LT-SNN: SELF-ADAPTIVE SPIKING NEURAL NETWORK FOR EVENT-BASED CLASSIFICATION AND OBJECT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks (SNNs) have received increasing attention due to its high biological plausibility and energy efficiency. The binary spike-based information propagation enables efficient sparse computation with event-based computer vision applications. Prior works investigated direct SNN training algorithm to overcome the non-differentiability of spike generation. However, most of the existing works employ a fixed threshold value for the membrane potential throughout the entire training process, which limits the dynamics of SNNs towards further optimizing the performance. The adaptiveness in the membrane potential threshold and the mismatched mechanism between SNN and biological nervous system remain under-explored in prior works. In this work, we propose LT-SNN, a novel SNN training algorithm with self-adaptive learnable potential threshold to improve SNN performance. LT-SNN optimizes the layer-wise threshold value throughout SNN training, imitating the self-adaptiveness of the biological nervous system. To stabilize the SNN training even further, we propose separate surrogate gradient path (SGP), a simple-yet-effective method that enables the smooth learning process of SNN training. We validate the proposed LT-SNN algorithm on multiple event-based datasets, including both image classification and object detection tasks. Equipped with high adaptiveness that fully captures the dynamics of SNNs, LT-SNN achieves state-of-the-art performance with compact models. The proposed LT-SNN based classification network surpasses SoTA methods where we achieved 2.71% higher accuracy together with 10.48× smaller model size. Additionally, our LT-SNN-YOLOv2 object detection model demonstrates 0.11 mAP improvement compared to the SoTA SNN-based object detection.

1 INTRODUCTION

In the biological nervous system, cortical neurons process information by encoding spatial-temporal inputs into action potentials for spike generation. Inspired by that, spiking neural networks (SNNs) accumulate the membrane potential by extracting information from the input features at each time step, and the resultant binary spikes (0 and 1) provides a sparse and succinct information representation. Such spatial-temporal computation promotes SNN as an attractive AI solution with both biological plausibility and energy efficiency in comparison to the conventional artificial neural networks (ANNs) (He et al., 2016). Furthermore, layer-wise processing with binary spikes elevates the computation efficiency, which benefits the energy-constrained applications such as edge computing.

Under the context of energy-efficient AI applications, the event-based camera or dynamic vision sensors (DVS) have emerged as an attractive and feasible solution for computer vision applications. Compared to the conventional frame-based camera, event cameras independently capture the absolute illumination changes of pixels, resulting in the asynchronous binary stream of events (Gallego et al., 2020). The captured event is characterized by binary pixels and temporal resolutions, leading to highly sparse and energy-efficient visual representations. Such binarized spatial-temporal information naturally fits the computation mechanism of SNNs, bridging the gap between computer vision and neuromorphic computing.

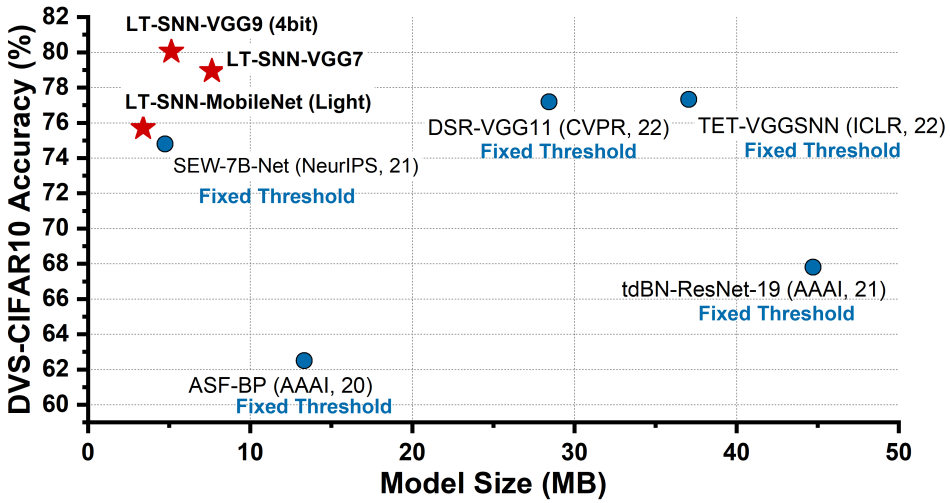

Figure 1: DVS-CIFAR10 classification accuracy of different SNN training methods. The proposed LT-SNN training algorithm achieves the state-of-art accuracy with the compact VGG models.

One of the major bottlenecks of SNN training is the non-differentiability of the spike generation. The infinite gradient of the step function impedes the gradient propagation during backward pass in training. Early research works relied on the ANN-to-SNN conversion (Diehl et al., 2015; Han et al., 2020) to obtain trained SNN models. However, additional training is required and sufficiently high accuracy was not achievable. To that end, various direct training methods have been proposed to obtain a SNN model with one-time training. Empowered by various surrogate gradient (SG) functions (Lee et al., 2016; Wu et al., 2019; Deng et al., 2021), the inaccessible gradient of the spike function is approximated and propagated during learning. However, the inaccurate approximation and heuristic SG selection hurts the training stability with deep models (e.g., ResNet (He et al., 2016)), which further motivated the temporal normalization method (Zheng et al., 2021) and output regularization techniques (Guo et al., 2022; Deng et al., 2021) to smooth the loss.

As the major inspiration of deep learning, the intricate nervous system achieves remarkable performance with a high degree of dynamics. Previous neuroscience works observed the location-dependent potential threshold (Kole & Stuart, 2008) in nervous systems, implying the adaptive firing procedure within the mechanism of spike generation. Accordingly, some recent works on SNN training introduced the learning dynamics into the training process, albeit to a limited degree. (Fang et al., 2021b) optimized the membrane time constant throughout training, with the requirements of large-sized models. DSR (Meng et al., 2022) proposed the threshold-associated spikes with learnable potential threshold. However, the deterministic ratio between firing range and potential threhsold of DSR limits the adaptiveness of SNN learning. (Sun et al., 2022) removes such constraints by directly passing the gradient of potential threshold through the SG function. Nevertheless, the instability of the straight-through surrogate gradient still results in sub-optimal performance compared to state-of-the-art (SoTA) SNN training with fixed threshold (Deng et al., 2021). Although (Deng et al., 2021) achieved the SoTA performance among prior works, the fixed threshold made the membrane potential to often overshoot, limiting the dynamics of SNNs. The limitations of all prior works motivates us to investigate the following question: *How can we optimize the potential threshold of SNNs with high stability and superior performance?*

To answer this question, we propose LT-SNN, a novel self-adaptive **SNN** training algorithm with **L**earnable **T**hreshold. Starting the training from scratch, LT-SNN fully optimizes the potential threshold without introducing any additional scaling or firing constraints. To achieve highly-stable training, we propose a simple-yet-effective technique, namely Separate Gradient Path (SGP). Compared to prior works, the proposed LT-SNN algorithm fully unleashes the advantage of biology-inspired adaptive potential threshold, leading to superior performance compared to all prior SNN algorithms. We validate LT-SNN on multiple event-based computer vision datasets with various model architectures. LT-SNN achieves the new state-of-the-art performance with light-weight or quantized models, as shown in Figure 1.

2 RELATED WORK

ANN-to-SNN conversion Given the non-differentiability of SNN training, early research works converted high-performance non-spiking ANN model into a spiking version (Diehl et al., 2015; Rueckauer et al., 2016). The major drawbacks of the conversion-based method is high computation latency (high number of time-steps) and the additional efforts for the overall training. Several methods have been proposed to improve the latency of the converted model (Han et al., 2020), but still did not reach the high accuracy of direct SNN training methods.

Direct SNN training The expensive training effort and high latency of the conversion-based schemes promotes the direct SNN training to be an attractive solution. BNTT (Neftci et al., 2019) treats the spatial-temporal computation of SNN as a special version of a recurrent neural network (RNN). The gradient of the spiking process is estimated by surrogate gradient (SG) functions in backpropagation. Driven by accuracy and different model architectures, various SG functions have been proposed, including but not limited to rectangle function (Wu et al., 2019), arctangent (Fang et al., 2021b;a), or triangle functions (Deng et al., 2021). However, the difference between the approximated gradient and the exact gradient largely limits the training stability of SNN, especially for the large-sized models. Motivated by that, tdBN (Zheng et al., 2021) introduces the batch-temporal normalization for deep SNN training, and SEW-ResNet (Fang et al., 2021a) directly passes the gradient via designed residual architecture. In addition to the architecture design, various regularization techniques were presented to stabilize SNN training by rectification of the membrane potential distribution (Guo et al., 2022) and backpropagation with spatio-temporal adjustment (Shen et al., 2022).

Biological dynamics in SNN The biology-inspired SNN training methods are also highlighted in recent research works. (Fang et al., 2021b) introduces the learnable time constant for direct SNN training, but the employed large-sized SNN model and extensive training efforts are computationally expensive. Recently proposed DSR (Meng et al., 2022) optimizes the potential threshold during training by multiplying the binary output spike with the threshold value, where the relationship between the firing range and threshold value is constrained by a deterministic ratio. However, optimizing the potential threshold value with such additional constraints limits the learnability of the SNN model. Furthermore, the threshold-dependent binary spikes require layer-wise scaling operation, which deteriorates the hardware compatibility with expansive high-precision multiplication.

3 BASICS OF SPIKING NEURAL NETWORKS

Inspired by the biological nervous system, spiking neural networks process the spatial-temporal information throughout layers. Mathematically, the widely-adopted Leaky Integrate-and-Fire (LIF) model characterizes the membrane potential accumulation with the following ordinary differential equation (ODE):

$$\tau \frac{du(t)}{dt} = -(u(t) - u_{reset}) + I(t), \tag{1}$$

where $u(t)$ represents the membrane potential at time t, V_{reset} is the reset potential after the spike operation, and I_t is the synapse current at time t and τ is the time constant. The ODE in Eq. 1 can be resolved as:

$$u_{t+1} = (1 - \frac{dt}{\tau})u_{t-1} + \frac{dt}{\tau} I, \tag{2}$$

where I is the pre-synaptic "current", characterized by the layer-wise output features of SNN model. Given the iteratively accumulated membrane potential, the binary spikes are generated by the step function:

$$S_t = \theta(u_t - V_{th}) = \begin{cases} 1 & \text{if } u_t \geq V_{th} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Where θ represents the Heaviside step function, and V_{th} represents the membrane potential threshold for spiking neurons. In the forward pass of SNN, the binary post-synaptic spikes are generated when the membrane potential exceeds the potential threshold V_{th}.

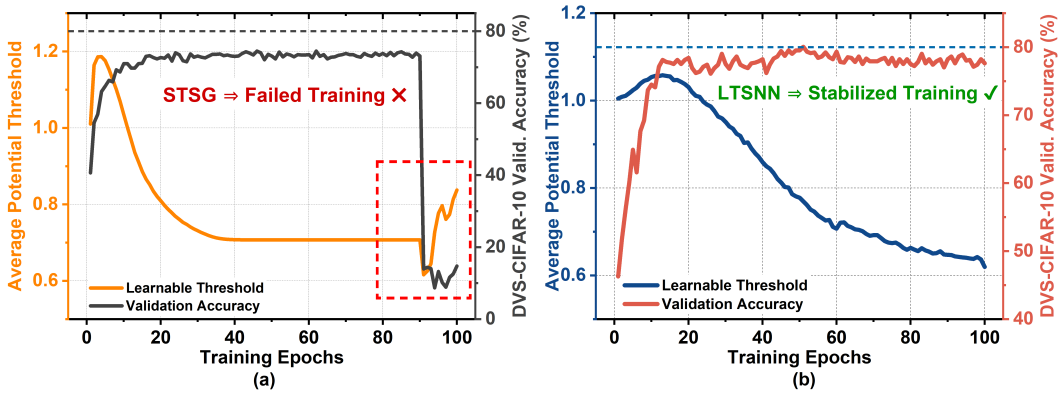

Figure 2: Validation accuracy and potential threshold of direct training SNN on DVS-CIFAR10 dataset with (a) STSG and (b) the proposed separate gradient path for LT-SNN.

As demonstrated by spatial-temporal backpropagation (STBP) (Wu et al., 2019), the backward propagation of SNN is characterized as:

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial S_t}\frac{\partial S_t}{\partial u_t}\frac{\partial u_t}{\partial I_t}\frac{\partial I_t}{\partial W} \tag{4}$$

As mentioned earlier, the non-differentiable spike function θ impedes the gradient computation of $\partial S_t/\partial u_t$. The directly-trained SNN incorporates the surrogate gradient function to approximate the intangible Dirac function. In this work, we choose triangle function for gradient approximation:

$$\frac{\partial S_t}{\partial u_t} = \theta'(u_t - V_{th}) = \max(0, 1 - |u_t - V_{th}|) \tag{5}$$

4 Challenges of Learning the Threshold of Membrane Potential

As described in Eq. 5, the non-differentiability between the output spike S and the membrane potential u can be alleviated by the surrogate gradient approximation. Mathematically, the gradient of the potential threshold can be approximated with a similar gradient surrogation:

$$|\frac{\partial L}{\partial V_{th}}| = \frac{\partial L}{\partial S_t}\frac{\partial S_t}{\partial V_{th}} = \frac{\partial L}{\partial S_t}\theta'(u_t - V_{th}) \tag{6}$$

The straight-through approximation in Eq. 6 is built upon the following hypothesis:

Hypothesis 1: *The surrogate gradient of $\frac{\partial S_t}{\partial u_t}$ and $\frac{\partial S_t}{\partial V_{th}}$ is transferrable, and the identical gradient surrogation is suitable for separate loss landscape with respect to u_t and V_{th}.*

In this work, we falsify **Hypothesis 1** by empirically demonstrating the learning instability caused by the straight-through surrogate gradient (STSG). Figure 2(a) depicts the validation accuracy and the layer-wise averaged potential threshold of the VGG-9 model which is trained from scratch with STSG. The drastic change in potential threshold and exploding validation accuracy represents the incompatibility of the STSG learning scheme. In summary, we have the following observation:

Observation 1: *The gradient approximation with respect to membrane potential and potential threshold is not interchangeable.*

Different from the STSG scheme, the recent DSR (Meng et al., 2022) scheme computes the gradient of potential threshold with the following threshold-associate firing procedure:

$$S_t = V_{th} \times \theta(u_t - \alpha V_{th}) \tag{7}$$

Where the fixed parameter $\alpha \in [0, 1]$ controls the threshold with respect to the firing range $[0, V_{th}]$. In other words, regardless of the threshold value, the membrane potential has to be a deterministic portion of the total firing range. However, such constraints in the training process largely limits the learnability of SNN. Based on the open-sourced implementation of DSR (Meng et al., 2022),

Table 1: The performance of DSR (Meng et al., 2022) is largely impacted by the value of α. The proposed LT-SNN outperforms DSR with 2.45% higher accuracy on DVS-CIFAR10 dataset.

Method	Model	Accuracy (%)	α	True Binary Spike
DSR (Meng et al., 2022)	VGG-11	77.19	0.3	✗
DSR (Meng et al., 2022)	VGG-11	75.29	1.0	✗
This work	VGG-9	**80.04**	-	✓

we unleash the optimization constraints of the potential threshold by setting $\alpha = 1.0$. Please note that, keeping the threshold and spike as identical values is commonly adopted in prior works (Deng et al., 2021). As shown in Table 1, the freely-learned potential threshold ($\alpha = 1.0$) exhibits large accuracy degradation compared to the constrained leanring ($\alpha = 0.3$), which implies the sub-optimal performance of the DSR (Meng et al., 2022) when the potential threshold is unconstrained.

On the other hand, associating the threshold value with the neuron output disables the true binary spikes (0 and 1) of SNN. Specifically, the output generated by Eq. 7 becomes $(0, V_{th})$ instead of a pure binary value $(0, 1)$. In the context of hardware computation, layer-wise varied spikes are equivalent to the layer-wise scaling on top of the true binary output, which often requires high-precision multipliers (Jacob et al., 2018) to maintain the accuracy. Unlike the conventional ANN with one-time scaling operation, the temporal computation of SNN requires step-wise high-precision multiplication, leading to the magnified energy cost and considerable hardware resource consumption. The instability of STSG and the imperfections of DSR (Meng et al., 2022) motivate us to investigate the following question:

Question 1: *How can we maximize the performance of SNN with the freely-optimized potential threshold, while maintaining the training stability and hardware compatibility?*

5 Proposed Method

To answer the question above, we propose *LT-SNN*, a novel SNN training algorithm that successfully resolves the contradiction between threshold optimization, training stability, and hardware compatibility. LT-SNN optimizes the layer-wise potential threshold during training, maximizing the biological plausibility of SNN without introducing any learning constraints. In the meantime, LT-SNN embraces the advantages of the adaptive potential threshold while maintaining the true binary spikes, bridging the research gap between biological inspiration and practical AI applications.

5.1 Separate Gradient Path

As summarized in **Observation 1**, the incompatibility of STSG implies the distinct loss landscape with respect to the gradient descent of the membrane potential and potential threshold. Motivated by that, we propose Separate Gradient Path (SGP), which treats the gradient computation of u_t and V_{th} with dedicated gradient approximations.

Specifically, SGP trains SNNs by introducing the Gradient Penalty Window (GPW), a simple-yet-effective method that optimizes the potential threshold without losing training stability. On top of the gradient approximation in Eq. 5, GPW is characterized as a non-linear function $\sigma(\cdot)$, which reshapes the surrogate gradient of the layer-wise potential threshold V_{th}. Mathematically, the GPW-aided separate gradient path is characterized as:

$$\frac{\partial S_t}{\partial u_t} = \theta'(u_t - V_{th}) = \max(0, 1 - |u_t - V_{th}|) \tag{8}$$

$$|\frac{\partial S_t}{\partial V_{th}}| = \theta'(u_t - V_{th})\sigma(u_t - V_{th}) = \max(0, 1 - |u_t - V_{th}|)\sigma(u_t - V_{th}) \tag{9}$$

In this work, we choose the Sigmoid function as the gradient penalty window for potential threshold:

$$\sigma(u_t - V_{th}) = \frac{1}{1 + e^{-(u_t - V_{th})}} \tag{10}$$

Table 2: Training results of VGG-9 model on DVS-CIFAR10 dataset with different SG schemes.

Method	Learnable Thre.	SG Func.	Top-1 Accuracy (%)
TET (Deng et al., 2021)	✗	Triangle	77.33
Penalty for all	✓	GPW for all	76.40
This work	✓	**SGP**	**80.04**

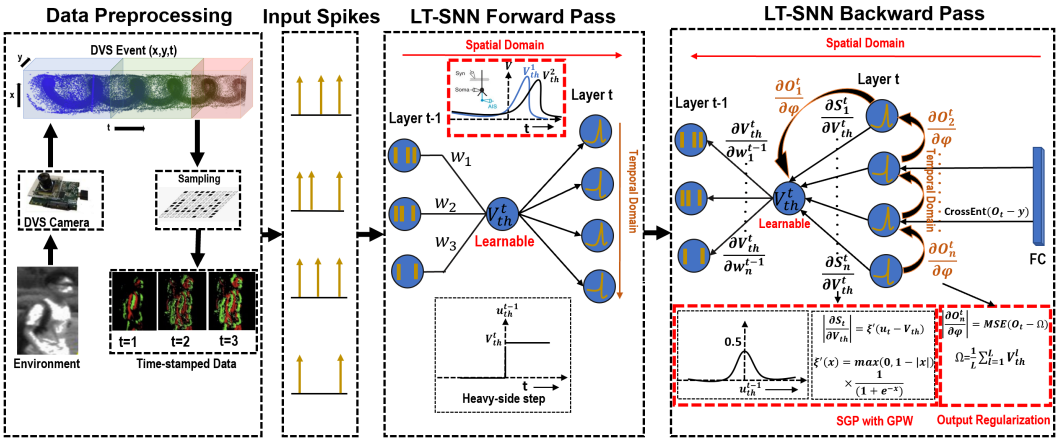

Figure 3: Implementation flow of LT-SNN.

For gradient computation of V_{th}, we accumulate the gradient computed in Eq. 9 to avoid the dimensionality mismatch:

$$\frac{\partial L}{\partial V_{th}} = \sum \left(\mathbb{1}\{u_t \geq V_{th}\} \times |\frac{\partial L}{\partial V_{th}}| \right) \tag{11}$$

Since the unfired neurons have no contribution to the final loss, the indicator function $\mathbb{1}\{u_t \geq V_{th}\}$ only keeps the gradient with respect to the active neurons in the forward pass.

We empirically prove the superiority of the proposed SGP algorithm based on different surrogate gradient scenarios. Table 2 summarizes the performance of the directly trained VGG9-SNN on the DVS-CIFAR10 dataset with different gradient computation paths. As shown in Table 2, penalizing all the surrogate gradient with GPW leads to sub-optimal performance. On the contrary, the proposed SGP algorithm achieves the best performance with distinct gradient computations. SGP exploits the optimal layer-wise potential threshold with the stabilized training process and fast convergence. Empowered by the proposed SGP scheme, LT-SNN achieves >80% validation accuracy within 50 training epochs, as depicted in Figure 2(b). Compared to the SoTA SNN training with fixed threshold (Deng et al., 2021), SGP embraces the advantage of the adaptive potential threshold learning and achieves superior accuracy. Furthermore, SGP preserves the true binary spikes (0 and 1) in the resultant model, maximizing the hardware compatibility without introducing any high-precision scaling for spike generation (Meng et al., 2022). The complete flow of the proposed SGP with GPW and output regularization is illustrated in Figure 3.

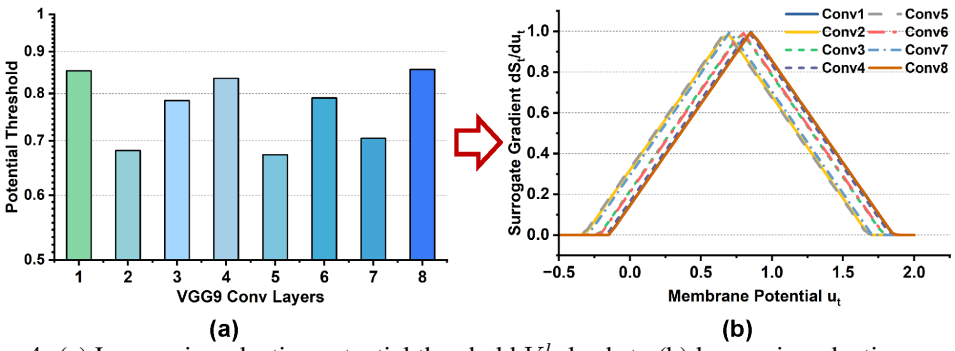

Figure 4: (a) Layer-wise adaptive potential threshold V_{th}^l leads to (b) layer-wise adaptive surrogate gradient gradient of the membrane potential u_t.

5.2 SELF-ADAPTIVE SNN LEARNING

The proposed SGP algorithm improves the performance of SNN with the unconstrained optimization of the potential threshold. However, the rationale behind the adaptiveness-induced optimal performance remains unclear, which motivates us to investigate the following question:

Question 2: *Why the adaptive potential threshold optimizes the performance of SNN model?*

As shown in Eq. 5, the gradient with respect to the membrane potential u_t is approximated based on the surrogate function θ', which is characterized by the non-linearity of the function and the shifting induced by the threshold V_{th}. With the proposed LT-SNN scheme, the adaptive potential threshold leads to the distinct surrogate gradient for membrane potential u_t of each layer, as shown in Figure 4. Compared to SNN training with fixed potential threshold, LT-SNN adjusts the layer-wise gradient information with the adaptive potential threshold.

5.3 OUTPUT REGULARIZATION

We also adopt and customize the temporal efficient training (TET) loss function for output regularization (Deng et al., 2021) to further improve the performance. The output of the LT-SNN will be regularized towards the averaged layer-wise potential threshold Ω:

$$L = \underbrace{\beta \frac{1}{T} \sum_{t=1}^{T} L_{CE}(O(t), y)}_{\text{TET Loss}} + \underbrace{(1-\beta) \frac{1}{T} \sum_{t=1}^{T} \text{MSE}(O(t), \Omega)}_{\text{MSE regularizer}} \tag{12}$$

$$\text{Where} \qquad \Omega = \frac{1}{L} \sum_{l=1}^{L} V_{th}^{l} \tag{13}$$

β and Ω are the weights of the moving average and target value of regularization, respectively. We choose Ω as the mean learnable threshold of the forward pass to regularize the output based on the layer-wise learnable threshold, where L represents the total number of layers in our SNN architecture.

6 EXPERIMENTAL RESULTS

In this section, we validate the proposed LT-SNN algorithm with multiple event-based computer vision datasets, including DVS-CIFAR10 (Li et al., 2017), N-Cars (Sironi et al., 2018), N-Caltech101 (Orchard et al., 2015) and Prophesee Automotive Gen1 (de Tournemire et al., 2020). Unlike prior works that only employ large-sized VGG or ResNet models (>11M parameters), the proposed algorithm is validated on compact VGG models (2.4-7.1M parameters) and light-weight MobileNet-V1 model (1.5M parameters) (Howard et al., 2017). Table 3 summarizes the architectures of the selected models.

Table 3: Convolutional SNN model architectures for LT-SNN training. "C3", "DW", "MP2" , "AP2" and "FC" represent 3×3 convolution layer, depth-wise separable block (Howard et al., 2017), 2×2 max-pooling, 2×2 average-pooling, and fully connected layer, respectively.

Model	Architecture
MobileNet-Light	32C3-64DW-64DW-AP2-128DW-128DW-AP2-256DW-AP2-FC256-FC10
VGG-7	32C3-32C3-AP2-64C3-64C3-AP2-128C3-128C3-AP2-256C3-256C3-AP2-FC256-FC10
VGG-9	32C3-64C3-64C3-AP2-128C3-128C3-AP2-256C3-256C3-AP2-512C3-512C3-AP2-FC512-FC10
Yolo-V2	32C3-MP2-64C3-MP2-128C3-64C1-128C3-MP2-256C3-128C1-256C3-MP2-512C3-256C1-512C3-256C1-MP2-1024C3-512C1-1024C3-AP2-FC512-FC576

SNN Training for Full-/Low-Precision Inference We perform LT-SNN training to execute inference with both full-precision and 4-bit precision. To train SNNs for low-precision inference, we adopt the Statistic-Aware Weight Binning (SAWB) quantizer (Choi et al., 2019) to compress the layer-wise weights of the LT-SNN-VGG-9 network down to 4-bit precision. We separately perform

Table 4: Experimental results of the proposed LT-SNN on DVS-CIFAR10 and CIFAR-10 datasets. Except "VGG-9 (4-bit)", 32-bit precision is used for all results of prior works and this work.

Dataset	Method	Architecture	Parameters	Simulation Length	Accuracy(%)
DVS-CIFAR-10	ASF-BP (Wu et al., 2021)	VGG-Like	15.2M	50	62.50
	tdBN (Zheng et al., 2021)	ResNet-19	11.18M	10	67.80
	ParamLIF (Fang et al., 2021b)	VGG-Like	17.4M	20	74.80
	RecDis (Guo et al., 2022)	ResNet-19	11.18M	10	72.42
	TET (Deng et al., 2021)	VGG-Like	9.27M	10	77.33
	DSR (Meng et al., 2022)	VGG-11	9.34M	30	75.70
	This work	MobileNet-V1 (light)	**1.28M**	**30**	**75.70**
	This work	VGG-7	**1.91M**	**30**	**78.93**
	This work	VGG-9 (4-bit)	**7.07M**	**30**	**80.04**
	This work	VGG-9	**7.07M**	**10**	**79.10**
	This work	VGG-9	**7.07M**	**8**	**78.30**
CIFAR-10	Hybrid (Rathi & Roy, 2020)	ResNet-20	12.3M	10	92.54
	tdBN (Zheng et al., 2021)	ResNet-19	11.18M	6	93.16
	DSR (Meng et al., 2022)[1]	ResNet-18	11.18M	6	91.89
	This work	ResNet-19	11.18M	**6**	**93.97**

[1] The experiment is rigorously performed based on the open-sourced DSR implementation with 6 time steps.

Table 5: Experimental results of the proposed LT-SNN on N-Caltech101 and N-Cars datasets.

Dataset	Method	Representation	Accuracy(%)	Pretrain
N-Caltech101	YOLE (Cannici et al., 2019)	Histogram	70.02	False
	EST (Gehrig et al., 2019)	Histogram	81.70	True
	AsyNet (Messikommer et al., 2020)	Histogram	76.10	False
	AEGNN (Schaefer et al., 2022)	Graph	66.80	False
	This work	**Binary Spike**	**77.71**	**False**
N-Cars	NDA (Li et al., 2022)	Binary Spike	91.9	False
	AsyNet (Messikommer et al., 2020)	Histogram	94.4	False
	YOLE (Cannici et al., 2019)	Histogram	92.7	False
	AEGNN (Schaefer et al., 2022)	Graph	94.5	False
	Object (Cordone et al., 2022)	Binary Spikes	92.4	False
	This work	**Binary Spike**	**95.02**	**False**

and compare the full-precision and quantized-aware training of our SNN architectures using hyper-parameters mentioned in Appendix A. Using our LT-SNN scheme, we achieve SoTA results for both full-precision and low-precision SNNs, as reported in Table 4 and Table 5.

Image Classification Task We validate the proposed LT-SNN algorithm on both event-based image classification datasets and conventional computer vision dataset with RGB images. The LT-SNN models are directly trained from scratch, and the detailed input preprocessing and experimental setup are summarized in Appendix A. Table 4 summarizes the performance of LT-SNN on DVS-CIFAR10 and CIFAR-10 (with RGB images) datasets. For DVS-CIFAR10 dataset, compared to the current SoTA method (Deng et al., 2021), the 4-bit VGG-9 model trained by proposed LT-SNN algorithm achieves 2.71% accuracy improvement with 1.31× less parameters and 10.48× model size reduction (MB). Furthermore, the proposed LT-SNN demonstrates consistently superior performance with different simulation length (from 8 time steps to 30 time steps). For the conventional RGB CIFAR-10 dataset, LT-SNN also achieves new SoTA performance with ResNet-19 architecture, where we adopted the tdBN (Zheng et al., 2021) scheme to stabilize the direct SNN training process. Based on the official implementation of DSR (Meng et al., 2022), we rigorously conduct the experiment of DSR on CIFAR-10 dataset with 6 simulated time steps.

In addition to the CIFAR datasets (Li et al., 2017), we evaluate the proposed LT-SNN algorithm on N-CalTech101 (Orchard et al., 2015) and N-Cars (Sironi et al., 2018) datasets. As shown in Table 5, the proposed LT-SNN achieves new SoTA performance on both datasets with 1.61% (N-Caltech101, without pre-training) and 0.52% (N-Cars) accuracy improvements. Compared to the conventional histogram-based computation (Messikommer et al., 2020; Cannici et al., 2019) that use

Table 6: Experimental results of the proposed LT-SNN on Prophesee Automotive Gen1 dataset.

Method	Model Architecture	Spiking Model	mAP
Asynet (Messikommer et al., 2020)	FB-Sparse	No	0.145
MatrixLSTM (Cannici et al., 2020)	ResNet-19	No	0.3
RED (Perot et al., 2020)	RetinaNet	No	0.41
VGG-11+SSD (Cordone et al., 2022)	**VGG+SSD-SNN**	Yes	0.187
This work	**Custom-YoloV2-SNN**	**Yes**	**0.298**

non-binary activations, the proposed LT-SNN enables end-to-end binary computation with binary spikes throughout the entire SNN, achieving superior performance with high hardware compatibility.

Impact of the Event Time Steps SNNs require iterative computation and membrane potential accumulation, which motivated prior works to exploit the computation reduction with less number of time steps to represent the incoming event. We also validated LT-SNN with different simulated time steps on the DVS-CIFAR10 dataset. As shown in Table 4, the proposed LT-SNN manages to achieve SoTA performance with reduced time steps and compact models, compared to the prior works. With the same 10 time steps as TET Deng et al. (2021), LT-SNN achieves 1.77% accuracy improvement, without using any data augmentation. With the extended 30 time steps, LT-SNN achieves 80.04% SoTA accuracy on DVS-CIFAR10 dataset.

Object Detection Task In addition to the classification datasets, we validate the proposed LT-SNN on large-sized Prophesee Automotive Gen1 dataset (de Tournemire et al., 2020) for object detection task. With 228,123 bounding boxes for cars and 27,658 for pedestrians, the Gen1 dataset (de Tournemire et al., 2020) is considered as the most complex event-based computer vision task. Unlike the prior works use accumulated histogram (Messikommer et al., 2020; Cannici et al., 2020; Cordone et al., 2022) or graph-based input representation (Schaefer et al., 2022), we translate DVS event to pure binary frames and synchronize them with artificial actual ground truths from (Perot et al., 2020) for network training. The detailed data preprocessing setup is summarized in Appendix A. The binary input events and intermediate spikes enables the end-to-end binarized computing for LT-SNN, elevates the computation efficiency with simplified data format. We develop LT-SNN-YoloV2 feature extraction network followed by Yolo loss to localize the objects and compute mean average precision (mAP) for detected bounding boxes against the ground truths as shown in Figure 5. Table 6 elucidates that, in comparison to the current SNN-based SoTA, our LT-SNN-Yolov2 records SoTA mAP of 0.298 on Prophesee Automotive Gen1 dataset.

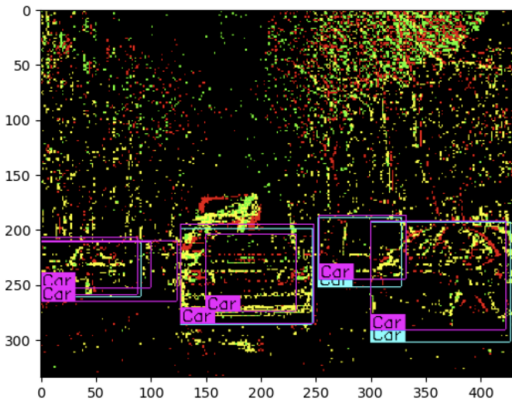

Figure 5: Inference results of LT-SNN on Prophesee Automotive Gen1 dataset.

7 Conclusion

In this work, we propose a novel SNN training algorithm with learnable threshold (LT-SNN), which optimizes the layer-wise threshold with direct SNN training. Our proposed work incorporates non-synaptic learning of biological nervous system to improve data propagation through SNNs. As one of the first studies on this topic, the proposed method unleashes the firing constraints that were imposed in prior works, without introducing any additional high-precision scaling factor to the spike generation. The proposed method with full and low-precision training has been verified on multiple event-based image classification and object detection datasets. LT-SNN improves the state-of-the-art accuracy for DVS-CIFAR10 dataset by 2.8% together with 10.48× smaller model size. For object detection on Prophesee Automotive Gen1 dataset, the LT-SNN outperforms SNN-based SoTA mAP by 0.11 with end-to-end binary computation.

REFERENCES

Marco Cannici, Marco Ciccone, Andrea Romanoni, and Matteo Matteucci. Asynchronous Convolutional Networks for Object Detection in Neuromorphic Cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 0–0, 2019.

Marco Cannici, Marco Ciccone, Andrea Romanoni, and Matteo Matteucci. A Differentiable Recurrent Surface for Asynchronous Event-based Data. In *European Conference on Computer Vision (ECCV)*, pp. 136–152, 2020.

Jungwook Choi, Swagath Venkataramani, Vijayalakshmi Viji Srinivasan, Kailash Gopalakrishnan, Zhuo Wang, and Pierce Chuang. Accurate and efficient 2-bit quantized neural networks. *Conference on Machine Learning and Systems (SysML)*, 1:348–359, 2019.

Loïc Cordone, Benoît Miramond, and Philippe Thierion. Object Detection with Spiking Neural Networks on Automotive Event Data. *arXiv preprint arXiv:2205.04339*, 2022.

Pierre de Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A Large Scale Event-based Detection Dataset for Automotive. *arXiv preprint arXiv:2001.08499*, 2020.

Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal Efficient Training of Spiking Neural Network via Gradient Re-weighting. In *International Conference on Learning Representations (ICLR)*, 2021.

Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, High-accuracy Spiking Deep Networks Through Weight and Threshold Balancing. In *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2015.

Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep Residual Learning in Spiking Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:21056–21069, 2021a.

Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating Learnable Membrane Time Constant to Enhance Learning of Spiking Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2661–2671, 2021b.

Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2020.

Daniel Gehrig, Antonio Loquercio, Konstantinos G Derpanis, and Davide Scaramuzza. End-to-end Learning of Representations for Asynchronous Event-based Data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5633–5643, 2019.

Yufei Guo, Xinyi Tong, Yuanpei Chen, Liwen Zhang, Xiaode Liu, Zhe Ma, and Xuhui Huang. RecDis-SNN: Rectifying Membrane Potential Distribution for Directly Training Spiking Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 326–335, 2022.

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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13558–13567, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*, 2017.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-arithmetic-only Inference. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2704–2713, 2018.

Maarten H. P. Kole and Greg J. Stuart. Is Action Potential Threshold Lowest in the Axon? *Nature Neuroscience*, 11(11):1253–1255, 2008.

Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training Deep Spiking Neural Networks Using Backpropagation. *Frontiers in Neuroscience*, 10:508, 2016.

Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-DVS: An Event-stream Dataset for Object Classification. *Frontiers in Neuroscience*, 11:309, 2017.

Yuhang Li, Youngeun Kim, Hyoungseob Park, Tamar Geller, and Priyadarshini Panda. Neuromorphic Data Augmentation for Training Spiking Neural Networks. *arXiv preprint arXiv:2203.06145*, 2022.

Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Training High-Performance Low-Latency Spiking Neural Networks by Differentiation on Spike Representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12444–12453, 2022.

Nico Messikommer, Daniel Gehrig, Antonio Loquercio, and Davide Scaramuzza. Event-based Asynchronous Sparse Convolutional Networks. In *European Conference on Computer Vision (ECCV)*, pp. 415–431, 2020.

Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. 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.

Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Frontiers in Neuroscience*, 9:437, 2015.

Etienne Perot, Pierre de Tournemire, Davide Nitti, Jonathan Masci, and Amos Sironi. Learning to Detect Objects with a 1 Megapixel Event Camera. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:16639–16652, 2020.

Nitin Rathi and Kaushik Roy. DIET-SNN: Direct Input Encoding with Leakage and Threshold Optimization in Deep Spiking Neural Networks. *arXiv preprint arXiv:2008.03658*, 2020.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, and Michael Pfeiffer. Theory and Tools for the Conversion of Analog to Spiking Convolutional Neural Networks. *arXiv preprint arXiv:1612.04052*, 2016.

Simon Schaefer, Daniel Gehrig, and Davide Scaramuzza. AEGNN: Asynchronous event-based graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12371–12381, 2022.

Guobin Shen, Dongcheng Zhao, and Yi Zeng. Backpropagation with Biologically Plausible Spatiotemporal Adjustment for Training Deep Spiking Neural Networks. *Patterns*, pp. 100522, 2022.

Amos Sironi, Manuele Brambilla, Nicolas Bourdis, Xavier Lagorce, and Ryad Benosman. HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1731–1740, 2018.

Hongze Sun, Wuque Cai, Baoxin Yang, Yan Cui, Yang Xia, Dezhong Yao, and Daqing Guo. A Synapse-Threshold Synergistic Learning Approach for Spiking Neural Networks. *arXiv preprint arXiv:2206.06129*, 2022.

Hao Wu, Yueyi Zhang, Wenming Weng, Yongting Zhang, Zhiwei Xiong, Zheng-Jun Zha, Xiaoyan Sun, and Feng Wu. Training Spiking Neural Networks with Acumulated spiking flow. In *Proceedings of the AAAI conference on Artificial Intelligence (AAAI)*, volume 35, pp. 10320–10328, 2021.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct Training for Spiking Neural Networks: Faster, Larger, Better. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, pp. 1311–1318, 2019.

Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going Deeper with Directly-trained Larger Spiking Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pp. 11062–11070, 2021.

A Detailed Experimental Setup

Data Preprocessing To ensure hardware efficient LT-SNN based training, we convert DVS-events to binary frames in the event pre-processing stage. For classification data, we perform sampling over different time steps and transform events to binary tensors of shape [Batch, Time, Channels, Height, Width]. Similarly, we follow the same convention to convert Prophesee Gen1 events to binary frames. Then the generated binary frames are synchronized with artificial ground truth from (Perot et al., 2020). Finally, the events and annotations are translated to the tensors of shape [Batch, Time, Channels, Height, Width] and [Batch, Number of boxes, Bounding box] respectively.

Training Parameters We train our proposed LT-SNN based classification and object detection architectures using PyTorch. Regarding hyperparameter selection, we use the Adam optimizer where the learning rate is set to 0.001 and cross entropy loss plus mean square error is computed between detected and target classes for back-propagation and output regularization. For output regularization, the moving average weight β is set to 0.45 for both the full-precision and low-precision training. Finally, each LT-SNN based architecture from Table 3 was trained for 200 epochs and we observe robust convergence of our proposed LT-SNN based architectures in less than 50 epochs.

B ALGORITHM

Algorithm 1 LT-SNN Algorithm

Initialize: Input spike as S_t^0 ; Membrane potential as u_t^n ; Learnable potential threshold as V_{th}^0; Output spike as O_t^0 and regularization ratio as β
for $t \leftarrow 1$ *to* T **do**
  Forward Pass:
  Compute u_t^1, O_t^1 using Eq. 2
  for $n \leftarrow 1$ *to* N **do**
    Forward Pass:
    Compute u_t^n, O_t^n using Eq. 3
  end
end
$p =$ Prediction(Y, O_t^n)
Loss:
Compute Ω using Eq. 13 with detached V_{th}.
$L_{LT_SNN} = L_{CE}(Y, O_t^n) + (1-\beta)MSE(O_t^n, \Omega)$
Backward Pass:
Initialize:
$\partial L_{LT_SNN}/\partial O_t^n \leftarrow 0$
for $t \leftarrow T$ *to* 1 **do**
  $\partial L_{LT_SNN}/\partial O_t^n =$ Spatial_Gradient$(\partial L_{CE}/\partial O_t^n)+$ Output_Regularization$(\partial MSE/\partial O_t^n)$
  for $n \leftarrow N$ *to* 1 **do**
    Compute gradient using Eq. 6
    Compute $\partial L_{CE}/\partial V_{th}^n$ using Eq. 9 and Eq. 10
    Mask $\partial L_{CE}/\partial V_{th}^n$ with active pixel density using Eq. 11
  end
end