High-Performance Temporal Reversible Spiking Neural Networks with $\mathcal{O}(L)$ Training Memory and $\mathcal{O}(1)$ Inference Cost

Jiakui Hu[*1] **Yao Man**[*2] **Xuerui Qiu**[2 3] **Yuhong Chou**[4] **Yuxuan Cai**[5] **Ning Qiao**[6] **Yonghong Tian**[1 7] **Bo Xu**[2] **Guoqi Li**[† 2 8]

Abstract

Multi-timestep simulation of brain-inspired Spiking Neural Networks (SNNs) boost memory requirements during training and increase inference energy cost. Current training methods cannot simultaneously solve both training and inference dilemmas. This work proposes a novel Temporal Reversible architecture for SNNs (T-RevSNN) to jointly address the training and inference challenges by altering the forward propagation of SNNs. We turn off the temporal dynamics of most spiking neurons and design multi-level temporal reversible interactions at temporal turn-on spiking neurons, resulting in a $\mathcal{O}(L)$ training memory. Combined with the temporal reversible nature, we redesign the input encoding and network organization of SNNs to achieve $\mathcal{O}(1)$ inference energy cost. Then, we finely adjust the internal units and residual connections of the basic SNN block to ensure the effectiveness of sparse temporal information interaction. T-RevSNN achieves excellent accuracy on ImageNet, while the memory efficiency, training time acceleration, and inference energy efficiency can be significantly improved by $8.6\times$, $2.0\times$, and $1.6\times$, respectively. This work is expected to break the technical bottleneck of significantly increasing memory cost and training time for large-scale SNNs while maintaining high performance and low inference energy cost. Source code and models are available at: https://github.com/BICLab/T-RevSNN.

[*]Equal contribution [1]Peking University, Beijing, China [2]Institute of Automation, Chinese Academy of Sciences, Beijing, China [3]School of Future Technology, University of Chinese Academy of Sciences [4]Department of Computing, The Hong Kong Polytechnic University, HongKong, China [5]01.AI, Beijing, China [6]SynSense AG Corporation, Zurich, Switzerland [7]Peng Cheng Laboratory, Shenzhen, Guangzhou, China [8]Key Laboratory of Brain Cognition and Brain-inspired Intelligence Technology, Beijing, China. Correspondence to: Guoqi Li <guoqi.li@ia.ac.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Spiking Neural Networks (SNNs) are promised to be a low-power alternative to Artificial Neural Networks (ANNs), thanks to their brain-inspired neuronal dynamics and spike-based communication (Roy et al., 2019; Schuman et al., 2022). Spiking neurons fuse spatio-temporal information through intricate dynamics and fire binary spikes as outputs when their membrane potential crosses the threshold (Maass, 1997). Spike-based communication enables the event-driven computational paradigm of SNNs and has extremely high power efficiency in neuromorphic chips such as TrueNorth (Merolla et al., 2014), Loihi (Davies et al., 2018), Tianjic (Pei et al., 2019), and Speck (Yao et al., 2024b).

By discretizing the nonlinear differential equation of spiking neurons into iterative versions (Wu et al., 2018; Neftci et al., 2019), SNNs can be trained directly using BackPropagation Through Time (BPTT) and surrogate gradient (Wu et al., 2018; Neftci et al., 2019). Multi-timestep simulation requires a memory of $\mathcal{O}(L \times T)$ for SNN training, where L is the layer and T is the timestep. In particular, in static vision tasks, such as image classification, it is common practice (Wu et al., 2019; Deng et al., 2020; Kim et al., 2022; Yao et al., 2023d; Rathi & Roy, 2023) to exploit the dynamics of SNNs by repeating the inputs to achieve high performance. In this case, the inference energy is $\mathcal{O}(T)$.

As an example, training spiking ResNet-19 with 10 timestep takes about $20\times$ more memory than ResNet-19 (Fang et al., 2023). Numerous efforts have been made to solve the memory dilemma. The mainstream idea is to decouple the training of SNNs from the timestep, e.g., modify the way the BPTT in the temporal (Xiao et al., 2022; Meng et al., 2023; Eshraghian et al., 2023), training with $T = 1$ and fine-tune to multiple timesteps (Lin et al., 2022; Yao et al., 2023d). With respect to scaling down timesteps during inference, one idea is to fine-tune a trained multi-timestep model to a single timestep (Chowdhury et al., 2021), or employ additional controllers to adjust inference timesteps adaptively (Yao et al., 2021; Li et al., 2023; Ding et al., 2024).

However, none of them can simultaneously achieve cheap training memory and low inference energy because they tend to optimize only in one direction. In this work, we

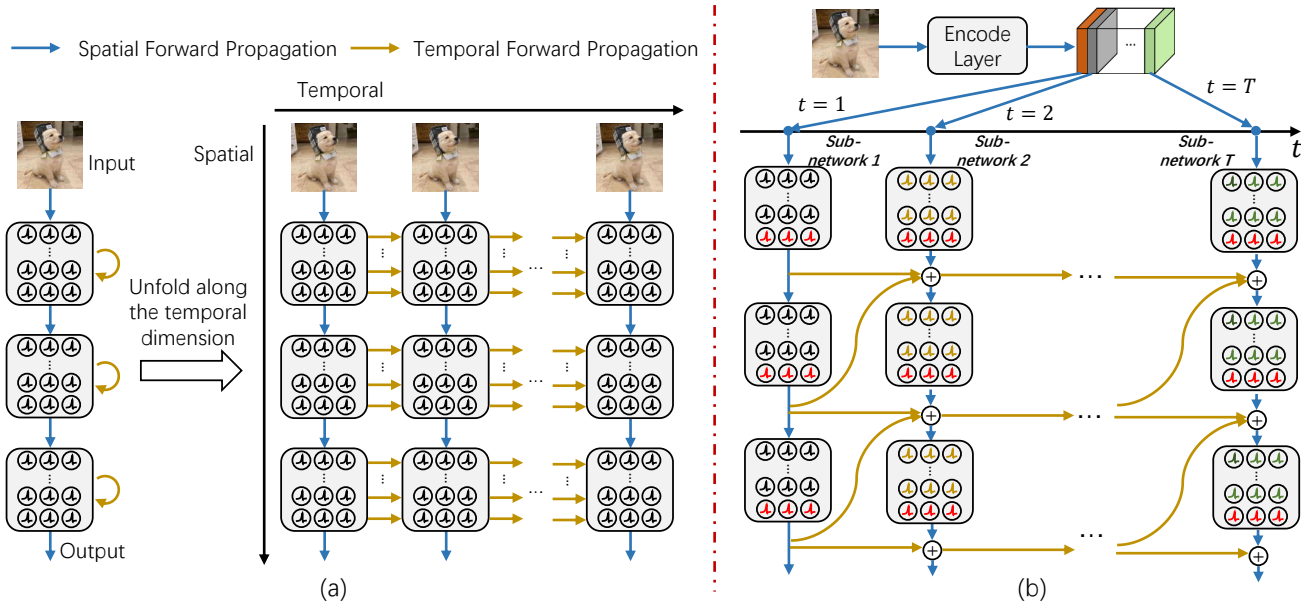

Figure 1. Illustration of the temporal forward of vanilla SNN and T-RevSNN. (a) Vanilla SNNs unfold along the temporal, reusing all parameters at each timestep. All spiking neurons accomplish the temporal dynamics. In the image classification task, images are input repeatedly. Thus, the memory and inference costs of vanilla SNNs are $\mathcal{O}(L \times T)$ and $\mathcal{O}(T)$, respectively. (b) In T-RevSNN, we only allow the key spiking neurons (red spiking neurons in the figure) to pass temporal information. Coupled with the multi-level temporal reversible design, the memory cost of T-RevSNN is $\mathcal{O}(L)$. Moreover, the image is encoded only once. The encoded features are divided into T groups, exploited as input for each timestep. Correspondingly, the entire SNN is also divided into T independent sub-networks, which only share parameters and transfer temporal information at key spiking neurons. Thus, the inference of T-RevSNN is $\mathcal{O}(1)$.

can shoot two birds with one stone via a novel temporal reversible architecture for SNNs. Our motivation is simple and intuitive. It has been shown that the BPTT of SNNs through the temporal contributes just a little to the final gradients. Xiao et al. (2022) and Meng et al. (2023) therefore do not calculate temporal gradients to improve training speed. That being the case, *can we only retain the temporal forward at key positions and turn off the temporal dynamics of other spiking neurons?*

Based on this simple idea, we design temporal reversible SNNs (Fig. 1(b)). *First*, to reduce training memory, we only turn on the temporal dynamics of the output spiking layer of each stage and design the temporal transfer to be reversible. Specifically, reversibility (Gomez et al., 2017) means that we do not need to store the membrane potentials and activations of all spiking neurons to compute gradients. *Second*, since turned-off spiking neurons have no temporal dynamics, we simply let these neurons no longer reuse parameters in the temporal dimension. In order not to increase parameters and energy cost, we encode the input only once and divide the input features and the entire network into T groups (Fig. 1(b)). *Third*, to improve performance, the information transfer between stages is carried out in a multi-level form. We also redesign the SNN blocks in the style of ConvNeXt (Liu et al., 2022) and adjust the residual connection to ensure the effectiveness of information transfer. In summary, the backward propagation of SNNs will be altered by their forward, resulting in cheap memory and inference costs concurrently.

We verify the effect of T-RevSNN on static (ImageNet-1k (Deng et al., 2009)) and neuromorphic datasets (CIFAR10-DVS (Li et al., 2017) and DVS128 Gesture (Amir et al., 2017)). Our contributions are:

- We redesign the forward propagation of SNNs simply and intuitively to simultaneously achieve low training memory, low power, and high performance.
- We have made systematic designs in three aspects to implement the proposed idea, including multi-level temporal-reversible forward information transfer of key spiking neurons, group design of input encoding and network architecture, and improvements in SNN blocks and residual connections.
- On ImageNet-1k, our model achieves state-of-the-art accuracy on CNN-based SNNs with minimal memory and inference cost and the fastest training. Compared to current Transformer-based SNN with state-of-the-art performance, i.e., spike-driven Transformer (Yao et al., 2023b), our model achieves close accuracy, while the memory efficiency, training time acceleration, and inference energy efficiency can be significantly improved by $8.6\times$, $2.0\times$ and $1.6\times$, respectively.

2. Related Work

Training of SNNs. The two primary deep learning techniques for SNNs are ANN-to-SNN conversion (ANN2SNN) (Wu et al., 2021a; Hu et al., 2023a) and direct training (Wu et al., 2018; Neftci et al., 2019). The ANN2SNN method approximates the activations in ANNs using firing rates in SNNs. It can achieve accuracy close to ANNs, but correct firing rate estimation takes a lot of timesteps (Sengupta et al., 2019; Rathi et al., 2020; Hu et al., 2023b; Wu et al., 2021a; Wang et al., 2023b; Guo et al., 2023). The direct training approach uses the surrogate gradient to solve the non-differentiable problem caused by binary spike firing. Through the BPTT, SNNs' training only requires a few timesteps. There is a performance gap between directly training SNNs and ANNs, which has been shrinking (Fang et al., 2021a; Hu et al., 2024; Yao et al., 2023d; Zhou et al., 2023; Yao et al., 2023b; Qiu et al., 2023). In this work, we focus on direct training.

Improvement of Architecture and Training in SNNs. SNNs have become larger and deeper. This is due to improvements in architecture and training. Spiking ResNet (Hu et al., 2024; Fang et al., 2021a) and Transformer (Zhou et al., 2023; Yao et al., 2023b) are two benchmark architectures, inspired by classic ResNet (He et al., 2016b;a) and Transformer (Vaswani et al., 2017; Dosovitskiy et al., 2021), respectively. Various training methods have been proposed to improve accuracy, such as normalization (Zheng et al., 2021; Kim & Panda, 2021; Duan et al., 2022), gradient optimization (Perez-Nieves & Goodman, 2021; Lian et al., 2023), attention (Yao et al., 2021; Qiu et al., 2024; Deng et al., 2024; Yao et al., 2023a;c), membrane optimization (Guo et al., 2022a;b;c). The memory required to train large-scale SNNs is much higher than that of ANNs because of the storage of activation values over multiple timesteps. Some works are devoted to reducing the memory cost for training, including Online Training Through Time (**OTTT**) (Xiao et al., 2022), Spatial Learning Through Time (**SLTT**) (Meng et al., 2023), etc. We compare our T-RevSNN with these training optimization methods in Section 4.4.

Reversible Neural Networks. The goal of reversible neural networks (Gomez et al., 2017) is to reduce memory cost when training the network. The core idea is that the activations of each layer can be calculated based on the subsequent reversible layer's activations. This enables a reversible neural network to perform backpropagation without storing the activations in memory, with the exception of a few non-reversible layers. The reversible idea has been applied to various architectures, such as Recurrent Neural Networks (RNNs) (MacKay et al., 2018), ResNet (Sander et al., 2021; Cai et al., 2023), vision transformer (Mangalam et al., 2022), U-Net (Brügger et al., 2019), etc. Recently, the idea of reversibility has been applied to SNN (Zhang & Zhang, 2024), which is spatially reversible and we named it **S-RevSNN**. In contrast, our method is temporal-reversible.

3. Preliminaries

Spiking neurons have neuronal dynamics that enable spatio-temporal information processing. The dynamics of spiking neurons can be characterized (Xiao et al., 2022):

$$V^l[t+1] = (1-\frac{1}{\tau_m})V^l[t] + \frac{1}{\tau_m}\mathbf{W}^l S^{l-1}[t+1], \quad (1)$$

$$S^l[t+1] = \Theta(V^l[t+1] - V_{th}), \quad (2)$$

where $l = 1, \cdots, L$ is the layer index, t is the timestep, $S^{l-1}[t]$ and $V^l[t-1]$ are the spatial input spikes and the temporal membrane potential input respectively, $\mathbf{W}^l$ is synaptic weight, τ_m is the decay factor, V_{th} is the threshold. The Heaviside step function is denoted by $\Theta(x)$. For every $x \geq 0$, $\Theta(x) = 1$; otherwise, $\Theta(x) = 0$. The input spatio-temporal information is incorporated into $V^l[t]$ and then compared with the threshold to decide whether to fire spikes.

Spatio-Temporal BackPropagation (STBP) is a combination of BPTT and surrogate gradient (Wu et al., 2018; Neftci et al., 2019), where the latter is used to solve the non-differentiable of spike signals. Specifically, the gradients for weight $\mathbf{W}^l$ at timestep T is calculated by

$$\nabla_{\mathbf{W}^l}\mathcal{L} = \sum_{t=1}^{T}(\frac{\partial \mathcal{L}}{\partial V^l[t]})^\top S^{l-1}[t]^\top \quad (3)$$

where $\mathcal{L}$ is the loss. It can be calculated by:

$$\frac{\partial \mathcal{L}}{\partial V^l[t]} = \frac{\partial \mathcal{L}}{\partial S^l[t]}\frac{\partial S^l[t]}{\partial V^l[t]} + \frac{\partial \mathcal{L}}{\partial V^l[t+1]}\frac{\partial V^l[t+1]}{\partial V^l[t]}, \quad (4)$$

where

$$\frac{\partial \mathcal{L}}{\partial S^l[t]} = \frac{\partial \mathcal{L}}{\partial V^{l+1}[t]}\frac{\partial V^{l+1}[t]}{\partial S^l[t]} + \frac{\partial \mathcal{L}}{\partial V^l[t+1]}\frac{\partial V^l[t+1]}{\partial S^l[t]}. \quad (5)$$

The non-differentiable term $\frac{\partial S^l[t]}{\partial V^l[t]}$ is the spatial gradient. It can be replaced by the surrogate function. $\frac{\partial V^l[t+1]}{\partial S^l[t]}$ and $\frac{\partial V^l[t+1]}{\partial V^l[t]}$ are the temporal gradients that need to be calculated. According to Eq. 3 and Eq. 4, exploiting the STBP algorithm requires storing the activations and membrane potentials of all spiking neurons at all timesteps.

4. Methods

4.1. Motivation

Previous works (Xiao et al., 2022; Meng et al., 2023) observed that the spatial gradients dominate the final calculation of derivatives. Thus, they abandon the calculation

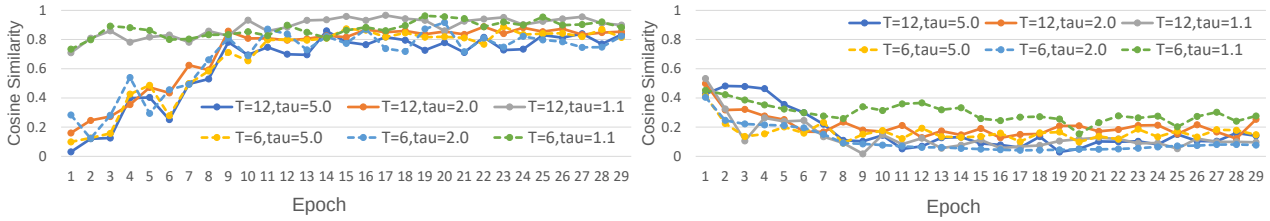

Figure 2. Left and Right sub-figures are the cosine similarity between the spatial gradients calculated by baseline and case 1/2, respectively. In case 1/2, we only retain/discard the temporal gradients of the last layer of spiking neurons in each stage. We use spiking ResNet-18 to train on CIFAR-10. T and τ are Timestep and decay, respectively. The temporal gradients of the final layer of each stage are more significant (Case 1, left sub-figure, high cosine similarity) than those of spiking neurons in preceding stages (Case 2, right sub-figure, low cosine similarity). Due to space constraints, cosine similarity calculation details are given in the supplementary material.

of temporal gradients to speed up training. In contrast, our idea is to turn off the forward propagation of most spiking neurons in time, so that the computation of the temporal gradient is naturally drastically reduced. Building upon these insights, we analyze which spiking neurons are relatively important for temporal information transfer.

We set up the following experiments. We first use STBP to train SNN on CIFAR-10. The calculated gradient is called the "baseline gradient", which needs to maintain the spatial and temporal computational graph of the previous timestep. In general, a typical neural network is divided into four stages, and the feature levels of each stage are various (He et al., 2016b). An intuitive idea is that the transfer of temporal information at the junction of two stages in SNNs may be important. Thus, we train two other SNNs using STBP. One (*Case 1*) discards most of the temporal gradients during training, retaining only the temporal gradients of the last layer in each stage. The other (*Case 2*) adopts the opposite behavior, discarding only the temporal gradient of the last spiking neuron layer of each stage.

Then, we compare the difference between the baseline, case 1, and case 2 by calculating the cosine similarity of the spatial gradient across the network. Results are shown in Fig. 2. The similarity in the left sub-figure (comparison between baseline and case 1) has always maintained a high level, especially as the training epochs increase. In contrast, The similarity in the right sub-figure (comparison between baseline and case 2) is always small. Thus, the temporal gradients of the final layer of each stage are more significant than those of spiking neurons in preceding stages.

4.2. Temporal-Reversible Spike Neuron

Based on the above analysis, we only turn on the temporal transfer of the spiking neurons in the last layer of each stage, as shown in Fig. 3. Thus spiking neurons in T-RevSNN have two states: turn-on or turn-off in the temporal dimension. The forward of temporal turn-off spiking neurons is

$$V^l[t] = \mathbf{W}^l S^{l-1}[t], \tag{6}$$

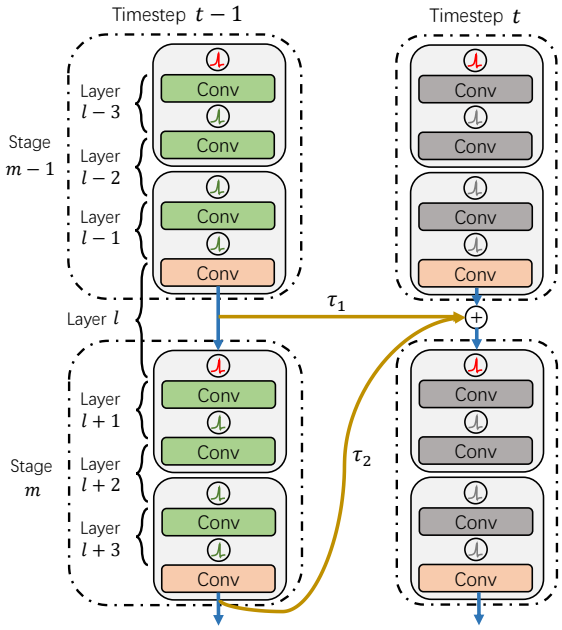

Figure 3. Temporal-reversible connection in T-RevSNN.

and the weight update depends solely on the spatial gradient:

$$\nabla_{\mathbf{W}^l}\mathcal{L} = \sum_{t=1}^{T}\left(\frac{\partial \mathcal{L}}{\partial V^{l+1}[t]}\frac{\partial V^{l+1}[t]}{\partial S^l[t]}\frac{\partial S^l[t]}{\partial V^l[t]}\right)^\top S^{l-1}[t]^\top \tag{7}$$

When updating turn-off spiking neuron weights $\mathbf{W}^l$, the gradient needs to be calculated on the sub-network at each timestep. We disable weight reuse of turn-off spiking neurons in the temporal dimension. Therefore, the gradients for weight $\mathbf{W}^l[t]$ at layer l, timestep t can be simplified by

$$\nabla_{\mathbf{W}^l[t]}\mathcal{L} = \left(\frac{\partial \mathcal{L}}{\partial V^{l+1}[t]}\frac{\partial V^{l+1}[t]}{\partial S^l[t]}\frac{\partial S^l[t]}{\partial V^l[t]}\right)^\top S^{l-1}[t]^\top, \tag{8}$$

which means that the update of all weights at timestep t only needs to perform $\mathcal{O}(L)$ calculation.

For temporal turn-on spiking neurons, the forward function is Eq. 1 and the weight update depends on both spatial and temporal gradients (i.e., Eq. 4). Inspired by the concept of

reversibility (Gomez et al., 2017), we observe that Eq. 1 is naturally reversible. Eq. 1 can be rewritten as:

$$V^l[t] = (1-\frac{1}{\tau_m})^{-1}(V^l[t+1] - \frac{1}{\tau_m}\mathbf{W}^l S^{l-1}[t+1]). \quad (9)$$

Subsequently, a reversible transformation can be established between $V^l[t+1]$ and $V^l[t]$. This means that when calculating the gradient at the first timestep, there is no need to store the membrane potentials and activations at all timesteps. We only need to store $V^l[T]$, and we can reversely deduce the membrane potential and activation values of all previous timesteps through Eq. 9. This reduces the memory required for multi-timestep training of SNNs. Regarding to the time complexity, the gradient process of the temporal turn-on spiking neurons is consistent with STBP, i.e., $\mathcal{O}(T^2)$.

Multi-level feature interaction has been proven to improve the model effectiveness in architectural design (Lin et al., 2017; Gao et al., 2019). The spatio-temporal neuronal dynamics of SNNs already naturally contain two levels of features, where spatial inputs come from low-level features (the previous layer at the same timestep) and temporal inputs come from same-level features (the previous timestep in the same layer). And, existing residual connections in SNNs (Fang et al., 2021a; Hu et al., 2024) also bring low-level features. But there's more we can do. In this work, we establish stronger multi-level connections between SNNs at adjacent timesteps. We incorporate the high-level features in deeper layers of the previous timestep into the information fusion of the current timestep. Generally, we can construct forward information passing as follows:

$$V^l[t+1] = \sum_{i=l}^{L}(1-\frac{1}{\tau_i})V^i[t] + \frac{1}{\tau_m}\mathbf{W}^l S^{l-1}[t+1], \quad (10)$$

where τ_i means the decay factor at different layers and $\tau_l = \tau_m$. Eq. 10 maintains temporal reversibility as follows:

$$V^l[t] = (1-\frac{1}{\tau_m})^{-1}(\sum_{i=l+1}^{L}(1-\frac{1}{\tau_i})(V^i[t+1] - \frac{1}{\tau_m}\mathbf{W}^l S^{l-1}[t+1])). \quad (11)$$

Note, as shown in Fig. 3, we only build high-level feature incorporation between stages. In multi-level interaction, we use up/down sampling to achieve feature alignment.

4.3. Basic SNN Block

As shown in Fig. 4, we design the basic SNN Block following (Liu et al., 2022) to extract local information effectively. It consists of two depth-Wise separable convolutions (DWConv/PWConv) and a residual connection, where the kernel size of DWConv is 5×5. We drop all Batch Normalization (BN) (Ioffe & Szegedy, 2015) modules to reduce the memory cost caused by storing the mean and variance of features (Xiao et al., 2022). To address the instability of training caused by loss of BN, the weights of all layers in the network are properly normalized (Brock et al., 2021). In addition, the ReZero (Bachlechner et al., 2021) technique is used on the Membrane Shortcut [1] (Hu et al., 2024) to enhance the network's capacity to meet dynamic isometrics after initialization and to facilitate efficient network training. To guarantee that only addition operations occur in inference, we can merge the scale of ReZero (i.e., α in Fig. 4) into the weight $\mathbf{W}^l$ at layer l through re-parameterization.

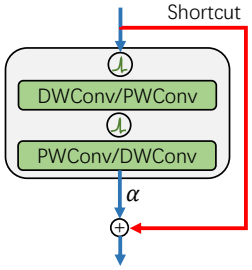

Figure 4. Basic SNN Block, following ConvNeXt-style (Liu et al., 2022).

4.4. Training and Inference Complexity Analysis

The memory and computation required by the STBP algorithm to calculate the gradient from *the output of the last layer at the last timestep* to *the input of the first layer at the first timestep* constitute the memory and time complexity of training SNNs. We analyze the training memory and time complexity of the proposed T-RevSNN and other SNN training optimization methods in Table 1 and Fig. 5.

Training Methods	Training		Inference
	Memory	Time	Energy
STBP (Neftci et al., 2019)	$\mathcal{O}(LT)$	$\mathcal{O}(LT)$	$\mathcal{O}(T)$
OTTT (Xiao et al., 2022)	$\mathcal{O}(L)$	$\mathcal{O}(LT)$	$\mathcal{O}(T)$
SLTT-K (Meng et al., 2023)	$\mathcal{O}(L)$	$\mathcal{O}(LK)$	$\mathcal{O}(T)$
S-RevSNN (Zhang & Zhang, 2024)	$\mathcal{O}(T)$	$\mathcal{O}(T)$	$\mathcal{O}(T)$
T-RevSNN turn-off (Ours)	$\mathcal{O}(L)$	$\mathcal{O}(L)$	$\mathcal{O}(1)$
T-RevSNN turn-on (Ours)		$\mathcal{O}(T)$	

Table 1. Training and inference complexity analysis.

STBP (Wu et al., 2018; Neftci et al., 2019) needs to retain the activations of all spiking neurons at all timesteps, the training memory is $\mathcal{O}(LT)$. The temporal gradient is propagated between the two layers of spiking neurons. When updating the weights of layer l, T gradients need to be cal-

[1] In addition to Membrane Shortcut (Hu et al., 2024), another type of residual commonly used in SNNs is Spike-Element-Wise (SEW) (Fang et al., 2021a), which cannot utilize this technique due to its nonlinear operations and inability to be re-parameterization.

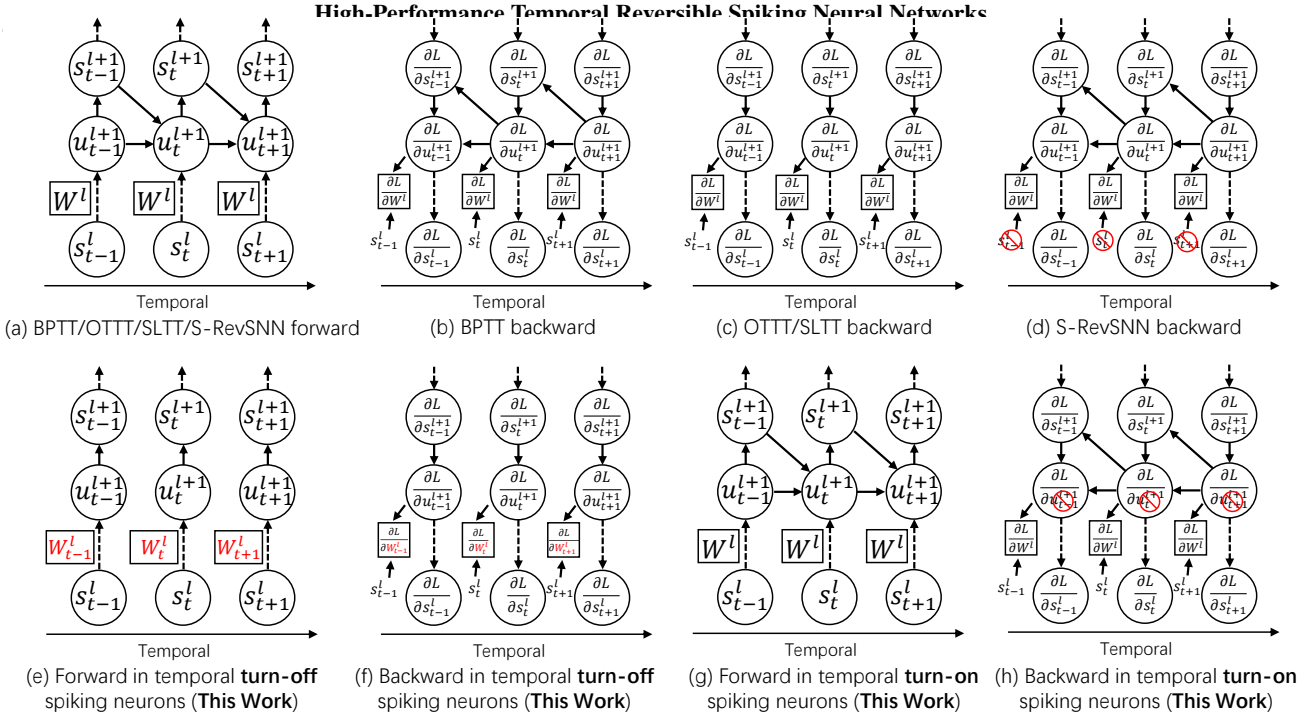

Figure 5. Illustration of the forward and backward. (a) Existing training methods do not change the forward of SNNs. (b) The backward of STBP unfolds simultaneously along the temporal and spatial dimensions. (c) The backward of OTTT/SLTT only unfolds along the spatial dimension. (d) The backward of S-RevSNN unfolds along the temporal and spatial dimensions but is reversible in spatial. (e) and (f) show the forward and backward of the temporal turn-off spiking neurons in the proposed method, respectively. (g) and (h) give the forward and backward of the temporal turn-on spiking neurons in T-RevSNN, which are basically consistent with the forward and backward in (a) and (b). The difference is that the backward in (h) is reversible, so only the membrane potentials of the last timestep needs to be stored.

culated and summed up. It implies that the time complexity is $\mathcal{O}(T)$ per layer, so the overall training time complexity of STBP is $\mathcal{O}(LT)$. Please see Fig. 5 (b).

OTTT (Xiao et al., 2022) does not calculate the temporal gradient and can be trained online. Each timestep only needs to retain activations at the current timestep, so the training memory and time are $\mathcal{O}(L)$ and $\mathcal{O}(LT)$, respectively.

SLTT-K (Meng et al., 2023) only calculates the gradients on the most important $K(K <= T)$ timesteps based on OTTT, thus the training memory and time complexity are $\mathcal{O}(L)$ and $\mathcal{O}(LK)$, respectively. Please see Fig. 5 (c).

S-RevSNN (Zhang & Zhang, 2024) retains the temporal gradient. Due to spatial reversibility, S-RevSNN only needs to store the activations of the spatially reversible layer at each timestep. Thus, the training memory and time complexity are both $\mathcal{O}(T)$. Please see Fig. 5 (d).

T-RevSNN (This work). We discard most of the temporal connections and are temporally reversible, so training memory of T-RevSNN is consistent with OTTT/SLTT, i.e., $\mathcal{O}(L)$. The training time of T-RevSNN is related to spiking neurons (turn-on/off), see Fig. 5 (e) to (h).

The optimization of OTTT/SLTT/S-RevSNN does not change the forward propagation, so their inference is all $\mathcal{O}(T)$. The proposed T-RevSNN redign the input encoding and split the network, thus the inference is $\mathcal{O}(1)$.

5. Experiments

We evaluate the proposed T-RevSNN on static ImageNet-1k (Deng et al., 2009) and neuromorphic CIFAR10-DVS (Li et al., 2017), DVS128 Gesture (Amir et al., 2017). We generally continued the experimental setup in (Yao et al., 2023b). The input size of ImageNet is 224×224. The batch size is set to 512 during 300 training epochs with a cosine-decay learning rate whose initial value is 0.002. The optimizer is Lamb. Standard data augmentation techniques, like random augmentation, are also employed in training. Details of the datasets, training and experimental setup, and power analysis are given in the supplementary material.

5.1. Main Results

5.1.1. ImageNet-1k

We comprehensively compare the proposed T-RevSNN with existing work in the five aspects of parameters, training time, memory, inference energy, Top-1 accuracy. In Table 2, we divide the existing direct training SNNs into three types, namely spiking ResNet, spiking Transformer, and SNNs with training optimization. We test various SNNs on 6 NVIDIA-A100-40GB devices and report their memory and training time, to demonstrate the superiority of T-RevSNN.

Methods	Architecture	Param (M)	Time step	Training Time (min/ep)	Memory (MB/img)	Energy (mJ)	Acc (%)
STBP-tdBN (Zheng et al., 2021)	ResNet-34	21.8	6	29.6	186.1	6.4	63.7
SEW ResNet (Fang et al., 2021a)	SEW-ResNet-50	25.6	4	10.0	596.9	4.9	67.8
MS ResNet (Hu et al., 2024)	MS-ResNet-34	21.8	6	11.2	267.1	5.1	69.4
TEBN (Duan et al., 2022)	ResNet-34	21.8	4	16.3	260.1	6.4*	64.3
TET (Deng et al., 2022)	SEW-ResNet-34	21.8	4	12.5	221.0	4.0*	68.0
Spikformer	Spikeformer-8-384	16.8	4	14.2	580.8	7.7	70.2
(Zhou et al., 2023)	Spikeformer-8-512	29.7	4	16.7	767.8	11.6	73.4
Spike-driven Transformer	Spikeformer-8-384	16.8	4	15.4	548.9	3.9	72.3
(Yao et al., 2023b)	Spikeformer-8-512	**29.7**	4	18.8	730.0	4.5	**74.6**
OTTT (Xiao et al., 2022)	ResNet-34	21.8	6	24.2	84.1	6.0*	64.2
SLTT (Meng et al., 2023)	ResNet-34	21.8	6	18.1	71.7	6.0*	66.2
	ResNet-50	25.6	6	23.4	117.3	7.2*	67.0
S-RevSNN (Zhang & Zhang, 2024)	Rev-ResNet-37	24.4	4	7.9	127.1	-	-
	Rev-SFormer-8-384	17.2	4	16.3	389.6	-	-
T-RevSNN (This work)	ResNet-18 (384)	15.2	4	**6.1**	**57.5**	**1.7**	69.8
	ResNet-18 (512)	**29.8**	4	9.1	85.7	2.8	**73.2**

Table 2. Evaluation on ImageNet-1K. We divide the existing direct training SNNs into three types, from top to bottom: spiking ResNet, spiking Transformer, and SNNs with training optimization (OTTT/SLTT/S-RevSNN). We measure the memory consumption of all models and the time it takes to complete an epoch. The memory per image (MB/img) is measured as the peak GPU memory each image occupies during training, following S-RevSNN (Zhang & Zhang, 2024). All experiments were tested under float16 automatic mixed precision, distributed data-parallel, and no gradient checkpoint. * We estimate the power of these models based on the corresponding spiking ResNet, e.g., we assume that the power of one timestep of SEW-ResNet-50 is $4.0/4 = 1.0$mJ. In our model, 384/512 represents the channel number in the last stage, which is used to control the parameters.

Compared with Spiking ResNet Baselines. There are two main types of spiking ResNet baselines in the SNN domain, SEW-ResNet (Fang et al., 2021a) and MS-ResNet (Hu et al., 2024). The proposed T-RevSNN achieves the best accuracy among the spiking ResNet series, with the lowest training memory and time, and inference energy. For example, **T-RevSNN (This work)** vs. SEW-ResNet-50 vs. MS-ResNet-34: Param, **29.8M** vs. 25.6M vs. 21.8M; Acc, **73.2%** vs. 67.8% vs. 69.4%; Memory, **85.7 (MB/img)** vs. 596.9 (MB/img) vs. 267.1 (MB/img); Traning-Time, **9.1 min/ep** vs. 10.0 min/ep vs. 11.2 min/ep; Inference, **2.8mJ** vs. 4.9mJ vs. 5.1 mJ. Thus, the proposed T-RevSNN shows a full range of advantages over existing spiking ResNets.

Compared with Spiking Transformer Baselines. Spiking Transformer is a type of architecture that has become popular in SNNs recently. Typical works include SpikFormer(Zhou et al., 2023) and Spike-driven Transformer (Yao et al., 2023b). The results in Table 2 show that under similar parameters (30M), the accuracy of T-RevSNN (73.2%) is the same as SpikFormer (73.4%), but 1.4% lower than Spike-driven Transformer (74.6%). *We think this may be a performance gap caused by the architecture (Transformer vs. CNN).* In this work, our SNN blocks are entirely composed of convolutional layers. In contrast, Spike-driven Transformer contains the spiking self-attention operator. We tried replacing the spiking CNN block with spiking Transformer in Zhou et al. (2023) and Yao et al. (2023b), but the network would not converge. And, T-RevSNN has overwhelming advantages in training speed and memory.

Compared with Training Optimization SNNs. As can be seen in Table 2, our model is better than existing training optimization SNNs in accuracy, training speed and memory, power. S-RevSNN (Zhang & Zhang, 2024) does not report accuracy on ImageNet, so there is no relevant data.

Overall, T-RevSNN shows significant advantages over existing SNNs regarding training speed, memory requirements, and inference power. T-RevSNN also performs competitively in performance. Although the accuracy is lower than Spike-driven Transformer, we argue that this is a gap caused by the architecture and can be solved in the future.

5.1.2. CIFAR10-DVS and Gesture

Unlike static datasets that encode repeated inputs, neuromorphic datasets input different information at each timestep (Deng et al., 2020). Therefore, in the neuromorphic task, the inference complexity of all SNN models is $\mathcal{O}(1)$. We present the details of these two datasets as well as the preprocessing and experimental setup in the supplementary material. We still use the network architecture in Fig. 1(b). Different from encoding only once in static image tasks, here we encode the input at each timestep, which is consistent with prior SNNs. Experiments on CIFAR10-DVS and Gesture are shown in Table 3, we achieve performance comparable to SOTA results with lower memory.

Methods	CIFAR10-DVS		DVS128 Gesture		
	Timestep	Top-1 Acc	Timestep	Top-1 Acc	Memory
LIAF-Net (Wu et al., 2021b)	10	70.4	60	97.6	-
Rollout (Kugele et al., 2020)	48	66.8	240	97.1	-
tdBN (Zheng et al., 2021)	10	67.8	40	96.9	-
PLIF (Fang et al., 2021b)	20	74.8	20	97.6	-
SEW ResNet (Fang et al., 2021a)	16	74.4	16	97.9	81.5
MS ResNet (Hu et al., 2024)	10	76.0	10	94.8	79.4
Dspike (Li et al., 2021)	10	75.4	-	-	-
(She et al., 2021)	-	-	20	98.0	-
DSR (Meng et al., 2022)	10	77.3	-	-	-
Spikformer (Zhou et al., 2023)	16	**80.6**	16	97.9	51.8
Spike-Driven Transformer (Yao et al., 2023b)	16	80.0	16	**99.3**	51.4
OTTT (Xiao et al., 2022)	10	76.6	20	96.9	-
SLTT (Meng et al., 2023)	10	77.2	20	97.9	-
S-RevSNN (Zhang & Zhang, 2024)	10	75.5	10	94.4	25.0
T-RevSNN (This work)	10	77.4	10	94.4	**10.3**
	16	**79.2**	16	**97.9**	18.9

Table 3. Evaluation on CIFAR10-DVS (Li et al., 2017) and Gesture (Amir et al., 2017). Memory is evaluated on the Gesture.

5.2. Ablation Studies

We conduct various ablation studies on ImageNet to analyze the proposed T-RevSNN. Generally, we set the training epoch to 100 and keep the parameter to 15.2M.

Timestep. In our design, we divide the parameters of the entire network into T groups (sub-networks). In Table 4, we analyze the impact of different T on accuracy, training speed, and memory. Since we fix the total number of parameters, increasing the T means that the sub-network at each timestep becomes smaller. The training memory decreases accordingly (training memory complexity of T-RevSNN is $\mathcal{O}(L)$), but the training time will increase. And, the relationship between accuracy and timestep is not linear.

Timestep	Acc	Memory (MB/img)	Training Time (min/ep)
2	68.1	75.6	**4.9**
4	68.6	57.5	6.1
8	**69.8**	49.8	9.9
16	63.4	**42.4**	16.0

Table 4. Ablation study with timestep T.

Multi-level Temporal Reversible. In Eq. 10, we design multi-level feature fusion for T-RevSNN. We fuse the features of the next stage into the current stage. Otherwise, T-RevSNN's accuracy will lose 1.2% (68.6% $\rightarrow$ 67.4%).

Scaled Residual Connection contributes to the convergence speed and final accuracy of the model, as shown in Table 5.

Scaled Residual	Acc	Epochs to 60% Acc
✓	68.6	25
✗	68.3	32

Table 5. Ablation study with residual connection.

SNN Block Structure is critical to final performance. In our design, we apply the idea of temporal reversibility to the ConvNeXt-style (Liu et al., 2022) architecture. In Table 6, we exploit the idea to MS-ResNet (Hu et al., 2024) and S-RevSNN (Zhang & Zhang, 2024). When applying temporal reversibility directly on MS-ResNet-34, performance will drop a little. Moreover, we can see that temporal and spatial reversibility are orthogonal. The combination of the two can further improve training efficiency. Since S-RevSNN does not provide baseline accuracy on ImageNet, we only observe training efficiency here.

Block	Acc	Memory (MB/img)	Training Time (min/ep)
MS-ResNet-34 (Baseline)	68.3	267.1	11.2
+ T-RevSNN	66.7	88.1	7.4
S-RevSNN-37 (Baseline)	-	127.1	7.9
+ T-RevSNN	-	55.4	6.8

Table 6. Ablation study with SNN Block Structure.

6. Conclusion

We propose T-RevSNN to address the training memory and inference energy challenges posed by multiple timestep simulations for SNNs. Our idea is simple and intuitive. Since the temporal gradients in SNNs are not important, can we only keep the temporal forward direction of key positions and turn off the temporal dynamics of other spiking neurons? To this end, we make systematic designs, including multi-level temporal-reversible forward information transfer of key spiking neurons, group design of input encoding and network architecture, and improvements in SNN blocks and residual connections. Extensive experiments are conducted on static and neuromorphic datasets, verifying the advantages of T-RevSNN in training, inference and accuracy. We hope our investigations pave the way for further research on large-scale SNNs and more applications of SNNs.

Acknowledgement. This work was partially supported by National Science Foundation for Distinguished Young Scholars (62325603), National Natural Science Foundation of China (62236009, U22A20103, 62441606, 62394311, 62425101, 62332002, 62088102), and CAAI-MindSpore Open Fund, developed on OpenI Community.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., and Modha, D. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7243–7252, 2017.

Bachlechner, T., Majumder, B. P., Mao, H., Cottrell, G., and McAuley, J. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, pp. 1352–1361. PMLR, 2021.

Brock, A., De, S., Smith, S. L., and Simonyan, K. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pp. 1059–1071. PMLR, 2021.

Brügger, R., Baumgartner, C. F., and Konukoglu, E. A partially reversible u-net for memory-efficient volumetric image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part III 22*, pp. 429–437, 2019.

Cai, Y., Zhou, Y., Han, Q., Sun, J., Kong, X., Li, J., and Zhang, X. Reversible column networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Oc2vlWU0jFY.

Chowdhury, S. S., Rathi, N., and Roy, K. One timestep is all you need: Training spiking neural networks with ultra low latency. *arXiv preprint arXiv:2110.05929*, 2021.

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

Deng, H., Zhu, R., Qiu, X., Duan, Y., Zhang, M., and Deng, L.-J. Tensor decomposition based attention module for spiking neural networks. *Knowledge-Based Systems*, pp. 111780, 2024.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

Deng, L., Wu, Y., Hu, X., Liang, L., Ding, Y., Li, G., Zhao, G., Li, P., and Xie, Y. Rethinking the performance comparison between snns and anns. *Neural Networks*, 121: 294–307, 2020.

Deng, S., Li, Y., Zhang, S., and Gu, S. Temporal efficient training of spiking neural network via gradient re-weighting. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_XNtisL32jv.

Ding, Y., Zuo, L., Jing, M., He, P., and Xiao, Y. Shrinking your timestep: Towards low-latency neuromorphic object recognition with spiking neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Duan, C., Ding, J., Chen, S., Yu, Z., and Huang, T. Temporal effective batch normalization in spiking neural networks. *Advances in Neural Information Processing Systems*, 35: 34377–34390, 2022.

Eshraghian, J. K., Ward, M., Neftci, E. O., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D. S., and Lu, W. D. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9): 1016–1054, 2023.

Fang, W., Yu, Z., Chen, Y., Huang, T., Masquelier, T., and Tian, Y. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34:21056–21069, 2021a.

Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., and Tian, Y. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2661–2671, October 2021b.

Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., Huang, L., Zhou, H., Li, G., and Tian, Y. Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40): eadi1480, 2023.

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., Daniilidis, K., and Scaramuzza, D. Event-based vision:

A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022.

Gao, S.-H., Cheng, M.-M., Zhao, K., Zhang, X.-Y., Yang, M.-H., and Torr, P. Res2net: A new multi-scale backbone architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):652–662, 2019.

Gomez, A. N., Ren, M., Urtasun, R., and Grosse, R. B. The reversible residual network: Backpropagation without storing activations. *Advances in Neural Information Processing Systems*, 30, 2017.

Guo, Y., Chen, Y., Zhang, L., Liu, X., Wang, Y., Huang, X., and Ma, Z. IM-loss: Information maximization loss for spiking neural networks. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022a. URL https://openreview.net/forum?id=Jw34v_84m2b.

Guo, Y., Chen, Y., Zhang, L., Wang, Y., Liu, X., Tong, X., Ou, Y., Huang, X., and Ma, Z. Reducing information loss for spiking neural networks. In *European Conference on Computer Vision*, pp. 36–52. Springer, 2022b.

Guo, Y., Tong, X., Chen, Y., Zhang, L., Liu, X., Ma, Z., and Huang, X. Recdis-snn: Rectifying membrane potential distribution for directly training spiking neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 326–335, 2022c.

Guo, Y., Huang, X., and Ma, Z. Direct learning-based deep spiking neural networks: a review. *Frontiers in Neuroscience*, 17:1209795, 2023.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M. (eds.), *Computer Vision – ECCV 2016*, pp. 630–645, Cham, 2016a. Springer International Publishing.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016b.

Horowitz, M. 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)*, pp. 10–14. IEEE, 2014.

Hu, Y., Tang, H., and Pan, G. Spiking deep residual networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(8):5200–5205, 2023a. doi: 10.1109/TNNLS.2021.3119238.

Hu, Y., Zheng, Q., Jiang, X., and Pan, G. Fast-snn: Fast spiking neural network by converting quantized ann. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):14546–14562, 2023b. doi: 10.1109/TPAMI.2023.3275769.

Hu, Y., Deng, L., Wu, Y., Yao, M., and Li, G. Advancing spiking neural networks toward deep residual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456. PMLR, 2015.

Kim, Y. and Panda, P. Revisiting batch normalization for training low-latency deep spiking neural networks from scratch. *Frontiers in Neuroscience*, 15:773954, 2021.

Kim, Y., Park, H., Moitra, A., Bhattacharjee, A., Venkatesha, Y., and Panda, P. Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks? In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 71–75, 2022.

Kugele, A., Pfeil, T., Pfeiffer, M., and Chicca, E. Efficient processing of spatio-temporal data streams with spiking neural networks. *Frontiers in Neuroscience*, 14: 439, 2020.

Li, H., Liu, H., Ji, X., Li, G., and Shi, L. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:309, 2017.

Li, Y., Guo, Y., Zhang, S., Deng, S., Hai, Y., and Gu, S. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34:23426–23439, 2021.

Li, Y., Geller, T., Kim, Y., and Panda, P. Seenn: Towards temporal spiking early-exit neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Lian, S., Shen, J., Liu, Q., Wang, Z., Yan, R., and Tang, H. Learnable surrogate gradient for direct training spiking neural networks. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 3002–3010, 8 2023.

Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Lin, Y., Hu, Y., Ma, S., Yu, D., and Li, G. Rethinking pretraining as a bridge from anns to snns. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022. doi: 10.1109/TNNLS.2022.3217796.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022.

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

MacKay, M., Vicol, P., Ba, J., and Grosse, R. B. Reversible recurrent neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

Mangalam, K., Fan, H., Li, Y., Wu, C.-Y., Xiong, B., Feichtenhofer, C., and Malik, J. Reversible vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10830–10840, 2022.

Meng, Q., Xiao, M., Yan, S., Wang, Y., Lin, Z., and Luo, Z.-Q. Training high-performance low-latency spiking neural networks by differentiation on spike representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12444–12453, 2022.

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

Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., Jackson, B. L., Imam, N., Guo, C., Nakamura, Y., et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

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

Panda, P., Aketi, S. A., and Roy, K. Toward scalable, efficient, and accurate deep spiking neural networks with backward residual connections, stochastic softmax, and hybridization. *Frontiers in Neuroscience*, 14:653, 2020.

Pei, J., Deng, L., et al. Towards artificial general intelligence with hybrid tianjic chip architecture. *Nature*, 572(7767): 106–111, 2019.

Perez-Nieves, N. and Goodman, D. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems*, 34:11795–11808, 2021.

Qiu, X., Zhu, R.-J., Chou, Y., Wang, Z., Deng, L.-j., and Li, G. Gated attention coding for training high-performance and efficient spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 601–610, 2024.

Qiu, X.-R., Wang, Z.-R., Luan, Z., Zhu, R.-J., Wu, X., Zhang, M.-L., and Deng, L.-J. Vtsnn: A virtual temporal spiking neural network. *Frontiers in Neuroscience*, 17: 1091097, 2023.

Rathi, N. and Roy, K. 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(6):3174–3182, 2023. doi: 10.1109/TNNLS.2021.3111897.

Rathi, N., Srinivasan, G., Panda, P., and Roy, K. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1xSperKvH.

Roy, K., Jaiswal, A., and Panda, P. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

Sander, M. E., Ablin, P., Blondel, M., and Peyré, G. Momentum residual neural networks. In *International Conference on Machine Learning*, pp. 9276–9287. PMLR, 2021.

Schuman, C. D., Kulkarni, S. R., Parsa, M., Mitchell, J. P., Kay, B., et al. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, 2(1):10–19, 2022.

Sengupta, A., Ye, Y., Wang, R., Liu, C., and Roy, K. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in Neuroscience*, 13:95, 2019.

She, X., Dash, S., and Mukhopadhyay, S. Sequence approximation using feedforward spiking neural network for spatiotemporal learning: Theory and optimization methods. In *International Conference on Learning Representations*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Wang, D., Wu, B., Zhao, G., Yao, M., Chen, H., Deng, L., Yan, T., and Li, G. Kronecker cp decomposition with fast multiplication for compressing rnns. *IEEE Transactions on Neural Networks and Learning Systems*, 34(5):2205–2219, 2023a.

Wang, W., Long, J., Wen, C., and Huang, J. Recent advances in distributed adaptive consensus control of uncertain nonlinear multi-agent systems. *Journal of Control and Decision*, 7(1):44–63, 2020.

Wang, Z., Fang, Y., Cao, J., Zhang, Q., Wang, Z., and Xu, R. Masked spiking transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1761–1771, 2023b.

Wightman, R. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Wu, J., Xu, C., Han, X., Zhou, D., Zhang, M., Li, H., and Tan, K. C. Progressive tandem learning for pattern recognition with deep spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (11):7824–7840, 2021a.

Wu, X., He, W., Yao, M., Zhang, Z., Wang, Y., and Li, G. Mss-depthnet: Depth prediction with multi-step spiking neural network. *arXiv preprint arXiv:2211.12156*, 2022a.

Wu, Y., Deng, L., Li, G., Zhu, J., and Shi, L. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12: 331, 2018.

Wu, Y., Deng, L., Li, G., Zhu, J., Xie, Y., and Shi, L. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1311–1318, 2019.

Wu, Y., Wang, D.-H., Lu, X.-T., Yang, F., Yao, M., Dong, W.-S., Shi, J.-B., and Li, G.-Q. Efficient visual recognition: A survey on recent advances and brain-inspired methodologies. *Machine Intelligence Research*, 19(5): 366–411, 2022b.

Wu, Z., Zhang, H., Lin, Y., Li, G., Wang, M., and Tang, Y. Liaf-net: Leaky integrate and analog fire network for lightweight and efficient spatiotemporal information processing. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021b. doi: 10.1109/TNNLS.2021.3073016.

Xiao, M., Meng, Q., Zhang, Z., He, D., and Lin, Z. Online training through time for spiking neural networks. *Advances in Neural Information Processing Systems*, 35: 20717–20730, 2022.

Xu, Y., Wei, H., Lin, M., Deng, Y., Sheng, K., Zhang, M., Tang, F., Dong, W., Huang, F., and Xu, C. Transformers in computational visual media: A survey. *Computational Visual Media*, 8:33–62, 2022.

Yang, H., Lam, K.-Y., Xiao, L., Xiong, Z., Hu, H., Niyato, D., and Vincent Poor, H. Lead federated neuromorphic learning for wireless edge artificial intelligence. *Nature Communications*, 13(1):4269, 2022.

Yao, M., Gao, H., Zhao, G., Wang, D., Lin, Y., Yang, Z., and Li, G. Temporal-wise attention spiking neural networks for event streams classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10221–10230, 2021.

Yao, M., Hu, J., Zhao, G., Wang, Y., Zhang, Z., Xu, B., and Li, G. Inherent redundancy in spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16924–16934, 2023a.

Yao, M., Hu, J., Zhou, Z., Yuan, L., Tian, Y., XU, B., and Li, G. Spike-driven transformer. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=9FmolyOHi5.

Yao, M., Zhang, H., Zhao, G., Zhang, X., Wang, D., Cao, G., and Li, G. Sparser spiking activity can be better: Feature refine-and-mask spiking neural network for event-based visual recognition. *Neural Networks*, 166:410–423, 2023c.

Yao, M., Zhao, G., Zhang, H., Hu, Y., Deng, L., Tian, Y., Xu, B., and Li, G. Attention spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9393–9410, 2023d.

Yao, M., Hu, J., Hu, T., Xu, Y., Zhou, Z., Tian, Y., XU, B., and Li, G. 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*, 2024a. URL https://openreview.net/forum?id=1SIBN5Xyw7.

Yao, M., Richter, O., Zhao, G., Qiao, N., Xing, Y., Wang, D., Hu, T., Fang, W., Demirci, T., De Marchi, M., Deng, L., Yan, T., Nielsen, C., Sheik, S., Wu, C., Tian, Y., Xu, B., and Li, G. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15(1):4464, May 2024b. ISSN 2041-1723. URL https://doi.org/10.1038/s41467-024-47811-6.

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

Zhang, H. and Zhang, Y. Memory-efficient reversible spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Zheng, H., Wu, Y., Deng, L., Hu, Y., and Li, G. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11062–11070, 2021.

Zhou, Z., Zhu, Y., He, C., Wang, Y., YAN, S., Tian, Y., and Yuan, L. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=frE4fUwz_h.

Limitation of this work are Transformer-based T-RevSNN, larger scale models, more vision or long sequence tasks, and we will work on them in future work. The experimental results in this paper are reproducible. We explain the details of model training and configuration in the main text and supplement it in the appendix. Our codes and models will be available on GitHub after review.

A. Details of Cosine Similarity Calculation

In Fig. 2, the Left and Right sub-figures are the cosine similarity between the spatial gradients calculated by baseline and case 1/2, respectively. In case 1/2, we only retain/discard the temporal gradients of the last layer of spiking neurons in each stage. We use spiking ResNet-18 to train on CIFAR-10 for 30 epochs with different hyperparameters (e.g., time step T, decay τ). Moreover, we take the mean gradient value of each layer in every epoch during the training process as a vector and use the cosine similarity formula (Eq. 12) to compare the differences between various training methods.

$$\mathcal{F}(A,B) = \frac{A \cdot B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(B_i)^2}}, \tag{12}$$

where $\mathcal{F}(\cdot)$ is the cosine similarity between vector A and B. $||A||$ and $||B||$ are the Euclidean norms of A and B.

B. Energy Consumption for T-RevSNN.

The T-RevSNN architecture can transform matrix multiplication into sparse addition, which can be implemented as an addressable addition on neuromorphic chips. Moreover, when evaluating algorithms, the SNN field often ignores specific hardware implementation details and estimates theoretical energy consumption for a model (Panda et al., 2020; Yin et al., 2021; Yang et al., 2022; Wang et al., 2023a; Yao et al., 2024a). This theoretical estimation is just to facilitate the qualitative energy analysis of various SNN and ANN algorithms. In the encoding layer, convolution operations serve as MAC operations that convert analog inputs into spikes, similar to direct coding-based SNNs (Wu et al., 2019). Conversely, in SNN's architecture, the Convolution (Conv) and Fully-Connected (FC) layers transmit spikes and perform AC operations to accumulate weights for postsynaptic neurons. The inference energy cost of T-RevSNN can be expressed as follows:

$$E_{total} = E_{MAC} \cdot FL^1_{conv} + E_{AC} \cdot T \cdot \Big(\sum_{n=2}^{N} FL^n_{conv} \cdot fr^n + \sum_{m=1}^{M} FL^m_{fc} \cdot fr^m\Big), \tag{13}$$

where N and M are the total number of Conv and FC layers, E_{MAC} and E_{AC} are the energy costs of MAC and AC operations, and fr^m, fr^n, FL^n_{conv} and FL^m_{fc} are the firing rate and FLOPs of the n-th Conv and m-th FC layer. The spike firing rate is defined as the proportion of non-zero elements in the spike tensor. Previous SNN works (Horowitz, 2014; Rathi et al., 2020) assume 32-bit floating-point implementation in 45nm technology, where E_{MAC} = 4.6pJ and E_{AC} = 0.9pJ for various operations.

C. Experiment Details.

Datasets. We employ two types of datasets: static image classification and neuromorphic classification. The former includes ImageNet-1K (Deng et al., 2009). The latter contains CIFAR10-DVS (Li et al., 2017) and DVS128 Gesture (Amir et al., 2017).

ImageNet-1K is the most typical static image dataset, which is widely used in the field of image classification. It offers a large-scale natural image dataset of 1.28 million training images and 50k test images, with a total of 1,000 categories. CIFAR10-DVS is an event-based neuromorphic dataset converted from CIFAR10 by scanning each image with repeated closed-loop motion in front of a Dynamic Vision Sensor (DVS). There are a total of 10,000 samples in CIFAR10-DVS, with each sample lasting 300ms. The temporal and spatial resolutions are μs and 128×128, respectively. DVS128 Gesture is an event-based gesture recognition dataset, which has the temporal resolution in μs level and 128×128 spatial resolution. It records 1342 samples of 11 gestures, and each gesture has an average duration of 6 seconds.

Data Preprocessing. In the static datasets, as shown in Fig. 1 (b), what differs from previous work is that we do not repeatedly feed the data into the SNN, but encode it and divide it into equal parts according to the number of timesteps. Then the features are each sent to the sub-networks. At the same time, due to the reversible connection of information between different timesteps, there is no information loss (Gomez et al., 2017; Cai et al., 2023) between these timesteps, so we do not need to average the output of all time steps before calculating the loss function. We only use the information from the last timestep to calculate the cross-entropy loss.

In contrast, neuromorphic (i.e. event-based) datasets can fully exploit the energy-efficient advantages of SNNs with spatio-temporal dynamics. Specifically, neuromorphic datasets are generated by event-based (neuromorphic) cameras, such as DVS (Wang et al., 2020; Gallego et al., 2022; Wu et al., 2022b;a; Xu et al., 2022). Compared to conventional cameras, DVS represents a new paradigm shift in visual information acquisition, encoding the time, location,

and polarity of brightness changes for each pixel into event streams with μs-level temporal resolution. Events (spike signals with address information) are generated only when the brightness of the visual scene changes. This is consistent with the event-driven nature of SNNs. Only when there is an event input will some spiking neurons of SNNs be triggered to participate in the computation. Typically, event streams are pre-processed into image sequences as input to SNNs. Details can be found in previous work (Yao et al., 2021).

Experimental Steup. The experimental setup in this work generally follows (Yao et al., 2023b). The experimental settings of ImageNet-1K have been given in the main text. Here we mainly give the network settings on neuromorphic datasets. The training epoch for these datasets is 300. The batch size is 16 for Gesture and CIFAR10-DVS. The learning rate is initialized to 0.0003 for Gesture, and 0.01 for CIFAR10-DVS. All of them are reduced with cosine decay. We follow (Yao et al., 2023b) to apply data augmentation on Gesture and CIFAR10-DVS. In addition, the network structures used in CIFAR10-DVS and Gesture are T-RevSNN ResNet-18 (256).

Detailed configurations and hyper-parameters. We use the open-source timm (Wightman, 2019) repository for hyper-parameter configuration. We give here a detailed hyperparameter configuration to improve the reproducibility of T-RevSNN.

Hyper-parameter	ImageNet
Model size	15M/30M
Embed dim	[64, 128, 256, 384] [96, 192, 384, 512]
Timestep	4
Epochs	300
Resolution	224*224
Batch size	4096
Optimizer	LAMB
Base Learning rate	2e-4
Learning rate decay	Cosine
Warmup eopchs	20
Weight decay	0.01
Rand Augment	m9-mstd0.5-inc1
Mixup	None
Cutmix	None
Label smoothing	0.1

Table 7. Hyper-parameters for image classification on ImageNet-1K.