LM-HT SNN: Enhancing the Performance of SNN to ANN Counterpart through Learnable Multi-hierarchical Threshold Model

Zecheng Hao[1], **Xinyu Shi**[1,2], **Yujia Liu**[1], **Zhaofei Yu**[1,2]* **& Tiejun Huang**[1,2]
[1] School of Computer Science, Peking University
[2] Institute for Artificial Intelligence, Peking University

Abstract

Compared to traditional Artificial Neural Network (ANN), Spiking Neural Network (SNN) has garnered widespread academic interest for its intrinsic ability to transmit information in a more energy-efficient manner. However, despite previous efforts to optimize the learning algorithm of SNNs through various methods, SNNs still lag behind ANNs in terms of performance. The recently proposed multi-threshold model provides more possibilities for further enhancing the learning capability of SNNs. In this paper, we rigorously analyze the relationship among the multi-threshold model, vanilla spiking model and quantized ANNs from a mathematical perspective, then propose a novel LM-HT model, which is an equidistant multi-threshold model that can dynamically regulate the global input current and membrane potential leakage on the time dimension. The LM-HT model can also be transformed into a vanilla single threshold model through reparameterization, thereby achieving more flexible hardware deployment. In addition, we note that the LM-HT model can seamlessly integrate with ANN-SNN Conversion framework under special initialization. This novel hybrid learning framework can effectively improve the relatively poor performance of converted SNNs under low time latency. Extensive experimental results have demonstrated that our model can outperform previous state-of-the-art works on various types of datasets, which promote SNNs to achieve a brand-new level of performance comparable to quantized ANNs. Code is available at `https://github.com/hzc1208/LMHT_SNN`.

1 Introduction

Recognized as the third generation of artificial neural networks [33], Spiking Neural Network (SNN) is increasingly receiving significant academic attention due to its enormous potential in biological plausibility and high energy efficiency. As the information transmission between the pre-synaptic and post-synaptic layers relies on the discrete spike signal, which will be only emitted when the membrane potential of the corresponding neuron exceeds the firing threshold, SNNs have a unique event-driven property compared to conventional Artificial Neural Network (ANN). By utilizing this property, researchers have pointed out that SNNs can achieve significant advantages in terms of energy consumption on neuromorphic hardware [35, 5, 37]. Currently, SNNs have further fulfilled a role in multiple application scenarios including object detection [22], natural language processing [32], and 3D recognition [24].

Spatial-Temporal back-propagation (STBP) with surrogate gradients is currently the most mainstream supervised learning algorithm suitable for SNNs. Although previous works have attempted to further enhance the learning ability of SNNs by delving into various optimization strategies, including

*Corresponding author: yuzf12@pku.edu.cn

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3233

gradient adjustment [29, 8, 14] and structural improvement [54, 51, 44, 50], there is still a certain performance gap between ANNs and SNNs.

Recently, the STBP learning algorithm based on multi-threshold models [41, 42] is considered as another potential way to improve the performance of SNNs. In this scenario, multiple levels of the firing threshold enable SNNs to transmit richer information at each time-step. Unfortunately, we think that current related works have not accurately recognized the mathematical essence of multi-threshold models as well as their relationship with ANNs and SNNs. In this paper, we innovatively propose a learnable multi-hierarchical and equidistant threshold model based on global input information, which is called LM-HT model. On the one hand, we note that our LM-HT model can equivalently represent the information of the vanilla model over multiple consecutive time-steps within a single step. Furthermore, we can convert the LM-HT model into a vanilla single threshold model through a layer-by-layer reparameterization scheme. On the other hand, the STBP method based on the LM-HT model can be transformed into the training modes of the vanilla STBP and quantized ANNs under different parameter initialization conditions, respectively. The main contribution of this work has been summarized as follows:

- We point out that the essence of the equidistant multi-threshold model is to simulate the spike firing situation of the vanilla spiking model within specific time windows. Specially, when the input current follows a completely uniform distribution on the time dimension, its spike firing rate is mathematically equivalent to the activation output of quantized ANNs.
- We propose an advanced LM-HT model, which can enhance the performance of SNNs to the level of ANNs and be transformed into a vanilla single threshold model losslessly during the inference stage. By adopting different parameter initialization schemes, the LM-HT model can further establish a bridge between the vanilla STBP and quantized ANNs training.
- We further design a brand-new hybrid training framework based on the LM-HT model, which is enable to effectively improve the performance degradation problem of traditional ANN-SNN Conversion methods regardless of the time latency degree involved.
- Experimental results have indicated that our model can fulfill state-of-the-art learning performance for various types of datasets. For instance, we achieve the top-1 accuracy of 81.76% for CIFAR-100, ResNet-19 within merely 2 time-steps.

2 Related Works

STBP supervised training. STBP is the most prevailing recurrent-mode learning algorithm in the field of SNN direct training. Wu *et al.* [47] tackled the non-differentiable problem existed in the spike firing process by utilizing surrogate gradients and achieved gradient smoothing calculation between layers. Deng *et al.* [8] and Guo *et al.* [14] respectively proposed brand-new target loss functions by analyzing the temporal distribution of the spike sequence and membrane potential. Furthermore, various temporal-dependent batch normalization layers [54, 10, 15] and advanced spiking neuron models [51, 44] have been pointed out, which enhances the capability and stability of SNN learning. The researchers also designed a variety of residual blocks [11, 20] and Transformer structures [55, 49] suitable for SNNs, promoting the development of STBP training towards the domains of deep and large-scale models. In addition, some variant and extended learning methods based on STBP have also received widespread attention. Temporal Coding [36] and Time-to-First-Spike (TTFS) [21] algorithm conduct one-time back-propagation based on the specific firing moment. Meng *et al.* [34] introduced the idea of online learning into vanilla STBP algorithm, which significantly saves training memory overhead by eliminating the gradient chains between different time-steps. Fang *et al.* [12] proposed a spiking neuron model that supports parallel computing in forward propagation, which also provides inspiration for this work.

ANN-SNN Conversion. ANN-SNN Conversion is another widely used method for obtaining high-performance SNNs with limited computational resources, which establishes a mathematical mapping relationship between activation layers and the Integrate-and-Fire (IF) models. Cao *et al.* [3] first proposed a two-step conversion learning framework, which replaces the activation functions of pre-trained ANNs with the IF models layer by layer. On this basis, Han *et al.* [16] and Li *et al.* [28] classified and summarized the relevant errors existed in the conversion process. Deng *et al.* [7] and Bu *et al.* [2] further reduced the conversion errors through deriving the optimal values for the bias term and initial membrane potential. For the critical conversion error caused by uneven spike firing

sequences, multiple optimization strategies have been proposed successively, including memorizing the residual membrane potential [17], firing negative spikes [43, 25], calibrating offset spikes [18] and hybrid finetuning training [45]. Currently, ANN-SNN Conversion has been further applied to the training of large-scale visual and language models [46, 32].

Spiking neural models with multi-threshold. The current proposed multi-threshold models can be generally divided into two categories: one emits signed spikes [22, 52, 43], while the other emits multi-bit spikes [27, 41, 42, 24]. However, these works generally consider using multi-threshold models to reduce ANN-SNN Conversion errors and lack further theoretical analysis. In this paper, we have the foresight to recognize the mathematical equivalence relationship between equidistant multi-threshold models and quantized ANNs under the conditions of using the soft-reset mechanism and uniform input current, achieving the current optimal performance in the domain of STBP learning.

3 Preliminaries

The spiking neuron models for SNNs. The Leaky-Integrate-and-Fire (LIF) model is one of the most commonly used models in the current SNN community. The following equations have depicted the dynamic procedure of the LIF model in a discrete form:

$$\boldsymbol{m}^l_{LIF}(t) = \lambda^l_{LIF}\boldsymbol{v}^l_{LIF}(t-1) + \boldsymbol{I}^l(t),\ \boldsymbol{I}^l(t) = \boldsymbol{W}^l\boldsymbol{s}^{l-1}_{LIF}(t)\theta^{l-1}. \tag{1}$$

$$\boldsymbol{v}^l_{LIF}(t) = \boldsymbol{m}^l_{LIF}(t) - \boldsymbol{s}^l_{LIF}(t)\theta^l,\ \boldsymbol{s}^l_{LIF}(t) = \begin{cases} 1, & \boldsymbol{m}^l_{LIF}(t) \geq \theta^l \\ 0, & \text{otherwise} \end{cases}. \tag{2}$$

Eq.(1) describes the charging process: $\forall t \in [1, T]$, $\boldsymbol{m}^l_{LIF}(t)$ and $\boldsymbol{v}^l_{LIF}(t-1)$ respectively represent the membrane potential before and after the charging at the t-th time-step. $\boldsymbol{I}^l(t)$ denotes the input current and λ^l_{LIF} characterizes the leakage degree of the membrane potential. When $\lambda^l_{LIF} = 1$, the LIF model will degenerate into a more specialized model called the IF model. Eq.(2) depicts the reset and firing process: $\boldsymbol{s}^l_{LIF}(t)$ indicates the spike emitting situation and θ^l is the firing threshold. Here we adopt the soft-reset mechanism, which means that the reset amplitude of the membrane potential is equal to the value of θ^l.

STBP learning algorithm for SNNs. The gradient calculation mode of STBP is inspired by the back-propagation Through Time (BPTT) algorithm in Recurrent Neural Network (RNN), which will propagate along the spatial and temporal dimensions of SNNs simultaneously. Following equations have described the specific propagation process:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l_{LIF}(t-1)} = \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^l_{LIF}(t-1)}\frac{\partial \boldsymbol{s}^l_{LIF}(t-1)}{\partial \boldsymbol{m}^l_{LIF}(t-1)}}_{\text{spatial dimension}} + \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l_{LIF}(t)}\frac{\partial \boldsymbol{m}^l_{LIF}(t)}{\partial \boldsymbol{v}^l_{LIF}(t-1)}\frac{\partial \boldsymbol{v}^l_{LIF}(t-1)}{\partial \boldsymbol{m}^l_{LIF}(t-1)}}_{\text{temporal dimension}}, \tag{3}$$

Here $\mathcal{L}$ denotes the target loss function. From Eq.(2) one can note that the mathematical relationship between $\boldsymbol{s}^l_{LIF}(t)$ and $\boldsymbol{m}^l_{LIF}(t)$ is equivalent to $\boldsymbol{s}^l_{LIF}(t) = H(\boldsymbol{m}^l_{LIF}(t) - \theta^l)$, where $H(\cdot)$ denotes Heaviside step function. As Heaviside function is non-differentiable, researchers consider using a surrogate function, which is approximate to Heaviside function but differentiable, to handle the term $\frac{\partial \boldsymbol{s}^l_{LIF}(t)}{\partial \boldsymbol{m}^l_{LIF}(t)}$ in the back-propagation chain. For example, $\frac{\partial \boldsymbol{s}^l_{LIF}(t)}{\partial \boldsymbol{m}^l_{LIF}(t)} = \text{sign}\left(\left|\boldsymbol{m}^l_{LIF}(t) - \theta^l\right| \leq \frac{\theta^l}{2}\right)$ describes the well-known rectangular surrogate function.

Quantized ANNs. The quantized ANN model is a widely used structure in the field of ANN-SNN Conversion. Compared to traditional ANNs, quantized ANNs usually use the following Quantization-Clip-Floor-Shift (QCFS) function [28, 2] as their activation function:

$$\boldsymbol{a}^l = \frac{\vartheta^l}{T_q}\text{clip}\left(\left\lfloor \frac{\boldsymbol{W}^l\boldsymbol{a}^{l-1}T_q + \varphi^l}{\vartheta^l} \right\rfloor, 0, T_q\right). \tag{4}$$

Here $\boldsymbol{a}^l$ and φ^l represent the activation output and shift factor, while T_q and ϑ^l denote the quantization level and learnable scaling factor. If we set $T_q = T, \vartheta^l = \theta^l, \boldsymbol{a}^l = \sum_{t=1}^{T}\boldsymbol{s}^l_{IF}(t)\theta^l/T, \boldsymbol{v}^l_{IF}(0) = \varphi^l$, one can find that the so-called QCFS function actually simulate the average spike firing rate of the IF model (we set $\boldsymbol{r}^l_{IF}(T_q) = \sum_{t=1}^{T_q}\boldsymbol{s}^l_{IF}(t)\theta^l/T_q$) under the condition of the uniform input current and soft-reset mechanism. This conclusion suggests that SNNs have the potential to maintain the same level of performance as ANNs under specific conditions.

 https://doi.org/10.52202/079017-3233

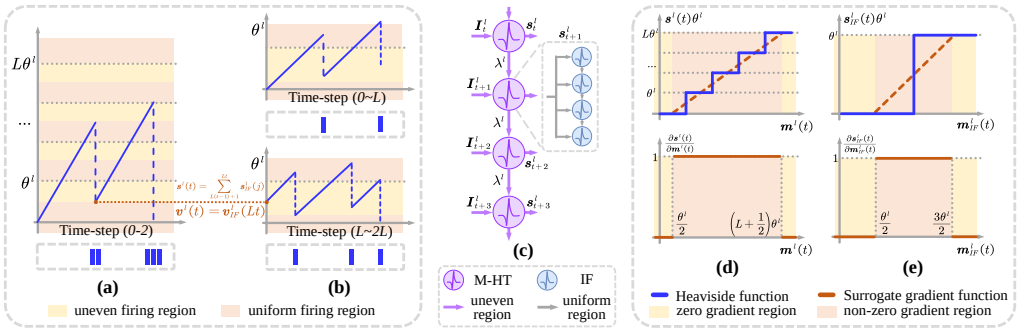

Figure 1: Forward and backward propagation of the M-HT model. (a)-(c): mathematical relationship between the M-HT model and vanilla IF model. (d)-(e): surrogate gradient calculation for the M-HT model.

4 Methodology

4.1 The Multi-hierarchical Threshold (M-HT) Model

In this section, we first introduce the M-HT model, which has equidistant multi-level thresholds and will select the threshold closest to its current membrane potential at each time-step to achieve the process of firing spikes and resetting potential. Eqs. (5)-(6) describe the dynamic equations of the M-HT model.

$$\boldsymbol{m}^l(t) = \lambda^l \boldsymbol{v}^l(t-1) + \boldsymbol{I}^l(t),\ \boldsymbol{I}^l(t) = \boldsymbol{W}^l \boldsymbol{s}^{l-1}(t)\theta^{l-1}. \tag{5}$$

$$\boldsymbol{v}^l(t) = \boldsymbol{m}^l(t) - \boldsymbol{s}^l(t)\theta^l,\ \boldsymbol{s}^l(t) = \begin{cases} L, & \boldsymbol{m}^l(t) \geq L\theta^l \\ k, & k\theta^l \leq \boldsymbol{m}^l(t) < (k+1)\theta^l, k = 1,...,L-1 \\ 0, & \text{otherwise} \end{cases}. \tag{6}$$

Here L denotes the number of level for the firing threshold. Regarding the surrogate gradient calculation of the M-HT model, similar to the vanilla spiking models, we propose $\frac{\partial \boldsymbol{s}^l(t)}{\partial \boldsymbol{m}^l(t)} = \text{sign}\left(\frac{1}{2}\theta^l \leq \boldsymbol{m}^l(t) \leq (L+\frac{1}{2})\theta^l\right)$, which covers a wider range of the membrane potential, as shown in Fig.1(d)-(e). As the M-HT model has L different firing options at each time-step, we can consider the information transmitted by the M-HT model within one time-step as an information integration of the vanilla model for L time-steps. Therefore, we attempt to bridge a mathematical equivalent relationship between the M-HT and IF model:

Lemma 4.1. *$\forall t \in [1,T]$, if $\boldsymbol{v}^l(t-1) \in [0,\theta^l)$, the effect of inputting current $\boldsymbol{I}^l(t)$ into a M-HT model with L-level threshold at the t-th time-step, is equivalent to continuously inputting uniform current $\boldsymbol{I}^l(t)/L$ for L time-steps into a IF model with $\boldsymbol{v}^l_{IF}(0) = \boldsymbol{v}^l(t-1)$, i.e. $\boldsymbol{s}^l(t) = clip\left(\left\lfloor \frac{\boldsymbol{v}^l(t-1)+\boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, L\right) = \sum_{j=1}^{L} \boldsymbol{s}^l_{IF}(j)$.*

Lemma 4.1 indicates that the M-HT model under a single time-step can be used to simulate the total number of spikes emitted by the IF model under uniform input current within L consecutive time-steps. In addition, note that $\boldsymbol{s}^l(t)$ in Lemma 4.1 can also be calculated through clip $(\lfloor \cdot \rfloor, \cdot, \cdot)$, which is equivalent to the QCFS function mentioned before in quantized ANNs. The above conclusion preliminarily demonstrates that the M-HT model can achieve the same-level performance as pre-trained ANNs with L-level quantization under single-step condition.

4.2 The Representation Ability of the M-HT Model on Multiple Time-steps

Based on Lemma 4.1, we further consider the information representation of the M-HT model on multiple time-steps:

Theorem 4.2. *When $\lambda^l = 1, \boldsymbol{v}^l(0) \in [0,\theta^l)$, for a M-HT model with L-level threshold, after T time-steps, we will derive the following conclusions:*

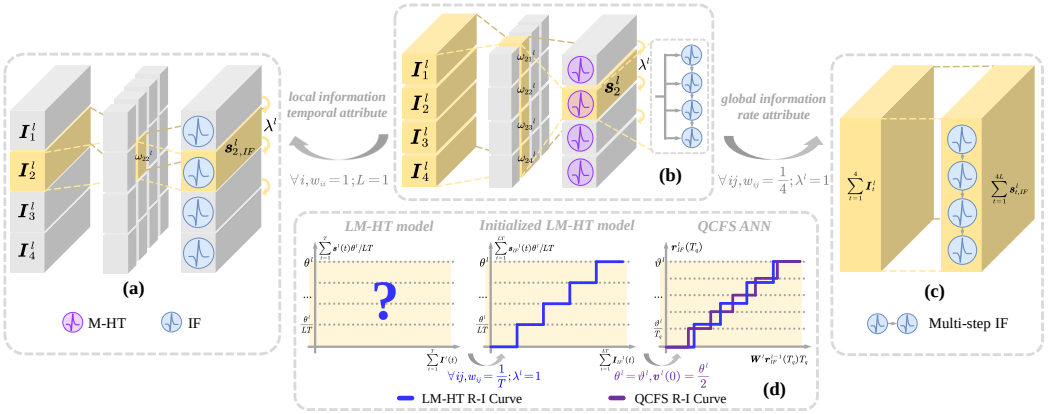

Figure 2: The STBP learning framework based on the LM-HT model. (a): vanilla STBP training. (b): STBP training with the LM-HT model. (c): direct training of quantized ANNs. (d): hybrid training with the LM-HT model, here R-I Curve denotes Rate-Input Curve.

(i) If we further assume $\forall t \in [1,T], \boldsymbol{I}^l(t) \in [0, L\theta^l)$*, we will have:* $\forall t \in [1,T], \boldsymbol{s}^l(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j), \boldsymbol{v}^l(t) = \boldsymbol{v}^l_{IF}(Lt), \sum_{t=1}^{T} \boldsymbol{s}^l(t) = \sum_{j=1}^{LT} \boldsymbol{s}^l_{IF}(j)$.
(ii) If we further assume $\boldsymbol{I}^l(1) = ... = \boldsymbol{I}^l(T)$*, we will have:* $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = clip\left(\left\lfloor \frac{\boldsymbol{v}^l(0)+\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT\right)$.
Here the IF model has uniform input currents $\boldsymbol{I}^l(1)/L, ..., \boldsymbol{I}^l(T)/L$ *respectively within every* L *steps and satisfies* $\boldsymbol{v}^l_{IF}(0) = \boldsymbol{v}^l(0)$.

The proofs of Lemma 4.1 and Theorem 4.2 have been provided in the Appendix. From Theorem 4.2(i) and Fig.1(a)-(c), one can find that the M-HT model is actually equivalent to dividing the spike firing sequence of the IF model on consecutive LT steps into T L-step time windows. Combining with the soft-reset mechanism, the M-HT model actually focuses on a specific time window of the vanilla IF model at each time-step and maintains an equal membrane potential with the IF model at the end of each time window (*i.e.* $\forall t \in [1,T], \boldsymbol{v}^l(t) = \boldsymbol{v}^l_{IF}(Lt)$). The M-HT model follows the assumption of uniform input current within each window, while maintaining the basic calculation properties of spiking neurons between different windows. When the input current follows a complete uniform distribution, according to Theorem 4.2(ii), the M-HT model can further simulate the output of an ANN with LT-level quantization.

4.3 The Learnable Multi-hierarchical Threshold (LM-HT) Model

The uniform and uneven firing regions in the M-HT model. For a specific spike firing rate, the M-HT model can often provide multiple spike firing sequences. For example, $[1,1], [0,2], [2,0]$ can all represent the situation where 2 spikes are emitted within 2 time-steps, while only $[1,1]$ can be viewed as a case of uniform firing situation. However, even when the input current is uniformly distributed, as the sum of spikes that cannot be divided by L in $[0, LT]$ is unable to be represented by a uniform spike output sequence, there are still uneven firing situations:

Corollary 4.3. *If* $\lambda^l = 1, \boldsymbol{v}^l(0) = 0$ *and* $\boldsymbol{I}^l(1) = ... = \boldsymbol{I}^l(T)$*, for a M-HT model with* L*-level threshold,* $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T)$ *is only satisfied when* $\boldsymbol{I}^l(1) \in [k\theta^l, k\theta^l + \theta^l/T), \forall k = 0, ..., L-1$ *or* $\boldsymbol{I}^l(1) \in (-\infty, 0) \cup [L\theta^l, +\infty)$.

The proof is provided in the Appendix. From Corollary 4.3, we can divide the input current into uniform and uneven firing regions according to the corresponding intervals, as shown in Fig.1(a)-(b). Note that the uneven spike sequences emitted by the l-th layer may further cause the input current of the $l+1$-th layer to no longer follow the uniform distribution. That is to say, as the number of layers increases, the uneven firing cases will tend to increase gradually without introducing extra regulation.

Learnable Temporal-Global Information Matrix and leaky parameters. Enhancing the uniform firing pattern can promote SNNs to achieve superior performance similar to quantized ANNs, while

https://doi.org/10.52202/079017-3233

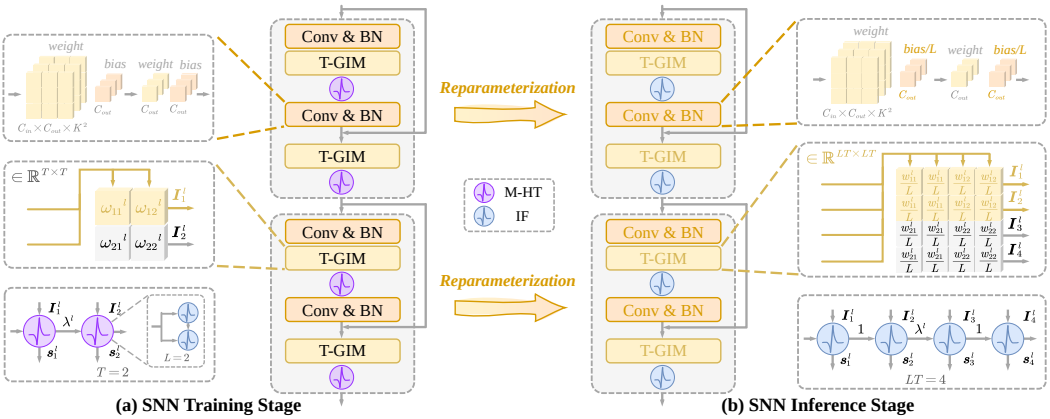

Figure 3: Reparameterization procedure of the LM-HT model.

uneven spike sequences retain more temporal and biological characteristics. Therefore, how to comprehensively utilize these two spike firing patterns becomes a critical problem. To address this issue, we first introduce the concept of Temporal-Global Information Matrix (T-GIM):

$$\forall t \in [1, T], \boldsymbol{I}^l(t) = \sum_{j=1}^{T} \omega_{tj}^l \boldsymbol{W}^l \boldsymbol{s}^{l-1}(j)\theta^{l-1}. \tag{7}$$

Here ω_{tj}^l is the element at row t and column j of the T-GIM $\boldsymbol{\Omega}^l$, $\boldsymbol{\Omega}^l \in \mathbb{R}^{T\times T}$. As shown in Eq.(7) and Fig.2(b), this brand-new input current adopts a multi-step current weighting form, allowing the model to simultaneously focus on the global information along the time dimension. Note that the new input current will follow a uniform distribution when $\forall i, j \in [1, T], \omega_{ij}^l = \frac{1}{T}$ and degrade to the vanilla input current when $\boldsymbol{\Omega}^l = \mathrm{diag}(1, ..., 1)$. For the first case mentioned above, if we further add the condition $\lambda^l = 1$, according to Theorem 4.2(ii), one can find that the output of the model will be consistent with the activation output of a LT-level quantized ANN layer by layer, as shown in Fig.2(b)-(c). For the second case, when $L = 1$, the model will degenerate into vanilla LIF model, as shown in Fig.2(a)-(b).

To enable the model to dynamically adjust the above calculation process, we set both $\boldsymbol{\Omega}^l$ and λ^l as learnable parameters. The initial values of $\boldsymbol{\Omega}^l$ and λ^l are set to $1/T$ and 1, respectively. During the training process, we choose the Sigmoid function $\sigma(\cdot)$ to control the parameters for fulfilling smooth gradient updates within a bounded learning range. We call this novel model as Learnable Multi-hierarchical Threshold (LM-HT) Model, which combines T-GIM and learnable attributes. We think the LM-HT model can regulate its spike firing pattern more flexibly and reasonably.

Since we can regulate the computational relationships between different time-steps through learnable $\boldsymbol{\Omega}^l$ and λ^l in the LM-HT model, during the back-propagation process, unlike Eq.(3), we detach the term $\frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l(t)} \frac{\partial \boldsymbol{m}^l(t)}{\partial \boldsymbol{v}^l(t-1)} \frac{\partial \boldsymbol{v}^l(t-1)}{\partial \boldsymbol{m}^l(t-1)}$ from the gradient calculation graph, thereby reducing redundant calculations and completely leaving the gradient propagation between different time-steps to $\boldsymbol{\Omega}^l$ and λ^l for control. The back-propagation calculation chains for the LM-HT model have been described as follow. Here $\odot$ denotes the Hadamard product.

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l(t)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^l(t)} \frac{\partial \boldsymbol{s}^l(t)}{\partial \boldsymbol{m}^l(t)}, \ \frac{\partial \boldsymbol{s}^l(t)}{\partial \boldsymbol{m}^l(t)} = \mathrm{sign}\left(\frac{1}{2}\theta^l \le \boldsymbol{m}^l(t) \le \left(L + \frac{1}{2}\right)\theta^l\right). \tag{8}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda^l} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l(t)} \odot \boldsymbol{v}^l(t-1), \ \frac{\partial \mathcal{L}}{\partial \omega_{ij}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{m}^l(i)} \odot \left(\boldsymbol{W}^l \boldsymbol{s}^{l-1}(j)\theta^{l-1}\right). \tag{9}$$

4.4 Hybrid Training based on the LM-HT Model

Although the traditional ANN-SNN Conversion frameworks have much lower computational overhead than STBP training algorithm, a serious performance degradation phenomenon often exists on the

Table 1: Ablation study for the LM-HT model on a subset of ImageNet-1k.

Model	T-GIM	Arch.	Acc.(%)	SOPs(G)	E.(mJ)
L=1,T=4	w/o	ResN-18	76.22	1.60	1.44
L=2,T=2	w/o		80.52	1.08	0.97
L=2,T=2	w/		80.56	0.73	0.66
L=1,T=4	w/o	ResN-34	65.62	3.20	2.88
L=2,T=2	w/o		82.18	2.42	2.18
L=2,T=2	w/		82.72	1.72	1.55

Table 2: Validation for the reparameterization procedure.

Arch.	Acc.(%)	SOPs(G)	E.(mJ)
Before reparameterization (L=2, T=2)			
VGG-13	61.64	0.26	0.23
ResN-18	64.44	0.52	0.46
After reparameterization (L=1, T=4)			
VGG-13	61.66	0.26	0.23
ResN-18	64.50	0.52	0.46

converted SNNs under low time latency [17]. To address this problem, previous researchers [39] considered adopting STBP training for a few epochs on the pre-trained ANN models to enhance the performance of the converted SNNs under fewer time-steps, which is called as hybrid training. In this work, we propose a brand-new hybrid training framework based on the LM-HT model.

We firstly choose QCFS function to train the quantized ANN models and then replace the QCFS function modules layer by layer with the LM-HT models under specific initialization ($\forall i,j \in [1,T], \omega_{ij}^l = \frac{1}{T}; \lambda^l = 1, \theta^l = \vartheta^l, \boldsymbol{v}^l(0) = \frac{\theta^l}{2}$), as shown in Fig.2(d). Combining with the conclusion pointed out by [2], one can note that the initialized LM-HT model and the QCFS function before substitution have an equivalence in terms of mathematical expectation, which has been described as the following theorem:

Theorem 4.4. *When* $\sum_{t=1}^{T} \boldsymbol{I}^l(t)/LT = \boldsymbol{W}^l \boldsymbol{r}_{IF}^{l-1}(T_q)$ *and* $\sum_{t=1}^{T} \boldsymbol{I}^l(t) \in [0, LT\theta^l]$*, if* $\forall i,j \in [1,T], \omega_{ij}^l = \frac{1}{T}$ *and* $\lambda^l = 1, \theta^l = \vartheta^l, \boldsymbol{v}^l(0) = \frac{\theta^l}{2}$*, for* L, T, T_q *with arbitrary values, we have:* $\mathbb{E}\left(\frac{\sum_{t=1}^{T} \boldsymbol{s}^l(t)\theta^l}{LT} - \frac{\vartheta^l}{T_q} clip\left(\left\lfloor \frac{\boldsymbol{W}^l \boldsymbol{r}_{IF}^{l-1}(T_q)T_q}{\vartheta^l} + \frac{1}{2} \right\rfloor, 0, T_q\right)\right) = 0.$

Theorem 4.4 indicates that regardless of whether the time-steps we choose during the STBP training phase is equal to the inference steps simulated in ANN-SNN Conversion, the average spike firing rate of the LM-HT models under the initial state of STBP training maintains a mathematical equivalence with that simulated by the QCFS function modules in the previous stage. Therefore, under this new training framework, we can adopt STBP algorithm to optimize the inference performance of SNN under any degree of time latency. The detailed pseudo-code has been provided in the Appendix.

4.5 Reparameterize the LM-HT model to vanilla LIF model

As discussed in Section 4.2, the mathematical essence of the LM-HT model is to simulate the spike firing situation of vanilla LIF neurons within each time window. Considering that the current neuromorphic hardware mainly supports single threshold models, we propose a reparameterization scheme that can transform the LM-HT model obtained during the training stage into a vanilla LIF model, which can further be deployed on hardware for inference.

As shown in Fig.3, for a L-level LM-HT model within T steps, we expand it into a vanilla LIF model within LT steps, where the membrane leakage factor between different time windows is set to λ^l. In addition, T-GIM will be extended from $\mathbb{R}^{T\times T}$ to $\mathbb{R}^{LT\times LT}$ and the parameters are averaged within each $L \times L$ sub-region, ensuring that the input current meets the precondition in Theorem 4.2. We also rectify the bias terms in synaptic layers, which involve addition operations at each time-step. By performing layer-by-layer reparameterization in the above manner, we will obtain a single threshold SNN model with theoretically lossless accuracy.

5 Experiments

To validate the effectiveness of our proposed STBP and hybrid training frameworks based on the LM-HT model, we consider multiple static and neuromorphic datasets with different data scale, including CIFAR-10(100) [23], ImageNet-200(1k) [6] and CIFAR10-DVS [26]. Consistent with the previous works, we also choose VGG [40] and ResNet [19] as the basic network architecture . We evaluate the computational overhead of SNNs based on the number of synaptic operations (SOPs) and the calculation standard for related energy consumption refers to [55]. In addition, as the information transmitted by our L-level LM-HT model within T time-steps remains at the same level as that of the

 https://doi.org/10.52202/079017-3233

Table 3: Comparison with previous state-of-the-art works.

Dataset	Method	Type	Architecture	Time-steps	Accuracy(%)
CIFAR-10	STBP-tdBN [54]	Direct Training	ResNet-19	4	92.92
	Dspike [29]	Direct Training	ResNet-18	4	93.66
	TET [8]	Direct Training	ResNet-19	4	94.44
	SLTT [34]	Online Training	ResNet-18	6	94.44
	GLIF [51]	Direct Training	ResNet-18	2, 4, 6	94.15, 94.67, 94.88
			ResNet-19	2, 4, 6	94.44, 94.85, 95.03
	LM-HT (L=2)	**Direct Training**	**ResNet-18**	**2**	**96.25**
			ResNet-19	**2**	**96.89**
CIFAR-100	Dspike [29]	Direct Training	ResNet-18	4	73.35
	TET [8]	Direct Training	ResNet-19	4	74.47
	SLTT [34]	Online Training	ResNet-18	6	74.38
	GLIF [51]	Direct Training	ResNet-18	2, 4, 6	74.60, 76.42, 77.28
			ResNet-19	2, 4, 6	75.48, 77.05, 77.35
	RMP-Loss [14]	Direct Training	ResNet-19	2, 4, 6	74.66, 78.28, 78.98
	LM-HT (L=2)	**Direct Training**	**ResNet-18**	**2**	**79.33**
			ResNet-19	**2**	**81.76**
ImageNet-200	DCT [13]	Hybrid Training	VGG-13	125	56.90
	Online-LTL [48]	Hybrid Training	VGG-13	16	54.82
	Offline-LTL [48]			16	55.37
	ASGL [44]	Direct Training	VGG-13	4, 8	56.57, 56.81
	LM-HT (L=2)	**Direct Training**	**VGG-13**	**2, 4**	**61.09, 61.75**
	LM-HT (L=4)			**2**	**62.05**
ImageNet-1k	STBP-tdBN [54]	Direct Training	ResNet-34	6	63.72
	TET [8]	Direct Training	ResNet-34	6	64.79
	MBPN [15]	Direct Training	ResNet-34	4	64.71
	RMP-Loss [14]	Direct Training	ResNet-34	4	65.17
	SEW ResNet [11]	Direct Training	ResNet-34	4	67.04
	GLIF [51]	Direct Training	ResNet-34	4	67.52
	LM-HT (L=2)	**Direct Training**	**ResNet-34**	**2**	**70.90**
CIFAR10-DVS	STBP-tdBN [54]	Direct Training	ResNet-19	10	67.80
	Dspike [29]	Direct Training	ResNet-18	10	75.40
	MBPN [15]	Direct Training	ResNet-19	10	74.40
	RMP-Loss [14]	Direct Training	ResNet-19	10	76.20
	LM-HT (L=2)	**Direct Training**	**ResNet-18**	**2, 4**	**80.70, 81.00**
	LM-HT (L=4)			**2**	**81.90**

vanilla LIF model within LT time-steps, to make a fair evaluation, we will compare the performance of the L-level LM-HT model within T steps with that of the previous works within LT steps.

5.1 Ablation & Validation Studies for the LM-HT Model

As shown in Tab.1, we investigate the impact of threshold levels and T-GIM for our proposed model. One can note that vanilla IF neuron ($L = 1, T = 4$) is not well suited for deep networks (*e.g.* ResNet-34) and causes relatively high energy consumption, while the M-HT series models ($L = 2, T = 2$) can effectively overcome the performance degradation problem on deep networks. When we further utilize T-GIM to regulate global information on the time dimension, the learning ability of our model is enhanced and the computational overhead in synaptic layers is significantly reduced.

We also validate the feasibility about the reparameterization procedure mentioned above. As shown in Tab.2 and Fig.3, by copying and reparameterizing the parameters of synapses, T-GIM and LM-HT neurons layer by layer, we obtain a single threshold model that maintained almost the same performance and power consumption as the original LM-HT model. This convertible property enables the LM-HT model to be more flexibly deployed on neuromorphic hardware.

5.2 Comparison with Previous SoTA Works

We first investigate the competitiveness of our proposed model in the domain of STBP learning. As shown in Tab.3, our comparative works incorporate previous state-of-the-art (SoTA) methods in various sub-domains of STBP training, including batchnorm layer optimization [54, 15], improved surrogate gradients [29], learning function design [8, 14], energy-efficient training [13, 48, 34] and advanced neuron models [51, 44].

Table 4: The performance of hybrid training based on the LM-HT model for CIFAR-100 dataset.

Method	Time-steps	VGG-16		ResNet-20	
		ANN Acc.(%)	SNN Acc.(%)	ANN Acc.(%)	SNN Acc.(%)
RMP [16]	32, 64, 128	71.22	- , - , 63.76	68.72	27.64, 46.91, 57.69
SNM [43]	32, 64, 128	74.13	71.80, 73.69, 73.95	-	-
SRP [17]	5, 6, 8	76.28	71.52, 74.31, 75.42	69.94	46.48, 53.96, 59.34
QCFS (T_q=4) [2]	2, 4, 8	76.11	63.33, 69.70, 74.12	63.90	38.04, 52.28, 61.77
LM-HT (L=2)	**2**	-	**75.97 (+6.27)**	-	**63.55 (+11.27)**
	4	-	**76.49 (+2.37)**	-	**64.87 (+3.10)**
LM-HT (L=4)	**2**	-	**76.38 (+2.26)**	-	**63.43 (+1.66)**
QCFS (T_q=8) [2]	2, 4, 8	77.31	64.85, 70.50, 74.63	69.56	19.76, 34.17, 55.50
LM-HT (L=2)	**2**	-	**76.31 (+5.81)**	-	**67.08 (+32.91)**
	4	-	**76.79 (+2.16)**	-	**69.00 (+13.50)**
LM-HT (L=4)	**2**	-	**76.08 (+1.45)**	-	**67.21 (+11.71)**

CIFAR-10 & CIFAR-100. For conventional static datasets, one can find that our solution demonstrates significant performance advantages. For ResNet-18 structure, we achieve the top-1 accuracies of 96.25% and 79.33% with merely 2 time-steps on CIFAR-10 and CIFAR-100 datasets, respectively. For ResNet-19 network with a larger parameter scale, our method fulfills the precisions of 96.89% and 81.76% within 2 time-steps, which at least outperforms other corresponding works with 2.04% and 3.48% under the same time latency. In addition, it is worth noting that our above results have even exceeded the performance of other works with more time-steps (*e.g.* 6 steps).

ImageNet-200 & ImageNet-1k. For large-scale datasets, we also confirm the superiority of the LM-HT model. For the two-level LM-HT model, we respectively reach the top-1 accuracies of 61.09% and 70.90% within 2 time-steps on ImageNet-200 and ImageNet-1k datasets, which is 4.52% higher than ASGL (4 steps) and 3.38% higher than GLIF (4 steps) under the same-level time overhead. For a larger training time-step, one can note that our method will also demonstrate a significant advantage. For example, the two-level LM-HT model reaches the precision of 61.75% with 4 time-steps, which has surpassed ASGL (8 steps) with 4.94%.

CIFAR10-DVS. We also evaluate the effectiveness of our approach on neuromorphic datasets. Compared to other previous methods, our proposed model can achieve better results on shallower networks with fewer time-steps. For instance, the two-level LM-HT model can achieve the accuracy of 80.70% after merely 2 time-steps.

5.3 Performance Analysis of Hybrid Training

In our hybrid training framework, we first choose [2] as the backbone for our ANN-SNN Conversion stage. Subsequently, we replace the QCFS function layer by layer with the initialized LM-HT model and conduct STBP training for merely 30 epochs. Furthermore, we also consider other advanced conversion methods [16, 43] and multi-stage error correction method [17] as our comparative works.

As shown in Tab.4, after conducting the STBP fine-tuning optimization with relatively low computational overhead, we note that the performance of the converted SNNs under different quantization levels has been significantly improved and surpass other previous methods, especially under low time latency. For instance, compared to the ResNet-20 network after eight-level quantization (*i.e.* T_q=8), the two-level LM-HT model has achieved a performance improvement of 32.91% and 13.50% with 2 and 4 time-steps, respectively.

6 Conclusions

In this paper, we first investigate the mathematical equivalence among the multi-threshold model, vanilla spiking model and quantized ANNs, then propose an advanced STBP training method based on the LM-HT model, which has been proven to cover the representation range of vanilla STBP and quantized ANNs training frameworks, thereby promoting SNNs to achieve superior performance at the same level as quantized ANNs. Furthermore, the LM-HT model can achieve lossless transformation towards single threshold models or quantized ANNs under specific parameter configuration. Numerous experimental results have verified the effectiveness of our method. We believe that our work will further promote in-depth research on advanced spiking neural model.

 https://doi.org/10.52202/079017-3233

Acknowledgements

This work was supported by STI 2030-Major Projects 2021ZD0200300, the National Natural Science Foundation of China under Grant No. 62176003 and No. 62088102, and by Beijing Nova Program under Grant No. 20230484362.

References

[1] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[2] Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks. In *International Conference on Learning Representations*, 2022.

[3] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015.

[4] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[5] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018.

[6] Jia Deng, Richard Socher, Lijia Li, Kai Li, and Feifei Li. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[7] Shikuang Deng and Shi Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. In *International Conference on Learning Representations*, 2021.

[8] Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. *International Conference on Learning Representations*, 2022.

[9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[10] Chaoteng Duan, Jianhao Ding, Shiyan Chen, Zhaofei Yu, and Tiejun Huang. Temporal effective batch normalization in spiking neural networks. In *Advances in Neural Information Processing Systems*, 2022.

[11] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. In *Advances in Neural Information Processing Systems*, 2021.

[12] Wei Fang, Zhaofei Yu, Zhaokun Zhou, Ding Chen, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. In *Advances in Neural Information Processing Systems*, 2023.

[13] Isha Garg, Sayeed Shafayet Chowdhury, and Kaushik Roy. DCT-SNN: Using dct to distribute spatial information over time for low-latency spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[14] Yufei Guo, Xiaode Liu, Yuanpei Chen, Liwen Zhang, Weihang Peng, Yuhan Zhang, Xuhui Huang, and Zhe Ma. RMP-Loss: Regularizing membrane potential distribution for spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[15] Yufei Guo, Yuhan Zhang, Yuanpei Chen, Weihang Peng, Xiaode Liu, Liwen Zhang, Xuhui Huang, and Zhe Ma. Membrane potential batch normalization for spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[16] Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy. RMP-SNN: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[17] Zecheng Hao, Tong Bu, Jianhao Ding, Tiejun Huang, and Zhaofei Yu. Reducing ann-snn conversion error through residual membrane potential. In *AAAI Conference on Artificial Intelligence*, 2023.

[18] Zecheng Hao, Jianhao Ding, Tong Bu, Tiejun Huang, and Zhaofei Yu. Bridging the gap between anns and snns by calibrating offset spikes. In *International Conference on Learning Representations*, 2023.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[20] Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks towards deep residual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[21] Saeed Reza Kheradpisheh and Timothée Masquelier. Temporal backpropagation for spiking neural networks with one spike per neuron. *International Journal of Neural Systems*, 30(06):2050027, 2020.

[22] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: Spiking neural network for energy-efficient object detection. In *AAAI Conference on Artificial Intelligence*, 2020.

[23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[24] Yuxiang Lan, Yachao Zhang, Xu Ma, Yanyun Qu, and Yun Fu. Efficient converted spiking neural network for 3d and 2d classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[25] Chen Li, Lei Ma, and Steve Furber. Quantization framework for fast spiking neural networks. *Frontiers in Neuroscience*, 16, 2022.

[26] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in Neuroscience*, 2017.

[27] Yang Li and Yi Zeng. Efficient and accurate conversion of spiking neural network with burst spikes. In *International Joint Conference on Artificial Intelligence*, 2022.

[28] Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration. In *International Conference on Machine Learning*, 2021.

[29] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. In *Advances in Neural Information Processing Systems*, 2021.

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[31] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[32] Changze Lv, Jianhan Xu, and Xiaoqing Zheng. Spiking convolutional neural networks for text classification. In *International Conference on Learning Representations*, 2023.

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

 https://doi.org/10.52202/079017-3233

[34] Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhiquan Luo. Towards memory and time-efficient backpropagation for training spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[35] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[36] Hesham Mostafa. Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):3227–3235, 2017.

[37] Jing Pei, Lei Deng, Sen Song, Mingguo Zhao, Youhui Zhang, Shuang Wu, Guanrui Wang, Zhe Zou, Zhenzhi Wu, Wei He, et al. Towards artificial general intelligence with hybrid tianjic chip architecture. *Nature*, 572(7767):106–111, 2019.

[38] Xuerui Qiu, Rui-Jie Zhu, Yuhong Chou, Zhaorui Wang, Liang-jian Deng, and Guoqi Li. Gated attention coding for training high-performance and efficient spiking neural networks. In *AAAI Conference on Artificial Intelligence*, 2024.

[39] Nitin Rathi and Kaushik Roy. 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, 2021.

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

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

[42] Xiaoting Wang, Yanxiang Zhang, and Yongzhe Zhang. MT-SNN: Enhance spiking neural network with multiple thresholds. *arXiv preprint arXiv:2303:11127*, 2023.

[43] Yuchen Wang, Malu Zhang, Yi Chen, and Hong Qu. Signed neuron with memory: Towards simple, accurate and high-efficient ANN-SNN conversion. In *International Joint Conference on Artificial Intelligence*, 2022.

[44] Ziming Wang, Runhao Jiang, Shuang Lian, Rui Yan, and Huajin Tang. Adaptive smoothing gradient learning for spiking neural networks. In *International Conference on Machine Learning*, 2023.

[45] Ziming Wang, Yuhao Zhang, Shuang Lian, Xiaoxin Cui, Rui Yan, and Huajin Tang. Towards high-accuracy and low-latency spiking neural networks with two-stage optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[46] Ziqing Wang, Yuetong Fang, Jiahang Cao, Qiang Zhang, Zhongrui Wang, and Renjing Xu. Masked spiking transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

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

[48] Qu Yang, Jibin Wu, Malu Zhang, Yansong Chua, Xinchao Wang, and Haizhou Li. Training spiking neural networks with local tandem learning. In *Advances in Neural Information Processing Systems*, 2022.

[49] Man Yao, Jiakui Hu, Zhaokun Zhou, Yuan Li, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[50] Man Yao, Guangshe Zhao, Hengyu Zhang, Yifan Hu, Lei Deng, Yonghong Tian, Bo Xu, and Guoqi Li. Attention spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9393–9410, 2023.

[51] Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. GLIF: A unified gated leaky integrate-and-fire neuron for spiking neural networks. In *Advances in Neural Information Processing Systems*, 2022.

[52] Qiang Yu, Chenxiang Ma, Shiming Song, Gaoyan Zhang, Jianwu Dang, and Kay Chen Tan. Constructing accurate and efficient deep spiking neural networks with double-threshold and augmented schemes. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1714–1726, 2022.

[53] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[54] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *AAAI Conference on Artificial Intelligence*, 2021.

[55] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Yuan Li. Spikformer: When spiking neural network meets transformer. In *International Conference on Learning Representations*, 2023.

 https://doi.org/10.52202/079017-3233

A Appendix

A.1 Proof of Theorem

A.1.1 Proof of Lemma 4.1 & Theorem 4.2

Before the proof of Theorem 4.2, we first need to introduce Lemma A.1:

Lemma A.1. *Assume a continuous T-step input current $\boldsymbol{I}^l(1), ..., \boldsymbol{I}^l(T)$, for a LM-HT model with L-level threshold, when $\forall t \in [1, T], \boldsymbol{I}^l(t) \in [0, L\theta^l)$ and $\boldsymbol{v}^l(0) \in [0, \theta^l), \lambda^l = 1$, we will have $\boldsymbol{v}^l(T) \in [0, \theta^l)$.*

Proof. $\forall t \in [0, T)$, if $\boldsymbol{v}^l(t) \in [0, \theta^l)$, as $\boldsymbol{m}^l(t+1) = \boldsymbol{v}^l(t) + \boldsymbol{I}^l(t)$, we have $\boldsymbol{m}^l(t+1) \in [0, (L+1)\theta^l)$. Therefore, after the firing process $\boldsymbol{v}^l(t+1) = \boldsymbol{m}^l(t+1) - \boldsymbol{s}^l(t)\theta^l$, one can note that $\boldsymbol{v}^l(t+1) \in [0, \theta^l)$. According to the idea of mathematical induction, if we directly set $\boldsymbol{v}^l(0) \in [0, \theta^l)$, we can have $\boldsymbol{v}^l(T) \in [0, \theta^l)$. □

Theorem 4.2. *When $\lambda^l = 1, \boldsymbol{v}^l(0) \in [0, \theta^l)$, for a M-HT model with L-level threshold, after T time-steps, we will derive the following conclusions:*
(i) If we further assume $\forall t \in [1, T], \boldsymbol{I}^l(t) \in [0, L\theta^l)$, we will have: $\forall t \in [1, T], \boldsymbol{s}^l(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j), \boldsymbol{v}^l(t) = \boldsymbol{v}^l_{IF}(Lt), \sum_{t=1}^{T} \boldsymbol{s}^l(t) = \sum_{j=1}^{LT} \boldsymbol{s}^l_{IF}(j)$.
(ii) If we further assume $\boldsymbol{I}^l(1) = ... = \boldsymbol{I}^l(T)$, we will have: $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = clip\left(\left\lfloor \frac{\boldsymbol{v}^l(0) + \sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT\right)$.
Here the IF model has uniform input currents $\boldsymbol{I}^l(1)/L, ..., \boldsymbol{I}^l(T)/L$ respectively within every L steps and satisfies $\boldsymbol{v}^l_{IF}(0) = \boldsymbol{v}^l(0)$.

Proof. (i) If we consider the pre-condition in Theorem 4.2 and combine Eq.(1) with Eq.(2), $\forall t \in [1, LT]$, we will have:

$$\boldsymbol{v}^l_{IF}(t) - \boldsymbol{v}^l_{IF}(t-1) = \boldsymbol{I}^l\left(\left\lceil \frac{t}{L} \right\rceil\right)/L - \boldsymbol{s}^l_{IF}(t)\theta^l. \tag{S1}$$

Similarly, if we set $\lambda^l = 1$ and incorporate Eq.(5), $\forall t \in [1, T]$, we will have:

$$\boldsymbol{v}^l(t) - \boldsymbol{v}^l(t-1) = \boldsymbol{I}^l(t) - \boldsymbol{s}^l(t)\theta^l. \tag{S2}$$

Then we accumulate Eq.(S1) along the time dimension and obtain the following equation:

$$\boldsymbol{v}^l_{IF}(Lt) - \boldsymbol{v}^l_{IF}\left(L(t-1)\right) = \boldsymbol{I}^l(t) - \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j)\theta^l. \tag{S3}$$

As $\boldsymbol{I}^l(t) \in [0, L\theta^l)$, according to Lemma A.1, when $\boldsymbol{v}^l(t-1) = \boldsymbol{v}^l_{IF}\left(L(t-1)\right) \wedge \boldsymbol{v}^l(t-1) \in [0, \theta^l)$, we will have $\boldsymbol{v}^l(t) \in [0, \theta^l)$ and $\boldsymbol{v}^l_{IF}(Lt) \in [0, \theta^l)$. Considering $\boldsymbol{s}^l(t), \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j) \in \mathbb{N}$, if $\boldsymbol{s}^l(t) \neq \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j)$, one can note that $|\sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j)\theta^l - \boldsymbol{s}^l(t)\theta^l| = |(\boldsymbol{v}^l(t) - \boldsymbol{v}^l(t-1)) - (\boldsymbol{v}^l_{IF}(Lt) - \boldsymbol{v}^l_{IF}\left(L(t-1)\right))| = |\boldsymbol{v}^l(t) - \boldsymbol{v}^l_{IF}(Lt)| \geq \theta^l$, which will violate the conclusion in Lemma A.1. Therefore, we can finally deduce that $\boldsymbol{s}^l(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}^l_{IF}(j)$. Then we can further have $\boldsymbol{v}^l(t) = \boldsymbol{v}^l_{IF}(Lt)$ and $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = \sum_{j=1}^{LT} \boldsymbol{s}^l_{IF}(j)$.

(ii) If we accumulate Eq.(S2) along the time dimension and divide θ^l on both sides, we will have the following equation:

$$\frac{\boldsymbol{v}^l(T) - \boldsymbol{v}^l(0)}{\theta^l} = \frac{\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} - \sum_{t=1}^{T} \boldsymbol{s}^l(t). \tag{S4}$$

If $\boldsymbol{I}^l(1) < 0$ or $\boldsymbol{I}^l(1) \geq L\theta^l$, it is obvious that we will have $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = \text{clip}\left(\left\lfloor \frac{\boldsymbol{v}^l(0) + \sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT\right) = 0$ or $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = \text{clip}\left(\left\lfloor \frac{\boldsymbol{v}^l(0) + \sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT\right) = L$.

If $\boldsymbol{I}^l(1) \in [0, L\theta^l)$, according to Lemma A.1, we will have $\boldsymbol{v}^l(T) \in [0, \theta^l)$. As $\sum_{t=1}^{T} \boldsymbol{s}^l(t) \in \mathbb{N}$, based on Eq.(S4), we can finally deduce that $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = \frac{\boldsymbol{v}^l(0)+\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} - \frac{\boldsymbol{v}^l(T)}{\theta^l} = \left\lfloor \frac{\boldsymbol{v}^l(0)+\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor = \text{clip}\left(\left\lfloor \frac{\boldsymbol{v}^l(0)+\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT \right)$.

One can note that Lemma 4.1 is actually a special case of Theorem 4.2 under the condition of $T = 1$, therefore Lemma 4.1 is also proven. □

A.1.2 Proof of Corollary 4.3

Corollary 4.3. *If $\lambda^l = 1, \boldsymbol{v}^l(0) = 0$ and $\boldsymbol{I}^l(1) = ... = \boldsymbol{I}^l(T)$, for a M-HT model with L-level threshold, $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T)$ is only satisfied when $\boldsymbol{I}^l(1) \in [k\theta^l, k\theta^l + \theta^l/T), \forall k = 0, ..., L-1$ or $\boldsymbol{I}^l(1) \in (-\infty, 0) \cup [L\theta^l, +\infty)$.*

Proof. If $\boldsymbol{I}^l(1) < 0$ or $\boldsymbol{I}^l(1) \geq L\theta^l$, it is obvious that we will have $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T) = 0$ or $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T) = L$. Otherwise, based on the conclusion $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = \text{clip}\left(\left\lfloor \frac{\boldsymbol{v}^l(0)+\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} \right\rfloor, 0, LT \right)$ in Theorem 4.2(ii), when $\boldsymbol{I}^l(1) \in [k\theta^l, k\theta^l + \theta^l/T), \forall k = 0, ..., L-1$, we will have $\sum_{t=1}^{T} \boldsymbol{s}^l(t) = kT, \forall k = 0, ..., L-1$. Note that $\forall T' \in [1, T]$, we can further have $\sum_{t=1}^{T'} \boldsymbol{s}^l(t) = kT', \forall k = 0, ..., L-1$. Therefore, it can be concluded that $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T) = k$. Instead, if $\boldsymbol{I}^l(1) \in [0, L\theta^l) \wedge \boldsymbol{I}^l(1) \notin [k\theta^l, k\theta^l + \theta^l/T), \forall k = 0, ..., L-1$, we will have $\sum_{t=1}^{T} \boldsymbol{s}^l(t) \neq kT, \forall k = 0, ..., L-1$. Therefore, $\boldsymbol{s}^l(1) = ... = \boldsymbol{s}^l(T)$ does not hold true.

□

A.1.3 Proof of Theorem 4.4

Theorem 4.4. *When $\sum_{t=1}^{T} \boldsymbol{I}^l(t)/LT = \boldsymbol{W}^l \boldsymbol{r}_{IF}^{l-1}(T_q)$ and $\sum_{t=1}^{T} \boldsymbol{I}^l(t) \in [0, LT\theta^l]$, if $\forall i, j \in [1, T], \omega_{ij}^l = \frac{1}{T}$ and $\lambda^l = 1, \theta^l = \vartheta^l, \boldsymbol{v}^l(0) = \frac{\theta^l}{2}$, for L, T, T_q with arbitrary values, we have: $\mathbb{E}\left(\frac{\sum_{t=1}^{T} \boldsymbol{s}^l(t)\theta^l}{LT} - \frac{\vartheta^l}{T_q} \textit{clip}\left(\left\lfloor \frac{\boldsymbol{W}^l \boldsymbol{r}_{IF}^{l-1}(T_q) T_q}{\vartheta^l} + \frac{1}{2} \right\rfloor, 0, T_q \right) \right) = 0$.*

Proof. If $\forall i, j \in [1, T], \omega_{ij}^l = \frac{1}{T}$, $\lambda^l = 1, \theta^l = \vartheta^l$ and $\boldsymbol{v}^l(0) = \frac{\theta^l}{2}$, combining with the conclusion mentioned in Theorem 4.2(ii), we will have $\frac{\sum_{t=1}^{T} \boldsymbol{s}^l(t)\theta^l}{LT} = \frac{\theta^l}{LT} \text{clip}\left(\left\lfloor \frac{\sum_{t=1}^{T} \boldsymbol{I}^l(t)}{\theta^l} + \frac{1}{2} \right\rfloor, 0, LT \right)$. According to the conclusion pointed out in [2], we have known that $\mathbb{E}\left(\frac{\theta^l}{LT} \text{clip}\left(\left\lfloor \frac{\boldsymbol{x}^l LT}{\theta^l} + \frac{1}{2} \right\rfloor, 0, LT \right) - \frac{\vartheta^l}{T_q} \text{clip}\left(\left\lfloor \frac{\boldsymbol{x}^l T_q}{\vartheta^l} + \frac{1}{2} \right\rfloor, 0, T_q \right) \right) = 0$, here $\boldsymbol{x}^l \in [0, \theta^l]$. Therefore, we directly set $\boldsymbol{x}^l = \sum_{t=1}^{T} \boldsymbol{I}^l(t)/LT = \boldsymbol{W}^l \boldsymbol{r}_{IF}^{l-1}(T_q)$ and then we will draw the final conclusion.

□

A.1.4 Computational Equivalence about the Reparameterization Process

Theorem A.2. *$\forall t, i \in [1, T], \forall j \in [L(t-1)+1, Lt], \forall k \in [L(i-1)+1, Li]$, when $\boldsymbol{s}^{l-1}(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}_{IF}^{l-1}(j)$, if $\hat{b_j^l} = b_t^l/L, \hat{\omega_{jk}^l} = \omega_{ti}^l/L$, we will have $\boldsymbol{I}^l(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{I}_{IF}^l(j)$. Here $\hat{b^l}, \hat{\omega^l}$ denote the rectified bias term and T-GIM layer after the reparameterization process.*

Proof. Firstly, it is obvious that $\boldsymbol{I}^l(t), \boldsymbol{I}_{IF}^l(j)$ can be rewritten as $\boldsymbol{I}^l(t) = \sum_{i=1}^{T} \omega_{ti}^l (\boldsymbol{W}^l \boldsymbol{s}^{l-1}(i) + b_i^l)$ and $\boldsymbol{I}_{IF}^l(j) = \sum_{i=1}^{LT} \hat{\omega_{ji}^l} (\boldsymbol{W}^l \boldsymbol{s}_{IF}^{l-1}(i) + \hat{b_i^l})$. Considering the precondition $\hat{b_j^l} = b_t^l/L, \hat{\omega_{jk}^l} = \omega_{ti}^l/L$,

https://doi.org/10.52202/079017-3233

we will have:

$$\begin{aligned}\boldsymbol{I}_{IF}^{l}(j) &= \sum_{i=1}^{T} \sum_{k=L(i-1)+1}^{Li} \hat{\omega_{jk}^{l}}(\boldsymbol{W}^{l}\boldsymbol{s}_{IF}^{l-1}(k) + \hat{b_{k}^{l}}) \\ &= \sum_{i=1}^{T} \frac{\omega_{ti}^{l}}{L} \sum_{k=L(i-1)+1}^{Li} (\boldsymbol{W}^{l}\boldsymbol{s}_{IF}^{l-1}(k) + \frac{b_{i}^{l}}{L}). \end{aligned} \tag{S5}$$

Then we can further have:

$$\begin{aligned}\sum_{j=L(t-1)+1}^{Lt} \boldsymbol{I}_{IF}^{l}(j) &= \sum_{j=L(t-1)+1}^{Lt} \sum_{i=1}^{T} \frac{\omega_{ti}^{l}}{L} \sum_{k=L(i-1)+1}^{Li} (\boldsymbol{W}^{l}\boldsymbol{s}_{IF}^{l-1}(k) + \frac{b_{i}^{l}}{L}) \\ &= \sum_{i=1}^{T} \frac{\omega_{ti}^{l}}{L} \sum_{j=L(t-1)+1}^{Lt} (\boldsymbol{W}^{l} \sum_{k=L(i-1)+1}^{Li} \boldsymbol{s}_{IF}^{l-1}(k) + b_{i}^{l}) \\ &= \sum_{i=1}^{T} \frac{\omega_{ti}^{l}}{L} \sum_{j=L(t-1)+1}^{Lt} (\boldsymbol{W}^{l}\boldsymbol{s}^{l-1}(i) + b_{i}^{l}) \\ &= \sum_{i=1}^{T} \omega_{ti}^{l}(\boldsymbol{W}^{l}\boldsymbol{s}^{l-1}(i) + b_{i}^{l}) \\ &= \boldsymbol{I}^{l}(t). \end{aligned} \tag{S6}$$

Due to the fact that the calculation process of the spike sequences passing through Conv & BN and T-GIM layers can be abstractly described by Theorem A.2, we can conclude that the sum of the input currents within the corresponding time windows before and after reparameterization remains unchanged. The spike sequences obtained by passing the input currents through the spiking neuron layer will also satisfy the precondition of Theorem A.2 ($\boldsymbol{s}^{l}(t) = \sum_{j=L(t-1)+1}^{Lt} \boldsymbol{s}_{IF}^{l}(j)$). Therefore, we can prove the computational equivalence before and after the reparameterization process. □

A.2 Comparison with Other Advanced Network Backbones

As shown in Tab.S1, we have made comparison with related advanced works [20, 55, 49, 38] on CIFAR datasets. One can find that our LM-HT model has superior scalability and can demonstrate its effectiveness on multiple different backbones. For example, for CIFAR-100 dataset, compared to GAC-SNN [38], we achieve an accuracy improvement of 1.35% on MS-ResNet-18. For Transformer-4-384 architecture, our method also outperforms Spikformer [55] and Spike-driven Transformer [49] in terms of performance.

Table S1: Comparison with previous methods based on advanced backbones and attention mechanism.

Dataset	Method	Architecture	Time-steps	Accuracy(%)
CIFAR-10	MS-ResNet [20]	MS-ResNet-18	6	94.92
	GAC-SNN [38]	MS-ResNet-18	4	96.24
	Spikformer [55]	Transformer-4-384	4	95.51
	Spike-driven Transformer [49]	Transformer-4-384	4	95.6
	Ours	**MS-ResNet-18**	**4**	**96.38**
		Transformer-4-384	**4**	**95.82**
CIFAR-100	MS-ResNet [20]	MS-ResNet-18	6	76.41
	GAC-SNN [38]	MS-ResNet-18	4	79.83
	Spikformer [55]	Transformer-4-384	4	78.21
	Spike-driven Transformer [49]	Transformer-4-384	4	78.4
	Ours	**MS-ResNet-18**	**4**	**81.18**
		Transformer-4-384	**4**	**79.03**

A.3 Experimental Configuration

For static datasets, we attempt to suppress the possible overfitting phenomenon by utilizing data augmentation techniques including AutoAugment [4] and Cutout [9]. For CIFAR10-DVS dataset, we resize each image to 48×48 pixels and split it into 10 frames. For ImageNet-1k dataset, we further consider MS-ResNet architecture [20] and Mixup technique [53] to strengthen the generalization ability of our network. We respectively try to use SGD [1] and AdamW [30] as our optimizers. The corresponding initial learning rate and weight decay are set to $0.025, 5 \times 10^{-4}$ for SGD on CIFAR-10(100), $0.0125, 5 \times 10^{-4}$ for SGD on ImageNet-200 and $0.02, 0.01$ for AdamW on CIFAR10-DVS. For ImageNet-1k dataset, we use SGD as our optimizer and set the corresponding weight decay as 0. Furthermore, in the hybrid training framework, our initial learning rate and weight decay are both set to 5×10^{-4}. The QCFS pretrained models are selected from related open-source checkpoints and self-implementation. For all experimental cases, we choose the Cosine Annealing scheduler [31] to dynamically regulate the learning rate. Our experiments are implemented on NVIDIA RTX A5000 and 4090.

A.4 The Pseudo-Code of Hybrid Training Algorithm

Algorithm 1 Hybrid training framework based on the LM-HT model.

Require: Pretrained QCFS ANN model $f_{\text{ANN}}(\boldsymbol{W}, T_q, \vartheta)$ with L_N layers; Dataset D; Number of time-steps choosed for STBP training T.
Ensure: SNN model $f_{\text{SNN}}(\boldsymbol{W}, \boldsymbol{\Omega}, L, \lambda, \theta)$.

# Convert ANN to SNN
for $l = 1$ to L_N **do**
  $f_{\text{SNN}}.\boldsymbol{W}^l = f_{\text{ANN}}.\boldsymbol{W}^l$
  $f_{\text{SNN}}.\theta^l = f_{\text{ANN}}.\vartheta^l$
  $f_{\text{SNN}}.\boldsymbol{\Omega}^l = \frac{1}{T}$
  $f_{\text{SNN}}.\lambda^l = 1$
  $f_{\text{SNN}}.\boldsymbol{v}^l(0) = f_{\text{SNN}}.\theta^l/2$
end for
# STBP training based on the LM-H model
# Set $f_{\text{SNN}}.\boldsymbol{\Omega}^l$, $f_{\text{SNN}}.\lambda^l$ as learnable parameters and $f_{\text{SNN}}.\theta^l$ as scalars
for (**Image**,**Label**) in D **do**
  for $l = 1$ to L_N **do**
    if Is the first layer **then**
      for $t = 1$ to T **do**
        $\boldsymbol{I}^l(t) = \boldsymbol{I}^l(t) \times L$
      end for
    end if
    LM-HT model performs forward propagation based on Eqs.(5)-(6) and Eq.(7)
    LM-HT model performs back-propagation based on Eqs.(8)-(9)
  end for
end for
return $f_{\text{SNN}}(\boldsymbol{W}, \boldsymbol{\Omega}, L, \lambda, \theta)$

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: We clearly point out the contributions and scope of this work in the abstract and introduction sections.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.

https://doi.org/10.52202/079017-3233

- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [NA] .

   Justification: We find no limitation which needs to be emphasized here.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes] .

   Justification: We provide the corresponding assumptions and proofs in the Appendix section.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.

- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes] .

   Justification: We provide the detailed experimental configuration in the Appendix section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: We provide the data and code with sufficient instructions in the supplemental materials.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

https://doi.org/10.52202/079017-3233

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: We specify the training and test details in the Appendix section and supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No] .

   Justification: The experiments choose a shared random seed to ensure fairness and reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes] .

   Justification: We make description about the computation resources in the Appendix section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes] .

   Justification: The research conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA] .

    Justification: We find no societal impact which needs to be emphasized here.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

https://doi.org/10.52202/079017-3233

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: We think that this paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes] .

    Justification: We make proper statements and citations for relevant existing assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: We choose public datasets and models in this work.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-3233