MMDEND: Dendrite-Inspired Multi-Branch Multi-Compartment Parallel Spiking Neuron for Sequence Modeling

Kexin Wang[1,2], Yuhong Chou[3,1], Di Shang[1,2], Shijie Mei[1,2],
Jiahong Zhang[1,2], YanBin Huang[1,2], Man Yao[1], Bo Xu[1,2], and Guoqi Li[1,2*]
[1]The Key Laboratory of Cognition and Decision Intelligence for Complex Systems,
Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3]The Hong Kong Polytechnic University
[1]wangkexin2021@ia.ac.cn, *guoqi.li@ia.ac.cn

Abstract

Vanilla spiking neurons are simplified from complex biological neurons with dendrites, soma, and synapses, into single somatic compartments. Due to limitations in performance and training efficiency, vanilla spiking neurons face significant challenges in modeling long sequences. In terms of performance, the oversimplified dynamics of spiking neurons omit long-term temporal dependencies. Additionally, the long-tail membrane potential distribution and binary activation discretization errors further limit their capacity to model long sequences. In terms of efficiency, the serial mechanism of spiking neurons leads to excessively long training times for long sequences. Though parallel spiking neurons are an efficient solution, their number of parameters is often tied to the hidden dimension or sequence length, which makes current parallel neurons unsuitable for large architectures. To address these issues, we propose **MMDEND**[1]: a **M**ulti-Branch **M**ulti-Compartment Parallel Spiking **Dend**ritic Neuron. Its proportion-adjustable multi-branch, multi-compartment structure enables long-term temporal dependencies. Additionally, we introduce a **S**caling-**S**hifting Integer **F**iring (**SSF**) mechanism that fits the long-tail membrane potential distribution, retains efficiency, and mitigates discretization errors. Compared with parallel neurons, MMDEND achieves better long-sequence modeling capability with fewer parameters and lower energy consumption. Visualization also confirms that the SSF mechanism effectively fits long-tail distributions.

1 Introduction

Vanilla spiking neurons are simplified abstractions of biological neurons, simulating the integrate-fire-reset dynamics. Advancements in training algorithms (Wu et al., 2018; Duan et al., 2022) have allowed spiking neurons to achieve success in many tasks while maintaining energy efficiency (Lv et al., 2023; Li et al., 2023; Zhu et al., 2023; Wang et al., 2024; Zhao et al., 2021; Rajagopal et al., 2023; Yao et al., 2024; Zhou et al., 2022). However, spiking neurons face significant challenges in modeling long sequences (Fang et al., 2024; Stan and Rhodes, 2024) due to limitations in both performance and efficiency.

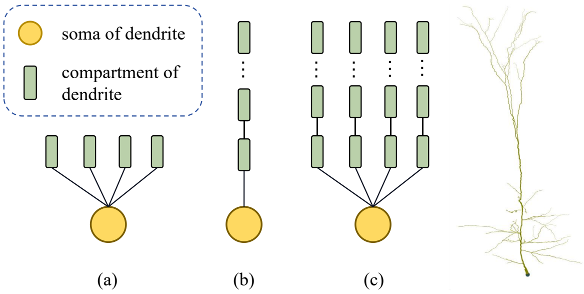


Figure 1: Types of Dendritic Neuron Modeling

In terms of performance, the overly simplified temporal dynamics are a key factor limiting the ability of spiking neurons to model long sequences (Fang et al., 2024; Stan and Rhodes, 2024). A typical biological neuron structure includes components such as dendrites, synapses, soma, and axon (Spruston, 2008). According to the modeling of this typical structure, current neuron models can be categorized into point neurons and fine-grained neurons. Vanilla spiking neurons, such as LIF (Maass, 1997), are a classic example of point neurons, where the neuron is simplified to a single soma. Due to this simplification, point neurons have limited temporal dynamics, making it difficult to capture long-term dependencies (Legenstein and Maass, 2011). On the other hand, the fine-grained neurons incorporate a more comprehensive biological neuron structure and exhibit long-term dependent temporal dynamics (Chen et al., 2024; Zheng et al., 2024). Specifically, the multi-branch, multi-compartment structure of dendrites in biological neurons has demonstrated exceptional capabilities in processing temporal signals (London

[1] https://github.com/WKX933/MMDEND

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 27459–27470
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

and Häusser, 2005). Although recent research has explored applying dendritic dynamics to sequential tasks (Zheng et al., 2024; Chen et al., 2024; Egrioglu et al., 2022; Egrioglu and Bas, 2024), the complexity of dendritic structures makes it challenging to balance detailed modeling with computational efficiency. Therefore, most of these works focus on either the multi-compartment or multi-branch structures as shown in Figure.1(b) and (a), without fully leveraging the dendritic dynamics.

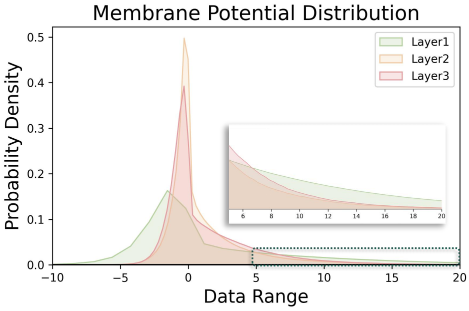


Figure 2: Long-tail Membrane Potential Probability Distribution.

Another issue that impacts the performance of spiking neurons is that binary activation leads to discretization errors and is difficult to fit long-tail distributions. Spiking neurons typically transmit binary spikes, which often require multiple time steps to mitigate the binary discretization errors. However, the multiple extended time steps result in exponentially higher training costs (Guo et al., 2024). To mitigate the discretization errors, while also taking the training efficiency into account, existing works for sequence tasks opt not to extend time steps. Instead, they employ dedicated firing mechanisms (Guo et al., 2024; Luo et al., 2024), such as negative spike activation, learnable spike activation and integer activation, to replace binary encoding. However, these methods fire within a fixed and symmetric range, which limits their ability to adapt to the asymmetric long-tail distribution of membrane potentials, as shown in Figure 2.

In terms of efficiency, the challenge is that the serial mechanism inherent in spiking neurons results in excessively long training time for long sequence tasks (Fang et al., 2024). Therefore, recent research has focused on either eliminating or improving the nonlinear reset mechanism and transitioning to parallel mechanisms (Chen et al., 2024; Fang et al., 2024). However, the number of parameters in current parallel neurons is often tied to the hidden dimension or sequence length, which makes these works more like modeling of layers rather than neurons and is unsuitable for large models.

Based on the above analysis, in this work, for sequence modeling tasks, we propose a **M**ulti-Branch **M**ulti-Compartment Parallel Spiking **Dend**ritic Neuron (MMDEND). As for performance, the multi-branch, multi-compartment structure of MMDEND provides long-term dependent dynamics. Expanding from a single branch to multiple branches may introduce exponential computational complexity. MMDEND achieves adjustable multi-branch proportions by grouping inputs, which allows it to enhance performance while reducing computational complexity. To overcome the limitations of binary firing, we introduce a Scaling-Shifting Integer Firing (**SSF**) mechanism that effectively fits the long-tail membrane potential distribution. SSF uses single-step integer training and multi-step spike inference, ensuring efficiency in both training and inference (Luo et al., 2024). To ensure efficient parallelism, dendritic dynamics are modeled using State-Space Modeling (SSM) and nonlinear firing is removed in the soma (Fang et al., 2024). Unlike traditional parallel neurons, the number of parameters in MMDEND is independent of both the channel and sequence length. Our main contributions can be summarized as follows:

- We propose **MMDEND**, a multi-branch, multi-compartment parallel spiking dendritic neuron with long-term dependency dynamics. The multi-branch proportion is adjustable for computing saving and task performance.

- We propose SSF mechanism that dynamically adapts to long-tail membrane potential distributions through translation and scaling. SSF adopts single-step integer firing during training and multi-step spiking firing during inference for efficiency.

- MMDEND achieves better long sequence modeling capability than parallel neurons with fewer parameters and lower energy consumption. Visualization confirms that SSF mechanism effectively fits long-tail distributions.

2 Related work

Spiking Neuron For Sequence Modeling. Due to the serial temporal mechanisms, lengthy training times pose a bottleneck for spiking neuron performance in long sequence modeling. To tackle this challenge, (Fang et al., 2024) introduced PSN, which eliminates the nonlinear reset mechanism to enable parallelism in spiking neurons and incorporates learnable time decay constants to compensate for neural dynamics. (Chen et al., 2024), inspired by pyramidal cells, proposed PMSN, which revisits the reset mechanism in spiking neurons while achieving multi-compartment parallelism. Considering the superior performance of SSM in processing temporal signals, (Stan and Rhodes, 2024) replaced the LIF dynamics with SSM. This approach is similar to the multi-compartment modeling used in PMSN, but lacks the soma component. Notably, the neuron size of these works depends on the sequence length or the hidden dimensions, which makes these works more like layers modeling rather than neurons.

Dendrite Modeling. Dendritic neurons are a type of biological neuron in the brain, characterized by their excellent temporal computation abilities and nonlinear expression properties (Chen and Liu, 2022; Wu et al., 2023). (Zheng et al., 2024) combines dendrites with spiking neural networks to propose DH-LIF, which effectively learns temporal features at different scales through heterogeneous timing factors on various dendritic branches. (Ji et al., 2022) modeled the dendritic neuron from four levels: synaptic, dendrite, membrane, and soma, with the dendritic component employing a multi-branch architecture. (Chen et al., 2024) proposed a single-branch multi-compartment model. These studies consider either multi-branch or multi-compartment characteristics alone, lacking comprehensive modeling of the full dendritic architecture.

Spiking Firing Mechanism. Multiple time steps are typically used to compensate for the information loss caused by binary firing, but this approach significantly increases computational costs. Recent work attempts to compensate for the loss from the firing mechanism. (Sun et al., 2022) introduced dual-thresholds and used integer firing. (Guo et al., 2024) proposed ternary spikes with negative activation and designed learnable peak amplitudes to adapt to different membrane potential distributions across layers. (Luo et al., 2024) proposed ILIF with positive integer firing during traing and spiking firing during inference.

3 Preliminaries

Spiking Neurons. LIF is a classic spiking neuron with a charge-fire-reset dynamic, and we take LIF as an example to introduce the spiking neurons. The dynamic process of LIF can be calculated as:

$$H_t = (1 - \frac{1}{\tau})V_{t-1} + \frac{1}{\tau}X_t \tag{1}$$

$$S_t = \Theta(H_t - V^{th}) \tag{2}$$

$$V_t = V^{re}S_t + H_t(1 - S_t) \tag{3}$$

The sequence from Eq.(1) to (3) describes the key processes in the LIF neuron model: charging, firing, and resetting. In these equations, X_t indicates the input current at each time step t, while H_t refers to the post-charge membrane potential. τ is the time dynamic factor. The spike tensor at time t is denoted by S_t. Θ is the step function, and V^{th} is the threshold voltage beyond which firing occurs. After firing, the membrane potential resets to V^{re}. In this work, we replace the charging dynamics with dendritic and soma dynamics. Additionally, we substitute the firing and resetting mechanisms with the SSF mechanism.

State Space Model. The SSM is a method for describing and analyzing dynamic systems, applicable to systems described by first-order or higher-order differential equations (Kalman, 1960). Its classical formulation can be expressed as:

$$\dot{h_t} = Ah_t + Bx_t \tag{4}$$

$$y_t = Ch_t + Dx_t \tag{5}$$

where A, B, C, D represent control matrices. Typically, before performing computer simulations, discretization methods are employed, such as the zero-order hold method (ZOH) (DeCarlo, 1989) for discretization. The discretized form of Eq. (4) can be expressed as:

$$h_t = \bar{A}h_{t-1} + \bar{B}x_t \tag{6}$$

where $\bar{A} = exp(\Delta A)$, $\bar{B} = (\Delta A)^{-1}(exp(\Delta A) - I) \cdot \Delta B$, Δ represents the sampling interval from continuous to discrete. Since our modeling of the dendrites starts with Kirchhoff's current law with first-order differential equations, the SSM is well-suited for the dendrites modeling and facilitates the parallelization of dendrites.

4 Method: MMDEND

In this work, we propose MMDEND, an adjustable multi-branch, multi-compartment parallel spiking neuron inspired by dendritic neurons. Starting from the dendritic model constructed via cable theory in Figure 1(c), we introduce the dendritic branch modeling and the soma modeling. Finally, we will present the SSF mechanism in detail.

4.1 Single-branch Multi-compartment Dendrite SSM Modeling

According to cable theory and PMSN (Chen et al., 2024), each branch of a dendrite can be modeled as a series of single-compartment circuits. As shown in the left part of Figure 3, each compartment includes a leakage resistor R_L, a cell membrane capacitor C_m, and a current source representing external input. R_k denotes the axial resistance between the k-th and $(k-1)$-th compartments. V_k represents the voltage value of the k-th compartment. The current continuity equation for compartment k is:

$$\frac{dv_{jk}}{dt} = \frac{v_{j(k-1)}}{\tau^f_{jk}} - \frac{v_{jk}}{\tau_{jk}} + \frac{v_{j(k+1)}}{\tau^p_{jk}} + \gamma_k I_j \tag{7}$$

where v_{jk} represents the voltage of the k-th compartment in the j-th branch ($j \in \{1, \ldots, J\}$, $k \in \{1, \ldots, K\}$), J and K denote the total number of branches and compartments, respectively. $\tau^f_{jk} = R_{j(k-1)}C_m$ and $\tau^p_{jk} = R_{jk}C_m$ represent the influence of adjacent compartments on the current compartment's membrane potential, and $\frac{1}{\tau_{jk}} = \frac{R_{jk}+R_{ljk}}{C_m R_{jk}+C_m R_{ljk}}$ denotes the time constant of the current compartment's temporal dynamics. $\gamma_k = \frac{r_k}{C_m}$, where r_k represents the decay coefficient that varies with the distance between the input current and the compartment. It is important to note that we decouple the last compartment of the dendrite from the soma, so each branch for the input $\mathbf{I}_j \in \mathbb{R}^{D'}$ can be described as:

$$\dot{\mathbf{V}}^{\mathbf{c}}_j = \begin{bmatrix} -\frac{1}{\tau_{j1}} & \frac{1}{\tau^p_{j1}} & 0 & \cdots \\ \frac{1}{\tau^f_{j2}} & -\frac{1}{\tau_{j2}} & \frac{1}{\tau^p_{j2}} & \cdots \\ \vdots & & \ddots & \\ 0 & \cdots & \frac{1}{\tau^f_{jK}} & -\frac{1}{\tau_{jK}} \end{bmatrix} \mathbf{V}^c_j + \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_K \end{bmatrix} \mathbf{I}_j \tag{8}$$

$$\mathbf{V}^{dend}_j = \begin{bmatrix} 0 & 0 & \ldots & 1 \end{bmatrix} \mathbf{V}^c_j + \gamma^o_j \mathbf{I}_j \tag{9}$$

where V^{dend}_j is the terminal voltage of branch j, determined by the voltage of the last compartment and the decoupling compensation term $\gamma^o_j \mathbf{I}_j$.

Each dendritic branch described above is a Single Input Single Output (SISO) continuous SSM system. We employ the ZOH method for discretization. It is important to note that the state transition matrix has very high computational complexity when performing exponential operations as a density matrix, making it difficult for long sequence operations. Therefore, we perform an eigenvalue decomposition of the state transition matrix $A = P\Lambda P^{-1}$. To ensure the transition matrix can be diagonalized and to enhance the expressiveness of the diagonalized matrix, we perform calculations in the complex domain. The terminal voltage V^{dend}_j of the dendritic branch can be calculated as:

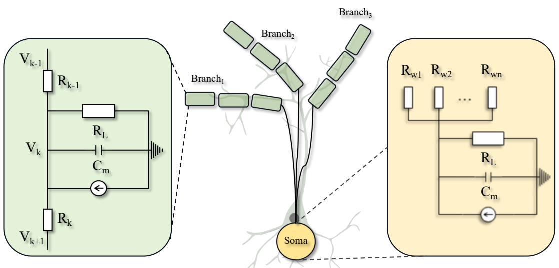


Figure 3: Dendritic Neuron Modeling.

$$\mathbf{V}_j[t] = \hat{\mathbf{A}}\mathbf{V}_j[t-1] + \hat{\mathbf{\Gamma}}\mathbf{I}_j[t] \tag{10}$$

$$\mathbf{V}^{dend}_j[t] = \hat{C}\mathbf{V}_j[t] + \gamma^o_j \mathbf{I}_j[t] \tag{11}$$

where $\mathbf{V}_j = P^{-1}\mathbf{V}^c_j, \mathbf{V}_j \in R^{K\times D'}$, the state transition matrix $\hat{\mathbf{A}} = exp\left(\mathbf{\Lambda dt}\right), \hat{\mathbf{A}} \in R^{K\times K}$, the distance coefficient matrix $\hat{\Gamma} = \Lambda^{-1}(\hat{A} - I)P^{-1}\Gamma$, $\Gamma = \left[\gamma_1, \ldots, \gamma_K\right]^T, \hat{\Gamma} \in R^{K\times 1}$, the output matrix $\hat{C} = \begin{bmatrix} 0 & 0 & \ldots & 1 \end{bmatrix} P$. $\mathbf{\Lambda}$, $\mathbf{dt}$, $\hat{\mathbf{C}}$, $\mathbf{\Gamma}$, and γ^o_j are all learnable parameters. In the hidden state expressions of each compartment, there is no nonlinear representation. Therefore, Eq.(10) can be expressed in a parallel form as:

$$\mathbf{V}_j[t] = \sum_{q=0}^{t} \hat{\mathbf{A}}^{t-q}\hat{\mathbf{\Gamma}}\mathbf{I}_j[q] \tag{12}$$

The parallel form of each branch, Eq.(12) can be efficiently implemented through FFT convolution.

4.2 Multi-branch Multi-compartment Dendrite Modeling

Extending to multi-branch can lead to an exponential increase in computational cost. To address this, we group the inputs along the hidden dimension, enabling the proportion of the multi-branch adjustable. First, we divide the input $I \in \mathbb{R}^D$ into J groups, with each group having a window length of D', where $D \geq D' \geq \frac{D}{J}$, and a stride of S. This results in the input current for each branch $I_{1,\ldots,J} \in \mathbb{R}^{D'}$. Grouping the input allows the number of dendritic branches per channel to dynamically vary between $\{1, \ldots, J\}$, adapting to tasks of different difficulty levels. To simulate the high nonlinear expressiveness of dendritic neurons, nonlinear activation is applied to the output current of each branch:

$$\mathbf{V}_j^{dend}[t] = f\left(g_j\left(\mathbf{I}\left[j*S:j*S+D'\right]\right)\right) \tag{13}$$

where f is the nonlinear function, and g_j denotes the dynamic process of dendritic branch j.

4.3 Soma Modeling

As shown in the right part of Figure 3, the soma includes axial resistances R_{wj} for each branch, soma leakage resistance R_L, soma capacitance C_m, and a current source determined by the input. The current continuity equation for the soma is:

$$\frac{dv^s}{dt} = -\frac{v^s}{\tau^s} + \sum_{j=1}^{M}\frac{v_j^{dend}}{\tau_j^s} + \gamma^s I \tag{14}$$

where v^s is the membrane potential of the soma, τ^s and τ_j^s are the time constants determined by the axial resistances R_{wj} and structural parameters of soma, and $\gamma^s = \frac{\tau_s}{C_m}$. For the soma, we also use ZOH for discretization,

$$\mathbf{V}^s[t] = \beta\mathbf{V}^s[t-1] + \alpha(\sum_{j=1}^{M}\frac{\mathbf{V}_j^{dend}}{\tau_j^s} + \gamma^s\mathbf{I}) \tag{15}$$

$$\mathbf{S}[t] = SSF(\mathbf{V}^s[t]) \tag{16}$$

where $\beta = \exp\left(-\frac{dt}{\tau^s}\right)$, and $\alpha = \tau^s(1-\beta)$. SSF is a firing mechanism capable of dynamically adapting to long-tail membrane potential distributions.

4.4 Scaling-Shifting Integer Firing

We propose SSF to dynamically adapt to asymmetric long-tail membrane potential distributions and negative membrane potentials. The SSF mechanism consists of two main components: membrane potential fitting and efficient integer firing.

Membrane Potential Fitting. SSF uses threshold as a measure of membrane potential to determine the integer value or number of spikes that can be triggered. To tackle the long-tail distribution and negative membrane potentials, we introduce an offset ϕ_p and a scaling factor ϕ_s in the firing mechanism. These parameters translate and scale the membrane potential to the effective encoding range $[-U, U]$, $U \in \mathbb{Z}^+$, ensuring information completeness. The membrane potential fitting process of SSF can be written as:

$$S[t] = \lfloor clip(\frac{\mathbf{V}^s - \phi_p}{\phi_s}, -U, U)/V^{th}\rfloor \tag{17}$$

where $\lfloor\cdot\rfloor$ is the floor function, $clip(*, -U, U)$ represents clipping within the range $[-U, U]$, and V^{th} is the threshold.

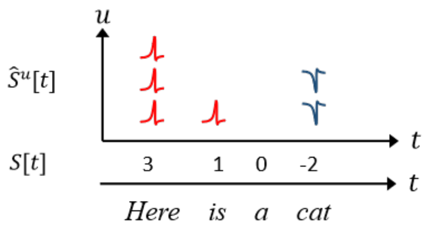


Figure 4: Scaling-Shifting Integer Firing Mechanism.

Efficient Integer Firing. SSF adopts single-step integer activation during training, and multi-time-step binary activation during inference as shown in Figure 4. During training, the $S[t]$ in Eq.(17) are integers $\in [-\bar{U}, \bar{U}]$, $\bar{U} = \lfloor\frac{U}{V^{th}}\rfloor$. During inference, to retain the advantage of low energy consumption, SSF employs a $\bar{U}$ time steps binary firing $S[t, 1:\bar{U}] \in \{-1, 0\}$ or $\{0, 1\}$. SSF satisfied:

$$S[t] = \sum_{u=1}^{\bar{U}} S[t, u] \tag{18}$$

Therefore, taking layer l as an example, it is easy to prove the equivalence of training and inference:

$$W^l S[t] = W^l \sum_{u=1}^{\bar{U}} S[t, u] = \sum_{u=1}^{\bar{U}} W^l S[t, u] \tag{19}$$

where W^l is the model weight of layer l.

5 Experiments

In this section, we demonstrate the modeling capability of MMDEND on general sequences in 5.1, its high expressiveness on long sequence language modeling tasks in 5.2 and large-scale language modeling tasks in 5.3. In 5.4, we analyze MMDEND's efficiency in energy consumption and training. Additionally, we validate the effectiveness of each component of MMDEND in 5.5. The experimental setup is detailed in the Appendix A, B, C.

5.1 General Sequence Modeling

To demonstrate the versatility of MMDEND, as shown in Table **??**, we compare it with expressive

Dataset	Timesteps	Approach	Parallel Training	Parameters	Accuracy
S-Cifar10 & S-Cifar100	32	LIF	N	0.51M	81.50% / 55.45%
		LIF wo reset	N	0.51M	79.50% / 53.33%
		GLIF (Yao et al., 2022)	N	0.51M	83.66% / 58.92%
		KLIF (Jiang and Zhang, 2023)	N	0.51M	83.26% / 57.37%
		SPSN (Fang et al., 2024)	Y	0.51M	86.70% / 62.11%
		masked PSN (Fang et al., 2024)	Y	0.52M	85.81% / 60.69%
		PSN (Fang et al., 2024)	Y	0.52M	88.45% / 62.21%
		PMSN (Chen et al., 2024)	Y	0.54M	90.97% / 66.08%
		MMDEND (Ours)	**Y**	**0.51M**	**92.71% / 67.65%**
SSC	250	SRNN (Cramer et al., 2020a)	N	0.11M	50.90%
		TC-LIF-FF (Zhang et al., 2024)	N	0.11M	63.46%
		TC-LIF-RNN (Zhang et al., 2024)	N	0.11M	61.09%
		ALIF (Yin et al., 2021)	N	0.73M	74.20%
		PSN (Fang et al., 2024)*	Y	0.32M	43.71%
		masked PSN (Fang et al., 2024)*	Y	0.32M	68.04%
		SPSN (Fang et al., 2024)*	Y	0.13M	71.50%
		MMDEND (Ours)	**Y**	**0.13M**	**75.63%**

* Our reproduced results based on publicly available codebases

Table 1: Comparison of Performance on General Sequential Tasks.

spiking neurons on spatial-temporal and speech tasks. For the spatial-temporal tasks, we use the column-by-column mode of the S-CIFAR10 and S-CIFAR100 as (Fang et al., 2024). For the speech tasks, we experiment on the spike speech benchmark SSC (Cramer et al., 2020b). Compared to serial neurons, parallel neurons exhibit significant performance advantages. Moreover, MMDEND outperforms the SOTA PMSN by 1.74% and 1.57% in the spatial-temporal tasks with fewer parameters, indicating that MMDEND has better general sequence modeling capabilities.

5.2 Long Sequence Language Modeling

To validate the effectiveness of MMDEND in long sequence modeling, we combine parallel neurons with the S4 model and compare it on the classic long sequence benchmark Long Range Arena (LRA). The subtask lengths in LRA range from 1k to 4k. As shown in Table 2, MMDEND outperforms the baselines on all the long sequence tasks, with an average improvement of 10% over SPSN and at least 22.5% over MPSN and PSN. It is worth to note that MMDEND can effectively handle **ultra-long sequence tasks** like PATHX, while both MPSN and SPSN failed on the PATHX task.

5.3 Large Scale Language Modeling

To substantiate the efficacy of MMDEND on large-scale language models and extensive datasets, we integrated parallel spiking neurons with the 350M GLA (Yang et al., 2023), conducting pre-training on 1B tokens from the Pile (Gao et al., 2020) dataset. We report our findings on the widely recognized LLM evaluation benchmark CommonSense Reasoning (Davis and Marcus, 2015). Due to the substantial additional parameters introduced by PSN, we compared MMDEND with SPSN and MPSN. As delineated in Table 3, MMDEND shows superior adaptability for large-scale tasks.

5.4 Energy and Training Efficiency Analysis

Energy efficiency Analysis. Introducing more complex temporal dynamics in neurons may raise concerns about increased energy consumption. To address this, we compare MMDEND with the PSN family and report energy consumption across varying numbers of compartments and branches. The energy consumption calculation formula is:

$$E_{MAC} * Flops_{neu} + T * R * E_{AC} * Flops_{layer}$$

where $Flops_{neu}$ is the Flops of spiking neuron, $Flops_{layer}$ is the Flops of a fully connected layer, T is the length of sequence, R is the firing rate. The

Architecture	AAN	CIFAR	IMDB	PATHFINDER	LISTOPS	PATHX	AVG
S4-PSN	0.834	0.787	0.633	0.658	0.399	0.507	0.636
S4-MPSN	0.809	0.787	0.672	0.812	0.390	0.502	0.662
S4-SPSN	0.864	0.856	0.857	0.926	0.568	0.503	0.762
S4-MMDEND	**0.900**	**0.878**	**0.886**	**0.943**	**0.599**	**0.963**	**0.861**

Table 2: Long Sequence Moding Experiments on Long Range Arena Benchmark.

	LOGIQA	WSC273	BOOLQ	PIQA	HS	WG	ARC-easy	OBQA	AVG
GLA-MPSN	**0.238**	0.490	0.378	0.527	0.258	**0.509**	0.255	0.254	0.363
GLA-SPSN	0.236	0.494	0.435	0.535	0.258	0.485	0.292	0.248	0.373
GLA-MMDEND	0.227	**0.513**	**0.475**	**0.540**	**0.260**	0.487	**0.292**	**0.254**	**0.381**

Table 3: Large Scale Experiments on CommonSense Reasoning Benchmark.

MODEL	**Flops**
PSN	DT^2
SPSN/MPSN	DWT
MMDEND	dend: $\sum_{i=1}^{n} f_i i D(3K+1)T$ soma: $2\sum_{i}^{n} f_i i DT + 3DT$
Fully Connected	D^2T

Table 4: Flops of spiking neurons and layers. D is the hidden dimension, T is the sequence length, W is the window length, K is the number of compartments, n is the number of branches, f_i is the portion of i branches.

detailed calculation of FLOPs is shown in Table 4. $E_{add} = 0.9pJ$ and $E_{mac} = 4.6pJ$ are the energy consumption of add and MAC operations at 45nm process nodes for full precision (FP32) SynOps.

As shown in Figure 5, due to the grouping mechanism, the energy consumption of MMDEND does not increase significantly as the number of branches and compartments increases, and remains much lower than that of PSN families. Compared to MMDEND with binary firing, the energy consumption of MMDEND with SSF shows only a slight increase. Furthermore, when compared to the version without a grouping mechanism, MMDEND saves about 30% in energy, with savings growing as the number of branches increases.

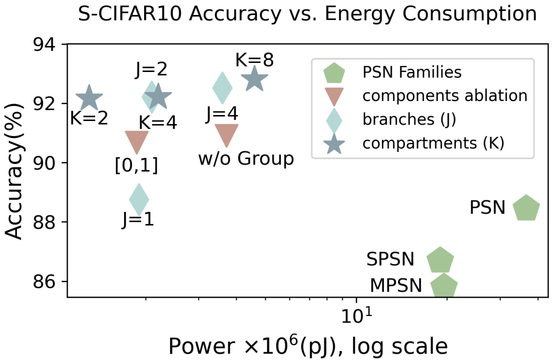


Figure 5: Energy Consumption and Performance on S-CIFAR10.

Training Efficiency Analysis. MMDEND is more suitable for long sequences and large-scale tasks compared to vanilla spiking neurons, not only because of its long-term dependent temporal dynamics but also due to its training efficiency. In Table 5, we compare the time cost of MMDEND and LIF as sequence length increases. Unlike LIF, whose time cost grows significantly with longer sequences, MMDEND's time cost remains stable regardless of sequence length.

Time (ms)	L128	L256	L512	L1024
LIF	10.98	21.41	44.67	83.56
MMDEND	0.78	0.77	0.78	0.77

Table 5: Forward time comparison under different sequence lengths.

5.5 Component Analysis of MMDEND

To verify the effectiveness of each component of MMDEND, we conducted ablation and visualization experiments on the S-CIFAR10 dataset in this subsection.

Branch and Compartment. As shown on the right part of Figure 6, we exhibit the performance variation from a single branch to 8 branches. It is evident that as the number of branches increases, the performance improves. Notably, there is a significant performance improvement when increasing from a single branch to two branches. Similarly,

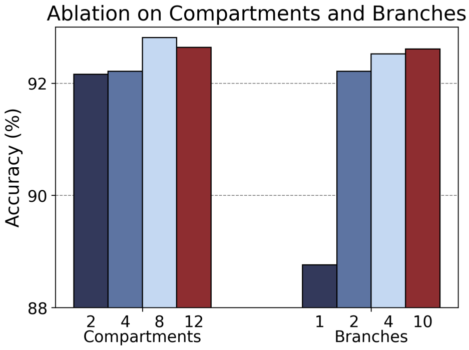


Figure 6: Ablation Study on Compartments and Branches.

in the compartments ablation experiments, performance improves with more compartments, though overly large compartments can hinder MMDEND's expression. Therefore, it is essential to choose the number of branches and compartments according to the task complexity.

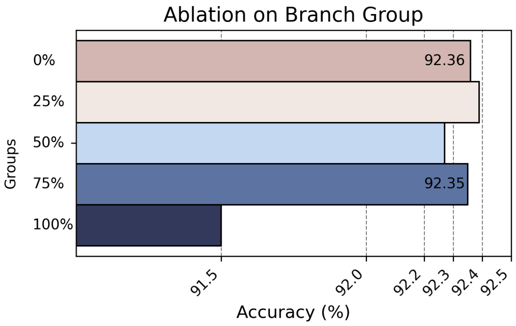


Figure 7: Ablation Study on Branch Group.

Group Dendrite Branch. As shown in Figure 7, the percentages represent the proportion of each channel sharing branch. An interesting observation is that sharing branches from 0% to 25% improves performance, but further increasing the shared proportion gradually decreases performance, indicating that the multi-branch proportion can be adjusted based on the task difficulty, and reducing information redundancy.

MMDEND-SSF Variants	Accuracy
baseline (ranging from $[-4, 4]$)	92.21
w/o scaling and translation	91.34
w/o scaling and translation & integer	90.67
Ranging from $[-1, 1]$ *(i.e., ternary spiking)*	91.04
Ranging from $[-3, 3]$	92.68

Table 6: Ablation on SSF Mechanism.

Scaling-Shifting Integer Firing. To investigate the impact of the translation-scaling mechanism, integer firing, and the firing range on performance within the SSF mechanism, we present the ablation results in Table 6. Removing the translation-scaling coefficients from the SSF mechanism resulted in a 0.87% decrease in accuracy. Furthermore, replacing integer firing with binary firing (i.e. 0-1 firing without reset), led to an additional 0.67% drop in performance. We also observed that as the firing range expanded from $[-1, 1]$ to $[-3, 3]$, performance gradually improved, but it slightly declined when the range was extended to $[-4, 4]$.

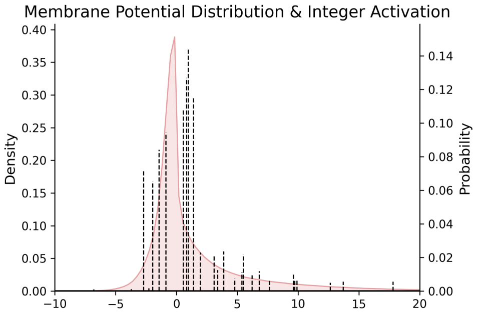


Figure 8: Visualization of long-tail soma membrane potential and spike activations.

Membrane Potential Distribution and SSF. To demonstrate that the SSF mechanism effectively addresses the issue of binary spike activation failing to fit asymmetrical long-tail distributions, we present the membrane potential distribution and SSF spike firing distribution in the soma, as shown in Figure 8. The SSF effectively covers the asymmetrical long-tail membrane potential distributions, preserving the completeness of the information.

6 Conclusion

In this work, we propose MMDEND to overcome the challenges that traditional spiking neurons face in long-sequence tasks. MMDEND is designed based on the circuitry of dendritic neurons derived from cable theory achieving long-term dependent temporal dynamics. We introduce the SSF mechanism, which dynamically adapts to long-tail membrane potential distributions by adjusting scale and shift parameters. SSF balances efficiency and low energy consumption by using integer activation during training and event-driven operations during inference. To achieve efficient parallelization, we model the dendrites using SSM and eliminate the nonlinear firing in the soma. Results show that

MMDEND outperforms all the serial and parallel spiking neuron baselines on general sequence, long-sequence, and large-scale tasks, proving the effectiveness and efficiency of dendritic dynamics.

7 Limitation

Since the modeling process of MMDEND starts from a single branch and extends to multiple branches, although we use grouping to prevent an exponential increase in computation, the multi-branch structure still inevitably leads to some increase in computational cost and energy consumption. We look forward to future work that will directly model the multi-branch, multi-compartment structure to eliminate this limitation.

8 Acknowledgments

This work was partially supported by National Key Research and Development Program of China (2023YFF1204200), CAS Project for Young Scientists in Basic Research (YSBR-116), National Natural Science Foundation of China (62325603, 62236009, U22A20103), Beijing Science and Technology Plan (Z241100004224011).

References

Xinyi Chen, Jibin Wu, Chenxiang Ma, Yinsong Yan, Yujie Wu, and Kay Chen Tan. 2024. Pmsn: A parallel multi-compartment spiking neuron for multi-scale temporal processing. *arXiv preprint arXiv:2408.14917*.

Yuwen Chen and Jiang Liu. 2022. Polynomial dendritic neural networks. *Neural Computing and Applications*, 34(14):11571–11588.

Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. 2020a. 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.

Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. 2020b. 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.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

Raymond A. DeCarlo. 1989. *Linear Systems: A State Variable Approach with Numerical Implementation*. Prentice-Hall, Inc.

Chaoteng Duan, Jianhao Ding, Shiyan Chen, Zhaofei Yu, and Tiejun Huang. 2022. Temporal effective batch normalization in spiking neural networks. *Advances in Neural Information Processing Systems*, 35:34377–34390.

Erol Egrioglu and Eren Bas. 2024. A new deep neural network for forecasting: Deep dendritic artificial neural network. *Artificial Intelligence Review*, 57(7):171.

Erol Egrioglu, Eren Baş, and Mu-Yen Chen. 2022. Recurrent dendritic neuron model artificial neural network for time series forecasting. *Information Sciences*, 607:572–584.

Wei Fang, Zhaofei Yu, Zhaokun Zhou, Ding Chen, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. 2024. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. *Advances in Neural Information Processing Systems*, 36.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Albert Gu, Karan Goel, and Christopher Re. 2021. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*.

Yufei Guo, Yuanpei Chen, Xiaode Liu, Weihang Peng, Yuhan Zhang, Xuhui Huang, and Zhe Ma. 2024. Ternary spike: Learning ternary spikes for spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12244–12252.

Junkai Ji, Cheng Tang, Jiajun Zhao, Zheng Tang, and Yuki Todo. 2022. A survey on dendritic neuron model: Mechanisms, algorithms and practical applications. *Neurocomputing*, 489:390–406.

Chunming Jiang and Yilei Zhang. 2023. Klif: An optimized spiking neuron unit for tuning surrogate gradient slope and membrane potential. *arXiv preprint arXiv:2302.09238*.

Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems.

Robert Legenstein and Wolfgang Maass. 2011. Branch-specific plasticity enables self-organization of nonlinear computation in single neurons. *Journal of Neuroscience*, 31(30):10787–10802.

Tianlong Li, Wenhao Liu, Changze Lv, Jianhan Xu, Cenyuan Zhang, Muling Wu, Xiaoqing Zheng, and Xuanjing Huang. 2023. Spikeclip: A contrastive language-image pretrained spiking neural network. *arXiv preprint arXiv:2310.06488*.

Michael London and Michael Häusser. 2005. Dendritic computation. *Annu. Rev. Neurosci.*, 28(1):503–532.

Xinhao Luo et al. 2024. Integer-valued training and spike-driven inference spiking neural network for high-performance and energy-efficient object detection. *arXiv preprint arXiv:2407.20708*.

Changze Lv, Tianlong Li, Jianhan Xu, Chenxi Gu, Zixuan Ling, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2023. Spikebert: A language spikformer trained with two-stage knowledge distillation from bert. *arXiv preprint arXiv:2308.15122*.

Wolfgang Maass. 1997. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

RKPMTKR Rajagopal, R Karthick, P Meenalochini, and T Kalaichelvi. 2023. Deep convolutional spiking neural network optimized with arithmetic optimization algorithm for lung disease detection using chest x-ray images. *Biomedical Signal Processing and Control*, 79:104197.

Nelson Spruston. 2008. Pyramidal neurons: dendritic structure and synaptic integration. *Nature Reviews Neuroscience*, 9(3):206–221.

Matei-Ioan Stan and Oliver Rhodes. 2024. Learning long sequences in spiking neural networks. *Scientific Reports*, 14(1):21957.

Congyi Sun, Qinyu Chen, Yuxiang Fu, and Li Li. 2022. Deep spiking neural network with ternary spikes. In *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 251–254. IEEE.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*.

Kexin Wang, Jiahong Zhang, Yong Ren, Man Yao, Di Shang, Bo Xu, and Guoqi Li. 2024. Spikevoice: High-quality text-to-speech via efficient spiking neural network. *arXiv preprint arXiv:2408.00788*.

Xundong Wu, Pengfei Zhao, Zilin Yu, Lei Ma, Ka-Wa Yip, Huajin Tang, Gang Pan, and Tiejun Huang. 2023. Mitigating communication costs in neural networks: The role of dendritic nonlinearity. *arXiv preprint arXiv:2306.11950*.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. 2018. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12:331.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. 2023. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*.

Man Yao, JiaKui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo XU, and Guoqi Li. 2024. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In *The Twelfth International Conference on Learning Representations*.

Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. 2022. Glif: A unified gated leaky integrate-and-fire neuron for spiking neural networks. *Advances in Neural Information Processing Systems*, 35:32160–32171.

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

Shimin Zhang, Qu Yang, Chenxiang Ma, Jibin Wu, Haizhou Li, and Kay Chen Tan. 2024. Tc-lif: A two-compartment spiking neuron model for long-term sequential modelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16838–16847.

Jianqing Zhao, Xiaohu Zhang, Jiawei Yan, Xiaolei Qiu, Xia Yao, Yongchao Tian, Yan Zhu, and Weixing Cao. 2021. A wheat spike detection method in uav images based on improved yolov5. *Remote Sensing*, 13(16):3095.

Hanle Zheng, Zhong Zheng, Rui Hu, Bo Xiao, Yujie Wu, Fangwen Yu, Xue Liu, Guoqi Li, and Lei Deng. 2024. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15(1):277.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, YAN Shuicheng, Yonghong Tian, and Li Yuan. 2022. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K Eshraghian. 2023. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*.

A Computing Setting

For the Highly Expressive Language Modeling experiment, we used 8 A100 GPUs. For all other experiments, we completed them on a single A100 GPU.

B Dataset

The datasets in this work are as follows:

Wikitext-103 is a widely used NLP dataset that includes over 100000 Wikipedia articles, totaling approximately 103 million words. In our experiments, we follow the setup described in (Merity et al., 2016), where the training set, validation set, and test set consist of 28475, 60, and 60 articles, respectively.

Long Range Arena (LRA) is a dataset and benchmark designed to evaluate the ability of models to handle long sequences (Tay et al., 2021). LRA aims to test model performance in managing long-range dependencies and includes tasks such as text classification, image classification, retrieval, list operations, and pathfinding. In our experiments, the sequence length distribution ranges from 1K to 4K.

S-Cifar10 and S-Cifar100 are image sequence classification tasks derived from CIFAR-10 and CIFAR-100. In this task, each image with size 32×32 is segmented into a column-by-column sequence from left to right.

Spiking Speech Command (SSC) is a speech recognition dataset specifically designed for the neuromorphic computing field. Unlike traditional speech datasets, the SSC dataset uses spike encoding to convert audio signals into spike sequences. Each spike input consists of 700 channels, encompassing 35 different word categories.

C Experiment Setting

In this subsection, we will introduce the model architecture and hyperparameter settings in each experiment.

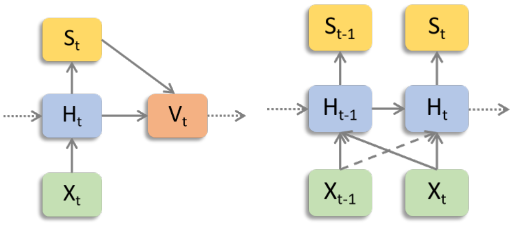
Model Architectures are shown as follows:

- For the Wikitext task, we followed the model architecture in (Gu et al., 2021), which consists of 16 transformer layers with a hidden size of 512. We replaced the activation layer in each transformer block with a 2-branch, 4-compartment MMDEND.

- For the LRA task, we followed the model architecture described in (Gu et al., 2021), which consists of 6 S4 blocks. We replaced the activation layer in each block with a 2-branch, 4-compartment MMDEND.

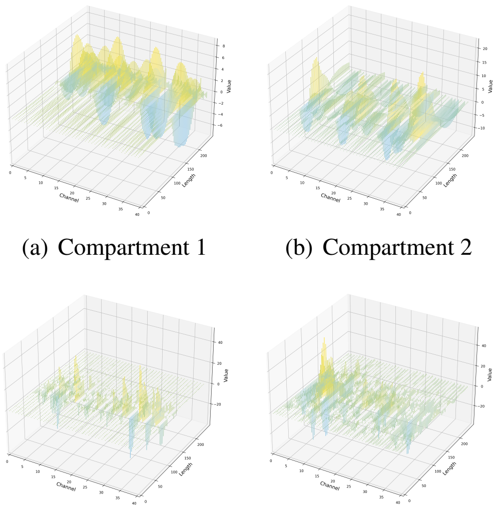
- For S-CIFAR10 and S-CIFAR100, we used the same model architecture setup as (Fang et al., 2024), which includes one convolutional layer and two linear layers. Sequence modeling between layers is performed using MMDEND. S-CIFAR10 utilizes a MMDEND with 8 compartments and 6 branches, while S-CIFAR100 employs a MMDEND with 8 compartments and 4 branches.

- For the SSC task, we used a four-layer linear network with a hidden size of 128. Sequence modeling between layers was performed using a MMDEND with 8 compartments and 4 branches.



(a) serial (b) parallel

Figure 9: serial and parallel spiking neuron.



(a) Compartment 1 (b) Compartment 2

(c) Compartment 3 (d) Compartment 4

Figure 10: Visualization of dendritic compartment membrane potential information patterns on the SSC dataset.

Hyperparameters Our hyperparameter Settings in each experiment are shown in the Table 7.

D Supplementary Preliminaries

Parallel spiking neuron When the nonlinearity is removed from Eq. (3), such that $V_t = H_t$, the membrane potentials at each time step $H = \{H_0, H_1, \dots, H_{T-1}\}$ as shown in Figure 9(b) can be calculated in parallel as (Fang et al., 2024):

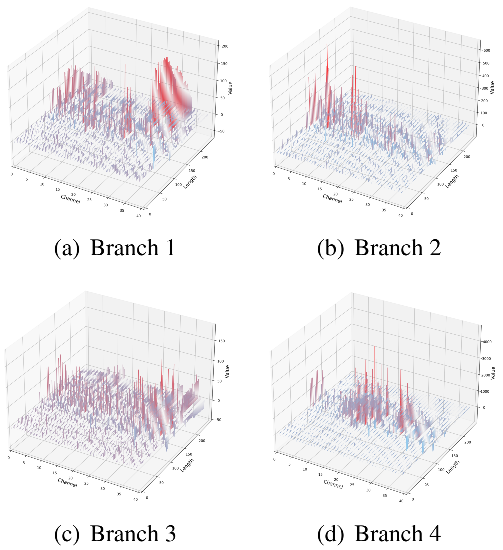
$$H_t = \frac{1}{\tau}\sum_{i=0}^{t}(1-\frac{1}{\tau})^{t-i} \cdot X_t = \sum_{i=0}^{t} W_{t,i} X_t \quad (20)$$

where $W_{t,i} = \frac{1}{\tau}(1-\frac{1}{\tau})^{t-i}$, which determines the temporal dynamics of the parallel neurons. Eq.

Table 7: Long Sequence Moding Experiments on Long Range Arena Benchmark.

Dataset	Learning Rate	Weight Decay	Batchsize	Epoch	Compartment	Branch
AAN	0.01	0.05	64	20	4	2
CIFAR	0.01	0.05	50	200	4	2
IMDB	0.01	0.05	16	32	4	2
PATHFINDER	0.004	0.05	64	200	4	2
LISTOPS	0.01	0.05	32	40	4	2
PATHX	0.001	0.05	16	50	4	2
Wikitext-103	5e-4	0.0	32	40	4	2
S-Cifar10	0.1	0.0	128	256	8	6
S-Cifar100	0.1	0.0	128	256	8	4
SSC	0.01	0.0	32	200	8	4

(20) can be efficiently implemented using the FFT convolution.



(a) Branch 1

(b) Branch 2

(c) Branch 3

(d) Branch 4

Figure 11: Visualization of dendritic branch membrane potential information patterns on the SSC dataset.

E Information Patterns of Dendritic Compartments and Branches

In this subsection, we visualize the membrane potentials across multiple branches to demonstrate their functions. Additionally, we exhibit the consistency between the soma membrane potential and the spike distribution under the SSF mechanism. As shown in Figure 11, we present the membrane potential distribution of MMDEND across four branches on the SSC dataset. We observed that different branches exhibit channel-specific characteristics. Specifically, Figures 11(a), 11(b), and 11(d) demonstrate concentrated responses to the anterior, posterior, and central segments of the channel, respectively, while Figure 11(c) shows a uniform response across the entire channel. Additionally, we also found that different compartments exhibit sequence-specific characteristics, which can be found in the Appendix E.

Unlike the information distribution observed in dendritic branches, the membrane potential distribution across dendritic compartments, as shown in Figure 10, exhibits different response patterns to various positions within the input sequence. For instance, Figures 10(a) and 10(b) show responses concentrated in the latter and middle-latter parts of the sequence, while Figures 10(c) and 10(d) demonstrate concentrated responses in the middle and early-middle parts of the sequence.