Supplementary Materials for: Accelerating Training of Spiking Neural Networks with Stabilized Spiking Flow

Jingtao Wang[1,4,5], Zengjie Song[2], Yuxi Wang[3],
Jun Xiao[1*], Yuran Yang[6], Shuqi Mei[6], Zhaoxiang Zhang[1,3,4,5*]
[1]University of Chinese Academy of Sciences [2]Xi'an Jiaotong University
[3]Centre for Artificial Intelligence and Robotics, HKISI-CAS
[4]Institute of Automation, Chinese Academy of Sciences
[5]State Key Laboratory of Multimodal Artificial Intelligence Systems [6]Tencent
{wangjingtao2021, zhaoxiang.zhang}@ia.ac.cn, zjsong@hotmail.com, xiaojun@ucas.ac.cn

A. Experiment Details

In this section we give a detailed description of settings for experiments in Table 1. As introduced in the main text, we apply SSF in TET [2], RecDis [4], SEW [3], tdBN [9] frameworks and evaluate on CIFAR-10 [5], CIFAR-100 [5], Tiny-ImageNet [1] and DVS-CIAFAR10 [6] datasets. As for TET framework, we adopt PreAct-ResNet-18 structure on CIFAR-10 and CIFAR-100, while using VGG-11 on DVS-CIFAR10. RecDis employs the same network structures with TET. For tdBN, we apply ResNet-19 structure proposed in [9] on all datasets. We employ SEW framework on Tiny-ImageNet dataset with SEW-ResNet-34 network. To reduce information loss, we set V_{th} of the output layer big enough to make sure that no spike would be generated, and use final membrane potential as outputs in all experiments shown in Table 1.

Aside from some shared hyper-parameters like λ, V_{th}, batch size, and learning rate, four different frameworks all have their unique hyper-parameters. We set these exclusive parameters carefully according to their original papers. In all experiments shown in Table 1, we fix batch size to 128. We train SNNs with SGD optimizer with learning rate set to 0.01 and use cosine annealing as the learning rate schedule. λ and V_{th} vary between different experiments, of which detailed settings are shown in Table S1. For V_{th}, we set it to 1 except the special need of RecDis. For λ, we either adopt the setting used in origin works (like TET) or select a suitable value to make sure a better performance. Our method needs a big enough λ to quickly stabilize output spiking flow. We conduct experiments to study the influence of λ, results are shown in Table S3.

*Corresponding authors.

Table S1. Settings for all experiments in Table 1.

Dataset	Framework	λ	V_{th}
CIFAR-10	tdBN	0.90	1.0
	tdBN-ours	0.90	1.0
	RecDis	0.80	0.5
	RecDis-ours	0.90	0.5
	TET	0.50	1.0
	TET-ours	0.95	1.0
CIFAR-100	tdBN	0.90	1.0
	tdBN-ours	0.80	1.0
	RecDis	0.80	0.5
	RecDis-ours	0.80	0.5
	TET	0.50	1.0
	TET-ours	0.95	1.0
Tiny-ImageNet	SEW	1.00	1.0
	SEW-ours	1.00	1.0
DVS-CIFAR10	tdBN	0.80	1.0
	tdBN-ours	0.80	1.0
	RecDis	0.80	0.5
	RecDis-ours	0.80	0.5
	TET	0.50	1.0
	TET-ours	0.85	1.0
	TET-CE-ours	0.95	1.0

B. Soft Reset Function

In addition to hard reset function introduced in Section 3.1, soft reset function is also a widely used reset function in SNN. After generating a spike, soft reset sets membrane potential to $u[t]-V_{th}$ rather than u_{reset} as in hard reset. We discuss soft reset function based on the LIF model described in Section 3.1. Formally, we denote the forward process of

soft reset function as [7]:

$$u_i^l[t] = \begin{cases} \lambda\left[u_i^l[t-1] - o_i^l[t-1]V_{th}\right] + I_i^l[t], \ t \neq 0, \\ I_i^l[0], \ t = 0. \end{cases} \tag{S1}$$

Based on the same idea, we generate stabilized input and output flows from the whole input and output electricity. Considering the attenuation, the whole input electricity is $\sum_{t=0}^{T} \lambda^{T-t} I_i^l[t]$. From Eq. (S1) we can build the relationship between the whole input and output electricity, which can be denoted as:

$$\begin{aligned} \sum_{t=0}^{T} \lambda^{T-t} u_i^l[t] = &\sum_{t=0}^{T-1} \lambda^{T-t}\left[u_i^l[t-1] - o_i^l[t]V_{th}\right] \\ &+ \sum_{t=0}^{T} \lambda^{T-t} I_i^l[t]. \end{aligned} \tag{S2}$$

With some transitions, we obtain:

$$u_i^l[T] = \sum_{t=0}^{T} \lambda^{T-t} I_i^l[t] - \sum_{t=0}^{T-1} \lambda^{T-t} o_i^l[t] V_{th}. \tag{S3}$$

Eq. (S3) can be further divided to:

$$u_i^l[T] - V_{th} o_i^l[T] = \sum_{t=0}^{T} \lambda^{T-t} I_i^l[t] - V_{th} \sum_{t=0}^{T} \lambda^{T-t} o_i^l[t]. \tag{S4}$$

The left part of Eq. (S4) stands for electricity remaining in spiking neurons after time step T, while the right part represents the whole input electricity reducing the whole output electricity, considering the decay of electricity at each time step. Different from the situation of hard reset, left electricity can be much higher than V_{th} in Eq. (S4), so we can not just dismiss it. However, the left electricity has two resources. One is the accumulated electricity in last several time steps, dubbed V_{last}, which is less than V_{th} and can be ignored. The other one is that when input electricity surpasses V_{th}, there will still remain some electricity after generating a spike. We name this part of left electricity in each time step as V_{exceed}. Considering that input spike trains can be viewed as stabilized input flows, input electricity of each time step equals to a constant real number. So V_{exceed} is either greater than 0 or equals to 0 in every time step.

In situation when $V_{exceed} > 0$, input electricity exceeds V_{th} in each time step, so neuron i will generate a spike every time step, which meaning that the output spike train reaches it's upper bound. In this condition, we have no way to distinguish different input spike trains as they are all beyond output trains' representation ability. To be more specific, spiking neuron clamps the whole input electricity to $[0, \sum_{t=0}^{T} \lambda^{T-t} V_{th}]$. On the other side, when $V_{exceed} = 0$, input electricity is in the range of $[0, \sum_{t=0}^{T} \lambda^{T-t} V_{th}]$, and the left part of Eq. (S4) only consists of V_{last}, which can be dismissed with long enough time steps.

In the same way, we define $FI = \frac{\sum_{t=0}^{T} \lambda^{T-t} I_i^l[t]}{\sum_{t=0}^{T} \lambda^{T-t}}$ as stabilized input flow, $FO = \frac{\sum_{t=0}^{T} \lambda^{T-t} o_i^l[t]}{\sum_{t=0}^{T} \lambda^{T-t}}$ as stabilized output flow. Average spiking electricity R is not necessary for electricity lost in each spiking process all equaling to V_{th}. With these definitions, the relationship between input and output spiking flows can be derived as:

$$FO = Clamp(\frac{FI}{V_{th}}, 0, 1). \tag{S5}$$

The backward process resembles Section 3.3. Formally, we have:

$$\frac{\partial L}{\partial I_i^l[t]} = \frac{\partial L}{\partial FO} \frac{\partial FO}{\partial FI} \frac{\partial FI}{\partial I_i^l[t]}, \tag{S6}$$

where $\frac{\partial L}{\partial FO} = \frac{\sum_{t=0}^{T} \lambda^{T-t} \frac{\partial L}{\partial I_i^{l+1}[t]}}{\sum_{t=0}^{T} \lambda^{T-t}}$, $\frac{\partial FI}{\partial I_i^l[t]} = 1$ as inputs naturally experience attenuation during inference. Specially, we have:

$$\frac{\partial FO}{\partial FI} = \begin{cases} 1, \ 0 < FI < V_{th}, \\ 0, \ others. \end{cases} \tag{S7}$$

To demonstrate that SSF can be applied to soft reset function, we arrange experiments training with both classic SG and SSF methods. Considering that all these four frameworks are not suit to soft reset function, we conduct experiments about soft reset function on a new framework using network structure similar to [7]. We set V_{th} to 1 and use both IF and LIF neuron models. We set λ to 0.95 when training SNNs with LIF model. Results are shown in Table S2, from which we can see that SSF method achieves nearly the same performance in all experiments compared to classic SG.

Table S2. Experiment results on different neuron models.

Model	Reset Function	Method	Accuracy
IF	hard	SG	94.77%
	hard	SSF	94.49%
	soft	SG	93.77%
	soft	SSF	93.44%
LIF	hard	SG	94.99%
	hard	SSF	94.90%
	soft	SG	94.00%
	soft	SSF	94.29%

C. Supplementary Experiments

In this section we provide some supplementary experiments to better illustrate the characteristic of our method. In Section 4, we have shown SSF's impressive effect on speeding up the training of SNN in one epoch. To analysis the acceleration efficiency from a global view, we also record

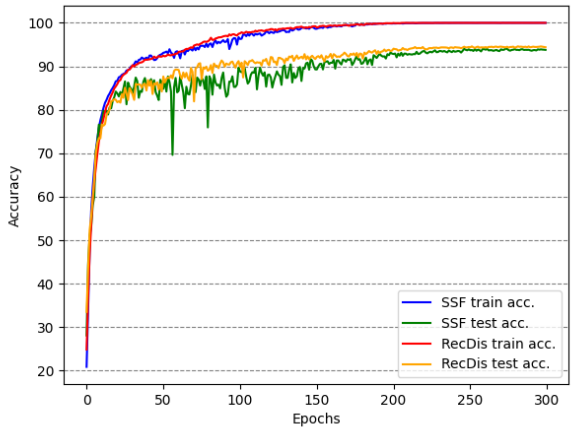

Figure S1. Training and test accuracy curves of training process with original RecDis frameworks and accelerated RecDis frameworks.

and depict the accuracy curves of RecDis and RedDis-ours trained on CIFAR-10 in Fig. S1. We can easily observe that epochs needed to converge are nearly the same between the original version and the accelerated one, while our SSF obviously decreases training time spent in each epoch by a large margin.

As mentioned before, SSF needs big enough λ to guarantee a satisfying result, which is due to the nature of stabilized spiking flow. Shown by [8], an input neuron's first spike will make the output flow to be unstable in a period of time, which increases the difference between output stabilized spiking flow and actual output spike train. Meanwhile, spiking neurons with smaller λ need more time steps to accumulate membrane potential, which makes the unstable time grow longer and thus leading to lower performance.

We have conducted experiments studying the influence of membrane potential decay rate λ in TET framework on CIFAR-10 dataset. We keep other settings same as described before with λ decreasing from 1 to 0.5, and change TET loss to cross entropy loss to remove the effect of loss function. Results are shown in Table S3, from which we can see that there exists a gap between $\lambda = 0.8$ and $\lambda = 0.7$. For λ higher or lower than 0.7, it has little effects on obtained SNNs' performance.

Table S3. Influence of membrane potential decay rate λ.

λ	1.0	0.9	0.8	0.7	0.6	0.5
acc	95.2%	95.5%	94.9%	93.2%	93.6%	93.1%

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[2] 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*, 2022. 1

[3] 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*, 34:21056–21069, 2021. 1

[4] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 326–335, 2022. 1

[5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1

[6] 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. 1

[7] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12444–12453, 2022. 2

[8] Xueyuan She, Saurabh Dash, and Saibal Mukhopadhyay. Sequence approximation using feedforward spiking neural network for spatiotemporal learning: Theory and optimization methods. In *International Conference on Learning Representations*, 2021. 3

[9] 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*, volume 35, pages 11062–11070, 2021. 1