Rethinking the Membrane Dynamics and Optimization Objectives of Spiking Neural Networks

Hangchi Shen[1,2], **Qian Zheng**[3,4], **Huamin Wang**[1,2,*], **Gang Pan**[3,4]
[1]College of Artificial Intelligence, Southwest University
[2]Chongqing Key Laboratory of Brain Inspired Computing and Intelligent Chips
[3]The State Key Lab of Brain-Machine Intelligence, Zhejiang University
[4]College of Computer Science and Technology, Zhejiang University
`stephen1998@email.swu.edu.cn, hmwang@swu.edu.cn,`
`{qianzheng, gpan}@zju.edu.cn`

Abstract

Despite spiking neural networks (SNNs) have demonstrated notable energy efficiency across various fields, the limited firing patterns of spiking neurons within fixed time steps restrict the expression of information, which impedes further improvement of SNN performance. In addition, current implementations of SNNs typically consider the firing rate or average membrane potential of the last layer as the output, lacking exploration of other possibilities. In this paper, we identify that the limited spike patterns of spiking neurons stem from the initial membrane potential (IMP), which is set to 0. By adjusting the IMP, the spiking neurons can generate additional firing patterns and pattern mappings. Furthermore, we find that in static tasks, the accuracy of SNNs at each time step increases as the membrane potential evolves from zero. This observation inspires us to propose a learnable IMP, which can accelerate the evolution of membrane potential and enables higher performance within a limited number of time steps. Additionally, we introduce the last time step (LTS) approach to accelerate convergence in static tasks, and we propose a label smooth temporal efficient training (TET) loss to mitigate the conflicts between optimization objective and regularization term in the vanilla TET. Our methods improve the accuracy by 4.05% on ImageNet compared to baseline and achieve state-of-the-art performance of 87.80% on CIFAR10-DVS and 87.86% on N-Caltech101. The code is available at `https://github.com/StephenTaylor1998/IMP-SNN`.

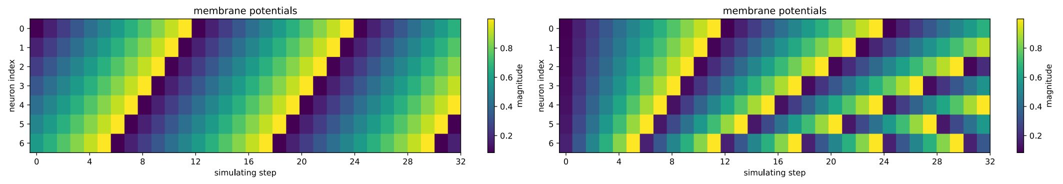
1 Introduction

In recent years, deep learning technology based on artificial neural networks (ANNs) has made significant breakthroughs in many fields [1, 2, 3, 4, 5]. However, as we all known, its high training and inference costs have become a major obstacle to restrict its further widespread applications. To overcome these challenges, a new generation of neural network architectures named Spiking Neural Networks (SNNs) [6] have been developed, which may be a feasible path to refer to the efficient dynamics mechanism of biological nervous systems [7]. SNNs leverage the dynamic mechanisms of membrane potential integration and fire from biological neural networks, which can process time-varying input data by using a single model [8] as biological neurons, maintaining linearly increasing computational complexity. Therefore, the advantages of energy efficiency and biologically plausible dynamical mechanisms [7, 9, 10] make them a bridge between the fields of brain science and artificial intelligence, which is widely regarded as the next generation of ANNs [6]. In addition, it is worth mentioning that SNNs can cleverly avoid multiplication operations on neuromorphic chips

[*]Corresponding author.

[11, 12, 13] by spike-based computing, achieving synaptic computation and membrane potential accumulation solely through addition operations, which can significantly enhance their computational efficiency.

It should be noted that training SNNs to achieve the comparable performance with the same ANN architecture remains a formidable challenge at present. The conversion method of ANNs to SNNs (ANN2SNN) [14, 15, 16, 17, 18] has proved to be an effective approach for obtaining high-performance SNNs. However, these methods mainly incorporate knowledge from the ANN's learning of static inputs into the converted SNNs, which motivates us to concern about the biological plausibility of these SNNs as they do not require the acquisition of spatiotemporal information. Another alternative is to leverage surrogate gradient [19, 20, 21] and backpropagation [22] through time to train high-performing SNNs. This method only requires four time steps to achieve an accuracy surpassing that of conversion methods requiring hundreds of time steps, making it the most promising training approach for currently available energy-efficient SNNs. Therefore, due to the development of direct training methods, SNNs have been extended to various tasks, including image classification[23, 24, 25], image reconstruction [26, 27, 28], object detection [29], natural language processing [30, 31], etc, and have demonstrated significant energy efficiency advantages in these fields. It is a pity that these training methods and architecture designs typically consider the firing rate or average membrane potential of the last layer as the output, instead of fully exploring the impacts of the membrane potential on the model performance and the training process [32, 33].



(a) Adjusting IMP.

(b) Adjusting input intensity.

Figure 1: Membrane potentials and spikes (yellow) generated by adjusting IMP and input intensity.

In this paper, we find that the initial membrane potential (IMP) has different effects on the membrane potential integration and spike firing compared to input intensity(Figure 1). Then, we discover that the novel spike patterns can be generated by adjusting the IMP of spiking neurons and additional mappings between firing patterns can be established, which motivates us to propose a learnable IMP. Furthermore, we find that in static tasks, the variation of the output accuracy at each time step for SNNs is directly related to the membrane potential, as it is the only time-varying term. Then we can explain why the convergence speed of temporal efficient training (TET) [34] in static tasks is significantly lower than standard direct training (SDT) [20]. To address this problem, we propose a method called last time steps (LTS) to achieve faster convergence in static tasks. And then, we propose a label-smoothed TET loss for neuromorphic tasks, which can outperform vanilla TET on neuromorphic datasets [35, 36]. It is worth noting that the proposed novel IMP method can obtain significant performance without any additional adjustments for the network structure and training method. By simply setting the IMP of the original LIF neurons to a learnable version, we achieve SOTA accuracies of up to 87.8% and 86.32% on CIFAR10DVS [35] and NCaltech101 [36], respectively. Furthermore, our LTS method improves the accuracy of SEW-ResNet50 [37] on the ImageNet1k [38] dataset to 71.83%, surpassing the vanilla SEW-ResNet152, 69.26%. The main contributions are as follows:

1) We discover that SNNs can generate new spike patterns by adjusting the IMP values, and prove that on the static tasks, the variation of SNN accuracy at each time step is only caused by the change of membrane potential. In addition, we innovatively introduce the learnable IMP in SNNs to accelerate the evolution of membrane potential.

2) To alleviate the slow convergence of TET on the static tasks, we propose the LTS method, which can accelerate the rate of convergence on the static tasks. Additionally, we construct a label-smoothed TET loss to further enhance the performances of SNNs on the neuromorphic tasks.

3) Compared with the baselines on the neuromorphic datasets and the large-scale static dataset ImageNet1k, our methods can achieve significant improvements. Moreover, there is almost no difference in the computational overhead and inference speed compared to the original models.

2 Related Works

2.1 Neuron Dynamics Modeling

The leaky integrate and analog fire spiking neuron [39] was proposed to replace binary spike with analog values for transmission, alleviating the issue of decreased performance in SNNs. The parametric leaky integrate-and-fire spiking neuron [40] was introduced to design a learnable dynamic model, enabling each neuron to learn optimal membrane time constants, thus increasing neuronal diversity. The gated leaky integrate-and-fire neuron [41] was employed for a channel-wise parameterization approach to fully parameterize the spiking neuron, including learnable decay mechanisms, potential thresholds, reset voltages, input conductance, and gating factors. The multi-level firing method [42] was used to enhance the expressive ability and achieve more efficient gradient propagation by integrating neurons with different thresholds to realize multi-level firing. Parallel spiking neurons [43] were discussed to remove the membrane potential reset process and redefine the dynamic mechanism in a non-iterative manner, addressing the difficulty of ordinary spiking neurons in learning long-term dependencies.

2.2 Direct Training Methods

The binary spikes emitted by spiking neurons during the forward phase are generated by a step function, which is a non-differentiable activation. In the backward propagation phase, this step function can be replaced with a surrogate gradient [19] to achieve direct training. The most common direct training method currently is backpropagation through time (BPTT) [22], which treats spiking neural networks (SNNs) as a special type of recurrent neural network (RNN). In this approach, gradients are propagated backward along the temporal dimension, which requires more computational resources and memory compared to their corresponding ANNs [44]. tdBN [21] explored normalization methods for spiking neural networks (SNNs) and achieved direct training of large-scale SNNs on the ImageNet dataset for the first time. Based on this, a more effective normalization method called TEBN [45] was proposed, which rescales the presynaptic inputs at each time step using distinct weights. TET [34] enabling spiking neural networks (SNNs) to converge to flatter minima compared to SDT [20], which enhances generalization capabilities. OTTT [46] and SLTT [47] simplified the gradient calculations along the temporal dimension in BPTT, significantly reducing memory and computational costs.

3 Analysis of Membrane Dynamics

In this section, we first investigate how the initial membrane potential affects neuronal spike patterns, and then analyze how the dynamic evolution of membrane potential drives improved SNN performance. These analyses underscore the critical role of membrane dynamics in SNNs and provide new insights into its impact on the SNN's representational capacity and convergence.

3.1 Preliminary of Spiking Neurons and Loss Functions

Spiking neurons are the fundamental unit of SNNs, used to simulate the dynamic behavior of brain neurons. Their operation is described by dynamical equations that are related to membrane potential and input current. The leaky integrate-and-fire (LIF) [7] model is one of the commonly used spiking neuron models, and its iterative form of the dynamical equations can be represented as follows:

$$h[t] = (1-\tau)s[t] + I[t], \quad h \in \mathbb{R}^{T\times N}, \quad I \in \mathbb{R}^{T\times N} \tag{1}$$

$$o[t] = h[t] > V_{th}, \qquad o \in \{0,1\}^{T\times N}, \quad V_{th} \in \mathbb{R} \tag{2}$$

$$s[t+1] = h[t] - o[t], \qquad s \in \mathbb{R}^{T\times N}, \quad s[0] \in \{0\}^{N} \tag{3}$$

where $I[t]$ represents the neural input current at time t, τ represents the membrane potential decay coefficient. When $\tau = 0$, it reduces to the Integrate-and-Fire (IF) [7] neuron. $s[t]$ represents the state of the membrane potential at time step t, and $s[0]$ is the state of the IMP, which is typically set to a constant 0. $h[t]$ represents the change in membrane potential during time step t, $o[t]$ indicates whether the neuron fires a spike at time t, and V_{th} represents the firing threshold of the neuron.

The most commonly used loss functions in direct training SNNs are SDT [20] and TET [34]. The SDT loss function $\mathcal{L}_{\text{SDT}}$ is defined as:

$$\mathcal{L}_{\text{SDT}} = \mathcal{L}_{\text{CE}}(\frac{1}{T} \times \sum_{t}^{T} y[t], y_{gt}), \tag{4}$$

here, T represents the total number of time steps, $y[t]$ represents the raw output of the model at each time step, y_{gt} represents the ground truth label, and $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss. SDT aggregate the outputs of the SNN by taking the mean of the outputs from all time steps, then calculate the loss based on the voting result. In TET, the averaging step is placed after the calculation of cross entropy loss:

$$\mathcal{L}_{\text{TET}} = \frac{1}{T} \times \sum_{t}^{T} \mathcal{L}_{\text{CE}}(y[t], y_{gt}), \tag{5}$$

the $\mathcal{L}_{\text{TET}}$ calculates the loss for each time step individually and then aggregates the losses from each time step to obtain the final loss. This approach can effectively improve the performance of SNN on neuromorphic datasets.

3.2 Membrane Dynamics Related to IMP

The membrane potential is commonly reset to zero before the next task in the current implementations of SNNs. However, through some experiments and the analysis of experimental results, we find that novel firing patterns and pattern mappings can be generated by adjusting IMP.

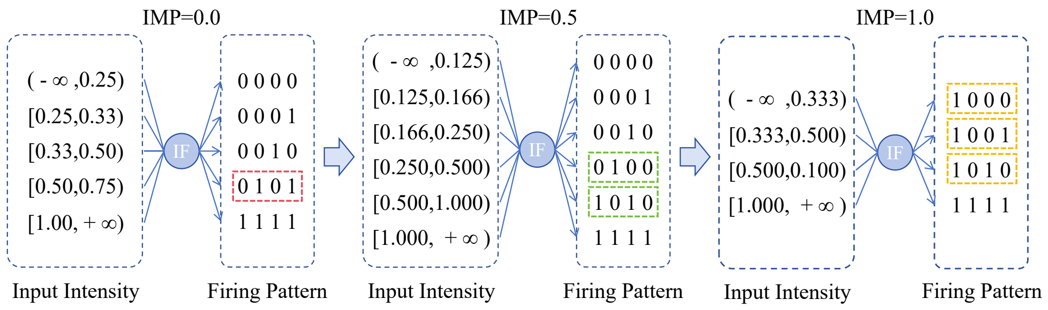
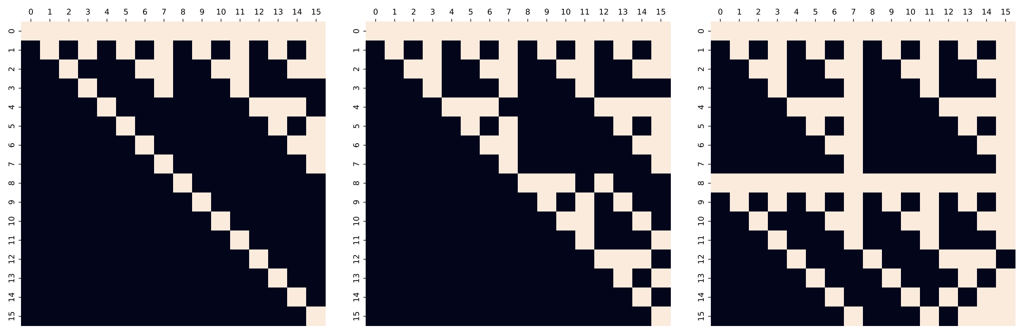


Figure 2: All firing patterns that IF neurons can generate under constant intensity input and 4 time steps. The red box highlights the disappearing firing patterns, while the green and yellow boxes denote the additional firing patterns due to the IMP change.

Observation 1: ***Novel firing patterns under constant intensity input can be generated by adjusting IMP.*** Figure 2 displays the firing patterns (spike sequences) generated by an IF neuron under constant input intensity. The number of patterns varies depending on the IMP. When the value of IMP increases from 0.0 to 0.5, the pattern '0101' disappears, while the new patterns '0100' and '1010' emerge, resulting in increased pattern diversity. Conversely, setting the IMP to 1.0 leads to the emergence of new patterns '1000', '1001', and '1010', but reduces the overall number of patterns. These additional patterns that cannot be generated if IMP is set to 0, can benefit static tasks by enabling SNNs to encode more information during the processing of static inputs.

Observation 2: ***New mappings of firing patterns can be generated by adjusting IMP.*** Apart from its impact on input encoding, our primary focus is whether the model's capabilities can be enhanced by modifying IMP. In ANNs, artificial neurons can map any single input variable to any value by adjusting the weights. Similarly, we hope that spiking neurons can also map input sequences to as many firing patterns as possible, thereby enabling the network to have better representation capabilities. From figure 3, we observe that every output pattern has at least one available pattern mapping. However, we can notice that black areas present in the figure, which means that no matter how the synaptic weights are adjusted, mapping among these spike patterns still cannot be established. In other words, spiking neurons can theoretically generate all firing patterns, but they cannot map any



(a) IMP=0.0 (b) IMP=0.25 (c) Learnable IMP

Figure 3: Pattern mapping of IF neuron over 4 time steps. The horizontal and vertical axes in the figure represent all possible spike patterns (16 total) that IF neurons may receive and emit. The white squares indicate that IF neuron can receive the spike pattern from the horizontal axis and emit a spike pattern on the vertical axis, known as pattern mapping.

specific pattern to all patterns. Figure 3b illustrates that through adjusting the IMP value from 0 to 0.25 can generate additional pattern mappings. Furthermore, Figure 3c demonstrates that when IMP is learnable, it exhibits a greater potential for establishing mappings among spike patterns. Therefore, we believe that the learnable IMP can effectively improve the expression capacity of spiking neurons.

3.3 Membrane Potential Evolution in Static Tasks

We further explore the effect of membrane potential variations on static tasks through a image classification task. We define SNN as follows to focus on its performance at each time step,

$$(s[t+1], y[t]) \leftarrow f(x[t], s[t], \theta), \tag{6}$$

where $f(\cdot)$ represents the network computation, θ represents the network weights, $s[t]$ represents the set of membrane potentials of all neurons in SNN with time step t, $x[t]$ represents the input at time step t, and $y[t]$ is the corresponding output. Assuming $x = x(t)$ is a constant input intensity for $t = 1, 2, ..., T$, then we can simplify Eq. 6 as:

$$y[t] = f(x, s[t], \theta). \tag{7}$$

On the static tasks, the temporal variations are determined solely by the state of the membrane potential $s[t]$, resulting in corresponding changes for the output $y[t]$.

Observation 3: ***In static tasks, the accuracy of SNNs at the each time step is sensitive to the current MP.*** Figure 4 demonstrates the test accuracy of SNNs at each time step on cifar10 dataset [48], which shows that the accuracy is extremely low at T=1, only 10.76%. However, as the time step T increases, the model accuracy exhibits an upward trend, exceeding 90%. It is worth noting that on the static tasks, since the input x and weight θ are fixed at each time step, the model's output at each time step is entirely determined by the current state of the membrane potential $s[t]$. For instance, when the membrane potential evolves to a "sufficiently good" state, such as at $t = 4$, the SNNs only requires the current membrane potential $s[4]$ and input x to achieve an accuracy of 92.05%, which is close to the model's final performance of 92.36%. Therefore, these findings prompt us to reconsider how to accelerate the evolution of the membrane potential to enhance the SNNs performances within a limited number of time steps.

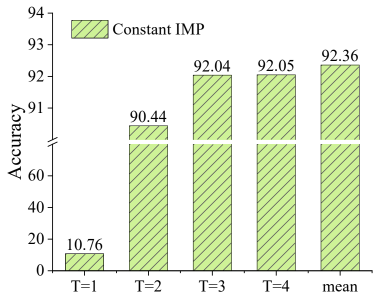
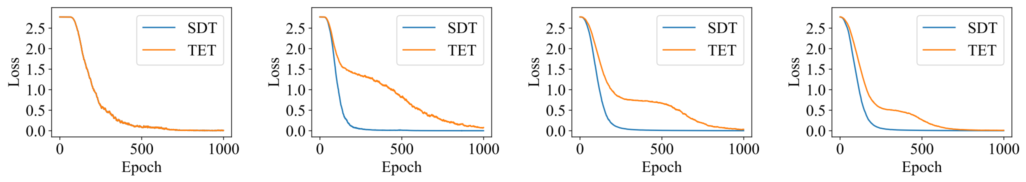


Figure 4: The test accuracy at each time step on the CIFAR10 dataset.

Observation 4: ***TET performs well on the neuromorphic tasks but exhibits slow convergence on the static tasks.*** We compare the SDT loss and the TET loss on the static datasets and the neuromorphic

Table 1: Test accuracy of TET and SDT on the static and neuromorphic datasets.

Loss Function	**Static Dataset**(SEW-R18) CIFAR10/100	ImageNet100	ImageNet1k	**Neuromorphic Dataset**(VGG11) CIFAR10DVS	DVSG128	NCaltech101
SDT Loss	**94.56/76.58**	**78.42**	**63.21**	84.3	98.26	85.78
TET Loss	94.33/76.40	77.80	62.92	**85.6**	**98.61**	**86.32**



(a) T=1 (b) T=2 (c) T=4 (d) T=6

Figure 5: The convergence speed of TET and SDT on the static data.

datasets, and find that TET loss has a significant advantage on the neuromorphic datasets, but it is not superior to SDT loss on the static datasets, as shown in Table 1. We believe that this phenomenon arises from the constant intensity input on the static datasets. By applying the TET loss Eq. 5 to the SNNs with the static input Eq. 7, we have:

$$\mathcal{L}_{\text{TET}} = \frac{1}{T} \times \sum_{t}^{T} \mathcal{L}_{\text{CE}}(y[t], y_{gt}),\ where\ y[t] = f(s[t], x, \theta). \tag{8}$$

It can be observed that on static tasks, the membrane potential $s[t]$, as the only time-varying term in the dynamic system, evolves gradually as the time step progresses. However, according to **Observation 3**, we know SNNs are sensitive to $s[t]$, which means that it is difficult to output the same result for different $s[t]$. The optimization goal of the TET loss is to make the network output the correct results for every time step, i.e. $y[t] = y_{gt}$. In this case, TET requires more iterations to build a flat "landscape", slowing down the convergence, as shown in Figure 5. Additionally, if the SNNs are insensitive to the value of $s[t]$ and even tends to the same value for all time steps, i.e. $y[1] = y[2] = ... = y[T] = \frac{1}{T}\sum_t^T y[t]$, then the output of the first time step is sufficient to represent the results of all T time steps. In this case, the computations for the subsequent time steps become redundant and meaningless.

4 Methods

4.1 Learnable IMP

The membrane potential in the current dynamic model of SNNs is typically initialized to a uniform constant value (usually 0). Based on Observation 1 and Observation 2, we find that the learnable IMP can enhance the expressive power of SNNs. Based on Observation 3, the learnable IMP can allow the membrane potential to start from a non-zero value, which may help improve the performances of SNNs on the static tasks. We can assign an independent learnable IMP for each spiking neuron. According to Eq. 1, the membrane potential accumulation can be represented as follows,

$$h[t] = \hat{\boldsymbol{s}}[t] + x[t], \quad x \in \mathbb{R}^{T\times N},\ h \in \mathbb{R}^{T\times N},\ \hat{\boldsymbol{s}} \in \mathbb{R}^{T\times N},\ \hat{\boldsymbol{s}}[0] \in \mathbb{R}^{N}, \tag{9}$$

where $\hat{\boldsymbol{s}}[0]$ represents the state of the IMP that is extended from zero to the real number. As a counterpart to the method of setting IMP to 0, in the initialization process, we set IMP to a uniform distribution with an expected value of 0,

$$\hat{\boldsymbol{s}}[0] = \mathit{Uniform}(-\lambda, \lambda)^{N}, \tag{10}$$

where λ is a hyper-parameter used to control the boundaries of the uniform distribution, ensuring the IMP remains within an appropriate range. Since the current implementation of SNNs requires memory allocation to store membrane potential states, replacing 0 with a trained IMP during the inference process will not incur additional computational overhead.

4.2 LTS Method for the Static Tasks

We propose a new post-processing representation method called last time step (LTS) to alleviate the convergence difficulties of TET on the static tasks, based on Observation 3 and Observation 4. This approach masks all outputs of the SNNs before the LTS and only retains the output of the LTS as the result of the entire model, which ensures that the SNNs can generate the most "high-quality" membrane potential without interference before the LTS T,

$$y[T] = f(s[T], x, \theta). \tag{11}$$

where $y[T]$ represents the LTS's output of the SNNs. When using only the LTS as the output, both SDT and TET losses yield the same representation,

$$\mathcal{L}_{\text{LTS}} = \mathcal{L}_{\text{CE}}(y[T], y_{gt}). \tag{12}$$

4.3 Label Smooth TET Loss for the Neuromorphic Tasks

Based on the results in Table 1, we recommend using TET loss to achieve better performances when dealing with the neuromorphic tasks. In the original version of TET [34], an additional mean squared error (MSE) regularization term was added to control the firing level of the last layer of the model, given by $\mathcal{L}_{\text{REG}} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\text{MSE}}(y[t], \phi)$, where ϕ denotes the target firing level. The coefficient λ controls the proportion of the two losses, and the complete loss is defined as $\mathcal{L}_{\text{Total}} = (1-\lambda)\mathcal{L}_{\text{TET}} + \lambda\mathcal{L}_{\text{REG}}$. We think that this setup will prevent the training loss of the model from converging to zero, because when $\mathcal{L}_{\text{REG}} = 0$, the model will output a constant value 1 at every time steps, rendering the model unable to perform the classification task. On the other hand, when $\mathcal{L}_{\text{TET}} = 0$, $\mathcal{L}_{\text{MSE}} > 0$. Considering that $\mathcal{L}_{\text{MSE}}$ plays a role similar to a smoothing process in the loss function, we propose removing $\mathcal{L}_{\text{REG}}$ and replacing the cross-entropy loss with a label smooth cross-entropy loss, as shown by the following equation:

$$\mathcal{L}_{\text{TET-S}} = \frac{1}{T} \times \sum_{t}^{T} \mathcal{L}_{\text{CE}}(f(s[t], x, \theta), \hat{y}_{gt}),\ where\ \hat{y}_{gt} = (1-\epsilon)y_{gt} + \frac{\epsilon}{K}, \tag{13}$$

here, y_{gt} represents the ground truth, $\hat{y}_{gt}$ represents the smoothed label, ϵ represents the smoothing factor, and K represents the number of classes. It can be observed that $\mathcal{L}_{\text{TET-S}}$ can effectively avoid the trade-off between firing level and classification accuracy for model training, and can theoretically allow the training loss to converge to zero.

5 Experiments

In this section, we demonstrate the effectiveness of our proposed method by extensive experiments. We compare the results of our method with other methods on both the neuromorphic dataset and the static dataset. Additional training procedures and other hyperparameter settings are provided in the appendix A.

5.1 Execution Speed Benchmark of IMP

We compare the execution speed and the memory consumption between the vanilla IF neurons and IF+IMP neurons in Figures 6. The number of neurons are set to 2^8, 2^{12}, 2^{16}, and 2^{20}, with time steps of 2, 4, 8, 16, and 32. All neurons are implemented by using spikingjelly and PyTorch, and the computations are performed on GPU. It can be observed that there is almost no difference (about $\pm 1.03\%$) in the execution time between the IF neurons with IMP and the vanilla IF neurons, including forward and backward propagation. In addition, since the computational consumption of SNNs is mainly caused by synaptic computation, the additional overhead caused by adding IMP can be neglected.

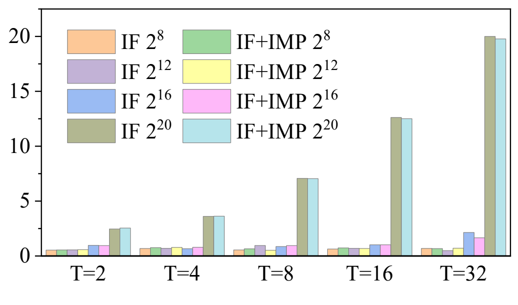
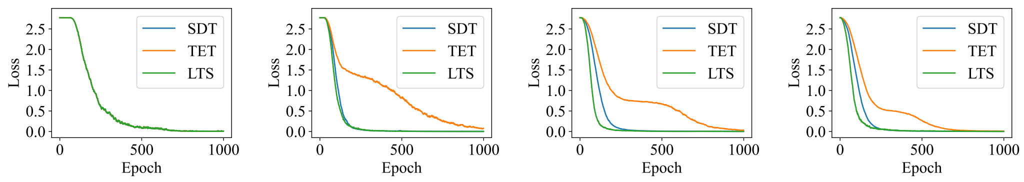


Figure 6: Execution time (ms) for the forward and backward pass of IF neurons, w/wo IMP.

5.2 Convergence Speed of LTS on the Static Data



(a) T=1 (b) T=2 (c) T=4 (d) T=6

Figure 7: Convergence speed of LTS, TET, and SDT on the static data.

We have conducted a validation of the convergence speed of SDT, TET, and LTS on the commonly used time steps (1, 2, 4, and 6), as shown in Figure 7. The application of LTS post-processing has resulted in an improvement in the convergence speed of SNNs.

5.3 Performances on the Neuromorphic Data Classification

Table 2: Comparison of our methods and other SOTA methods on the neuromorphic datasets. Size refers to the input resolution of SNNs.

Dataset	Method	SNN Architecture	Size	Time Steps	Accuracy(%)
CIFAR10-DVS	GLIF[41]	Wide 7B Net	48	16	78.10
	NDA[49]	VGG	48	10	79.60
	TET[34]	VGG	48	10	83.17
	TEBN[45]	VGG	48	10	84.90
	PSN[43]	VGG	48	10	85.90
	IMP(ours)	VGG	48	10	**85.90**
	IMP+TET-S(ours)	VGG	48	10	**87.10**
	IMP+TET-S(ours)	VGG	48	8	**87.80**
	PLIF[40]	PLIF Net	128	20	74.80
	TDBN[21]	ResNet-19	128	10	67.80
	Dspike[50]	ResNet-18	128	10	75.40
	KLIF[51]	PLIF Net	128	15	70.90
	SEW ResNet[52]	Wide 7B Net	128	16	74.40
	Spikformer[23]	Spikformer	128	10	78.90
	Spikformer[23]	Spikformer	128	16	80.90
	NDA[49]	VGG	128	10	81.70
	IMP(ours)	VGG	128	16	**86.30**
	IMP+TET-S(ours)	VGG	128	16	**87.00**
N-Caltech101	NDA[49]	VGG	48	10	78.20
	EventMix[53]	ResNet18	48	10	79.47
	ESP[54]	SNN7-LIFB	48	10	81.74
	TCJA[55]	TCJA-SNN	48	10	82.50
	TKS[56]	VGG-TKS	48	10	84.10
	IMP(ours)	VGG	48	10	**84.68**
	IMP+TET-S(ours)	VGG	48	10	**85.01**
	EventDrop[57]	VGG	128	10	74.04
	NDA[49]	VGG	128	16	83.70
	EventRPG[58]	VGG	128	10	85.62
	STR[59]	VGG	128	10	85.91
	IMP(ours)	VGG	128	16	**86.12**
	IMP+TET-S(ours)	VGG	128	16	**87.86**

We have applied our methods to a simple spiking VGG model and compare them with the SOTA SNNs on the neuromorphic datasets. Since the CIFAR10DVS and NCaltech101 datasets are not pre-divided into training and testing sets, we split these datasets in a 9:1 ratio. To ensure a fair comparison with other existed methods, we adopt the two configurations with resolutions of 48 and 128, respectively. The data preprocessing and training settings refer to the appendix A.8.

Table 2 reports the experimental results on the CIFAR10DVS and NCaltech101 datasets. On the CIFAR10DVS dataset, when setting the resolution to 48, the IMP method achieves the SOTA accuracy of 85.9%, which is 2.73% higher than the baseline [34] and is on par with the current SOTA method PSN. It should be noted that the removal of the reset process in PSN means that the spiking activities in the previous time steps will not affect the membrane potential values in the subsequent time steps. When we set the resolution to 128, the IMP method once again demonstrates its superiority, achieving the SOTA accuracy of 86.3%, which exceeds all other methods, including the current SOTA data augmentation method EventRPG[58]. In addition, we further explore the impact of using smoothed TET loss on model performance. The experimental results show that the performance of the model has been significantly improved under both configurations, achieving accuracies of 87.00% and 87.10% respectively. In addition, the accuracy can be further improved to 87.8% by setting the time step to 8. On the NCaltech101 dataset, our method can also demonstrate excellent performance. Specifically, when we set the resolution to 48, our method achieves the current SOTA accuracy of 85.01%. When switching to resolution of 128, our method further demonstrates its advantages, achieving the SOTA accuracy of 87.86%.

5.4 Performances on the Static Data Classification

Table 3: Comparison of our methods and other methods on the ImageNet1k dataset.

Method	Network Architecture	Reset	Params	Time Steps	Accuracy(%)
PSN[43]	SEW ResNet-18	×	11.69	4	67.63
	SEW ResNet-34	×	21.79	4	70.54
Dspike[50]	ResNet-34	✓	21.79	6	68.19
	VGG-16	✓	138.42	5	71.24
TET[34]	SEW ResNet-34	✓	21.79	4	68.00
TDBN[21]	ResNet-34	✓	21.79	6	67.05
TEBN[45]	SEW ResNet-34	✓	21.79	4	68.28
GLIF[41]	ResNet-34	✓	21.79	4	67.52
Spikformer[23]	Spikformer-6-512	✓	23.37	4	72.64
	Spikformer-8-512	✓	29.68	4	73.38
SEW ResNet[52]	SEW ResNet-18	✓	11.69	4	63.18
	SEW ResNet-34	✓	21.79	4	67.04
	SEW ResNet-50	✓	25.56	4	67.78
	SEW ResNet-101	✓	44.55	4	68.76
	SEW ResNet-152	✓	60.19	4	69.26
LTS	SEW ResNet-18	✓	11.69	4	64.33(+1.15)
	SEW ResNet-34	✓	21.79	4	68.10(+1.06)
	SEW ResNet-50	✓	25.56	4	71.24(+3.46)
IMP+LTS	SEW ResNet-18	✓	14.17	4	**65.38**(+2.20)
	SEW ResNet-34	✓	25.54	4	**68.90**(+1.86)
	SEW ResNet-50	✓	36.67	4	**71.83**(+4.05)

We apply our proposed IMP and LTS post-processing methods to the standard SEW-ResNet[52] architecture and compare them with the SOTA spiking neurons and the directly training SNN methods on the large-scale static dataset ImageNet1k [38].

Table 3 presents the detailed experimental results on the large-scale static dataset ImageNet1k. Specifically, by applying the LTS post-processing method to the SEW-ResNet18/34/50 models, we can obtain the accuracy improvements of 1.15%, 1.06%, and 3.46% compared to the baselines, respectively. These results demonstrate the effectiveness of LTS on the large-scale datasets. Furthermore, with the introduction of the learnable IMP, the accuracy can be further increased by 2.2%/1.86%/4.05%. With the LTS post-processing and learnable IMP, our SEW-ResNet50 achieves an accuracy of 71.83%, surpassing the accuracy of the vanilla SEW-ResNet152, 69.26%.

5.5 Further Ablation Studies

Table 4: Ablation Study on CIFAR10DVS and Imagenet100.

Dataset	Method	Spiking Network	Time-steps	Accuracy(%)
CIFAR10-DVS	SDT($\epsilon = 0.0$)	VGG	10	83.70
	TET($\epsilon = 0.0$)	VGG	10	84.90
	TET-S($\epsilon = 0.1$)	VGG	10	85.60
	TET-S($\epsilon = 0.01$)	VGG	10	**86.10**
	TET-S($\epsilon = 0.001$)	VGG	10	85.40
	IMP+SDT($\lambda = 0.0$)	VGG	10	83.70
	IMP+TET($\lambda = 0.0$)	VGG	10	85.90
	IMP+TET-S($\lambda = 0.0$)	VGG	10	**86.20**
	IMP+TET-S($\lambda = 0.2$)	VGG	10	**87.10**
	IMP+TET-S($\lambda = 0.4$)	VGG	10	**86.40**
ImageNet100	TET	SEW-ResNet18	4	78.50
	SDT	SEW-ResNet18	4	79.10
	LTS	SEW-ResNet18	4	**80.20**
	IMP+TET	SEW-ResNet18	4	78.70
	IMP+SDT	SEW-ResNet18	4	79.90
	IMP+LTS	SEW-ResNet18	4	**80.80**

Table 4 presents the results of a series of ablation studies conducted on the CIFAR10-DVS and ImageNet100 datasets, aimed at analyzing the impact of various factors on model performance. This helps understand the role of different components and parameters in the overall model, and aids in optimizing the model design. For the CIFAR10-DVS dataset, we explored methods including SDT, TET, and their variants TET-S and versions combined with IMP. VGG was used as the spiking neural network structure. The results show that for different ϵ values, the TET-S+IMP method achieved the best accuracy, with IMP+TET-S ($\lambda = 0.2$) reaching 87.10%, the highest among all methods on the CIFAR10-DVS dataset. For the ImageNet100 dataset, we tried TET, SDT, LTS, and their versions combined with IMP, using SEW-ResNet18 as the spiking neural network structure. On this dataset, the LTS method and its combined version with IMP, IMP+LTS, performed the best, reaching 80.80% accuracy.

6 Conclusions

We have proposed a learnable IMP by rethinking the membrane dynamics of SNNs to enhance the dynamics mechanism of spiking neurons. Additionally, we have presented a LTS post-processing method for the static tasks and a label-smoothed TET loss for the neuromorphic tasks. It is worth mentioning that our methods only require very minor modifications to the settings and loss functions of spiking neurons to effectively improve the performance of SNNs on the static tasks and the neuromorphic tasks. At the same time, almost no additional computational cost is required. Since our proposed method has broad compatibility with existing model structures and training methods, it can be widely applied on the existed methods to further improve their network performances.

Acknowledgement

This work was supported by Fundamental Research Funds for the Central Universities (Grant No. SWU021002), Project of Science and Technology Research Program of Chongqing Education Commission (Grant No. KJZD-K202100203), Key R&D Program of Zhejiang (2022C01048), and National Natural Science Foundation of China (Grant Nos. U1804158, 62376247, U20A20220, and 62334014).

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901, 2020.

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8748–8763, 2021.

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, 2021.

[5] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023.

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

[7] Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[8] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

[9] Ming ZHANG, Zonghua Gu, and Gang Pan. A survey of neuromorphic computing based on spiking neural networks. *Chinese Journal of Electronics*, 27(4):667–674, 2018.

[10] Duzhen Zhang, Tielin Zhang, Shuncheng Jia, Qingyu Wang, and Bo Xu. Tuning synaptic connections instead of weights by genetic algorithm in spiking policy network. *Machine Intelligence Research*, pages 1–13, 2024.

[11] Michael V DeBole, Brian Taba, Arnon Amir, Filipp Akopyan, Alexander Andreopoulos, William P Risk, Jeff Kusnitz, Carlos Ortega Otero, Tapan K Nayak, Rathinakumar Appuswamy, et al. Truenorth: Accelerating from zero to 64 million neurons in 10 years. *Computer*, 52(5):20–29, 2019.

[12] 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.

[13] De Ma, Xiaofei Jin, Shichun Sun, Yitao Li, Xundong Wu, Youneng Hu, Fangchao Yang, Huajin Tang, Xiaolei Zhu, Peng Lin, and Gang Pan. Darwin3: a large-scale neuromorphic chip with a novel ISA and on-chip learning. *National Science Review*, 11(5):nwae102, 03 2024.

[14] Bing Han and Kaushik Roy. Deep spiking neural network: Energy efficiency through time based coding. In *European Conference on Computer Vision (ECCV)*, pages 388–404. Springer, 2020.

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

[16] Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6316–6325. PMLR, 18–24 Jul 2021.

[17] Liuzhenghao Lv, Wei Fang, Li Yuan, and Yonghong Tian. Optimal ann-snn conversion with group neurons. *arXiv preprint arXiv:2402.19061*, 2024.

[18] Yangfan Hu, Qian Zheng, Xudong Jiang, and Gang Pan. Fast-snn: Fast spiking neural network by converting quantized ann. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):14546–14562, 2023.

[19] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10, 2016.

[20] 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, 2018.

[21] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 35(12):11062–11070, May 2021.

[22] 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.

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

[24] Hangchi Shen, Huamin Wang, Yuqi Ma, Long Li, Shukai Duan, and Shiping Wen. Multi-lra: Multi logical residual architecture for spiking neural networks. *Information Sciences*, 660:120136, 2024.

[25] Tao Chen, Chunyan She, Lidan Wang, and Shukai Duan. Memristive leaky integrate-and-fire neuron and learnable straight-through estimator in spiking neural networks. *Cognitive Neurodynamics*, pages 1–17, 2024.

[26] Lin Zhu, Siwei Dong, Jianing Li, Tiejun Huang, and Yonghong Tian. Retina-like visual image reconstruction via spiking neural model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1438–1446, June 2020.

[27] Zhanfeng Liao, Yan Liu, Qian Zheng, and Gang Pan. Spiking nerf: Representing the real-world geometry by a discontinuous representation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(12):13790–13798, Mar. 2024.

[28] Weixing Zhang, Zongrui Li, De Ma, Huajin Tang, Xudong Jiang, Qian Zheng, and Gang Pan. Spiking gs: Towards high-accuracy and low-cost surface reconstruction via spiking neuron-based gaussian splatting, 2024.

[29] Qiaoyi Su, Yuhong Chou, Yifan Hu, Jianing Li, Shijie Mei, Ziyang Zhang, and Guoqi Li. Deep directly-trained spiking neural networks for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6555–6565, October 2023.

[30] Malyaban Bal and Abhronil Sengupta. Spikingbert: Distilling bert to train spiking language models using implicit differentiation. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 38(10):10998–11006, Mar. 2024.

[31] Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*, 2023.

[32] Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31. Curran Associates, Inc., 2018.

[33] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: Spiking neural network for energy-efficient object detection. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 34(07):11270–11277, Apr. 2020.

[34] 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)*, 2022.

[35] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in neuroscience*, 11:244131, 2017.

[36] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9:159859, 2015.

[37] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 21056–21069. Curran Associates, Inc., 2021.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[39] Zhenzhi Wu, Hehui Zhang, Yihan Lin, Guoqi Li, Meng Wang, and Ye Tang. Liaf-net: Leaky integrate and analog fire network for lightweight and efficient spatiotemporal information processing. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11):6249–6262, 2022.

[40] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2661–2671, October 2021.

[41] Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. Glif: A unified gated leaky integrate-and-fire neuron for spiking neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:32160–32171, 2022.

[42] Lang Feng, Qianhui Liu, Huajin Tang, De Ma, and Gang Pan. Multi-level firing with spiking ds-resnet: Enabling better and deeper directly-trained spiking neural networks. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2471–2477, 2022.

[43] Wei Fang, Zhaofei Yu, Zhaokun Zhou, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. Parallel spiking neurons with high efficiency and long-term dependencies learning ability. *arXiv preprint arXiv:2304.12760*, 2023.

[44] Lei Deng, Yujie Wu, Xing Hu, Ling Liang, Yufei Ding, Guoqi Li, Guangshe Zhao, Peng Li, and Yuan Xie. Rethinking the performance comparison between snns and anns. *Neural Networks*, 121:294–307, 2020.

[45] Chaoteng Duan, Jianhao Ding, Shiyan Chen, Zhaofei Yu, and Tiejun Huang. Temporal effective batch normalization in spiking neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[46] Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Zhouchen Lin. Online training through time for spiking neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 20717–20730. Curran Associates, Inc., 2022.

[47] Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Towards memory- and time-efficient backpropagation for training spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6166–6176, October 2023.

[48] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[49] Yuhang Li, Youngeun Kim, Hyoungseob Park, Tamar Geller, and Priyadarshini Panda. Neuromorphic data augmentation for training spiking neural networks. In *European Conference on Computer Vision (ECCV)*, pages 631–649. Springer, 2022.

[50] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[51] Chunming Jiang and Yilei Zhang. KLIF: An optimized spiking neuron unit for tuning surrogate gradient slope and membrane potential. *arXiv e-prints*, page arXiv:2302.09238, February 2023.

[52] 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, 2021.

[53] Guobin Shen, Dongcheng Zhao, and Yi Zeng. Eventmix: An efficient data augmentation strategy for event-based learning. *Information Sciences*, 644:119170, 2023.

[54] Guobin Shen, Dongcheng Zhao, and Yi Zeng. Exploiting high performance spiking neural networks with efficient spiking patterns. *arXiv preprint arXiv:2301.12356*, 2023.

[55] Rui-Jie Zhu, Malu Zhang, Qihang Zhao, Haoyu Deng, Yule Duan, and Liang-Jian Deng. Tcja-snn: Temporal-channel joint attention for spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[56] Yiting Dong, Dongcheng Zhao, and Yi Zeng. Temporal knowledge sharing enable spiking neural network learning from past and future. *IEEE Transactions on Artificial Intelligence*, 2024.

[57] Fuqiang Gu, Weicong Sng, Xuke Hu, and Fangwen Yu. Eventdrop: Data augmentation for event-based learning. In *30th International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[58] Mingyuan Sun, Donghao Zhang, Zongyuan Ge, Jiaxu Wang, Jia Li, Zheng Fang, and Renjing Xu. Eventrpg: Event data augmentation with relevance propagation guidance. *arXiv preprint arXiv:2403.09274*, 2024.

[59] Dengyu Wu, Yi Qi, Kaiwen Cai, Gaojie Jin, Xinping Yi, and Xiaowei Huang. Direct training needs regularisation: Anytime optimal inference spiking neural network. *arXiv preprint arXiv:2405.00699*, 2024.

A Appendix / supplemental material

A.1 Broader Impacts

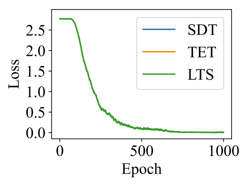
This paper focuses on the fundamental research of spiking neural networks, with the goal of revealing the impact of membrane dynamics on the network and optimizing its performance. Generally, there are no negative societal impacts in this work.

A.2 Limitations

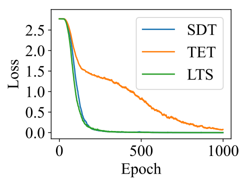
IMP has a small gradient during training, which makes it sensitive to initialization (Figure 4). In addition, learnable IMP may lead to an excessive number of parameters, as it assigns initial states to each neuron, although this has the same computational cost as setting the IMP to 0. The advantage of LTS may reduce when the time step is set too large, due to the supervision is only applied at the last time step. Therefore, we only recommend using LTS on static tasks with time steps less than 8 (Table 8), which should be able to handle most situations. Additionally, the performance of the combination of LTS and the latest spike transformer technology is not yet clear. Furthermore, we have not found a unified loss function that can achieve superior performance on both static tasks and neuromorphic tasks, which remains a challenge in the current research.

A.3 Convergence Speed

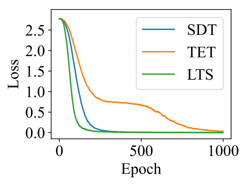
We compared the convergence speed of TET and SDT at different time steps (T=1,2,4,6,8,12,16,24,32). For static tasks, TET's convergence speed was lower than SDT's, and the difference in convergence speed diminished as the time step increased.



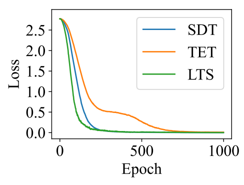
(a) T=1



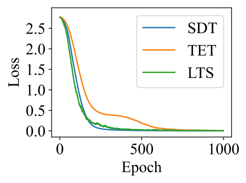
(b) T=2



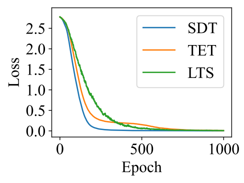
(c) T=4



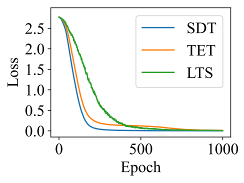
(d) T=6



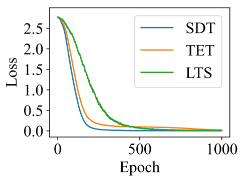
(e) T=8



(f) T=16



(g) T=24



(h) T=32

Figure 8: The convergence speed of SDT, TET and LTS on static data.

A.4 Compared with Transformer-based SNNs

The performance of SEW-ResNet-LTS can be close to some Transformer-based SNNs (Table 5).

Table 5: Accuracy and theoretical energy consumption compared with Transformer-based SNNs.

Model	Param (M)	SOPs (G)	Power (mJ)	Accuracy
Spikformer-8-384	16.81	6.82	7.73	70.24
Spikformer-6-512	23.37	8.69	9.42	72.46
Spike-driven 8-384	16.81	-	3.90	72.28
Meta-SpikeFormer	15.10	-	16.70	74.10
SEW-R50-LTS (ours)	25.56	3.10	2.79	71.24
SEW-R50-LTS+IMP (ours)	36.67	3.45	3.11	71.83

A.5 Energy Consumption

IMP does not incur significant additional theoretical power consumption, but can effectively improve the performance of SNNs (Table 6).

Table 6: Accuracy and theoretical energy consumption on ImageNet1k dataset.

Model	Training Method	Accuracy	SOPs(G)	Power(mJ)
SEW-ResNet18	TET	62.92	1.36055	1.22449
SEW-ResNet18	SDT	63.21	1.37418	1.23676
SEW-ResNet18	LTS	64.33	**1.21427**	**1.09285**
SEW-ResNet18	LTS+IMP	**65.38**	1.31371	1.18234
SEW-ResNet34	TET	67.98	3.59539	3.23585
SEW-ResNet34	SDT	68.10	3.52732	3.17459
SEW-ResNet34	LTS	68.10	**3.11694**	**2.80525**
SEW-ResNet34	LTS+IMP	**68.90**	3.12180	2.80962
SEW-ResNet50	TET	69.87	3.40181	3.06163
SEW-ResNet50	SDT	70.33	3.20071	2.88064
SEW-ResNet50	LTS	71.24	**3.10432**	**2.79389**
SEW-ResNet50	LTS+IMP	**71.83**	3.45325	3.10792

A.6 Performance of LTS on DVS Tasks

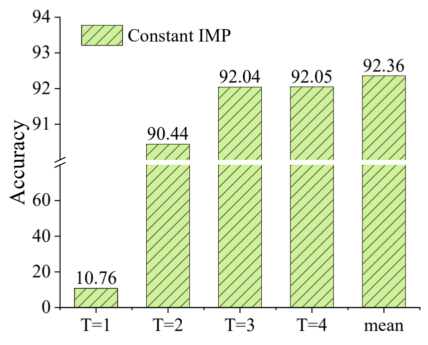
The LTS method can lead to information loss, especially on DVS tasks with a large number of time steps (Table 7). Therefore, we suggest considering the use of LTS only in static tasks, as the effectiveness of LTS relies on the assumption of having the same input at each time step.

Table 7: Accuracy on CIFAR10DVS dataset with different time-steps.

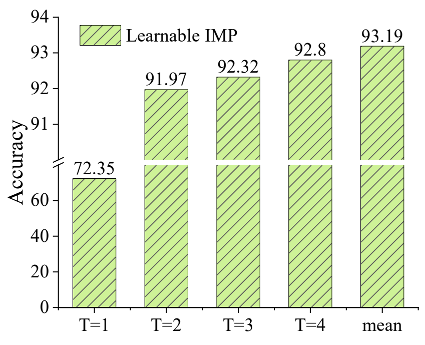
Model	Training Method	T=4	T=8	T=10	T=16
VGG	TET	83.8	85.0	85.8	86.4
VGG	SDT	83.4	84.3	84.4	85.1
VGG	LTS	83.7	83.0	82.9	82.3

A.7 Enhancing Performance by Learnable IMP

The learnable IMP can significantly improve the accuracy of the first time step and lead to better overall performance (Figure 9).



(a) ConstantIMP



(b) LearnableIMP

Figure 9: The test accuracy at each time step on the CIFAR10 dataset.

A.8 Experimental Configurations and Hyperparameter Settings

Table 8 lists the key parameters required for training on the static datasets ImageNet1k, ImageNet100, CIFAR10, and CIFAR100. Table 9 outlines the key parameters used for training on the neuromorphic datasets CIFAR10-DVS-128, CIFAR10-DVS-48, N-Caltech101-128, and N-Caltech101-48.

Table 8: Experimental configurations on static task.

hyper-parameter	ImageNet1K	ImageNet100	CIFAR10	CIFAR100
architecture	SEW-ResNet	SEW-ResNet	SEW-ResNet	SEW-ResNet
time steps	4	4	4	4
enable TEBN	No	No	No	No
detach reset	Yes	Yes	Yes	Yes
spiking neuron	IF+IMP	IF+IMP	IF+IMP	IF+IMP
sg function	Atan	Atan	Atan	Atan
membrane decay	-	-	-	-
optimizer	AdamW	AdamW	AdamW	AdamW
learning rate	0.001	0.001	0.001	0.001
weight decay	5e-4	5e-4	5e-4	5e-4
momentum	-	-	-	-
epoch	320	200	200	200
warm up	10	10	10	10
lr schedule	cosine	cosine	cosine	cosine
loss function	SDT/TET	SDT/TET	SDT/TET	SDT/TET
label smooth	-	0.1	0.1	0.1
data augment	standard	standard	standard	standard
enable cutmix	No	Yes	Yes	Yes
enable mixup	No	Yes	Yes	Yes
GPUs	4	1	1	1

Table 9: Experimental configurations on neuromorphic task.

hyper-parameter	CIFAR10DVS-48	CIFAR10DVS-128	NC101-48	NC101-128
architecture	VGG11	VGG11	VGG11	VGG11
time steps	10	16	10	16
enable TEBN	Yes	No	Yes	No
detach reset	Yes	Yes	Yes	Yes
spiking neuron	LIF+IMP	LIF+IMP	LIF+IMP	LIF+IMP
sg function	ZIF	sigmoid	ZIF	sigmoid
membrane decay	0.25	0.5	0.25	0.5
optimizer	SGD	AdamW	SGD	AdamW
learning rate	0.1	0.001	0.1	0.001
weight decay	5e-4	0.06	5e-4	0.06
momentum	0.9	-	0.9	-
epoch	200	200	150	150
warm up	0	30	0	30
lr schedule	cosine	cosine	cosine	cosine
loss function	SDT/TET	SDT/TET	SDT/TET	SDT/TET
label smooth	0.01	0.01	0.01	0.01
event augment	standard	NDA	standard	NDA
enable cutmix	No	Yes	No	Yes
enable mixup	No	Yes	No	Yes
GPUs	1	1	1	1

A.9 On-chip Learning

The following approach can be useful for implementing on-chip learning IMP: (1) Use an auxiliary neuron to distribute IMP (by firing a spike) to the other neurons at the initial time step. (2) Optimize the synaptic weights of this auxiliary neuron to adjust IMP.

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: The abstract and introduction clearly state our contributions in the field of spiking neural networks, including the discovery of special phenomena caused by SNN dynamics and the inspired improvement methods.

   Guidelines:

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

   Justification: We discussed the limitations of the proposed method in the appendix, which is enlightening for future work.

   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: The paper provides a complete proof of the proposed viewpoint and method.

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: The method section provides a detailed introduction to the method proposed in this paper, which can be reproduced by referring to the appendix and submitted code.

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: The dataset used in this article is publicly available, and the code will be made public to ensure that others can reproduce the experimental results.

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).
- 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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The appendix of the paper provides detailed experimental settings.

   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: [Yes]

   Justification: The paper accurately presents error bars for the execution speed benchmark. Notably, our experiments involved comparing our method's optimal performance with other approaches.

   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: The article provides the resource cost required for conducting experiments, as well as the execution time of our proposed method. Further detailed information is provided in the code.

   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 in this paper adheres to 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: This paper focuses on the fundamental research of spiking neural networks, with the goal of revealing the impact of membrane dynamics on the network and optimizing its performance. Generally, there are no negative societal impacts in this work.

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.
- 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: This paper focuses on the fundamental research of spiking neural networks, which does not involve the development or release of data or models that have a high risk for misuse.

    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: The creators or original owners of the assets (such as code, data, and models) used in this paper have been properly credited. Their contributions have been explicitly mentioned in an appropriate manner. Additionally, the license and terms of use for each asset have been explicitly stated and adhered to, including obtaining any necessary permissions or authorizations.

    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: [Yes]

    Justification: The experimental code will be made openly accessible, along with the necessary documents to facilitate reproducibility of the experimental results and utilization of the code for future 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 experiments or 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: The 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.