : \mathcal{N}(200, 50)$, $\tau_{m,out} : \mathcal{N}(20, 5)$	

PS-MNIST	BRF-RSNN		BHRF-RSNN	ALIF-RSNN	
	BPTT	TBPTT	BPTT	BPTT	TBPTT
Network	1 + 256 (fully recurrent) + 10		1 + 256 (fully recurrent) + 10	1 + 256 (fully recurrent) + 10	
Learning rate (Lr)	0.1	0.006	0.1	0.001	
Loss function	NLL		NLL	NLL	
Minibatch size	256		256	256	
Lr scheduling	LinearLR		LinearLR	LinearLR	
Optimizer	Adam		RAdam	Adam	
Epochs	300		200	300	
Parameter initialization	$\omega : \mathcal{U}(15, 85)$, $b' : \mathcal{U}(0.1, 1)$, $\tau_{m,out} : \mathcal{N}(20, 1)$		$\omega : \mathcal{U}(10, 50)$, $b' : \mathcal{U}(1, 6)$, $\alpha : Sigmoid(\mathcal{N}(0, 0.1))$	$\tau_m : \mathcal{N}(20, 5)$, $\tau_a : \mathcal{N}(200, 50)$, $\tau_{m,out} : \mathcal{N}(20, 5)$	

ECG-QT	BRF-RSNN	BHRF-RSNN	ALIF-RSNN
Network	4 + 36 (fully recurrent) + 6	4 + 36 (fully recurrent) + 6	4 + 36 (fully recurrent) + 6
Learning rate (Lr)	0.1	0.3	0.05
Loss function	NLL	NLL	NLL
Minibatch size	16	4	64
Lr scheduling	LinearLR	LinearLR	LinearLR
Optimizer	Adam	RAdam	Adam
Epochs	400	300	400
Parameter initialization	$\omega : \mathcal{U}(3, 5)$, $b' : \mathcal{U}(0.1, 1)$, $\tau_{m,out} : \mathcal{N}(20, 1)$	$\omega : \mathcal{U}(7, 11)$, $b' : \mathcal{U}(0.1, 1)$, $\alpha : Sigmoid(\mathcal{N}(0, 0.1))$	$\tau_m : \mathcal{N}(20, 0.5)$, $\tau_a : \mathcal{N}(7, 0.2)$, $\tau_{m,out} : \mathcal{N}(20, 0.5)$
Sub-seq. length	0	0	10

SHD	BRF-RSNN	BHRF-RSNN	ALIF-RSNN
Network	700 + 128 (fully recurrent) + 20	700 + 128 (fully recurrent) + 20	700 + 128 (fully recurrent) + 20
Learning rate (Lr)	0.075	0.1	0.075
Loss function	NLL	NLL	NLL
Minibatch size	32	32	32
Lr scheduling	LinearLR	LinearLR	LinearLR
Optimizer	Adam	RMSprop	Adam
Epochs	20	20	20
Parameter initialization	$\omega : \mathcal{U}(5, 10)$, $b' : \mathcal{U}(2, 3)$, $\tau_{m,out} : \mathcal{N}(20, 5)$	$\omega : \mathcal{U}(3, 10)$, $b' : \mathcal{U}(0.1, 1)$, $\alpha : Sigmoid(\mathcal{N}(0, 0.1))$	$\tau_m : \mathcal{N}(20, 5)$, $\tau_a : \mathcal{N}(150, 10)$, $\tau_{m,out} : \mathcal{N}(20, 5)$
Sub-seq. length	0	0	10

Due to the BHRF-RSNN being in the preliminary phase of its exploration, we initially used the sigmoid function for the leaky integrator decay constant α, as it ensured the decay to be in the range of 0 and 1.

A.13. Learning speed-up with forward gradient injection (FGI) and TorchScript

As a further exploration, we apply forward gradient injection (FGI) from Otte (2024) for modeling the surrogate gradient functions. Let x be a computation node, f a non-differentiable function (e.g. step function), g' a function that we want to use as derivative for f, and sg the stop gradient operator. Instead of overriding the `backward()` method, we can just write:

$$h = x \cdot \mathrm{sg}(g'(x)) \tag{44}$$

$$y = h - \mathrm{sg}(h) + \mathrm{sg}(f(x)) \tag{45}$$

With FGI + TorchScript we achieve speed-ups by a factor of more than two for all dataset as shown in Table 4, while maintaining performance.

Table 4. BRF-RSNN training speed up with model optimization methods over standard backward() baseline.

Optimization	Gradient	S-MNIST		PS-MNIST		ECG		SHD	
		Time (hrs)	Ratio	Time (hrs)	Ratio	Time (hrs)	Ratio	Time (min)	Ratio
-	backward()	23.8	$1\times$	31.0	$1\times$	8.5	$1\times$	40.9	$1\times$
TorchScript	backward()	11.3	$2.1\times$	13.8	$2.2\times$	4.1	$2.1\times$	16.4	$2.5\times$
TorchScript	FGI	10.2	$\mathbf{2.3}\times$	12.8	$\mathbf{2.4}\times$	3.5	$\mathbf{2.4}\times$	14.0	$\mathbf{2.9}\times$