Supplementary Materials

Pinsky-Rinzel Neuron Model

This section presents the generalized two-compartment spiking neuron model derived from the two-compartment Pinsky-Rinzel (P-R) pyramidal neuron model. The two-compartment P-R neuron model is designed to elucidate intricate biophysical mechanisms that underly complex bursting within CA3 pyramidal cells and enables lightweight computation. Its neuronal dynamic can be formulated in continuous time by the following equations:

$$C_m \frac{dV_s}{dt} = -I_{Na} - I_K - I_{Leak} + \frac{I_{link}}{P} + I_s \quad (17)$$

$$C_m \frac{dV_d}{dt} = -I_{NaP} - I_{KS} - I_{Leak} - \frac{I_{link}}{1-P} + I_d \quad (18)$$

where V_s and V_d are the membrane potentials of somatic and dendritic compartments. I_{Na} and I_K denote the pertinent currents in the somatic compartment, while the dendritic compartment encompasses the slow potassium current I_{KS} and persistent sodium current I_{NaP}. The input currents to the soma and dendrite are denoted by I_s and I_d respectively. Specifically, I_s is assumed to be 0 in this paper and the input currents are solely injected into the dendritic compartment. Additionally, the membrane capacitance and the proportion of cell area are represented by C_m and P respectively.

Table 3 provides the detailed calculations related to the ionic currents mentioned in the equations above. In particular, E_{Na}, E_K, and E_L signify equilibrium potentials, while g_{Na}, g_K, g_L, g_c, g_{NaP}, and g_{KS} represent conductances.

Table 3: Summary of the ionic current calculation in two-compartment P-R neuron model.

Ionic Current	**Calculation**
I_{Na}	$g_{Na}m^3h \cdot (V_s - E_{Na})$
I_K	$g_K n^4 \cdot (V_s - E_K)$
I_{Leak}	$g_L \cdot (V - E_L)$
I_{link}	$g_c \cdot (V_d - V_s)$
I_{NaP}	$g_{NaP}l^3h \cdot (V_d - E_{Na})$
I_{KS}	$g_{KS}q \cdot (V_d - E_K)$

We obtain the discrete-time formulations for Eqs. (17) and (18) by the Euler method as follows:

$$\begin{aligned} V_s[t+1] =& V_s[t] + \frac{dt}{C_m}(-I_{Na}[t] - I_K[t] - \\ & I_{Leak}[t] + \frac{I_{link}[t]}{P}) \end{aligned} \quad (19)$$

$$\begin{aligned} V_d[t+1] =& V_d[t] + \frac{dt}{C_m}(-I_{NaP}[t] - I_{KS}[t] - \\ & I_{Leak}[t] + \frac{I_{link}[t]}{1-P} + I_d[t]) \end{aligned} \quad (20)$$

The term I_{link} signifies the interaction between the somatic and dendritic compartments. Additionally, by substituting the expressions for ionic currents from the Table 3 into Eq. (19) and (20) and integrating the spiking operation for the somatic output membrane potential, we derive the overall dynamics of the generalized two-compartment spiking neuron model, as depicted in Eq. (7)-(9).

Energy Efficiency Analysis

We analyze the theoretical energy cost for LSTM, LIF, and TC-LIF recurrent networks based on their neuronal update functions. Table 4 presents the detailed calculation of theoretical energy cost for each model.

Experimental Details

Datasets

In this subsection, we introduce the dataset used for this work. These datasets cover a wide range of tasks, allowing us to assess the model's capabilities in handling different types of input data.

S-MNIST: The Sequential-MNIST (S-MNIST) dataset is derived from the original MNIST dataset, which consists of 60,000 and 10,000 grayscale images of handwritten digits for training and testing sets with a resolution of 28 $\times$ 28 pixels. In the S-MNIST dataset, each image is converted into a vector of 784 time steps, with each pixel representing one input value at a certain time step. This dataset enables us to evaluate the performance of our model in solving sequential image classification tasks.

PS-MNIST: The Permuted Sequential MNIST dataset (PS-MNIST) is a variation of the Sequential MNIST dataset, in which the pixels in each image are shuffled according to a fixed random permutation. This dataset provides a more challenging task than S-MNIST, as the input sequences no longer follow the original spatial order of the images. Therefore, when learning this dataset, the model needs to capture complex, non-local, and long-term dependencies between pixels.

GSC: The Google Speech Commands (GSC) has two versions, and we employ the 2nd version in this work. The GSC version 2 is a collection of 105,829 on-second-long audio clips of 35 different spoken commands, such as "yes", "no", "up", "down", "left", "right", etc. These audio clips are recorded by different speakers in various environments, offering a diversity of datasets to evaluate the performance of our model.

SHD: The Spiking Heidelberg Digits dataset is a spike-based sequence classification benchmark, consisting of spoken digits from 0 to 9 in both English and German (20 classes). The dataset contains recordings from twelve different speakers, with two of them only appearing in the test set. Each original waveform has been converted into spike trains over 700 input channels. The train set contains 8,332 examples, and the test set consists of 2,088 examples (no validation set). The SHD dataset enables us to evaluate the performance of our proposed model in processing and classifying speech data represented in spiking format.

Table 4: Computations on the energy cost of LIF, TC-LIF, and LSTM.

Neuron Model	Dynamics	Step Cost	Total Cost
LIF	$\mathcal{I}_t = \mathcal{W}^{m,n}X^m + \mathcal{W}^{n,n}\mathcal{S}^n_{t-1}$ $\mathcal{U}_t = \beta\mathcal{U}_{t-1} + \mathcal{I}_t - \mathcal{V}_{th}\mathcal{S}^n_{t-1}$	$(mnFr_{in} + nnFr_{out})E_{AC}$ $nFr_{out}E_{AC} + nE_{MAC}$	$mnFr_{in}E_{AC} + (nn+n)Fr_{out}E_{AC}$ $+nE_{MAC}$
TC-LIF	$\mathcal{I}_t = \mathcal{W}^{m,n}X^m + \mathcal{W}^{n,n}\mathcal{S}^n_{t-1}$ $\mathcal{U}^D_t = \mathcal{U}^D_{t-1} + \mathcal{I}_t + \beta_1\mathcal{U}^S_{t-1} - \gamma\mathcal{S}^n_{t-1}$ $\mathcal{U}^S_t = \mathcal{U}^S_{t-1} + \beta_2\mathcal{U}^D_t - \mathcal{V}_{th}\mathcal{S}^n_{t-1}$	$(mnFr_{in} + nnFr_{out})E_{AC}$ $nFr_{out}E_{AC} + nE_{MAC}$ $nFr_{out}E_{AC} + nE_{MAC}$	$mnFr_{in}E_{AC}$ $+(nn+2n)Fr_{out}E_{AC}$ $+2nE_{MAC}$
LSTM	$f_t = \sigma_g(\mathcal{W}_f x_t + U_f h_{t-1} + b_f)$ $i_t = \sigma_g(\mathcal{W}_i x_t + U_i h_{t-1} + b_i)$ $o_t = \sigma_g(\mathcal{W}_o x_t + U_o h_{t-1} + b_o)$ $\hat{c}_t = \sigma_c(\mathcal{W}_c x_t + U_c h_{t-1} + b_c)$ $c_t = f_t \odot c_{t-1} + i_t \odot \hat{c}_t$ $h_t = o_t \odot \sigma_h(c_t)$	$n(m+n+2)E_{MAC}$ $n(m+n+2)E_{MAC}$ $n(m+n+2)E_{MAC}$ $n(m+n+4)E_{MAC}$ $2nE_{MAC}$ $5nE_{MAC}$	$4(mn+nn)E_{MAC}$ $17nE_{MAC}$

SSC: The Spiking Speech Command dataset, another spike-based sequence classification benchmark, is derived from the Google Speech Commands version 2 dataset and contains 35 classes from a large number of speakers. The original waveforms have been converted to spike trains over 700 input channels. The dataset is divided into train, validation, and test splits, with 75,466, 9,981, and 20,382 examples, respectively. The SSC dataset allows us to assess the performance of our proposed spiking neuron model in processing and recognizing speech commands represented in spiking data.

Training with Surrogate Gradient

Training SNN poses challenges stemming from the non-differentiability of spike functions, denoted as $\Theta(x)$ in Eqs. (3), (9), and (12). This trait hinders the application of prevalent gradient-based optimization methods, notably backpropagation. The surrogate gradient approach offers a solution to this impediment by introducing a proxy gradient as an approximation for the gradient of the spike function, expressed as $\Theta'(x) \approx \theta'(x)$. While the actual gradient mostly holds a zero value, the surrogate gradient approximates non-zero values in regions of interest. This allows backpropagation to be applied, as the surrogate gradient provides the necessary feedback to update the network's weight.

In this work, we adopt the triangle function as $\theta'(x)$ to enable gradient-based training for SNN:

$$\frac{\partial \mathcal{S}[t]}{\partial \mathcal{U}[t]} = \theta'(\mathcal{U}[t] - \mathcal{V}_{th}) = \max(1 - |\mathcal{U}[t] - \mathcal{V}_{th}|, 0) \quad (21)$$

where $\mathcal{U}[t]$ denotes the membrane potential in Eq. (3) for single-compartment spiking neuron and the somatic membrane potential in Eqs. (9) and (12) for two-compartment spiking neurons.

Network Architecture

We perform experiments employing both feedforward and recurrent connection configurations. To maintain a fair comparison with existing works, we utilize network architectures exhibiting comparable parameters. These architectures and their corresponding parameters are summarized in Table 6.

TC-LIF Model Hyper-parameters

We outline the specific hyper-parameter settings for the TC-LIF neuron model in Table 5, encompassing the dendritic reset scalar γ, the spike threshold $\mathcal{V}_{th}$, and initial values for β_1 and β_2.

Table 5: Network hyper-parameters for TC-LIF.

Dataset	Network	γ	β_1, β_2	$\mathcal{V}_{th}$
S-MNIST	feedforward recurrent	0.5 0.5	(-0.5, 0.5) (-0.8, 0.4)	1.0 1.0
PS-MNIST	feedforward recurrent	0.7 0.5	(-0.5, 0.5) (-0.2, 0.8)	1.5 1.8
GSC	feedforward recurrent	0.6 0.7	(-0.5, 0.5) (-0.8, 0.8)	1.2 1.25
SHD	feedforward recurrent	0.5 0.5	(-0.5, 0.5) (-0.5, 0.5)	1.5 1.5
SSC	feedforward recurrent	0.5 0.5	(-0.5, 0.5) (-0.5, 0.5)	1.5 1.5

Table 6: Summary of network architectures and parameters.

Dataset	Network	Architecture	Parameters(K)
S-MNIST	feedforward recurrent	40-256-128-10/64-256-256-10 40-200-64-10/64-256-256-10	44.8/85.1 63.6/155.1
PS-MNIST	feedforward recurrent	40-256-128-10/64-256-256-10 40-200-64-10/64-256-256-10	44.8/85.1 63.6/155.1
GSC	feedforward recurrent	40-300-300-12 40-300-300-12	106.2 196.5
SHD	feedforward recurrent	700-128-128-20 700-128-128-20	108.8 141.8
SSC	feedforward recurrent	700-128-128-35 700-128-35	110.8 110.8

Training Configuration

We train the S-MNIST and PS-MNIST datasets for 200 epochs utilizing the Adam optimizer. Their initial learning rates are set to 0.0005 for both feedforward and recurrent networks with the learning rates decaying by a factor of 10 at epochs 60 and 80. For the GSC, SHD, and SSC datasets, we train the models for 100 epochs using the Adam optimizer. The initial learning rate of GSC datasets is 0.001 for both feedforward and recurrent networks, which decays by 10 at Epoch 60, 90, and 120. The initial learning rate is set to 0.0005, and 0.005 for feedforward and recurrent networks on the SHD dataset, with the learning rate decaying to 0.8 times its previous value after every 10 epochs. For the SSC dataset, the initial learning rates are 0.0001 for both feedforward and recurrent networks, and decay to 0.8 times their previous values every 10 epochs. We train S-MNIST, PS-MNIST, and GSC tasks on Nvidia Geforce GTX 3090Ti GPUs with 24GB memory, and train SHD and SSC tasks on Nvidia Geforce GTX 1080Ti GPUs with 12GB memory.

Table 7: Infinite norm values with corresponding β_1 and β_2 before and after training on S-MNIST.

Before		**After**		**Test Acc**
β_1, β_2	norm	β_1, β_2	norm	
(-0.2, 0.2)	1.352	(-0.184, 0.146)	1.262	98.40
(-0.2, 0.4)	1.688	(-0.188, 0.307)	1.539	99.07
(-0.2, 0.6)	2.008	(-0.203, 0.563)	1.948	99.15
(-0.2, 0.8)	2.312	(-0.202, 0.835)	2.360	89.36
(-0.4, 0.2)	1.304	(-0.379, 0.159)	1.248	99.01
(-0.4, 0.4)	1.576	(-0.370, 0.318)	1.480	99.15
(-0.4, 0.6)	1.816	(-0.383, 0.532)	1.751	99.06
(-0.4, 0.8)	2.024	(-0.380, 0.700)	1.949	99.04
(-0.6, 0.2)	1.256	(-0.621, 0.172)	1.219	99.08
(-0.6, 0.4)	1.464	(-0.621, 0.342)	1.399	99.01
(-0.6, 0.6)	1.624	(-0.634, 0.594)	1.587	98.96
(-0.6, 0.8)	1.736	(-0.612, 0.761)	1.704	98.82
(-0.8, 0.2)	1.208	(-0.812, 0.187)	1.194	98.97
(-0.8, 0.4)	1.352	(-0.815, 0.360)	1.321	99.20
(-0.8, 0.6)	1.432	(-0.812, 0.580)	1.416	98.64
(-0.8, 0.8)	1.448	(-0.821, 0.801)	1.418	98.99

Gradient Exploding Problem Analysis

We analyze the severity of the gradient exploding problem concerning the TC-LIF initialized in our predefined region. Specifically, recurrent SNNs are trained on the S-MNIST dataset with TC-LIF models initialized by different (β_1, β_2) in the second quadrant. Our analysis involves recording the values of β before and after training and calculating the infinite norms of the partial derivatives between adjacent time steps in the last hidden layer.

The result reveals that except for the model that is initialized at (-0.2, 0.8) with a convergent infinite norm of 2.36, the remaining models initialized within the second quadrant exhibit commendable performance on the test set. While an infinite norm of the partial derivative between successive time steps exceeding 1 suggests the potential for the exploding gradient problem during long-term BPTT, our results suggest that values slightly above 1 for the infinite norm do not notably impede convergence. Encouragingly, for the majority of β_1 and β_2 initialized in the second quadrant, the corresponding infinite norm values satisfy this condition. Hence, when initializing the TC-LIF model within the second quadrant, stable convergence for the proposed TC-LIF model can be promised, mitigating concerns regarding the gradient exploding problem.