SPIKEWEAVE: SPIKING SELF-ATTENTION ACROSS THOUSANDS OF TIMESTEPS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks promise energy-efficient sequence modeling through sparse, event-driven computation. However, existing spiking self-attention architectures rely on direct encoding, which repeatedly presents a static input over multiple timesteps. This introduces a simulation dimension decoupled from the input sequence and multiplies computational cost. Neuronal dynamics are thus spent on re-encoding a static input rather than on modeling temporal structure. Whether intrinsic neuronal dynamics can instead drive long-range sequence modeling remains largely unexplored. We propose SpikeWeave, a temporal spiking self-attention architecture that aligns each sequence position with a single neuronal time step. The network thus consumes successive elements natively through its own dynamics. To retain context over long horizons, we introduce a multiscale dendritic exponential moving average (DEMA) mechanism that maintains parallel leaky integrations at distinct decay rates. A separate somatic compartment converts the dendritic readout into spikes for query and key generation. Crucially, somatic resets leave dendritic states intact, decoupling spike generation from memory retention. For scalable training and state memory conservation, we develop a chunkwise spiking self-attention scheme in which dendritic states carry context across chunk boundaries. We evaluate SpikeWeave on long-range sequence modeling benchmarks, where it delivers strong performance among spiking architectures while preserving sparse, event-driven computation. We believe SpikeWeave takes a step toward scalable, energy-efficient sequence modeling with spiking neural networks.

1 INTRODUCTION

Spiking Neural Networks (SNNs) offer a brain-inspired computing paradigm based on discrete spike communication (Maass, 1997). Neuronal dynamics (Gerstner & Kistler, 2002; Izhikevich, 2003) and sparse spike communication (Davies et al., 2018) enable a favorable balance between temporal processing and energy-efficient event-driven computation. Recent efforts integrate self-attention with SNNs to form Spiking Transformers, bringing content-based interactions into this paradigm. Although most work focuses on visual recognition (Zhou et al., 2023; Yao et al., 2023), applications now include language (Zhu et al., 2024), time-series forecasting (Lv et al., 2024; Wu et al., 2025), and sequential decision making (Huang et al., 2025). Nevertheless, long-range modeling with Spiking Transformer remains comparatively underexplored. A central challenge is that the neuronal state must simultaneously track local dynamics and retain long-past information.

Inside a Spiking Transformer, spiking self-attention (SSA) forms query and key spikes through successive neuronal updates. Each update can consume a new sequence element or another presentation of the same input. Under direct encoding, commonly used in Spiking Transformers, these updates replay a fixed input for T timesteps (Figure 1a). Intuitively, this approach works well for static data. Spikformer handles static inputs by repeating a single frame across timesteps (Zhou et al., 2023). The same convention also extends to sequential inputs. Decision SpikeFormer replays an embedded trajectory along a separate timestep axis and applies temporal attention across these copies (Huang et al., 2025). Neuronal dynamics thus evolve in simulation time, while sequence positions occupy a separate token dimension that no single neuron traverses. For a sequence of length N, this requires $O(TN)$ neuronal updates, of which all but N re-encode an input already presented.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

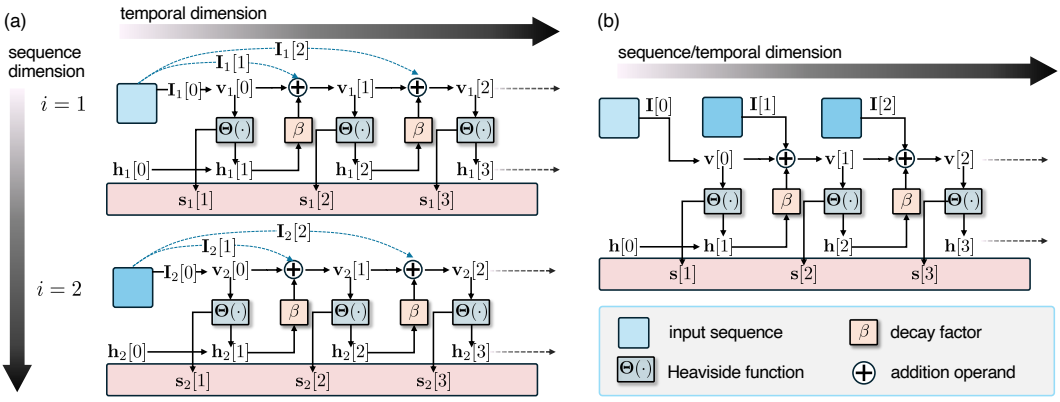


Figure 1: Simulation-time replication versus sequence-time binding. (a) Conventional Spiking Transformers with direct encoding treat sequence position and timestep as separate dimensions. At each position i, the same input representation is repeatedly presented across T timesteps to support the neuronal state evolution. (b) SpikeWeave binds neuronal updates to sequence positions: each new input triggers one update, and the resulting state is carried to the next position. For a sequence of length N and a fixed network size, these organizations require $O(TN)$ and $O(N)$ neuronal updates, respectively. Here, I denotes the input, v and h denote the pre- and post-reset membrane potentials, s denotes the emitted spike, β is the decay factor, and $\Theta(\cdot)$ is the Heaviside function.

The alternative is to operate in sequence time, where each arriving position updates the neuronal state (Figure 1b). Sequence modality interleaves both short fluctuations and long dependencies, requiring the state to encode multiple timescales simultaneously. However, a conflict arises when the same state responsible for temporal integration is also used to generate spikes. In a standard Leaky Integrate-and-Fire (LIF) neuron, the same membrane state integrates inputs and determines spike emission. When used to generate SSA queries and keys, each spike therefore partially or entirely erases the temporal trace, depending on the reset rule. Dendritic heterogeneity can decouple these roles, preserving dendritic traces across somatic resets while supporting multiple timescales (Zheng et al., 2024). Nonetheless, existing spiking sequence models circumvent this coupling by adopting linear recurrent memories. For instance, SpikeGPT (Zhu et al., 2024) and SpikingSSMs (Shen et al., 2025a) pursue long contexts through linear recurrence backbones. How to retain multiscale context while generating query and key spikes for SSA remains an open question.

We propose SpikeWeave, a temporal Spiking Transformer that advances its state once per sequence position, requiring $O(N)$ neuronal updates. Its multiscale dendritic exponential moving average (DEMA) layer assigns each feature channel multiple traces with distinct learnable time constants. A separate somatic readout integrates these traces to generate query and key spikes. Because somatic resets leave dendritic states intact, each query and key carries a multiscale summary of preceding inputs without erasing the stored history.

For sequences spanning thousands of positions, full self-attention incurs quadratic cost in sequence length. SpikeWeave controls this cost by partitioning the sequence into fixed-length chunks and restricting attention to positions within each chunk; with a fixed chunk size, the resulting attention computation scales linearly with sequence length. A fixed-dimensional dendritic state is carried from one chunk to the next and updated with the current chunk's inputs. Thus, context from earlier positions reaches later chunks through this recurrent summary.

We demonstrate competitive accuracy across sequential image, spiking speech, and long-range sequence benchmarks. Futheremore, we examine the contributions of dendritic memory and chunkwise attention through controlled experiments. These results highlight the strengths of SpikeWeave and demonstrate the effectiveness of its memory–attention design for sequence modeling.

Our contributions are summarized as follows:

- We introduce a temporal spiking architecture that processes inputs directly in sequence time, requiring $O(N)$ neuronal updates compared with $O(TN)$ under T-step direct encoding.

- We couple multiscale dendritic memory with chunkwise causal SSA. Dendritic traces preserve history across somatic resets and chunk boundaries, supporting long-range context with linear attention cost.
- We evaluate SpikeWeave on S/PS-MNIST, SHD, and five LRA tasks. It achieves 98.44% on PS-MNIST, 96.43% on SHD, and 82.40% average LRA accuracy, including the highest Text accuracy among the compared methods (87.64%).

2 Related Work

Spiking Transformer Architectures. Spiking Transformers adapt content-based interaction to event-driven representations. Spikformer (Zhou et al., 2023) introduced spiking self-attention with spike-form query (Q), key (K), and value (V) representations without softmax operation. While Spike-driven Transformer (Yao et al., 2023) replaces conventional self-attention with spike masking and sparse accumulation. Later work explores distinct design axes. Spike-driven Transformer V2 (Yao et al., 2024) develops the Meta-SpikeFormer architecture, while V3 (Yao et al., 2025) introduces Spike Firing Approximation training for scaling. Saccadic SSA (Wang et al., 2025b) combines spike-distribution relevance with saccadic temporal selection, Bipolar SSA (Wang et al., 2025a) uses ternary polarity-aware QK interactions with Shiftmax. LRF-Dyn (Zhang et al., 2026) combines local receptive fields with neuronal dynamics. Zhang et al. introduce MLPixer and SRB channel mixers (Zhang et al., 2025b) to enlarge global spatial receptive fields. Despite these advances, most retain a separate timestep axis. Direct encoding replays the same representation at each timestep, leaving neuronal updates decoupled from token order. SpikeWeave instead advances the neuronal state once at each sequence position while persisting long-horizon memory states.

Spiking sequence models. Spiking sequence models pursue long contexts through several mechanisms on various types of modality. SpikeGPT (Zhu et al., 2024) replaces self-attention with an RWKV-inspired spiking recurrent mixer. SpikingSSMs (Shen et al., 2025a) integrate spiking dynamics into SSM backbones and introduce a surrogate dynamic network for parallel training. SpikF (Wu et al., 2025) is an attention-free long-term forecasting model that patches inputs and selects frequency components in the Fourier domain. D-RF (Zhang et al., 2025a) adds multiple frequency-selective dendritic branches and an adaptive somatic threshold. Delay-based methods learn temporal offsets or memory paths. DLTC (Sun et al., 2024) adjusts spike timing through learnable delays and thresholds. DMU (Sun et al., 2025) augments a vanilla RNN with a delay line and delay gates. These works address long-range temporal processing through different architectures, but do not target the specific interface in which a persistent multiscale state influence spike generation in SSA.

Dendritic and multiscale temporal computation. Neuroscience studies show that dendrites can discriminate temporal input order, shape stimulus selectivity, and contribute to linking contextual memories formed close in time (Branco et al., 2010; Smith et al., 2013; Sivyer & Williams, 2013; Sehgal et al., 2025). Related neuromorphic models instantiate these ideas at different levels. Dendrify (Pagkalos et al., 2023) provides reduced compartmental SNN models. DH-SNNs (Zheng et al., 2024) learn heterogeneous dendritic time constants for multiscale dynamics, and DMP-SNN (Sun et al., 2026) introduces a compact slow state alongside fast spiking activity. SpikeWeave combines these ideas through a different interface. Its dendritic traces persist across somatic resets and condition the somatic readout that generates query and key spikes for chunkwise SSA.

3 Preliminaries

3.1 Spiking Self-Attention

Spiking self-attention (SSA) replaces continuous query, key, and value representations with spike variables while retaining content-dependent interactions (Zhou et al., 2023; Yao et al., 2023). Let $\mathbf{Q} \in \mathbb{R}^{L\times d_q}$, $\mathbf{K} \in \mathbb{R}^{L\times d_k}$, and $\mathbf{V} \in \mathbb{R}^{L\times d_v}$ denote the spike representations of a sequence of length L. Unlike conventional attention, SSA typically omits softmax normalization and forms interactions directly from the binary query–key products. With an optional scale η and an attention mask $\mathbf{M}$, its output is

$$\mathbf{Y} = (\eta(\mathbf{Q}\mathbf{K}^\top) \odot \mathbf{M})\mathbf{V}, \tag{1}$$

where $\odot$ denotes element-wise multiplication. For causal sequence modeling, $M_{i\ell} = 1$ only when key position ℓ is no later than query position i. The score matrix therefore preserves signed or binary event interactions without constructing a probability distribution.

3.2 Spiking and Multi-Dendritic Neurons

Spiking neuron. A layer of LIF neurons (Izhikevich, 2003; Gerstner & Kistler, 2002) maintains a vector membrane state that both integrates incoming currents and determines spike emission. A common discrete-time LIF parameterization can be described by the following processes:

$$\mathbf{v}^\ell[t] = \mathbf{h}^\ell[t-1] + f\big(\mathbf{w}^\ell, \mathbf{x}^{\ell-1}[t-1]\big) \qquad \text{(Charging function)}, \tag{2}$$

$$\mathbf{s}^\ell[t] = \Theta\big(\mathbf{v}^\ell[t] - \boldsymbol{\vartheta}^\ell\big) \qquad \text{(Firing function)}, \tag{3}$$

$$\mathbf{h}^\ell[t] = \begin{cases} \boldsymbol{\beta}^\ell \odot \mathbf{v}^\ell[t] - \boldsymbol{\vartheta}^\ell \odot \mathbf{s}^\ell[t], & \text{soft reset}, \\ \mathbf{v}^\ell[t] \odot \big(\mathbf{1} - \mathbf{s}^\ell[t]\big), & \text{hard reset}, \end{cases} \qquad \text{(Leak-and-reset function)}. \tag{4}$$

where $\mathbf{v}^\ell[t]$ and $\mathbf{h}^\ell[t]$ denote the pre-reset and post-reset membrane states, respectively. $\boldsymbol{\beta}^\ell$ is the vector of decay factors, $\boldsymbol{\vartheta}^\ell$ is the vector of thresholds, and $\mathbf{1}$ is an all-ones vector. The function f maps the input through the parameters $\mathbf{w}^\ell$. Θ is the element-wise Heaviside function. When the membrane potential exceeds the threshold, the neuron emits a spike; otherwise, it remains inactive.

Multi-dendritic neuron. A multi-dendritic neuron introduces multiple dendritic compartments or branch states that integrate inputs before somatic spike generation (London & Haeusser, 2005; Poirazi et al., 2003; Pagkalos et al., 2023). For layer ℓ with D feature channels and K dendritic branches, the branch dynamics and somatic input can be written as

$$\mathbf{d}_k^\ell[t] = \boldsymbol{\rho}_k^\ell \odot \mathbf{d}_k^\ell[t-1] + \mathbf{b}_k^\ell \odot \mathbf{x}^{\ell-1}[t-1], \qquad \text{(Branch update)}, \tag{5}$$

$$\mathbf{r}^\ell[t] = \sum_{k=1}^{K} \mathbf{w}_k^\ell \odot \mathbf{d}_k^\ell[t], \qquad \text{(Dendritic readout)}, \tag{6}$$

$$\mathbf{v}^\ell[t] = \mathbf{h}^\ell[t-1] + \mathbf{r}^\ell[t], \qquad \text{(Somatic input)}, \tag{7}$$

where $\mathbf{d}_k^\ell[t] \in \mathbb{R}^D$ is the state of branch $k = 1, \ldots, K.$, $\boldsymbol{\rho}_k^\ell \in (0,1)^D$ is its vector of decay factors, $\mathbf{b}_k^\ell, \mathbf{w}_k^\ell \in \mathbb{R}^D$ are feature-wise input and readout weights, and $\mathbf{r}^\ell[t] \in \mathbb{R}^D$ is the dendritic readout delivered to the soma. The firing and reset functions then follow equation 3–equation 4. When the branch decay vectors differ, the neuron can represent information over multiple temporal scales (Zheng et al., 2024). A two-compartment design can keep the dendritic states separate from the somatic membrane, allowing a model to preserve dendritic traces across a somatic reset (Zheng et al., 2024). SpikeWeave instantiates this separation with its DEMA branches and an independent somatic readout, as described next.

4 Method

In this section, we describe SpikeWeave as a sequence-time Spiking Transformer whose temporal memory and attention interactions are coupled through spikes. It combines multiscale dendritic memory with somatic spike generation and chunkwise causal spiking self-attention (SSA). We first present the overall architecture and the data flow through a layer (§ 4.1). We then introduce the multiscale dendritic exponential moving average (DEMA), detailing its dendritic recurrence and somatic spike generation (§ 4.2). Finally, we describe chunkwise causal SSA and its integration with gated residual updates and a residual spiking feed-forward block and explain how dendritic state is maintained across attention chunks (§ 4.3).

4.1 Overall Architecture

As illustrated in Fig. 2, SpikeWeave consists of an input embedding followed by L stacked blocks that combine multiscale temporal memory with causal SSA within fixed-length chunks. The embedding maps a length-N sequence to $\mathbf{X}^{(0)} \in \mathbb{R}^{N \times D}$. Each block contains a DEMA-based attention sublayer for sequence mixing and a gated spiking feed-forward network (FFN) for channel mixing.

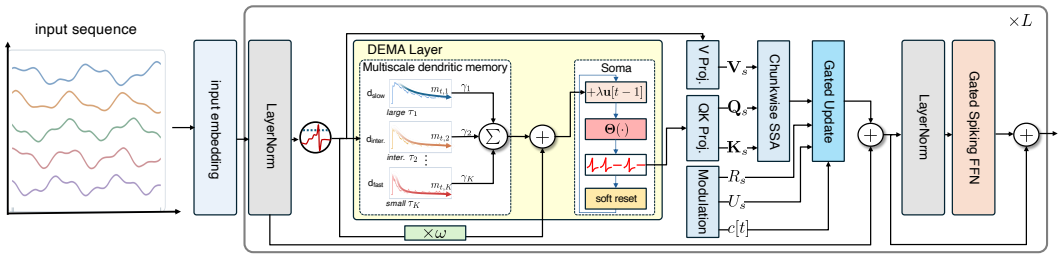


Figure 2: Overall architecture of SpikeWeave. An input embedding is followed by L stacked blocks, each containing a DEMA-based attention sublayer and a gated spiking feed-forward network. Multiscale dendritic states drive a separate spiking soma, whose output supplies the query/key projections and the modulation branch; value events are generated from the input events. The Q/K and V projection modules include linear maps, LayerNorm, and signed LIF neurons. Chunkwise causal SSA is followed by a gated update. Both sublayers use membrane potential residual connections.

Given the input $\mathbf{X}^{(\ell-1)}$ to block ℓ, the two sublayers are written as

$$\begin{aligned}\mathbf{H}^{(\ell)} &= \mathrm{LN}_{\mathrm{attn}}^{(\ell)}\left(\mathbf{X}^{(\ell-1)} + \mathcal{A}^{(\ell)}\left(\mathbf{X}^{(\ell-1)}\right)\right),\\ \mathbf{X}^{(\ell)} &= \mathrm{LN}_{\mathrm{ffn}}^{(\ell)}\left(\mathbf{H}^{(\ell)} + \mathcal{F}^{(\ell)}\left(\mathbf{H}^{(\ell)}\right)\right),\end{aligned} \tag{8}$$

where $\mathcal{A}^{(\ell)}$ comprises input spike generation, DEMA, chunkwise SSA, and the gated update, and $\mathcal{F}^{(\ell)}$ denotes the gated spiking FFN branch. For a single block, we omit the layer index and arrange sequence positions as rows. We use TSN and BSN to denote ternary and binary LIF, respectively.

DEMA-based spiking attention. The attention sublayer first converts the input representation into ternary events and processes them through the DEMA module:

$$\mathbf{E} = \mathrm{TSN}_{\mathrm{in}}(\mathbf{X}), \qquad \mathbf{S} = \mathrm{DEMA}(\mathbf{E}), \tag{9}$$

where $\mathbf{E} \in \{-1, 0, 1\}^{N\times D}$ and $\mathbf{S} \in \{0, 1\}^{N\times D}$. Here, DEMA includes the multiscale dendritic recurrence and its separate spiking soma. Somatic events generate queries and keys, while input events generate values:

$$\mathbf{Q}_s = \mathrm{TSN}_Q\left(\mathrm{LN}_Q\left(\mathbf{S}\mathbf{W}_Q^\top\right)\right), \mathbf{K}_s = \mathrm{TSN}_K\left(\mathrm{LN}_K\left(\mathbf{S}\mathbf{W}_K^\top\right)\right), \mathbf{V}_s = \mathrm{TSN}_V\left(\mathrm{LN}_V\left(\mathbf{E}\mathbf{W}_V^\top\right)\right), \tag{10}$$

where $\overline{\mathbf{W}}_Q$ and $\overline{\mathbf{W}}_K$ are the projection weights. For chunk b, causal SSA computes

$$\mathbf{Y}_b = \left[\alpha_{\mathrm{SSA}}\left(\mathbf{Q}_{s,b}\mathbf{K}_{s,b}^\top\right) \odot \mathbf{P}_b\right]\mathbf{V}_{s,b}, \tag{11}$$

where $\mathbf{P}_b$ is a binary mask that allows valid causal connections and α_{SSA} is a fixed scale This operation preserves signed query–key correlations without softmax. The chunk outputs are reassembled in sequence order and normalized token-wise using RMSNorm.

Spike-driven modulation and gated update. The modulation branch maps the same somatic events to a ternary context gate, a binary update gate, and a continuous candidate:

$$\mathbf{R}_s = \mathrm{TSN}_r\left(\mathbf{S}\mathbf{W}_r^\top\right), \qquad \mathbf{U}_s = \mathrm{BSN}_u\left(\mathbf{S}\mathbf{W}_u^\top\right), \qquad \mathbf{C} = \mathbf{S}\mathbf{W}_c^\top. \tag{12}$$

The t-th row of $\mathbf{C}$ corresponds to $\mathbf{c}_t^\top$ in Fig. 2. These signals form the attention update as

$$\mathbf{Z} = \mathrm{LN}_o\left(\mathrm{TSN}_{\mathrm{ctx}}\left(\widehat{\mathbf{Y}} \odot \mathbf{R}_s\right)\mathbf{W}_o^\top\right), \qquad \mathcal{A}(\mathbf{X}) = \mathbf{U}_s \odot (\mathbf{C} + \mathbf{Z}). \tag{13}$$

Thus, $\mathbf{R}_s$ modulates the context before spike generation, whereas $\mathbf{U}_s$ controls the combined candidate and projected context. The resulting update enters the first residual equation in Eq. equation 8.

Gated spiking FFN. The spiking FFN uses ternary input events to generate continuous content and a binary gate:

$$\begin{aligned}\mathbf{E}_f &= \mathrm{TSN}_f(\mathbf{H}), \mathbf{G} = \mathrm{BSN}_g\left(\mathrm{LN}_g\left(\mathbf{E}_f\mathbf{W}_g^\top\right)\right),\\ \mathcal{F}(\mathbf{H}) &= \left[\left(\mathbf{E}_f\mathbf{W}_1^\top\right) \odot \mathbf{G}\right]\mathbf{W}_2^\top.\end{aligned} \tag{14}$$

The content branch $\mathbf{E}_f\mathbf{W}_1^\top$ remains continuous, and $\mathbf{G}$ selects its components before the output projection. The second residual equation in Eq. equation 8 then adds $\mathbf{H}$ and performs LayerNorm.

4.2 MULTISCALE DENDRITIC EXPONENTIAL MOVING AVERAGE (DEMA)

DEMA maps the ternary input events $\mathbf{E}$ to binary somatic events $\mathbf{S}$ through K continuous dendritic branches per feature and a separate soma. The branches retain input history, and their weighted readout drives somatic spike generation.

Dendritic integration and readout. We omit the layer index and consider an unpadded sequence. At position t, $\mathbf{e}_t$ and $\mathbf{s}_t$ denote the input and somatic event vectors, respectively. Let $\mathbf{m}_t^{(k)} \in \mathbb{R}^D$ denote the state of dendritic branch k, whose d-th component is $m_{t,d}^{(k)}$. The complete state bank is $\mathbf{M}_t \in \mathbb{R}^{D\times K}$. For each branch, the input and retention coefficients are

$$\mathbf{p}^{(k)} = \sigma\left(\boldsymbol{\delta}^{(k)}\right), \qquad \boldsymbol{\rho}^{(k)} = \mathbf{1}_D - \mathbf{p}^{(k)} \odot \sigma\left(\boldsymbol{\alpha}^{(k)}\right), \tag{15}$$

where $\boldsymbol{\delta}^{(k)}, \boldsymbol{\alpha}^{(k)} \in \mathbb{R}^D$ are learnable parameter vectors. The branch state evolves as

$$\mathbf{m}_t^{(k)} = \boldsymbol{\rho}^{(k)} \odot \mathbf{m}_{t-1}^{(k)} + \left(\mathbf{p}^{(k)} \odot \boldsymbol{\beta}^{(k)}\right) \odot \mathbf{e}_t, \qquad \mathbf{m}_0^{(k)} = \mathbf{0}, \tag{16}$$

where $\boldsymbol{\beta}^{(k)} \in \mathbb{R}^D$ supplies the input gains. The somatic input current combines the branch states with an instantaneous input contribution:

$$\mathbf{r}_t = \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \boldsymbol{\gamma}^{(k)} \odot \mathbf{m}_t^{(k)} + \boldsymbol{\omega} \odot \mathbf{e}_t, \tag{17}$$

where $\boldsymbol{\gamma}^{(k)}, \boldsymbol{\omega} \in \mathbb{R}^D$ are learnable readout and direct-input weights.

Multiscale memory and EMA interpretation. For channel d in branch k, define the effective dendritic time constant $\tau_d^{(k)} = -\frac{1}{\log \rho_d^{(k)}}$. Here, $\tau_d^{(k)}$ is determined by the retention coefficient which is distinct from the somatic time constant τ_d^{s}. Larger $\tau_d^{(k)}$ gives slower decay and longer retention. The dendritic time constants are initialized on a logarithmic grid. This initialization does not constrain their range or ordering after training.

Define the normalized EMA state

$$z_{t,d}^{(k)} = \rho_d^{(k)} z_{t-1,d}^{(k)} + \left(1 - \rho_d^{(k)}\right) e_{t,d}, \qquad z_{0,d}^{(k)} = 0. \tag{18}$$

The corresponding dendritic state is a scaled EMA:

$$m_{t,d}^{(k)} = g_d^{(k)} z_{t,d}^{(k)}, \qquad g_d^{(k)} = \frac{p_d^{(k)} \beta_d^{(k)}}{1 - \rho_d^{(k)}} = \frac{\beta_d^{(k)}}{\sigma\left(\alpha_d^{(k)}\right)}. \tag{19}$$

Unrolling the branch recurrence gives the causal kernel representation

$$r_{t,d} = \omega_d e_{t,d} + \sum_{j=0}^{t-1} \mathcal{K}_d[j] e_{t-j,d}, \qquad \mathcal{K}_d[j] = \frac{1}{\sqrt{K}} \sum_{k=1}^{K} \gamma_d^{(k)} p_d^{(k)} \beta_d^{(k)} \left(\rho_d^{(k)}\right)^j, \tag{20}$$

whereas the outer exponent j denotes the decay over j sequence steps. Proposition 1 proves the equivalence between scaled EMA and a linear leaky compartment under piecewise-constant inputs.

4.3 CHUNKWISE CAUSAL SSA

Chunkwise causal SSA combines direct attention within each chunk with temporal state propagation across chunks, as illustrated in Fig. 3. We use the event sequences $\mathbf{Q}_s$, $\mathbf{K}_s$, and $\mathbf{V}_s$ defined in Section 4.1 and omit the layer index. For an unpadded sequence of length N, we form $J = \lceil N/n_c \rceil$ contiguous chunks with boundaries $t_b = \min(bn_c, N)$ and $t_0 = 0$. Chunk b contains positions $t_{b-1} + 1, \ldots, t_b$ and has length $n_b = t_b - t_{b-1}$.

Intra-chunk causal attention. Let $\mathbf{Q}_{s,b}$, $\mathbf{K}_{s,b}$, and $\mathbf{V}_{s,b}$ denote the spike matrices in chunk b. With the lower-triangular mask $(\mathbf{P}_b)_{ij} = \mathbb{I}[j \le i]$, its attention output is

$$\mathbf{A}_b = \alpha \left(\mathbf{Q}_{s,b}\mathbf{K}_{s,b}^{\top}\right) \odot \mathbf{P}_b, \mathbf{Y}_b = \mathbf{A}_b\mathbf{V}_{s,b}, \tag{21}$$

where $\alpha = \frac{1}{64}$ is the fixed scale factor. Thus, each query aggregates values from its own position and preceding positions within the same chunk. Across the sequence, the direct attention matrix has the block-diagonal form $\mathbf{A} = \mathrm{blkdiag}(\mathbf{A}_1, \ldots, \mathbf{A}_J)$, with lower-triangular diagonal blocks and zero cross-chunk entries. If the final chunk is padded for batched computation, all attention edges involving added padding are masked out.

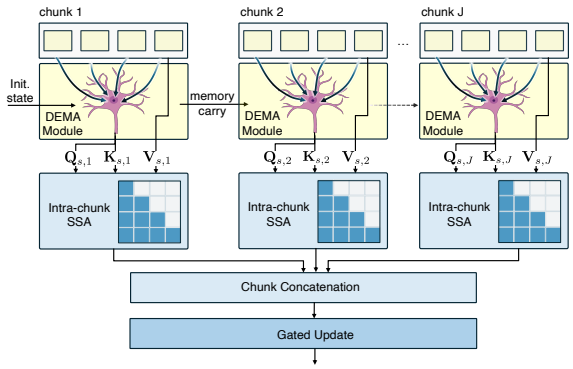


Figure 3: Chunkwise causal SSA. Within each chunk, causal SSA aggregates value events using DEMA-conditioned query and key events. DEMA carries temporal memory across chunk boundaries, providing inter-chunk context. The chunk outputs are concatenated in sequence order and passed to the gated update. Linear projections and spiking neuron layers are omitted for clarity.

Inter-chunk memory propagation. DEMA and the LIF populations retain their states across chunk boundaries. For position $i = 1, \ldots, n_b$ in chunk b, expanding the dendritic recurrence from the chunk boundary gives

$$\mathbf{m}^{(k)}_{t_{b-1}+i} = \underbrace{\left(\boldsymbol{\rho}^{(k)}\right)^{i} \odot \mathbf{m}^{(k)}_{t_{b-1}}}_{\text{inter-chunk memory}} + \underbrace{\sum_{r=1}^{i} \left(\boldsymbol{\rho}^{(k)}\right)^{i-r} \odot \mathbf{p}^{(k)} \odot \boldsymbol{\beta}^{(k)} \odot \mathbf{e}_{t_{b-1}+r}}_{\text{intra-chunk accumulation}}, \tag{22}$$

where powers are element-wise. The first term carries dendritic history across the boundary, while the second incorporates inputs from the current chunk. The carried state influences the somatic events that generate queries, keys, and modulation signals. The LIF neurons also preserve their membrane states across boundaries, providing additional temporal dependence. Consequently, earlier chunks can influence the current output through these states, although their keys and values are not directly accessed by the current attention block. Finally, the chunk outputs are concatenated in sequence order and any added padding is removed before the gated update described in Section 4.1. The context LIF then evolves over the restored sequence without resets at chunk boundaries. All states are initialized to zero at the beginning of each independent forward pass.

5 Experiments

We organize the evaluation around predictive accuracy and the use of temporal memory. We first consider S-MNIST, PS-MNIST, SHD, and LRA, then analyze a trained reference model through fixed-weight interventions. These analyses examine whether predictions depend on cross-chunk dendritic history, how branches with different timescales contribute, and which operations remain difficult.

5.1 Sequential Benchmark Comparisons

Sequential and spiking recognition. Table 1 compares SpikeWeave with spiking models for sequential image classification and auditory recognition. On S-MNIST and PS-MNIST, SpikeWeave achieves test accuracies of 99.46% and 98.44%, respectively, with 0.97M parameters. It obtains the highest PS-MNIST accuracy among the compared methods, while its S-MNIST accuracy is close to the 99.53% reported for PMSN. Relative to Spikformer and the Spike-driven Transformer, SpikeWeave achieves higher accuracy on both tasks with approximately 88% fewer parameters. It processes the 784 sequence positions directly as neuronal time, without an additional simulation-time dimension. On SHD, SpikeWeave achieves 96.43% accuracy using 100 time steps and 0.27M parameters. This is comparable to SpikCommander's 96.41% at the same number of time steps with

Table 1: Long-sequence classification results on sequential event-based benchmarks.

Datasets	Method	Neuron	Arch.	Param.	Parallel	SD	$T \times N$	Acc@1
S/PS-MNIST	(Zenke & Vogels, 2021) *Neural Comput.*	LIF	FF	85.1K	✗	✓	784×1	72.06 / 10.00
	(Yin et al., 2021) *Nat. Mach. Intell.*	ALIF	Rec.	156.3K	✗	✓	784×1	98.70 / 94.32
	(Fang et al., 2023) *NeurIPS*	PSN	FF	2.5M	✓	✓	784×1	97.90 / 97.76
	(Chen et al., 2026) *IEEE TNNLS*	PMSN	FF	156.4K	✓	✓	784×1	99.53 / 97.78
	(Zhou et al., 2023) *ICLR*	LIF	Attn.	8.05M	✗	✗	4×784	98.84 / 97.97
	(Yao et al., 2023) *NeurIPS*	LIF	Attn.	7.97M	✗	✓	4×784	98.77 / 97.80
	(Bal & Sengupta, 2025) *ICLR*	S6 (Prob.)	SSM	1.08M	✓	✗	784×1	99.43 / 98.40
	SpikeWeave *Ours*	MultiDen.	Attn.	0.97M	✓	✓	784×1	**99.46 / 98.44**
Spiking Heidelberg Digits	(Zenke & Vogels, 2021) *Neural Comput.*	LIF	Rec.	249.0K	✗	✓	250×1	84.00
	(Yin et al., 2021) *Nat. Mach. Intell.*	ALIF	Rec.	141.3K	✗	✓	250×1	84.40
	(Fang et al., 2023) *NeurIPS*	PSN	FF	232.5K	✓	✓	250×1	89.75
	(Wang et al., 2026) *AAAI*	LIF	SCommd.	0.19M	✓	✓	100×1	96.41
	SpikeWeave *Ours*	MultiDen.	Attn.	0.27M	✓	✓	100×1	**96.43**

Table 2: Comparison of model accuracy on the Long Range Arena (LRA) benchmark. The input length of each task is shown in parentheses.

Methods	Network	ListOps (2,048)	Text (4,096)	Retrieval (4,000)	Image (1,024)	Pathfinder (1,024)	Avg.
Non-spiking baselines							
–	Random	10.00	50.00	50.00	10.00	50.00	34.00
(Tay et al., 2021)	Transformer	36.37	64.27	57.46	42.44	71.40	54.39
(Gu et al., 2022)	S4 (Bidirectional)	59.60	86.82	90.90	88.65	94.20	84.03
Spiking state-space models							
(Stan & Rhodes, 2024)	Binary S4D	54.80	82.50	85.03	82.00	79.79	76.82
	$\rightarrow$ + GSU & GeLU	59.60	86.50	90.22	85.00	91.30	82.52
(Shen et al., 2025b)	SpikingSSMs	60.23	80.41	88.77	88.21	93.51	82.23
(Liu et al., 2024)	Spiking LMU	37.30	65.80	79.76	55.65	72.68	62.23
(Spieler et al., 2024)	ELM Neuron	44.55	75.40	84.93	49.62	71.15	65.13
(Huang et al., 2024)	SD-TCM	59.20	86.33	89.88	84.77	91.76	82.39
Spiking attention models							
SpikeWeave	**DEMA+Attn.**	**58.85**	**87.64**	**90.02**	**83.42**	**92.08**	**82.40**

0.19M parameters. The SHD result extends the evaluation from sequential pixel inputs to auditory spike sequences.

Long-range sequence modeling. Table 2 evaluates SpikeWeave on five LRA tasks spanning symbolic, textual, and visual inputs, with sequence lengths from $1{,}024$ to $4{,}096$ per input sequence. SpikeWeave achieves an average accuracy of 82.40%, comparable to Binary S4D with GSU and GELU (82.52%) and SD-TCM (82.39%), and below bidirectional S4 (84.03%).

Table 3: Fixed-weight interventions on 512 ListOps validation examples. Each mask suppresses four of 16 branches per channel and layer. Drops are in percentage points; intervals are paired 95% bootstrap CIs. The random condition uses one mask realization.

Condition	Acc.(%)	Drop(pp)	95% CI
Baseline	57.42	0.00	–
Reset dendrites at chunk boundaries	36.72	20.70	[16.60, 24.80]
Mask fastest 25%	58.40	−0.98	[−3.52, 1.37]
Mask slowest 25%	53.13	4.30	[1.56, 7.23]
Random mask 25%	49.61	7.81	[4.10, 11.33]

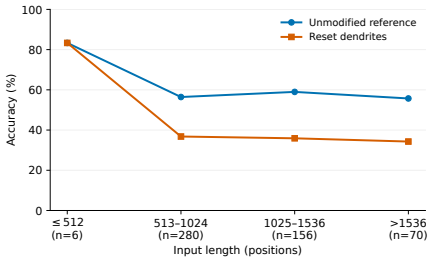


Figure 4: ListOps accuracy with and without dendritic resets at attention chunk boundaries, grouped by input length. Each chunk spans 512 positions. n denotes the number of validation examples in each length group.

5.2 Dependence on Cross-Chunk Dendritic Memory

Evaluation setting. We first test whether prediction depends on dendritic states carried across attention chunks. Both analyses below use a six-layer SpikeWeave reference variant with $D = 80$, $K = 16$, and chunk length $n_c = 512$. Its differences from the main architecture and the evaluation protocol are detailed in Appendix C. All interventions keep the weights fixed and use the same 512 ListOps validation examples. Paired 95% bootstrap intervals quantify uncertainty across examples, conditional on this checkpoint and the chosen intervention.

Removing the dendritic carry. We remove the inter-chunk carry in Eq. equation 22 by resetting dendritic states at every attention chunk boundary in all layers. Somatic and other LIF states are not directly reset. Attention connectivity and global mean pooling are unchanged. Accuracy falls from 57.42% to 36.72%, a 20.70-point drop with a paired 95% interval of [16.60, 24.80] (Table 3). This establishes that the checkpoint's performance depends on cross-chunk dendritic carry. Because DEMA supplies both query/key events and modulation signals, the result does not isolate the contribution of query/key addressing.

Length and activity diagnostics. All three multi-chunk groups lose approximately 20–23 points, with no monotonic increase in the effect across length groups (Fig. 4). The six single-chunk examples are unchanged, consistent with the absence of an internal chunk boundary. Their small number limits this observation to a consistency check. DEMA somatic event rates remain nonzero in every layer after reset (Appendix C.2). The loss therefore occurs without complete somatic silence, although event timing and downstream activity may still change.

6 Conclusion

We propose SpikeWeave that combines multiscale dendritic dynamics with chunkwise spiking self-attention. Dendritic traces preserve temporal context across somatic resets and condition spike generation. Local query–key interactions complement this recurrent memory, while fixed-length chunks keep the attention term linear in sequence length. In the evaluation of cross-chunk dendritic memory, resetting dendritic states at chunk boundaries reduces accuracy by 20.70%, showing that its predictions depend on cross-chunk dendritic carry. Branch suppression further reveals greater sensitivity to removing the slowest branches than the fastest branches.

Limitations. In SpikeWeave, context from earlier chunks is compressed into finite-dimensional states, which do not provide direct attention to individual earlier positions. The current implementation processes complete sequences and does not preserve states between separate chunk calls. Although we apply chunkwise SSA to reduce attention memory costs for very long sequences, chunking alone does not guarantee constant training memory. Our interventions provide initial evidence for the role of dendritic states in temporal memory. How individual dendrite retain information and interact across timescales remains to be fully characterized. Code will be made publicly available soon.

AI USE STATEMENT

The core research ideas, methodology, experimental design, and scientific content of the figures and tables originated with the authors. We used large language models to assist with software implementation and engineering tasks, as well as manuscript drafting, organization, language refinement, and LaTeX formatting. Additional assistance included literature searches, reference checking, and discussion of the interpretation of experimental results. The authors reviewed the AI-assisted contributions and take full responsibility for the accuracy, originality, and integrity of all methods, code, results, figures, and text in this work.

ETHICS STATEMENT

This work studies spiking neural architectures for sequence modeling using established, publicly available benchmark datasets. The study does not involve recruiting human participants or collecting new personal data. Our empirical findings are limited to the reported benchmark settings.

REPRODUCIBILITY STATEMENT

The main text describes the model architecture and evaluation setup. Supporting theoretical derivations and additional experimental details are provided in the appendices. The fixed-weight intervention protocol is documented in Appendix C, with additional diagnostics in Appendix C.2. The benchmark datasets used in this work are publicly available. The code will be made publicly available in a future release.

REFERENCES

Malyaban Bal and Abhronil Sengupta. P-SpikeSSM: Harnessing probabilistic spiking state space models for long-range dependency tasks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Sf4ep9Udjf.

Tiago Branco, Beverley A. Clark, and Michael Hausser. Dendritic discrimination of temporal input sequences in cortical neurons. *Science*, 329(5999):1671–1675, 2010. doi: 10.1126/science.1189664. URL https://doi.org/10.1126/science.1189664.

Xinyi Chen, Jibin Wu, Chenxiang Ma, Yinsong Yan, Hanwen Liu, Yujie Wu, and Kay Chen Tan. PMSN: A parallel multi-compartment spiking neuron for multiscale temporal processing. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, July 2026. doi: 10.1109/TNNLS.2026.3708510. URL https://ieeexplore.ieee.org/document/11604167. Early Access.

Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2744–2757, July 2022. ISSN 2162-2388. doi: 10.1109/tnnls.2020.3044364. URL http://dx.doi.org/10.1109/TNNLS.2020.3044364.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359. URL https://doi.org/10.1109/MM.2018.112130359.

Wei Fang, Zhaofei Yu, Zhaokun Zhou, Ding Chen, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53674–53687, 2023. doi: 10.52202/075280-2335. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a834ac3dfdb90da54292c2c932c997cc-Abstract-Conference.html.

Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Wei Huang, Qinying Gu, and Nanyang Ye. Decision spikeformer: Spike-driven transformer for decision making. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19241–19250, June 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Huang_Decision_SpikeFormer_Spike-Driven_Transformer_for_Decision_Making_CVPR_2025_paper.html.

Yulong Huang, Zunchang Liu, Changchun Feng, Xiaopeng Lin, Hongwei Ren, Haotian Fu, Yue Zhou, Hong Xing, and Bojun Cheng. Prf: Parallel resonate and fire neuron for long sequence learning in spiking neural networks. *arXiv preprint arXiv:2410.03530*, 2024. doi: 10.48550/arXiv.2410.03530. URL https://arxiv.org/abs/2410.03530.

E.M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6): 1569–1572, 2003. doi: 10.1109/TNN.2003.820440.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. ISSN 0018-9219. doi: 10.1109/5.726791. URL http://dx.doi.org/10.1109/5.726791.

Zeyu Liu, Gourav Datta, Anni Li, and Peter Anthony Beerel. LMUFormer: Low complexity yet powerful spiking model with legendre memory units. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=oEF7qExD9F.

Michael London and Michael Haeusser. Dendritic computation. *Annual Review of Neuroscience*, 28: 503–532, 2005. doi: 10.1146/annurev.neuro.28.061604.135703. URL https://doi.org/10.1146/annurev.neuro.28.061604.135703.

Changze Lv, Yansen Wang, Dongqi Han, Xiaoqing Zheng, Xuanjing Huang, and Dongsheng Li. Efficient and effective time-series forecasting with spiking neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 33624–33637. PMLR, 2024. URL https://proceedings.mlr.press/v235/lv24d.html.

Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. doi: 10.1016/S0893-6080(97)00011-7. URL https://doi.org/10.1016/S0893-6080(97)00011-7.

Michalis Pagkalos, Spyridon Chavlis, and Panayiota Poirazi. Introducing the dendrify framework for incorporating dendrites to spiking neural networks. *Nature Communications*, 14:131, 2023. doi: 10.1038/s41467-022-35747-8. URL https://doi.org/10.1038/s41467-022-35747-8.

Panayiota Poirazi, Terrence Brannon, and Bartlett W. Mel. Pyramidal neuron as two-layer neural network. *Neuron*, 37(6):989–999, 2003. doi: 10.1016/S0896-6273(03)00149-1. URL https://doi.org/10.1016/S0896-6273(03)00149-1.

Megha Sehgal, Daniel Almeida Filho, George Kastellakis, Sungsoo Kim, Jinsu Lee, Yang Shen, Shan Huang, Ayal Lavi, Giselle Fernandes, Irene Davila Mejia, Sunaina Soans Martin, Asli Pekcan, Melody Shana Wu, Won Do Heo, Panayiota Poirazi, Joshua T. Trachtenberg, and Alcino J. Silva. Compartmentalized dendritic plasticity in the mouse retrosplenial cortex links contextual memories formed close in time. *Nature Neuroscience*, 28:602–615, 2025. doi: 10.1038/s41593-025-01876-8. URL https://doi.org/10.1038/s41593-025-01876-8.

Shuaijie Shen, Chao Wang, Renzhuo Huang, Yan Zhong, Qinghai Guo, Zhichao Lu, Jianguo Zhang, and Luziwei Leng. Spikingssms: Learning long sequences with sparse and parallel spiking state space models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(19):20380–20388, 2025a. doi: 10.1609/aaai.v39i19.34245. URL https://ojs.aaai.org/index.php/AAAI/article/view/34245.

Shuaijie Shen, Chao Wang, Renzhuo Huang, Yan Zhong, Qinghai Guo, Zhichao Lu, Jianguo Zhang, and Luziwei Leng. SpikingSSMs: Learning long sequences with sparse and parallel spiking state space models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(19):20380–20388, 2025b. doi: 10.1609/aaai.v39i19.34245. URL https://doi.org/10.1609/aaai.v39i19.34245.

Benjamin Sivyer and Stephen R. Williams. Direction selectivity is computed by active dendritic integration in retinal ganglion cells. *Nature Neuroscience*, 16(12):1848–1856, 2013. doi: 10.1038/nn.3565. URL https://doi.org/10.1038/nn.3565.

Spencer L. Smith, Ian T. Smith, Tiago Branco, and Michael Hausser. Dendritic spikes enhance stimulus selectivity in cortical neurons in vivo. *Nature*, 503:115–120, 2013. doi: 10.1038/nature12600. URL https://doi.org/10.1038/nature12600.

Aaron Spieler, Nasim Rahaman, Georg Martius, Bernhard Schölkopf, and Anna Levina. The expressive leaky memory neuron: An efficient and expressive phenomenological neuron model can solve long-horizon tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vE1e1mLJ0U.

Matei-Ioan Stan and Oliver Rhodes. Learning long sequences in spiking neural networks. *Scientific Reports*, 14(1):21957, 2024. doi: 10.1038/s41598-024-71678-8. URL https://doi.org/10.1038/s41598-024-71678-8.

Pengfei Sun, Jibin Wu, Malu Zhang, Paul Devos, and Dick Botteldooren. Delay learning based on temporal coding in spiking neural networks. *Neural Networks*, 180:106678, 2024. doi: 10.1016/j.neunet.2024.106678. URL https://doi.org/10.1016/j.neunet.2024.106678.

Pengfei Sun, Jibin Wu, Malu Zhang, Paul Devos, and Dick Botteldooren. Delayed memory unit: Modeling temporal dependency through delay gate. *IEEE Transactions on Neural Networks and Learning Systems*, 36(6):10808–10818, 2025. doi: 10.1109/TNNLS.2024.3490833. URL https://doi.org/10.1109/TNNLS.2024.3490833.

Pengfei Sun, Zhe Su, Jascha Achterberg, Giacomo Indiveri, Dan F. M. Goodman, and Danyal Akarca. Algorithm–hardware co-design of neuromorphic networks with dual memory pathways. *Nature Machine Intelligence*, 8(6):901–912, 2026. doi: 10.1038/s42256-026-01255-3. URL https://www.nature.com/articles/s42256-026-01255-3.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Jiaqi Wang, Liutao Yu, Xiongri Shen, Sihang Guo, Chenlin Zhou, Leilei Zhao, Yi Zhong, Zhiguo Zhang, and Zhengyu Ma. SpikCommander: A high-performance spiking transformer with multi-view learning for efficient speech command recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(3):2119–2127, 2026. doi: 10.1609/aaai.v40i3.37194. URL https://ojs.aaai.org/index.php/AAAI/article/view/37194.

Shuai Wang, Malu Zhang, Jingya Wang, Dehao Zhang, Yimeng Shan, Jieyuan Zhang, Yichen Xiao, Honglin Cao, Haonan Zhang, Zeyu Ma, Yang Yang, and Haizhou Li. Bipolar self-attention for spiking transformers. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-3401. URL https://papers.nips.cc/paper_files/paper/2025/hash/9316cfcb0a81e53a1f35b4353f115571-Abstract-Conference.html.

Shuai Wang, Malu Zhang, Dehao Zhang, Ammar Belatreche, Yichen Xiao, Yu Liang, Yimeng Shan, Qian Sun, Enqi Zhang, and Yang Yang. Spiking vision transformer with saccadic attention. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=qzZsz6MuEq.

Wenjie Wu, Dexuan Huo, and Hong Chen. SpikF: Spiking Fourier network for efficient long-term prediction. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67342–67368. PMLR, 2025. URL https://proceedings.mlr.press/v267/wu25m.html.

Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer: When spiking neural network meets the transformer. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2798. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ca0f5358dbadda74b3049711887e9ead-Abstract-Conference.html.

Man Yao, Jiakui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. 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*, 2024. URL https://openreview.net/forum?id=1SIBN5Xyw7.

Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):2973–2990, 2025. doi: 10.1109/TPAMI.2025.3530246. URL https://arxiv.org/abs/2411.16061.

Bojian Yin, Federico Corradi, and Sander M. Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10): 905–913, 2021. doi: 10.1038/s42256-021-00397-w. URL https://doi.org/10.1038/s42256-021-00397-w.

Friedemann Zenke and Tim P. Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, 33(4):899–925, 2021. doi: 10.1162/neco_a_01367. URL https://doi.org/10.1162/neco_a_01367.

Dehao Zhang, Malu Zhang, Shuai Wang, Jingya Wang, Wenjie Wei, Zeyu Ma, Guoqing Wang, Yang Yang, and Haizhou Li. Dendritic resonate-and-fire neuron for effective and efficient long sequence modeling. In *Advances in Neural Information Processing Systems 38*, 2025a. doi: 10.52202/085713-3587. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/9a6ca401f890aeb2c272d2bb26196d3f-Abstract-Conference.html.

Dehao Zhang, Fukai Guo, Shuai Wang, Jingya Wang, Jieyuan Zhang, Yimeng Shan, Malu Zhang, Yang Yang, and Haizhou Li. Neural dynamics self-attention for spiking transformers. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jJedqisfOt.

Jieyuan Zhang, Xiaolong Zhou, Shuai Wang, Wenjie Wei, Hanwen Liu, Qian Sun, Malu Zhang, Yang Yang, and Haizhou Li. Unveiling the spatial-temporal effective receptive fields of spiking neural networks. In *Advances in Neural Information Processing Systems*, volume 38, pp. 35904–35927, 2025b. doi: 10.52202/085713-1206. URL https://papers.neurips.cc/paper_files/paper/2025/hash/3353b22e6b85a76d45d6b01aa4328be5-Abstract-Conference.html.

Hanle Zheng, Zhong Zheng, Rui Hu, Bo Xiao, Yujie Wu, Fangwen Yu, Xue Liu, Guoqi Li, and Lei Deng. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15:277, 2024. doi: 10.1038/s41467-023-44614-z. URL https://www.nature.com/articles/s41467-023-44614-z.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=frE4fUwz_h.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K. Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2302.13939.

A Additional Theoretical Analysis

A.1 EMA realization of the dendritic recurrence

Proposition 1 (EMA realization by leaky dendrites)**.** *Consider a dendritic branch in Eq. equation 16, with parameters fixed across sequence positions and zero initial state. Let $z_{t,d}^{(k)}$ and $g_d^{(k)}$ be defined in Eqs. equation 18 and equation 19, respectively, and let $\tau_d^{(k)} = -1/\log \rho_d^{(k)}$. Then*

$$m_{t,d}^{(k)} = g_d^{(k)} z_{t,d}^{(k)}. \tag{23}$$

Moreover, the branch recurrence is the exact unit-step discretization of

$$\tau_d^{(k)} \frac{\mathrm{d}m_d^{(k)}(\xi)}{\mathrm{d}\xi} = -m_d^{(k)}(\xi) + g_d^{(k)} e_d(\xi), \tag{24}$$

with $m_d^{(k)}(0) = 0$ and $e_d(\xi) = e_{t,d}$ on each interval $(t-1, t]$. Here, ξ is continuous sequence time, and $m_d^{(k)}(t) = m_{t,d}^{(k)}$.

Proof. The coefficient parameterization gives

$$1 - \rho_d^{(k)} = p_d^{(k)} \sigma\left(\alpha_d^{(k)}\right), \qquad g_d^{(k)} \left(1 - \rho_d^{(k)}\right) = p_d^{(k)} \beta_d^{(k)}.$$

Multiplying the normalized EMA recurrence by $g_d^{(k)}$ yields

$$g_d^{(k)} z_{t,d}^{(k)} = \rho_d^{(k)} g_d^{(k)} z_{t-1,d}^{(k)} + p_d^{(k)} \beta_d^{(k)} e_{t,d}.$$

This is the dendritic recurrence. The shared zero initial condition therefore establishes $m_{t,d}^{(k)} = g_d^{(k)} z_{t,d}^{(k)}$ by induction.

For the continuous formulation, suppress the feature and branch indices. With constant input e_t over $(t-1, t]$, the exact solution is

$$m(t) = e^{-1/\tau} m(t-1) + g\left(1 - e^{-1/\tau}\right) e_t.$$

Substituting $e^{-1/\tau} = \rho$ and $g(1-\rho) = p\beta$ recovers $m_t = \rho m_{t-1} + p\beta e_t$. □

This equivalence concerns the linear dendritic recurrence. Somatic thresholding and subtractive reset act on a separate membrane state and do not reset the dendritic traces.

B Details of the Sequential Benchmark

B.1 Datasets

Sequential and permuted sequential MNIST. S-MNIST and PS-MNIST convert each 28×28 MNIST image (Lecun et al., 1998) into a sequence of 784 scalar pixel values for ten-class digit classification. S-MNIST uses raster-scan order, whereas PS-MNIST applies a fixed permutation of the pixel positions, shared across examples and data splits. The latter removes the local ordering present in the original image while preserving its pixel values.

Spiking Heidelberg Digits. SHD (Cramer et al., 2022) provides auditory spike recordings for classification into 20 spoken-digit classes. Each recording consists of event times and indices from 700 input channels. The temporal representation supplied to the model must be specified separately from this original event representation.

Table 4: Current sequential-benchmark model presets. $N_{\max}$ is the configured input-length limit, D the residual width, d_v the value width, d_{ff} the FFN hidden width, and n_c the attention chunk length. All LRA rows use $L = 6$, $K = 16$, and $d_q = 64$. Retrieval lengths are per document. Dashes denote unconfirmed settings. Pre/Post denotes residual normalization placement; LN and BN denote LayerNorm and BatchNorm, respectively.

Task	$N_{\max}$	Classes	D	d_v	d_{ff}	n_c	Norm.
S/PS-MNIST	784	10	128	128	128	49	Post-LN
SHD	100	20	128	128	128	10	Post-LN
ListOps	2048	10	128	128	128	128	Post-LN
Text	4096	2	256	256	256	512	Pre-LN
Retrieval	4000	2	256	256	256	512	Pre-LN
Image	1024	10	512	512	512	128	Post-BN
Pathfinder	1024	2	256	256	256	128	Pre-LN

Long Range Arena. LRA (Tay et al., 2021) comprises ListOps, Text, Retrieval, Image and Pathfinder. These tasks cover nested-expression evaluation, sentiment classification, document matching, image classification, and visual path connectivity. The implementation reads preprocessed token sequences for ListOps, Text, and Retrieval, and complete grayscale pixel sequences for the three image tasks. Table 4 reports the configured input-length limits and model dimensions.

Token sequences are truncated to the configured limit and right-padded to the longest sequence in each mini-batch. For Retrieval, the two documents remain separate sequences; their neuronal states are not carried from one document to the other.

Image inputs retain all 1024 pixels for Image and Pathfinder. Pixel values $p \in [0, 255]$ are transformed as $(p/255 - \mu)/\sigma$, with $(\mu, \sigma) = (0.48, 0.24)$ for Image and $(0.5, 0.5)$ for Pathfinder and Path-X. These sequences are not truncated, and zero-valued pixels are treated as valid inputs.

B.2 Model Configurations

Shared backbone and task-specific dimensions. The current LRA presets use $L = 6$ encoder blocks, $K = 16$ dendritic branches per feature, and query/key width $d_q = 64$. Each block combines DEMA, chunkwise SSA, and a gated spiking FFN with a continuous content branch and a binary neuronal gate. SSA uses a fixed scale of $1/64$ and causal masks within chunks. Dendritic and LIF states start from zero for each independent input and evolve along sequence positions across chunk boundaries. There is no additional simulation-time dimension.

The normalization variants are listed explicitly in Table 4. Pre-normalized encoders additionally normalize the final sequence representation. In the Image configuration, BatchNorm computes training statistics over the batch and sequence axes and uses running statistics at evaluation; the causal restriction describes the attention mask. All configurations normalize SSA context currents by their per-position RMS with $\epsilon = 10^{-8}$ before context gating and spike generation.

Input embeddings and classification heads. Token inputs use learned embeddings, while normalized pixels use a learned scalar-to-D affine map. The final representation is averaged over valid positions, excluding padding, and passed to a classifier with one hidden layer, GELU activation, and a linear output projection. Its hidden width equals d_{ff} in the listed presets. Retrieval uses a shared encoder for both documents and concatenates their separately pooled representations before classification. Embedding dropout is 0.1, except for Text, which uses 0.2. All presets use residual-branch dropout of 0.1, classifier dropout of 0.2, and zero attention and hidden-event dropout.

Neuronal initialization. DEMA branch time constants are initialized on a logarithmic grid from 2 to 1024 sequence positions and subsequently learned independently for each channel and branch. Somatic time constants start at 2 and are constrained to $(1, 4096)$; the other LIF time constants start at 2 and are constrained to $(1, 128)$. Initial thresholds are 0.5 for the DEMA soma, 1.0 for attention input, query, key, value, and context neurons, 0.15 for the context and update gates, 1.5 for the FFN input neurons, and 0.5 for the FFN gate. All thresholds are learnable with a lower bound of 0.05. LIF neurons use subtractive reset, with gradients stopped through the emitted spike on the reset path.

Table 5: Current LRA batching and weight-decay presets. Retrieval batches count document pairs. Global batch is training batch per GPU times four; validation batch sizes are also per GPU.

Task	Train/GPU	Global train	Validation/GPU	Weight decay
ListOps	108	432	108	0.05
Text	50	200	50	0.10
Retrieval	16	64	28	0.05
Image	256	1024	280	0.05
Pathfinder	256	1024	280	0.05

B.3 Training Settings

Optimization. All the presets use AdamW with a peak learning rate of 3×10^{-3}, moment coefficients $(0.9, 0.98)$, and $\epsilon = 10^{-8}$. The learning rate increases linearly from 3×10^{-7} over 3000 optimizer updates, then decays linearly to zero at update $90{,}000$. This is the maximum training budget; early stopping can terminate training earlier. Weight decay is 0.1 for Text and 0.05 for the other LRA tasks. Dendritic decay logits, neuronal time-constant and threshold parameters, normalization parameters, and biases are excluded from weight decay. The objective is cross-entropy with label smoothing of 0.05, without an auxiliary spike-rate loss. The global gradient norm is clipped to 1.0.

Surrogate gradients. The forward pass uses discrete spike thresholds. Backpropagation uses the derivative of a sigmoid surrogate with slope 10 for attention and DEMA neurons and slope 5 for FFN neurons. Gradients passing through each attention/DEMA event operator are additionally multiplied by 0.5; FFN event operators use unit backward gain. These factors affect only the backward pass.

Batching and execution. The presets specify FP32 training with four distributed GPU processes, no gradient accumulation, and seed 131071. Table 5 lists per-process batch sizes and the corresponding global batch for complete updates. Epoch-end batches are retained and can be smaller. These settings specify one seed per task; they do not imply multiple-seed averages or deterministic GPU execution.

C Details of the Mechanistic Analyses

C.1 Reference Checkpoint and Evaluation Protocol

The mechanism analyses use the supplied ListOps reference checkpoint. It has pre-normalized residual blocks, an additional LIF in the FFN content branch, and no RMS normalization after SSA. These properties distinguish it from the architecture described in Section 4.1.

The saved configuration excludes dynamics, normalization, and bias parameters from weight decay. Early stopping monitors validation cross-entropy with patience 20 and a minimum improvement of 0.002; the selected checkpoint maximizes validation accuracy. The intervention subset is sampled from the same 2000-example validation set and uses identical example and batch orders across conditions. Thus, these analyses characterize a selected checkpoint rather than an independent test-set estimate of generalization.

Evaluation consistency. The unmodified reference achieves 57.42% on the intervention subset. An existing full-validation evaluation records 1144 correct predictions out of 2000 (57.20%), compared with 57.60% in the historical training log. The 0.40 percentage-point difference lies within the evaluation script's 0.50-point tolerance, but that tolerance does not establish equivalence to the historical forward computation. The mechanism results refer to the specified evaluation graph.

Paired uncertainty estimates. The intervals in Table 3 use 2000 paired bootstrap resamples of the 512 examples. The direct fast–slow and slow–random comparisons use the supplied additional analysis with $100{,}000$ paired resamples. Each paired resample retains the correspondence between the same examples under the compared conditions. The random-mask condition uses one mask realization; these intervals therefore exclude variation across training seeds and random masks. The

Table 6: Cross-entropy under the same fixed-weight interventions as Table 3. Lower is better.

Condition	Cross-entropy
Unmodified reference	1.1417
Reset dendrites at chunk boundaries	1.7852
Mask fastest 25%	1.1407
Mask slowest 25%	1.2399
Randomly mask 25%	1.4321

additional comparisons and subgroup analyses are exploratory and are not corrected for multiple comparisons.

C.2 Additional Intervention Diagnostics

Losses and prediction changes. Table 6 reports cross-entropy on the intervention subset. Under boundary reset, 122 predictions change from correct to incorrect and 16 change from incorrect to correct. The net loss of 106 correct predictions accounts for the 20.70 percentage-point accuracy drop. Somatic and other LIF states and global mean pooling remain available under this intervention.

Input-length groups. The 6 examples of length at most 512 have identical predictions and cross-entropies before and after reset. For lengths 513–1024 ($n = 280$), 1025–1536 ($n = 156$), and above 1536 ($n = 70$), accuracy drops by 19.64, 23.08, and 21.43 percentage points, respectively. These observational groups differ in their input composition, so the grouped results do not isolate a causal effect of sequence length.

Somatic activity. DEMA soma event rates range from 4.53% to 13.78% across layers in the unmodified reference and from 4.23% to 12.98% after boundary reset. The rates count nonzero binary soma outputs over valid positions and channels, excluding padding. They rule out complete somatic silence as a description of the reset condition, but do not exclude changes in event timing or in the underlying dendritic currents. They characterize this neuronal population, not whole-network activity or energy consumption.

Readout gains. For the branch contribution to the kernel in Eq. equation 20, the signed DC gain is

$$G_d^{(k)} = \frac{\gamma_d^{(k)} p_d^{(k)} \beta_d^{(k)}}{\sqrt{K}\left(1 - \rho_d^{(k)}\right)}. \tag{25}$$

The gain includes the branch readout weight and differs from the state-scaling gain $g_d^{(k)}$ in Eq. equation 19. The fast, slow, and random masks remove 10.34%, 35.44%, and 26.07% of the sum of absolute branch gains, respectively. These are infinite-horizon gains of the linear dendritic kernel, excluding the direct input term; they do not measure stored task information. In the first layer, the fast and slow masks reduce dendritic readout RMS by approximately 4.4% and 25.7%.

Direct mask comparisons. Accuracy under the slow mask exceeds that under the random mask by 3.52 percentage points, with a paired 95% interval of $[-0.20, 7.23]$. This comparison does not establish a consistent ordering between the two masks. Together with the fast–slow comparison in the main text, it describes the selected branch subsets under fixed weights and does not isolate timescale from readout strength.

Additional timescale statistics. Across $6 \times 80 \times 16 = 7680$ dendritic branches, the interquartile range of learned $\tau_d^{(k)}$ is 10.81–263.09 positions, and 7.42% have $\tau_d^{(k)} \geq 1024$. After training, 79.10% of branches remain within a factor of two of their initial time constants; the correlation between initial and learned log-timescales is 0.945. The layer-wise distributions do not show a monotonic increase with depth. Large time constants describe slow decay and do not by themselves establish retention of task-relevant information over the same distance.

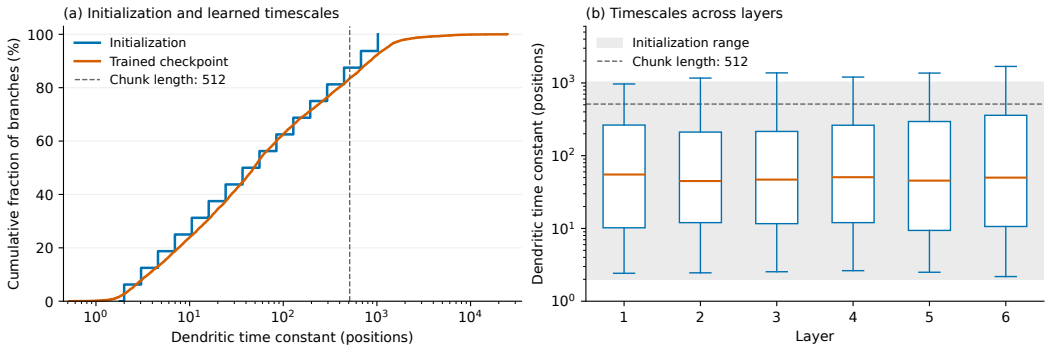


Figure 5: Dendritic time constants in the ListOps reference checkpoint. (a) Empirical cumulative distributions at initialization and after training. (b) Per-layer medians, interquartile boxes, and 5th–95th percentile whiskers. Dashed lines mark the 512-position attention chunk length; shading marks the initialization range.

C.3 Dendritic Timescales and Branch Contributions

Learned timescales. Having established dependence on dendritic carry, we next examine the learned timescales and the contributions of their associated branches. The median time constant is 48.51 positions, and 16.48% of branches satisfy $\tau_d^{(k)} \geq 512$, compared with 12.50% at initialization (Fig. 5). The initialization already spans 2–1024 positions on a logarithmic grid, so training adapts an existing multiscale memory bank. We assess branch contributions through selective suppression.

Selective branch suppression. Within each channel and layer, we rank branches by learned $\tau_d^{(k)}$ and suppress the fastest or slowest four by setting their readout coefficients $\gamma_d^{(k)}$ to zero in Eq. equation 17. The direct input term and $1/\sqrt{K}$ scaling remain unchanged. Suppressing the slowest branches reduces accuracy by 4.30 points. Suppressing the fastest branches increases accuracy by 0.98 points, but its confidence interval includes zero (Table 3). Accuracy under the fast mask exceeds that under the slow mask by 5.27 points, with a paired 95% interval of $[2.15, 8.40]$. The checkpoint is therefore more sensitive to suppressing its slowest branches than its fastest branches.