# SpikingSSMs: Learning Long Sequences with Sparse and Parallel Spiking State Space Models

**Shuaijie Shen**[*,1,2], **Chao Wang**[*,1,2], **Renzhuo Huang**[1,2], **Yan Zhong**[2,3], **Qinghai Guo**[2], **Zhichao Lu**[4], **Jianguo Zhang** [†,1,5], **Luziwei Leng**[†,2]

[1] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen
[2] ACSLab, Huawei Technologies Co., Ltd., Shenzhen
[3] School of Mathematical Sciences, Peking University, Beijing
[4] Department of Computer Science, City University of Hong Kong, Hong Kong
[5] Pengcheng Laboratory, Shenzhen
{shensj2024, wangc2023, huangrz2023}@mail.sustech.edu.cn, zhongyan@stu.pku.edu.cn, guoqinghai@huawei.com, zhichao.lu@cityu.edu.hk, zhangjg@sustech.edu.cn, lengluziwei@huawei.com

## Abstract

Known as low energy consumption networks, spiking neural networks (SNNs) have gained a lot of attention within the past decades. While SNNs are increasing competitive with artificial neural networks (ANNs) for vision tasks, they are rarely used for long sequence tasks, despite their intrinsic temporal dynamics. In this work, we develop spiking state space models (SpikingSSMs) for long sequence learning by leveraging on the sequence learning abilities of state space models (SSMs). Inspired by dendritic neuron structure, we hierarchically integrate neuronal dynamics with the original SSM block, meanwhile realizing sparse synaptic computation. Furthermore, to solve the conflict of event-driven neuronal dynamics with parallel computing, we propose a lightweight surrogate dynamic network which accurately predicts the after-reset membrane potential and compatible to learnable thresholds, enabling orders of acceleration in training speed compared with conventional iterative methods. On the long range arena benchmark task, SpikingSSM achieves competitive performance to state-of-the-art SSMs meanwhile realizing on average 90% of network sparsity. On language modeling, our network significantly surpasses existing spiking large language models (spikingLLMs) on the WikiText-103 dataset with only a third of the model size, demonstrating its potential as backbone architecture for low computation cost LLMs.

**Code** — https://github.com/shenshuaijie/SDN
**Extended version** — https://arxiv.org/abs/2408.14909

## Introduction

Recent years have witnessed the proliferation of real-world time-series datasets in various domains, which often require reasoning over tens of thousands of time steps (Tay et al. 2021a). Therefore, plenty of sequence models have emerged in recent years, which aim to model the long-range dependencies (LRDs) in sequential data to achieve human-level performance across diverse modalities, encompassing text, vision, audio, and video (Gu, Goel, and Re 2022). Among these methods, growing attention has been given to Transformer (Vaswani et al. 2017), since this architecture has led to remarkable developments in the areas of vision and speech. However, for an input sequence of length $L$, it requires the high-cost computational complexity of $\mathcal{O}(L^2)$ during training and inference in the module of self-attention, which is one of the core contextualizing components in the Transformer model. Although some Transformer variants (Kitaev, Kaiser, and Levskaya 2020; Zaheer et al. 2020; Katharopoulos et al. 2020; Choromanski et al. 2021) are proposed to reduce the compute and memory requirements, their performances on performing long-range reasoning remain considerably suboptimal (Gu, Goel, and Re 2022).

Recurrent neural networks (RNNs) (Schuster and Paliwal 1997; Sherstinsky 2020) have emerged early for learning on the variable-length input sequences, which requires only $\mathcal{O}(1)$ operations with respect to the sequence length. However, constrained hidden state space and gradient vanish problem have limited their learning of long sequences. To address this problem, innovative works such as RWKV (Peng et al. 2023) and state space models (SSMs) (Gu, Goel, and Re 2022; Gu and Dao 2023) are proposed by introducing an appropriate design of hidden states for handling LRDs with both training parallelizability and inference efficiency. RNNs owes part of its inspiration to cognitive and neurological computational principles (Lipton, Berkowitz, and Elkan 2015), which also serve as the foundation for another class of biologically-grounded architectures known as Spiking Neural Networks (SNNs) (Maass 1997). With their potential in low-energy computing, SNNs have gained a lot of attention within the past decades. Recently, they have been shown to be as efficient as artificial neural networks (ANNs) for vision tasks (Che et al. 2022; Zhou et al. 2022; Yao et al. 2024; Che et al. 2024) under convolution or Transformer architectures. However, despite the intrinsic temporal dynamics, SNNs are rarely used for long sequence tasks. Note that SNNs under convolution or Transformer architectures often need a certain simulation time

[*]These authors contributed equally.
[†]Corresponding author

window to improve spike-based representation, causing inference delays compared to their artificial counterparts. This disadvantage can be avoid for SNNs under RNN architecture since they can make use of the inherent temporal dimension for dynamic computing.

In this work, we explore an integration of spiking neurons with SSMs, and develop SpikingSSMs for long sequence learning, combining efficient parallel training and low-energy, spike-based sparse computation. Several recent works have proposed binary SSM (Stan and Rhodes 2024) or stochastic spiking SSM (Bal and Sengupta 2024). However, they have limited exploration or overlooked the intricate dynamics that characterize biological spiking neurons, leading to incomplete interpretability and performance degradation. To this end, we adopt the widely used Leaky Integrate-and-Fire (LIF) neuron with deterministic reset mechanisms (Gerstner et al. 2014). To reconcile the conflict of its asynchronous event-driven dynamics with parallel computing, we propose a surrogate dynamic network which accelerates training and is dispensable during inference without adding additional parameters to the network. Through an equivalence study we demonstrate the versatility of SDN for approximating parametric LIF neuron models and its potential as general purpose module for parallel computing SNNs. The key contributions of this study are summarized as follows:

- We introduce SpikingSSMs for long sequence tasks, which merge the strengths of SSMs in parallel computing and long sequence modeling with sparse computation of SNNs.
- To address the challenges posed by event-driven neuronal dynamics in the context of parallel computing, we propose a surrogate dynamic network (SDN) to approximate the dynamics of LIF neurons via a well-designed model, which extremely accelerates the training of SpikingSSMs with only negligible additional computation.
- We also highlight the equivalence of SDN for different thresholds and incorporate learnable thresholds into our model architecture, which further improves network performance.
- We evaluate our method on sequential and permuted sequential MNIST classification tasks, as well as the Long Range Arena (LRA) benchmark, where our model achieves competitive performance with state-of-the-art SSMs meanwhile with high sparsity. Additionally, in large-scale language modeling task on the WikiText-103 dataset. Our model sets a new record in the field of SNN, demonstrating its scalability.

# Related Work

## Long Sequence Modeling

The essential problem of sequence modeling is compressing context into a certain state. Driven by this problem, sequence models explore trade-offs between efficiency and effectiveness. For example, Attention mechanism (Vaswani et al. 2017; Dao et al. 2022; Dao 2023) does not compress context at all, i.e. it stores the entire context (i.e. the KV cache) during auto-regressive inference, which is effective but inefficient since this causes the slow linear-time inference and quadratic-time training (Sun et al. 2023; Yang et al. 2023). On the other hand, recurrent models compress context into a finite state, resulting in constant-time inference and linear-time training. However, their effectiveness is limited by how well this state has compressed the context and the fixed representation space (Peng et al. 2023; Qin et al. 2024). SSMs have emerged as compelling frameworks for sequence modeling. HiPPO (Gu et al. 2020) revolutionized this field by compressing long inputs into dynamic, polynomial-based representations using orthogonal polynomials. S4 (Gu, Goel, and Re 2022) further evolved this approach by introducing a low-rank correction, enabling stable diagonalization and simplifying operations with Cauchy kernels. A series of later works have further improved efficiency of the model using advanced techniques such as parallel scan (Smith, Warrington, and Linderman 2023), Fast Fourier Transform (FFT) (Fu et al. 2023; Duhamel and Vetterli 1990) and gating mechanism (Mehta et al. 2023). A very recent work, Mamba (Gu and Dao 2023) focuses on enhancing the selectivity of the state representation, balancing efficiency and effectiveness without compromising contextual information. Aided with hardware-optimized algorithms the model demonstrated strong performance on temporal tasks up to million-length sequences such as language modeling.

## SNNs for Sequence Modeling

With the improvement of SG training methods, SNNs adopting conventional RNN architectures have been applied to sequence classification tasks and achieved high accuracy (Bellec et al. 2018; Yin, Corradi, and Bohté 2021, 2023). However, limited by the architecture and serial processing, pure RNN-based SNNs are rarely applied to long sequence learning. To this end, enabling efficient parallel computing of SNN is critical. PSN (Fang et al. 2023) achieved it by removing the reset of spiking neuron, however with the cost of increased firing rate and insufficiency in network sparsity. PSU (Li et al. 2024b) proposed parallel spiking units which decoupled the integration-spiking-resetting process by introducing a probabilistic reset mechanism and effectively improved network sparsity. However, its learnable parameter is quadratic to the sequence length which impeded the scalability of the method. Leveraging on the Legendre Memory Units (LMU) for sequence modeling (Voelker, Kajić, and Eliasmith 2019), SpikingLMUFormer (Liu et al. 2024) augmented the LMU with convolutional layers and spiking activation, surpassing transformers in long sequence modeling. The recent progress of SSMs has also inspired works developing their spiking versions. Du, Liu, and Chua proposed SpikeS4 by simply stacking LIF neurons on S4 layers and applied for speech tasks. Binary S4D (Stan and Rhodes 2024) constructed binary SSM by directly applying spiking activation function on the summation of hidden states, which maintains parallel training but ignores neuronal dynamics and sparsity. A recent work (Bal and Sengupta 2024) proposed S6-based SNN which improved network sparsity by implementing a stochastic spiking neuron for SSM, however

the model exhibited significant accuracy drop compared to the original model, partially attributed to the stochastic noise in gradients. In this work, we adopt widely used deterministic reset dynamics for spiking neurons, and develop solutions to solve the conflict of their asynchronous event-driven feature with parallel computing.

### SNNs for Language Modeling

Motivated by the potential of constructing low-energy large language models, several recent works have explored combining SNNs with language models. SpikeGPT (Zhu et al. 2023) adopted spike activation for the output of RWKV (Peng et al. 2023) blocks and applied to large scale language modeling tasks. SpikeBERT (Lv et al. 2024) built upon Spikformer (Zhou et al. 2022) and distilled knowledge from the original BERT (Devlin et al. 2018). In this work, we develop large scale SNNs based on SSM architectures for language modeling.

## Method

### Preliminaries

**LIF Neuron** The LIF neuron is a simplified version of biological neuron models (Gerstner et al. 2014), which captures the "leaky-integrate-fire-reset" process and is widely used in SNNs for machine learning as it balances tractability and temporal dynamics. With $t$ denoting the time step, the LIF neuron is formulated by following equations:

$$u'_t = \tau u_{t-1} + I_t \tag{1}$$

$$s_t = H(u'_t - v_{\rm th}) \tag{2}$$

$$\text{Soft reset}: \ u_t = u'_t - s_t v_{\rm th} \tag{3}$$

$$\text{Hard reset}: \ u_t = u'_t(1 - s_t) + s_t u_r \tag{4}$$

where input currents $I$ are linearly integrated into the leaky membrane potential $u$ of the neuron, and then a spike $s$ is determined to be fired if the current $u$ surpasses a threshold $v_{\rm th}$, with $H$ denoting the Heaviside function. At last, the membrane potential is reset according to the soft reset mechanism (Eq. 3) or the hard reset mechanism (Eq. 4). The hard and soft reset mechanisms embody different neuronal memory strategies, where the hard reset forget the history after spiking and reset to a reset potential $u_r$ (we set it to 0 in this work), while the soft reset still keeps all the history subtracted by a reset after spiking. In order to realistically mimic biological neurons, the hard reset mechanism is most commonly used in spiking networks.

**Surrogate Gradient Training of SNN** Since the spikes are considered identical, the spiking activation function $H$ is defined as a Heaviside function which is non-differentiable at $x = 0$ and has a derivative value of 0 elsewhere. Therefore, surrogate gradient (SG) methods (Esser et al. 2016; Bellec et al. 2018) are proposed to solve this issue. The surrogate gradient function is defined as a soft relaxed function that approximates the original discontinuous gradient of the spiking activation function. Typical SG functions are usually differentiable everywhere and have a nonzero derivative value near the threshold, such as rectangular (Zheng et al. 2021) and triangular (Bellec et al. 2018) functions, etc.

**State Space Model** SSMs are broadly used in many scientific disciplines, which map a 1-dimensional signal $x$ to an N-dimensional latent signal $h$ and project it to a 1-dimensional output signal $y$. For a discrete input sequence $x_{1:L}$, through certain discretization rule (Gu et al. 2024) the SSMs can be defined by:

$$h_t = \bar{A} h_{t-1} + \bar{B} x_t \tag{5}$$

$$y_t = C h_t \tag{6}$$

with subscript $t$ denoting the time step. The parameters are the state matrix $\bar{A} \in \mathbb{R}^{N\times N}$ and other matrices $\bar{B} \in \mathbb{R}^{N\times 1}$, $C \in \mathbb{R}^{1\times N}$. Theoretically, $\bar{A}$ can be diagonalized for efficient computation (Gupta, Gu, and Berant 2022). Within a layer of the network, the input is always multidimensional rather than 1-dimension, therefore, an SSM layer handles multiple features by multiple independent SSMs (with different parameters). In parallel computing, the SSM can be expressed as the convolution between convolution kernels and input signals, with the initial condition $y_0 = 0$:

$$y_t = \sum_{k=1}^{t} C \bar{A}^{t-k} \bar{B} x_k \tag{7}$$

In practice, this computation can be further accelerated by FFT with time complexity $\mathcal{O}(L\log(L))$ (Gupta, Gu, and Berant 2022)].

### Spiking S4 Block

It has been shown that the diagonal version of SSM (Gupta, Gu, and Berant 2022) maintains performance while simplifying the model. Therefore, we choose the latest S4D model (Gu et al. 2024) as the backbone to verify our method. The output $y$ of the state space block is now activated by an LIF neuron, i.e. the $y_t = Ch_t$ is treated as the input current of the neuron:

$$u'_t = \tau u_{t-1} + y_t \tag{8}$$

$$s_t = H(u'_t - v_{\rm th}) \tag{9}$$

The spiking output is then feed into the FC layer of the next spiking S4 block, which undergoes addition operation with the weight matrix, realizing low-energy, sparse synaptic computation. The threshold largely controls the spiking rate of the neuron, inspired by previous works (Rathi and Roy 2021), we set it as a learnable parameter during training to optimize network performance. A comparison of different s4 blocks and the spiking s4 block is shown in Fig. 1. Interestingly, from a neurobiological perspective, the structure of the spiking S4 block resembles a multi-time scale dendritic neuron (London and Häusser 2005; Zheng et al. 2024), with $h$ representing dendrites and $y$ representing the soma which receives collective input from dendrites, both characterized by self-recurrent temporal dynamics.

### Surrogate Dynamic Network

Since $y$ can be calculated in parallel, given an input sequence $y_{1:T}$ to the spiking neuron, under hard reset, $u_t$ with $t \in [1, T]$ can be formulated as:

$$u_t = \sum_{i=1}^{t} \left[ \prod_{j=i}^{t-1} (1 - s_j) \cdot \tau^{t-i} \cdot y_i \right] \tag{10}$$

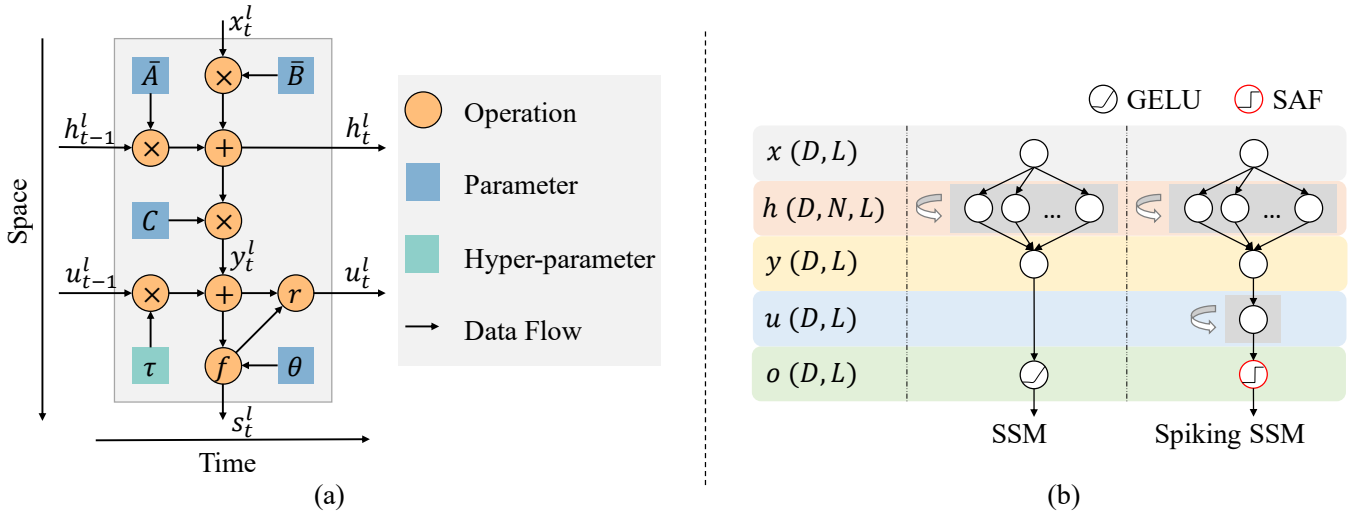


Figure 1: Architecture of SpikingSSM. (a) Forward computation graph of SpikingSSM in one layer. Operation $r$ denotes the reset mechanism. The learnable parameter $\theta$ denotes parameters that influence the spiking function $f$, such as the threshold. (b) Comparison of different SSMs. The original SSM outputs float point number. SpikingSSM replaces the non-linear function of original SSM with an LIF neuron, adding neuronal dynamics on a higher hierarchy. SAF denotes the spiking activation function. The left panel denotes the computation stage of different variables and their corresponding dimensions, with $D, N, L$ denoting the model dimension, the hidden dimension of SSM and the sequence length, respectively.

It can be seen that the membrane potential is determined by the past spiking history of the neuron which can not be computed in parallel, thus SNNs always adopt the form of iterative computing. The nonlineariy of spiking activation, especially the event-driven reset mechanism prevents parallel computing of SNNs, which makes them not practical for efficient training on modern hardware, especially for long sequence tasks. The neural networks, however, are designed for handling the mapping between inputs and outputs, and can be parallelized on modern hardware. Since spiking neurons with fixed parameters must produce the same outputs with the same inputs, the 'neuron' can be considered as a black-box that maps the input to spike sequence, which is exactly what neural networks are good at. Therefore, we propose using a pre-trained neural network, dubbed as Surrogate Dynamic Network, to predict the spike train in parallel. Specifically, we train a network $f$, which learns the neuronal dynamics that maps input to output spike trains. For example, a neural network to predict spike train based on all time-step input can be expressed as:

$$s_{1:T} = f(I_{1:T}) \tag{11}$$

where $I_{1:T}$ is the input current from time-step 1 to $T$, and $s_{1:T}$ is the corresponding spike train predicted by the network $f$.

Meanwhile, for the sake of efficiency, the network should be very small such that the forward inference can be done with low computation cost. As demonstrated in the experiment, a 3-layer network with 1-D convolution is sufficient to learn the neuron dynamics and accurately predict the spike, as shown in Fig. 2 (more details are presented in the experiment section). To further accelerate training and simplify the computational graph, we switch the trained SDN to inference mode without backpropagation during training of the task network, using its predicted spike train and the input to compute the membrane potential as equation 10. Finally, the spike is determined by the membrane potential as the output of the spiking S4 block. During test mode, the SDN can be either kept for parallel inference with linear time complexity with respect to the sequence length, or removed for real-time iterative inference with time complexity $\mathcal{O}(1)$, without adding additional parameters to the network, i.e. spiking neurons switch to the original reset mechanism. Note that in order to reduce the complexity of computational graph, the SDN can also be trained to predict the membrane potential after leaking, i.e., the '$\tau u_{t-1}$' term in equation 8. In this case, the computational graph has a similar form as in spatial learning through time (SLTT) (Meng et al. 2023), which has been demonstrated effective and more efficient than the traditional backpropagation through time (BPTT) for the training of SNN. More details of the derivation are provided in the supplement material.

## Learnable Threshold

The threshold determines the moment of spike generation and largely modulates the spiking rate of SNN. Previous works (Rathi and Roy 2021; WANG, Cheng, and Lim 2022) have shown that optimizing the threshold during the training of SNN can improve network performance. Can SDN approximate neuron dynamics with different threshold during the training of SpikingSSM? We demonstrate that this is feasible through an equivalence study. First, we identify some important properties of the threshold, given that both the initial membrane potential and the reset potential are 0.

**Property 1. The ratio of inputs and threshold determines the dynamic process of the neuron.**

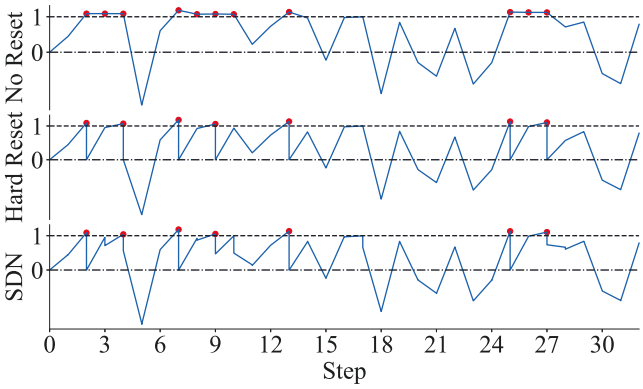


Figure 2: Comparison of membrane potential samples produced by different methods under the same input. The membrane potential predicted by the SDN (bottom) accurately approximates the ground truth produced by the spiking neuron (middle). Without reset the membrane potential significantly produces more spikes (top). The two black dashed lines denote the reset potential and the spiking threshold which are set to 0 and 1, respectively. Red points denote moments when spikes are generated, i.e. the membrane potential surpasses the threshold. Note that for the spiking neuron, the membrane potential is reset to 0 immediately once surpasses the threshold.

In other words, if we scale the threshold and inputs with the same factor, the spike train will remain unchanged. Formally, if $f$ represents the dynamic process of the neuron, we have:

$$s_{1:T} = f(I_{1:T}; v_{\text{th}}) = f(\alpha I_{1:T}; \alpha v_{\text{th}}) \tag{12}$$

**Property 2. The threshold scales the distribution of the input.**

For neurons with different threshold, the threshold functions as a scaling factor, therefore we can get a general SDN that only acts as a 'neuron' with $v_{\text{th}} = 1.0$ by fed scaled inputs $\frac{I_{1:T}}{v_{\text{th}}}$. Therefore, based on these properties, we can incorporate a trainable threshold for SDN by learning a scaling factor for the input.

## Experiments

In this section, we first introduce the architecture design, training and evaluation of SDN. In addition, we benchmark SpikingSSM assisted by SDN against traditional iterative training approaches on training speed. Next, we validate SpikingSSM on three benchmarks tasks of different scales, including classification on the sequential MNIST dataset and its permuted variant, long sequence modeling on the LRA dataset, and language modeling on the WikiText-103 dataset. Finally, we perform ablation studies of our method and analyze the computation cost of the model.

### Training and Evaluation of SDN

**Dataset** The training dataset for SDN contains the input currents and their corresponding target membrane potentials. The inputs $\in R^L$ are sampled from normal distribution $N(0, 1)$, where $L = 1024$ is the sequence length. The ground truth membrane potentials are provided by iterative LIF neuron with hard reset. The number of training and testing samples are $10^5$ and $10^4$, respectively.

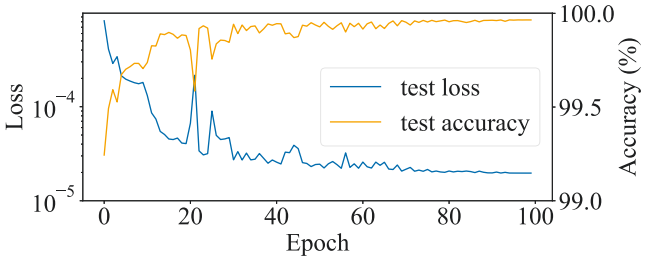


Figure 3: Training of SDN. The MSE loss and spiking accuracy on the test set are plotted here. Note that SDN already achieves sufficiently high accuracy after the first training epoch.

**Architecture of SDN** The SDN is a 4-layer CNN constructed by 1-D convolutions and 1-D batch normalizations, denoted by "C8k1s1p0g1-C8k8s1p8g8-Trunc-BN-relu+C8k1s1p0g1-BN+relu-C1k1s1p0g1", where "C", "k", "s", "p" and "g" denote output channel, kernel size, stride, padding and group, with the numbers following them indicating the value. The term "Trunc" signifies truncating the input to maintain a constant length, and the two "+" symbols denote the start and end of a residual connection.

In this case, the total number of parameters in SDN is less than 200, which is minor compared with the backbone network. More details about the architecture are provided in the supplement material.

**Fitting Ability** We train SDN on the generated dataset with mean square error (MSE) as the loss function for 100 epochs. For testing, we further evaluate the spiking accuracy from the predicted membrane potential by comparing with the spikes generated from the ground truth membrane potential. Fig. 3 shows that the loss of SDN converges and the model gradually attains high spiking accuracy. We also present samples of membrane potentials predicted by SDN for better illustration. As shown in Fig. 2, the membrane potential predicted by SDN closely approximates the ground truth. Without reset, the membrane potential significantly produces more spikes. Note that in some cases the network could mistakenly reset the membrane potential in a minor degree. This occasionally happens when the membrane potential is very close to the threshold, e.g. at the 3rd step. The result proves that SDN can accurately model the membrane potential dynamics of the LIF neuron. Although there is still minor difference between the predicted value and the ground truth, it has negligible impact on the final trained network performance, as demonstrated in the ablation study.

**Comparison on Training Speed** We compare the training speed of SpikingSSM assisted by SDN with traditional training methods based on iterative LIF neurons, including BPTT and the more recent SLTT with optimized computational graph. The inputs are 1-D sequences with varying lengths of $L = 1K, 2K, 4K, 8K$ with batch size of 64. The

| Method | Speed (ms) | | | |
|---|---|---|---|---|
| | $L = 1K$ | $L = 2K$ | $L = 4K$ | $L = 8K$ |
| BPTT | 1370 | 2900 | 8040 | 25600 |
| SLTT | 1210 | 2720 | 7740 | 25600 |
| Ours | 183 | 196 | 200 | 253 |
| Ratio | 7.5× | 15.0× | 40.2× | 101.2× |

Table 1: Comparison on training speed of different methods. The input has a batch size of 64. Training with SDN achieves significant acceleration, the speed up ratio amplifies with increasing sequence length.

.

time measurement is done on a single GPU. As shown in Table 1, the speed up ratio using SDN amplifies with increasing sequence length, achieving two orders of acceleration at $8K$. Therefore, SDN extremely accelerates the training of SpikingSSM, especially for long sequences.

## Long Sequence Tasks with SpikingSSM

**Sequential MNIST** The MNIST dataset (Yann and Cortes 1998) comprises 70,000 grayscale images of handwritten digits (0-9), divided as 60,000 training and 10,000 testing images each with a size of 28×28 pixels. The sequential MNIST (sMNIST) dataset (Le, Jaitly, and Hinton 2015) is created by flattening the original 2-dimensional images into sequences of 784 elements. And the permuted sequential MNIST (psMNIST) variant (Le, Jaitly, and Hinton 2015) applied a fixed permutation to the pixels, thereby distorting the temporal structure within the sequence. As shown in table 2, SpikingSSM demonstrates competitive performance with other works on both sMNIST and psMNIST datasets.

| Model | SNN | sMNIST | psMNIST |
|---|---|---|---|
| LMUformer | No | — | 98.55 |
| S4 | No | 99.63 | 98.70 |
| SpikingLMUformer | Yes | — | 97.92 |
| Binary-S4D | Yes | 99.4 | — |
| S6-based SNN | Yes | — | 98.4 |
| SpikingSSM | Yes | 99.6 | 98.4 |

Table 2: Performance comparison of SpikingSSM and other works on sMNIST and psMNIST datasets.

**LRA** The LRA benchmark (Tay et al. 2021b) is proposed for the purpose of benchmarking sequence models under the long-context scenario. LRA comprises six tasks featuring sequences that range from $1K$-$16K$ steps, spanning various modalities such as visual data, mathematics experssions, and text. These tasks are designed to assess model abilities in long-context understanding including text classification, document retrieval, image classification, pathfinder, and listops. Table 3 compares SpikingSSM against both non-spiking and spiking networks with transformer or SSM architectures. The SpikingSSM adopts a similar architecture as the original S4D model with parameter initialization as in S4D-Lin (Gu et al. 2024) (architecture details are provided in the supplement material). While maintaining a level of accuracy comparable to that of the original model, the SpikingSSM achieves almost 90% of average network sparsity. Our model also demonstrates a significant performance improvement over previous SNN sequence models. Notably, SpikingSSM successfully tackles the Path-X task, a highly challenging problem that requires reasoning over long-range dependencies within sequences of length 128 × 128, totaling 16,384 steps. Our SpikingSSM with a trainable threshold shows better overall performance and sparsity compared to a fixed threshold. Through further analysis, we find that the trainable threshold facilitates a bimodal distribution of the membrane potential, which reduces quantization error of spiking and improves information transmission of the SNN, consistent with previous findings (Guo et al. 2022) (details are provided in the supplement material).

**WikiText-103** The WikiText-103 dataset is a comprehensive collection of text from Wikipedia articles rated as Good or Featured, consisting of over 100 million tokens spanning a diverse range of topics and domains. We adopted the commonly used perplexity as the metric. Due to its composition of full articles, this dataset is particularly well-suited for models designed to capture long-term dependencies, making it a critical benchmark for word-level language modeling. In our experiments, we adopted a more parameter-efficient setup compared to the S4 model (details are provided in the supplement material). Despite utilizing significantly fewer parameters, the SpikingSSM, not only outperforms the pre-trained SpikeGPT, but also substantially narrows the performance gap with ANN networks.

## Ablation Study

To verify the important roles of SDN, we conduct an ablation study on whether replacing LIF neurons with SDN in SpikingSSM during training causes performance degradation. In addition, as a pre-trained network, SDN has learned to model the dynamics of LIF neurons, and this bias restricts SDN to act as the LIF neuron, but does this bias really help the performance of SpikingSSM? We build three models with identical architecture and same hyperparameters, expect the spiking activation function. 'LIF' uses the iterative LIF neuron, 'SDN-S' uses SDN that is trained from scratch with the SpikingSSM end-to-end, and 'SDN' uses the fixed pre-trained SDN. We train these three models on sCIFAR10 dataset (the IMAGE subset in LRA). Table 5 shows the results of these three models. The 'SDN' model achieves comparable performance to the iterative LIF neuron and greatly accelerates training. The 'SDN-S' model fail in achieving comparable performance to 'SDN', demonstrating that the bias of restricting SDN to act as the LIF neuron is beneficial.

## Computation Cost

Spiking networks are considered low energy cost because the activation of spiking neurons are binary value, and the multiplication between binary activation value and float number weight can be done via only addition operation in some neuromorphic chips, e.g., Speck(Richter et al. 2023). Therefore, the major operation synaptic Accumulation (AC)

| Model | SNN | LISTOPS | TEXT | RETRIEVAL | IMAGE | PATHFINDER | Path-X | AVG |
|---|---|---|---|---|---|---|---|---|
| Transformer | No | 36.37 | 64.27 | 57.46 | 42.44 | 71.40 | — | 53.66 |
| LMUFormer | No | 34.43 | 68.27 | 78.65 | 54.16 | 69.90 | — | 59.24 |
| S4D-Lin | No | **60.52** | **86.97** | **90.96** | **87.93** | **93.96** | **92.80*** | **85.52** |
| Spiking LMUFormer | Yes | 37.30 | 65.80 | 79.76 | 55.65 | 72.68 | — | 60.20 |
| Binary S4D | Yes | 54.80 | **82.50** | 85.03 | 82.00 | 82.60 | 61.20 | 74.69 |
| S6-based SNN | Yes | 55.70 | 77.62 | 88.48 | 80.10 | 83.41 | — | 72.55 |
| SpikingSSM-VF | Yes | 59.93 | 82.35 | 88.20 | 86.81 | **93.68** | 94.80 | 84.30 |
| (spiking rate) | | (0.13) | (0.10) | (0.06) | (0.22) | (0.07) | (0.10) | (0.11) |
| SpikingSSM-VT | Yes | **60.23** | 80.41 | **88.77** | **88.21** | 93.51 | **94.82** | **84.33** |
| (spiking rate) | | (0.14) | (0.06) | (0.06) | (0.15) | (0.08) | (0.10) | (0.10) |

Table 3: Performance comparison of SpikingSSM and previous works on the LRA dataset. *Since the original S4D-Lin failed in the Path-X task, we instead present the result of another close variant S4D-Inv. -VF and -VT denote fixed and trainable threshold, respectively. Furthermore, we take the 50% accuracy for the absence of Path-X accuracy as did in the work of S4D, then compute the overall average metrics across all tasks as AVG. The spiking rate for each task have also been calculated, which is indicated by blue font.

| Model | SNN | PPL | Parameters |
|---|---|---|---|
| Transformer | No | 20.51 | 231M |
| S4 | No | 20.95 | 249M |
| SpikeGPT | Yes | 39.75 | 213M |
| SpikingSSM | Yes | 33.94 | 75M |

Table 4: Performance comparison of SpikingSSMs with previous works on WikiText-103 dataset.

| Model | Accuracy (%) | Spiking Rate (%) | Speed (ms) |
|---|---|---|---|
| LIF | 85.45 | 12.08 | 1480 |
| SDN | 85.57 | 11.92 | 230 |
| SDN-S | 81.52 | 18.30 | 285 |

Table 5: Performance comparison on the sCIFAR10 dataset.

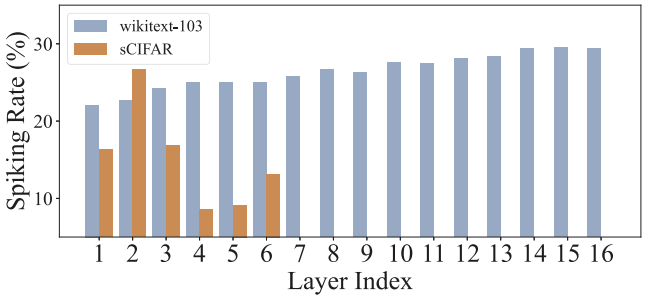


Figure 4: Spiking rate across all layers of SpikingSSMs on the sCIFAR10 and the WikiText-103 datasets.

in SNN has lower energy cost compared to the major operation Multiply-and-Accumulate (MAC) in ANN. Although the hardware implementation and the dynamics of spiking neurons are ignored, a theoretical energy consumption analysis gives an estimation of the efficiency of SNN. Refer to previous works(Richter et al. 2023; Li et al. 2024a), we assume the energy cost of MAC $E_{\mathrm{MAC}} = 4.6pJ$ and AC $E_{\mathrm{AC}} = 0.9pJ$(Horowitz 2014).

We define spiking rate as the ratio of the number of spikes to the total time steps of a neuron; the mean spiking rate of the whole network is the mean of spiking rate of all neurons in network. We denote spiking rate as the mean spiking rate. Fig. 4 shows the spiking rate of each layers. Note that the parameters and computation are mainly from the feature-mix layers, we list the MAC, AC and energy cost in these layers. For the WikiText-103 dataset with sample length $L = 8192$, our model has 16 layers, in which a linear layer projecting the spikes from $d = 1024$ to $d = 2048$. If all projections are done via multiplication between float numbers, it contains $275.2G$ MAC, and requires about $1.265J$. However, the inputs of these layers are binary numbers in our model, and the average spiking rate is less than 30%. According to the spiking rates in Fig. 4, our model contains $72.66G$ AC, and requires about $65.40mJ$.

## Conclusion

In conclusion, by hierarchically integrating the LIF neuronal dynamics with SSMs, we propose the SpikingSSM which shows competitive performance in long-sequence learning with efficient sparse computation of SNNs. For the efficient training of SNN with iterative LIF neurons, we propose a surrogate dynamic network to approximate the dynamics of LIF neurons with parallel computing, which extremely accelerates the training of SpikingSSMs. The SDN is switched to inference mode in training the task spiking networks with only negligible additional computation. We also demonstrate the versatility of SDN for approximating parametric LIF neuron models and its potential as general purpose module for parallel computing SNNs. The application of SpikingSSMs to various benchmark tasks, including the LRA and WikiText-103, not only showcase their competitive performance against previous works but also emphasizes their advantages in sparsity and low-energy requirements. This study contributes to the broader applicability of spiking neural networks, especially in fields requiring efficient processing of long sequence data.

## Acknowledgments

This work is supported in part by National Key Research and Development Program of China (2021YFF1200800), the National Natural Science Foundation of China (Grant No. 62276121, 12326604) and the Science and Technology Innovation 2030-Major Project (Brain Science and Brain-Like Intelligence Technology) under Grant 2022ZD0208700.

## References

Bal, M.; and Sengupta, A. 2024. Rethinking Spiking Neural Networks as State Space Models. arXiv:2406.02923.

Bellec, G.; Salaj, D.; Subramoney, A.; Legenstein, R.; and Maass, W. 2018. Long short-term memory and learning-to-learn in networks of spiking neurons. In *The Thirty-second Conference on Neural Information Processing Systems*. Curran Associates Inc.

Che, K.; Leng, L.; Zhang, K.; Zhang, J.; Meng, Q.; Cheng, J.; Guo, Q.; and Liao, J. 2022. Differentiable hierarchical and surrogate gradient search for spiking neural networks. In *The Thirty-sixth Conference on Neural Information Processing Systems*, 24975–24990. Curran Associates Inc.

Che, K.; Zhou, Z.; Yuan, L.; Zhang, J.; Tian, Y.; and Leng, L. 2024. Spatial-Temporal Search for Spiking Neural Networks. *arXiv preprint arXiv:2410.18580*.

Choromanski, K. M.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlos, T.; Hawkins, P.; Davis, J. Q.; Mohiuddin, A.; Kaiser, L.; Belanger, D. B.; Colwell, L. J.; and Weller, A. 2021. Rethinking Attention with Performers. In *International Conference on Learning Representations*.

Dao, T. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Dao, T.; Fu, D.; Ermon, S.; Rudra, A.; and Ré, C. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *The Thirty-sixth Conference on Neural Information Processing Systems*, 16344–16359. Curran Associates Inc.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Du, Y.; Liu, X.; and Chua, Y. 2024. Spiking structured state space model for monaural speech enhancement. In *2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, 766–770. IEEE.

Duhamel, P.; and Vetterli, M. 1990. Fast Fourier transforms: a tutorial review and a state of the art. *Signal processing*, (4): 259–299.

Esser, S. K.; Merolla, P. A.; Arthur, J. V.; Cassidy, A. S.; Appuswamy, R.; Andreopoulos, A.; Berg, D. J.; McKinstry, J. L.; Melano, T.; Barch, D. R.; di Nolfo, C.; Datta, P.; Amir, A.; Taba, B.; Flickner, M. D.; and Modha, D. S. 2016. Convolutional networks for fast, energy-efficient neuromorphic computing. *Proceedings of the National Academy of Sciences*, 113: 11441–11446.

Fang, W.; Yu, Z.; Zhou, Z.; Chen, D.; Chen, Y.; Ma, Z.; Masquelier, T.; and Tian, Y. 2023. Parallel Spiking Neurons with High Efficiency and Ability to Learn Long-term Dependencies. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 53674–53687. Curran Associates, Inc.

Fu, D. Y.; Dao, T.; Saab, K. K.; Thomas, A. W.; Rudra, A.; and Ré, C. 2023. Hungry hungry hippos: Towards language modeling with state space models. In *The eleventh International Conference on Learning Representations*.

Gerstner, W.; Kistler, W. M.; Naud, R.; and Paninski, L. 2014. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gu, A.; and Dao, T. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint arXiv:2312.00752*.

Gu, A.; Dao, T.; Ermon, S.; Rudra, A.; and Ré, C. 2020. Hippo: Recurrent memory with optimal polynomial projections. In *The Thirty-fourth Conference on Neural Information Processing Systems*, 1474–1487. Vancouver,Canada: Curran Associates Inc.

Gu, A.; Goel, K.; and Re, C. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. In *International Conference on Learning Representations*.

Gu, A.; Gupta, A.; Goel, K.; and Ré, C. 2024. On the parameterization and initialization of diagonal state space models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713871088.

Guo, Y.; Tong, X.; Chen, Y.; Zhang, L.; Liu, X.; Ma, Z.; and Huang, X. 2022. Recdis-snn: Rectifying membrane potential distribution for directly training spiking neural networks. In *2022 IEEE/CVF conference on computer vision and pattern recognition*, 326–335. IEEE.

Gupta, A.; Gu, A.; and Berant, J. 2022. Diagonal state spaces are as effective as structured state spaces. In *The Thirty-sixth Conference on Neural Information Processing Systems*, 22982–22994. Curran Associates Inc.

Horowitz, M. 2014. 1.1 Computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 10–14. IEEE.

Katharopoulos, A.; Vyas, A.; Pappas, N.; and Fleuret, F. 2020. Transformers are RNNs: fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Kitaev, N.; Kaiser, L.; and Levskaya, A. 2020. Reformer: The Efficient Transformer. In *International Conference on Learning Representations*.

Le, Q. V.; Jaitly, N.; and Hinton, G. E. 2015. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. arXiv:1504.00941.

Li, B.; Leng, L.; Shen, S.; Zhang, K.; Zhang, J.; Liao, J.; and Cheng, R. 2024a. Efficient Deep Spiking Multilayer Perceptrons With Multiplication-Free Inference. *IEEE Transactions on Neural Networks and Learning Systems*, 1–13.

Li, Y.; Sun, Y.; He, X.; Dong, Y.; Zhao, D.; and Zeng, Y. 2024b. Parallel Spiking Unit for Efficient Training of Spiking Neural Networks. In *2024 International Joint Conference on Neural Networks*, 1–8.

Lipton, Z. C.; Berkowitz, J.; and Elkan, C. 2015. A Critical Review of Recurrent Neural Networks for Sequence Learning. arXiv:1506.00019.

Liu, Z.; Datta, G.; Li, A.; and Beerel, P. A. 2024. LMUFormer: Low Complexity Yet Powerful Spiking Model With Legendre Memory Units. *arXiv preprint arXiv:2402.04882*.

London, M.; and Häusser, M. 2005. Dendritic computation. *Annu. Rev. Neurosci.*, 28: 503–532.

Lv, C.; Li, T.; Xu, J.; Gu, C.; Ling, Z.; Zhang, C.; Zheng, X.; and Huang, X. 2024. SpikeBERT: A Language Spikformer Learned from BERT with Knowledge Distillation. arXiv:2308.15122.

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

Mehta, H.; Gupta, A.; Cutkosky, A.; and Neyshabur, B. 2023. Long range language modeling via gated state spaces. In *The eleventh International Conference on Learning Representations*.

Meng, Q.; Xiao, M.; Yan, S.; Wang, Y.; Lin, Z.; and Luo, Z.-Q. 2023. Towards memory-and time-efficient backpropagation for training spiking neural networks. In *2023 IEEE/CVF International Conference on Computer Vision*, 6166–6176. Paris, France.

Peng, B.; Alcaide, E.; Anthony, Q.; Albalak, A.; Arcadinho, S.; Biderman, S.; Cao, H.; Cheng, X.; Chung, M.; Derczynski, L.; Du, X.; Grella, M.; Gv, K.; He, X.; Hou, H.; Kazienko, P.; Kocon, J.; Kong, J.; Koptyra, B.; Lau, H.; Lin, J.; Mantri, K. S. I.; Mom, F.; Saito, A.; Song, G.; Tang, X.; Wind, J.; Woźniak, S.; Zhang, Z.; Zhou, Q.; Zhu, J.; and Zhu, R.-J. 2023. RWKV: Reinventing RNNs for the Transformer Era. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 14048–14077. Singapore: Association for Computational Linguistics.

Qin, Z.; Li, D.; Sun, W.; Sun, W.; Shen, X.; Han, X.; Wei, Y.; Lv, B.; Luo, X.; Qiao, Y.; and Zhong, Y. 2024. TransNormerLLM: A Faster and Better Large Language Model with Improved TransNormer. arXiv:2307.14995.

Rathi, N.; and Roy, K. 2021. Diet-snn: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 34: 3174–3182.

Richter, O.; Xing, Y.; Marchi, M. D.; Nielsen, C.; Katsimpris, M.; Cattaneo, R.; Ren, Y.; Liu, L.-Y. D.; Sheik, S.; Demirci, T.; NingQiaoSynSense, A.; Swizerland; SynSense; China, P. R.; Circuits, B.-I.; Lab, S.; for Advanced Materials, Z. I.; of Groningen, U.; Netherlands; Systems, G. C.; Center, M. S.; and Netherlands. 2023. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15.

Schuster, M.; and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45: 2673–2681.

Sherstinsky, A. 2020. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404: 132306.

Smith, J. T.; Warrington, A.; and Linderman, S. W. 2023. Simplified state space layers for sequence modeling. In *The eleventh International Conference on Learning Representations*.

Stan, M.-I.; and Rhodes, O. 2024. Learning long sequences in spiking neural networks. *Scientific Reports*, 14: 21957.

Sun, Y.; Dong, L.; Huang, S.; Ma, S.; Xia, Y.; Xue, J.; Wang, J.; and Wei, F. 2023. Retentive network: A successor to transformer for large language models. arXiv:2307.08621.

Tay, Y.; Dehghani, M.; Abnar, S.; Shen, Y.; Bahri, D.; Pham, P.; Rao, J.; Yang, L.; Ruder, S.; and Metzler, D. 2021a. Long Range Arena : A Benchmark for Efficient Transformers. In *International Conference on Learning Representations*.

Tay, Y.; Dehghani, M.; Abnar, S.; Shen, Y.; Bahri, D.; Pham, P.; Rao, J.; Yang, L.; Ruder, S.; and Metzler, D. 2021b. Long Range Arena : A Benchmark for Efficient Transformers. In *The Ninth International Conference on Learning Representations*. Vienna,Austria.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, 6000–6010. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781510860964.

Voelker, A.; Kajić, I.; and Eliasmith, C. 2019. Legendre memory units: Continuous-time representation in recurrent neural networks. In *The Thirty-third Conference on Neural Information Processing Systems*. Curran Associates Inc.

WANG, S.; Cheng, T. H.; and Lim, M.-H. 2022. LTMD: Learning Improvement of Spiking Neural Networks with Learnable Thresholding Neurons and Moderate Dropout. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 28350–28362. Curran Associates, Inc.

Yang, S.; Wang, B.; Shen, Y.; Panda, R.; and Kim, Y. 2023. Gated linear attention transformers with hardware-efficient training. arXiv:2312.06635.

Yann, L.; and Cortes, C. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Yao, M.; Hu, J.; Hu, T.; Xu, Y.; Zhou, Z.; Tian, Y.; XU, B.; and Li, G. 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*. Vienna Austria.

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

Yin, B.; Corradi, F.; and Bohté, S. M. 2023. Accurate online training of dynamical spiking neural networks through forward propagation through time. *Nature Machine Intelligence*, 5: 518–527.

Zaheer, M.; Guruganesh, G.; Dubey, A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; and Ahmed, A. 2020. Big bird: transformers for longer sequences. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713829546.

Zheng, H.; Wu, Y.; Deng, L.; Hu, Y.; and Li, G. 2021. Going Deeper With Directly-Trained Larger Spiking Neural Networks. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, 11062–11070. online: AAAI Press.

Zheng, H.; Zheng, Z.; Hu, R.; Xiao, B.; Wu, Y.; Yu, F.; Liu, X.; Li, G.; and Deng, L. 2024. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15: 277.

Zhou, Z.; Zhu, Y.; He, C.; Wang, Y.; Yan, S.; Tian, Y.; and Yuan, L. 2022. Spikformer: When spiking neural network meets transformer. arXiv:2209.15425.

Zhu, R.-J.; Zhao, Q.; Li, G.; and Eshraghian, J. K. 2023. Spikegpt: Generative pre-trained language model with spiking neural networks. arXiv:2302.13939.