Multi-Level Firing with Spiking DS-ResNet: Enabling Better and Deeper Directly-Trained Spiking Neural Networks

Lang Feng[1] , **Qianhui Liu**[1] , **Huajin Tang**[1,2] , **De Ma**[1] and **Gang Pan**[1,2,*]

[1]College of Computer Science and Technology, Zhejiang University, Hangzhou, China
[2]Zhejiang Lab, Hangzhou, China

{langfeng, qianhuiliu, htang, made, gpan}@zju.edu.cn

Abstract

Spiking neural networks (SNNs) are bio-inspired neural networks with asynchronous discrete and sparse characteristics, which have increasingly manifested their superiority in low energy consumption. Recent research is devoted to utilizing spatio-temporal information to directly train SNNs by backpropagation. However, the binary and non-differentiable properties of spike activities force directly trained SNNs to suffer from serious gradient vanishing and network degradation, which greatly limits the performance of directly trained SNNs and prevents them from going deeper. In this paper, we propose a multi-level firing (MLF) method based on the existing spatio-temporal back propagation (STBP) method, and spiking dormant-suppressed residual network (spiking DS-ResNet). MLF enables more efficient gradient propagation and the incremental expression ability of the neurons. Spiking DS-ResNet can efficiently perform identity mapping of discrete spikes, as well as provide a more suitable connection for gradient propagation in deep SNNs. With the proposed method, our model achieves superior performances on a non-neuromorphic dataset and two neuromorphic datasets with much fewer trainable parameters and demonstrates the great ability to combat the gradient vanishing and degradation problem in deep SNNs.

1 Introduction

Spiking neural networks (SNNs) are developed to realize brain-like information processing [Maass, 1997], which use asynchronous binary spike signals to transmit information and have the ability to process information in both spatial domain (SD) and temporal domain (TD). Besides, the sparsity and event-driven properties position them as potential candidates for the implementation of low energy consumption on dedicated neuromorphic hardware. As an example, the energy consumed by SNNs to transmit a spike on neuromorphic hardware is only nJ or pJ [Diehl and Cook, 2015].

In terms of learning algorithms, existing unsupervised learning algorithms [Qi *et al.*, 2018; Liu *et al.*, 2020] are difficult to train deep SNNs. Currently, there are two main learning algorithms for deep SNNs training. One is ANN-SNN conversion learning [Sengupta *et al.*, 2019; Yan *et al.*, 2021; Hu *et al.*, 2021], which converts the pre-trained ANN model to the SNN model. Conversion learning can achieve deep SNNs training with competitive results, but it has to consume a large number of timesteps to ensure the coding resolution. Moreover, conversion learning cannot utilize the TD information, making it difficult to train neuromorphic datasets. The other is direct supervised learning [Wu *et al.*, 2018; Gu *et al.*, 2019; Liu *et al.*, 2022; Zheng *et al.*, 2021], which is the approach taken by this paper. Direct supervised learning has great potential to make full use of spatio-temporal information to train the network and can reduce the demand for timesteps. However, to achieve more efficient direct supervised learning for better and deeper directly-trained SNNs, there are still two challenging issues to overcome.

The first is gradient vanishing. Due to non-differentiable spike activities, approximate derivative [Neftci *et al.*, 2019] has to be adopted to make the gradient available, such as rectangle function and Gaussian cumulative distribution function [Wu *et al.*, 2018]. However, it will raise a problem that the limited width of the approximate derivative causes membrane potentials of a multitude of neurons to fall into the saturation area, where the approximate derivative is zero or a tiny value. Furthermore, the sharp features that have larger values in the feature map cannot be further enhanced due to falling into the saturation area to the right of the approximate derivative caused by excessive membrane potential. This greatly limits the performance of deep SNNs, and the neurons located in this saturation area caused by excessive membrane potential are termed to be *dormant* units in this paper. In the above cases, the gradient propagation will be blocked and unstable, therefore resulting in the gradient vanishing and increasing the difficulty of training deep SNNs.

The second is network degradation, which is terribly serious in deep directly-trained SNNs, even if residual structure [He *et al.*, 2016] is adopted. Therefore, existing training methods mainly expand SNNs in width to get improved performance, resulting in a large number of trainable parameters. The above non-differentiable spike activity is one of the reasons for network degradation, and the weak spatial ex-

---

*Corresponding author.

pression ability of binary spike signals is another significant factor. For the widely used spiking neuron models like leaky integrate-and-fire (LIF) model, the sharp feature with a larger value and the non-sharp feature with a smaller value will have the same output in the forward process if the corresponding membrane potentials both exceed the firing threshold. As a result, the loss of information caused by discrete spikes will make residual structures hard to perform identity mapping.

We take steps to address these two challenges for enabling better and deeper directly-trained deep SNNs. We first propose the multi-level firing (MLF) method. MLF expands the non-zero area of the rectangular approximate derivatives by allocating the coverage of approximate derivative of each level. In this way, the membrane potentials of neurons are more likely to fall into the area where the derivative is not zero, so as to alleviate gradient vanishing. Besides, with the activation function of neurons in MLF generating spikes with different thresholds when activating the input, the expression ability of the neurons can be improved. Second, we propose spiking dormant-suppressed residual network (spiking DS-ResNet). Spiking DS-ResNet can efficiently perform identity mapping of discrete spikes as well as reduce the probability of dormant unit generation, making it more suitable for gradient propagation. To demonstrate the effectiveness of our work, we perform experiments on a non-neuromorphic dataset (CIFAR10) and neuromorphic datasets (DVS-Gesture, CIFAR10-DVS). Our model achieves state-of-the-art performances on all datasets with much fewer trainable parameters. Experimental analysis indicates that MLF effectively reduces the proportion of dormant units and improves the performances, and MLF with spiking DS-ResNet allows SNNs to go very deep without degradation.

2 Related Work

Learning algorithm of deep SNNs   For deep SNNs, there are two main learning algorithms to achieve competitive performance: (1) indirect supervised learning such as ANN-SNN conversion learning; (2) direct supervised learning, the gradient descent-based backpropagation method.

The purpose of ANN-SNN conversion learning is to make the SNNs have the same input-output mapping as the ANNs. Conversion learning avoids the problem of the weak expression ability of binary spike signals by approximating the spike sequence the real-valued output of ReLU, with which the inevitable conversion loss arises. A lot of works focus on reducing the conversion loss [Han *et al.*, 2020; Yan *et al.*, 2021] and achieve competitive performances. However, conversion learning ignores the effective TD information and needs a large number of timesteps to ensure accuracy. As a result, it is often limited to non-neuromorphic datasets and has a serious inference latency.

In recent years, direct supervised learning of SNNs has developed rapidly. From spatial back propagation [Lee *et al.*, 2016] to spatial-temporal back propagation [Wu *et al.*, 2018; Gu *et al.*, 2019; Fang *et al.*, 2020], people have realized the utilization of spatial and temporal information for training. On this basis, [Zheng *et al.*, 2021] realized the direct training of large-size networks and achieved state-of-the-art performance on the neuromorphic datasets. However, existing methods didn't solve the problem of the limited width of approximate derivative and weak expression ability of binary spike signals, which makes the direct training of deep SNNs inefficient. Gradient vanishing and network degradation seriously restrict directly-trained SNNs from going very deep, which is what we want to overcome.

Gradient vanishing or explosion   Gradient vanishing or explosion is the shared challenge of deep ANNs and deep SNNs. For deep ANNs, there are quite a few successful methods to address this problem. Batch normalization (BN) [Ioffe and Szegedy, 2015] reduces internal covariate shift to avoid gradient vanishing or explosion. The residual structure [He *et al.*, 2016] makes the gradient propagate across layers by introducing shortcut connection, which is one of the most widely used basic blocks in deep learning.

For directly-trained deep SNNs, existing research on the gradient vanishing or explosion problem is limited. It is worth noting that the threshold-dependent batch normalization (t-dBN) method proposed by [Zheng *et al.*, 2021] can adjust the firing rate and avoid gradient vanishing or explosion to some extent, which is helpful for our further research on gradient vanishing. On this basis, we will combat the gradient vanishing problem in SD caused by the limited width of the approximate derivative.

Deep network degradation   Network degradation will result in a worse performance of deeper networks than that of shallower networks. For deep ANNs, one of the most successful methods to solve degradation problem is residual structure [He *et al.*, 2016]. It introduces a shortcut connection to increase the identity mapping ability of the network and enable the networks to reach hundreds of layers without degradation greatly expanding the depth of the networks.

For directly-trained deep SNNs, there are few efforts on the degradation problem. Even if tdBN has explored the directly trained deep SNNs with residual structure and made SNNs go deeper, the degradation of deep SNNs is still serious. Our work will try to fill this gap in the field of SNNs.

3 Preliminaries

In this section, we review the spatio-temporal back propagation (STBP) [Wu *et al.*, 2018] and the iterative LIF model [Wu *et al.*, 2019] to introduce the foundation of our work.

STBP realizes error backpropagation in both TD and SD for the direct training of SNNs. On this basis, [Wu *et al.*, 2019] develops the iterative LIF model into an easy-to-program version and accelerates the direct training of SNNs. Considering the fully connected network, the forward process of the iterative LIF model can be described as

$$x_i^{t+1,n} = \sum_{j=1}^{l(n-1)} w_{ij}^n o_j^{t+1,n-1} + b_i^n, \qquad (1)$$

$$u_i^{t+1,n} = k_\tau u_i^{t,n}(1 - o_i^{t,n}) + x_i^{t+1,n}, \qquad (2)$$

$$o_i^{t+1,n} = f(u_i^{t+1,n} - V_{th}) = \begin{cases} 1, & u_i^{t+1,n} \geq V_{th} \\ 0, & u_i^{t+1,n} < V_{th} \end{cases}, \qquad (3)$$
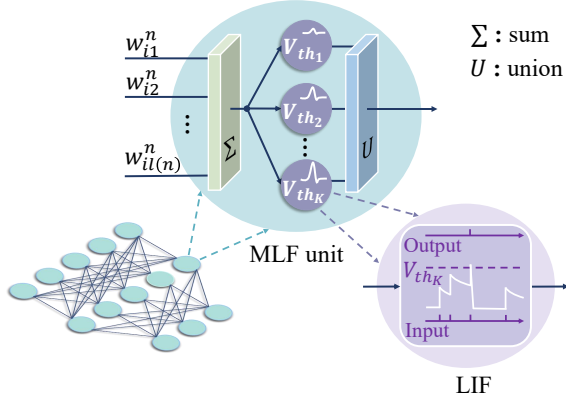
Figure 1: Illustration of MLF unit. A MLF unit contains multiple LIF neurons with different level thresholds. After receiving the input, these neurons will update the membrane potentials. Once the membrane potential of each level neuron reaches the corresponding threshold, a spike will be fired. The final output of MLF unit is the union of the spikes fired by all level neurons.

where k_τ is a decay factor. n and $l(n-1)$ denote the n-th layer and the number of neurons in the $(n-1)$-th layer respectively. t is time index. $u_i^{t,n}$ and $o_i^{t,n}$ are the membrane potential and the output of the i-th neuron in the n-th layer at time t respectively. $o_i^{t,n} \in (0, 1)$ is generated by the activation function $f(\cdot)$, which is the step function. V_{th} is the firing threshold. When the membrane potential exceeds the firing threshold, the neuron will fire a spike and the membrane potential is reset to zero. w_{ij}^n is the synaptic weight from the j-th neuron in the $(n-1)$-th layer to the i-th neuron in the n-th layer and b_i^n is the bias.

4 Method

4.1 The MLF Method

The Forward Process

As shown in Fig. 1, we replace LIF neurons with MLF units, which contain multiple LIF neurons with different level thresholds. The output is the union of all spikes fired by these neurons. The forward process can be described as

$$\mathbf{u}_i^{t+1,n} = k_\tau \mathbf{u}_i^{t,n} \odot (1 - \mathbf{o}_i^{t,n}) + x_i^{t+1,n}, \qquad (4)$$

$$\mathbf{o}_i^{t+1,n} = f(\mathbf{u}_i^{t+1,n} - \mathbf{V}_{th}), \qquad (5)$$

$$\hat{o}_i^{t+1,n} = s(\mathbf{o}_i^{t+1,n}), \qquad (6)$$

where $\mathbf{u}_i^{t,n} = (u_{i,1}^{t,n}, u_{i,2}^{t,n}, ..., u_{i,k}^{t,n}, ..., u_{i,K}^{t,n})$ and $\mathbf{o}_i^{t,n} = (o_{i,1}^{t,n}, o_{i,2}^{t,n}, .., o_{i,k}^{t,n}, ..., o_{i,K}^{t,n})$ denote the membrane potential vector and the output vector of the i-th MLF unit in the n-th layer at time t respectively. $\odot$ denotes the Hadamard product. k and K denote the k-th level and the number of levels respectively. $\mathbf{V}_{th} = (V_{th1}, V_{th2}, .., V_{thk}, ..., V_{thK})$ is the threshold vector. To facilitate the calculation of pre-synaptic input $x_i^{t,n}$, we define a spike encoder as $s(\mathbf{o}_i^{t,n}) = o_{i,1}^{t,n} + o_{i,2}^{t,n} + ... + o_{i,K}^{t,n}$, which is completely equivalent to union (see Appendix A). $\hat{o}_i^{t,n} = s(\mathbf{o}_i^{t,n})$ is the final output of the i-th MLF unit in the

n-th layer at time t. Then, $x_i^{t,n}$ can be computed by Eq. (1), where $o_i^{t,n}$ is replaced with $\hat{o}_i^{t,n}$.

Comparing Eq. (2)-(3) and Eq. (4)-(6), it can be seen that MLF unit doesn't introduce additional trainable parameters to the network, but just replaces LIF neurons with MLF units. Benefitting from the union of multiple spikes, MLF unit can distinguish some sharp features with large values and the non-sharp features with small values.

The Backward Process

To demonstrate that MLF can make the gradient propagation more efficient in SD, we next deduce the backward process of MLF method.

In order to obtain the gradients of weights and biases, we first derive the gradients of $o_{i,k}^{t,n}$, $\hat{o}_i^{t,n}$ and $u_{i,k}^{t,n}$, With L representing the loss function, the gradients $\partial L/\partial o_{i,k}^{t,n}$, $\partial L/\partial \hat{o}_i^{t,n}$ and $\partial L/\partial u_{i,k}^{t,n}$ can be computed by applying the chain rule as follows

$$\frac{\partial L}{\partial o_{i,k}^{t,n}} = \frac{\partial L}{\partial \hat{o}_i^{t,n}} + \frac{\partial L}{\partial u_{i,k}^{t+1,n}} u_{i,k}^{t,n}(-k_\tau), \qquad (7)$$

$$\frac{\partial L}{\partial \hat{o}_i^{t,n}} = \sum_{j=1}^{l(n+1)} \sum_{m=1}^{K} (\frac{\partial L}{\partial u_{j,m}^{t,n+1}} w_{ji}^n), \qquad (8)$$

$$\frac{\partial L}{\partial u_{i,k}^{t,n}} = \frac{\partial L}{\partial o_{i,k}^{t,n}} \frac{\partial o_{i,k}^{t,n}}{\partial u_{i,k}^{t,n}} + \frac{\partial L}{\partial u_{i,k}^{t+1,n}} k_\tau (1 - o_{i,k}^{t,n}), \qquad (9)$$

We can observe that gradient $\partial L/\partial o_{i,k}^{t,n}$ and $\partial L/\partial u_{i,k}^{t,n}$ come from two directions: SD (the left part in Eq. (7), (9)) and TD (the right part in Eq. (7), (9)). Gradient $\partial L/\partial \hat{o}_i^{t,n}$ comes from SD. Finally, we can obtain the gradients of weights $\mathbf{w}^n$ and biases $\mathbf{b}^n$ as follows

$$\frac{\partial L}{\partial \mathbf{w}^n} = \sum_{t=1}^{T} \sum_{k=1}^{K} \frac{\partial L}{\partial \mathbf{u}_k^{t,n}} \frac{\partial \mathbf{u}_k^{t,n}}{\partial \mathbf{w}^n} = \sum_{t=1}^{T} (\sum_{k=1}^{K} \frac{\partial L}{\partial \mathbf{u}_k^{t,n}}) \hat{\mathbf{o}}^{t,n-1^T}, \qquad (10)$$

$$\frac{\partial L}{\partial \mathbf{b}^n} = \sum_{t=1}^{T} \sum_{k=1}^{K} \frac{\partial L}{\partial \mathbf{u}_k^{t,n}} \frac{\partial \mathbf{u}_k^{t,n}}{\partial \mathbf{b}^n} = \sum_{t=1}^{T} (\sum_{k=1}^{K} \frac{\partial L}{\partial \mathbf{u}_k^{t,n}}), \qquad (11)$$

where T is the number of timesteps. Due to the non-differentiable property of spiking activity, $\partial o_k/\partial u_k$ cannot be derived. To solve this problem, we adopt the rectangular function $h_k(u_k)$ [Wu *et al.*, 2018] to approximate the derivative of spike activity, which is defined by

$$\frac{\partial o_k}{\partial u_k} \approx h_k(u_k) = \frac{1}{a} sign(|u_k - V_{thk}| < \frac{a}{2}), \qquad (12)$$

where a is the width parameter of the rectangular function.

Considering the gradient propagation in SD from n-th layer to $(n-1)$-th layer, the spatial propagation link can be described as: $\partial L/\partial \hat{o}_i^{t,n} \rightarrow (\partial L/\partial o_{i,1}^{t,n}, ..., \partial L/\partial o_{i,K}^{t,n}) \rightarrow (\partial L/\partial u_{i,1}^{t,n}, ..., \partial L/\partial u_{i,K}^{t,n}) \rightarrow \partial L/\partial \hat{o}_i^{t,n-1}$. If it is only one-level firing ($K = 1$), the model will become the standard STBP model. In this case, numerous neurons will fall into the saturation area outside the rectangular area, some of which
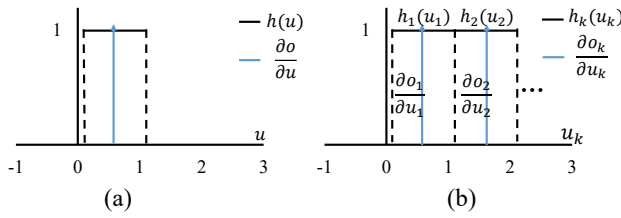
Figure 2: (a) The distribution of approximate derivative of one-level firing. (b) The distribution of approximate derivatives of multi-level firing.
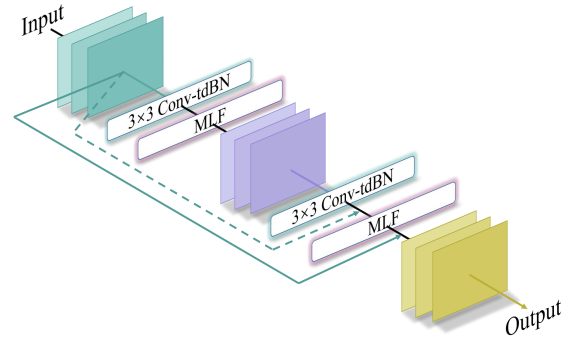

Figure 3: Illustration of spiking ResNet and spiking DS-ResNet. The dotted line connection represents spiking ResNet, and the solid line connection represents spiking DS-ResNet

will become dormant unit, shown in Fig. 2(a), and the corresponding $\partial o_{i,1}^{t,n}/\partial u_{i,1}^{t,n}$ will be zero due to the limited width of the approximate derivative. Consequently, most of $\partial L/\partial u_{i,1}^{t,n}$ will lose the gradients propagated from SD, and the spatial propagation links from n-th layer to $(n-1)$-th layer will be broken, which makes the gradient propagation blocked in SD.

In the case of multi-level firing ($K > 1$) and non-overlapping distribution of each level $h_k(u_k)$ where $V_{thk+1} - V_{thk} = a$, MLF units are less likely to fall into the dormant state after receiving inputs because of the wider non-zero area, as shown in Fig. 2(b). As a result, MLF can guarantee efficient gradient propagation through the spatial propagation links unless all levels of u_k fall into the areas outside the corresponding rectangular areas, and the adjustment of $\mathbf{w}^n$ and $\mathbf{b}^n$ can be accelerated during the training process.

The pseudo code for the overall training of the forward and backward process is shown in Appendix B.

4.2 Dormant-Suppressed Residual Network

Residual network (ResNet) [He *et al.*, 2016], as one of the most widely used basic blocks, has achieved great success in deep networks. To convert ResNet into spiking ResNet, we replace BN and ReLU by tdBN [Zheng *et al.*, 2021] and MLF units respectively, where tdBN is used to coordinate distribution difference and normalize the input distribution to $N(0, (V_{th_1})^2)$. In spiking ResNet, MLF activation is after the addition of the shortcut connection, as shown by the dotted line in Fig. 3. The addition will increase the values of the feature map before activation. In SNNs, the increase of the values will make inputs exceed the right side of the rectangular area in Fig. 2 resulting in more dormant units. Besides, due to the discrete property of the activation function, the shortcut connection cannot perform identity mapping well. Therefore, spiking ResNet still suffers from network degradation, which prevents the directly trained SNNs from going deeper.

To address the above two issues, we replace activation after addition structure with activation before addition structure and propose the spiking dormant-suppressed residual network (spiking DS-ResNet), as shown by the solid line in Fig. 3. Spiking DS-ResNet can reduce the proportion of dormant units at the output of each block, enabling more efficient gradient propagation and more efficient identity mapping of discrete spikes in very deep SNNs. We formalize this improvement in theorem 1.

Theorem 1. *Considering the values of feature map normalized by tdBN satisfy $x \sim N(0, (V_{th_1})^2)$, the probabilities of*

a spike from shortcut connection leading to dormant unit in spiking DS-ResNet and spiking ResNet are P^* and P respectively, and the abilities of identity mapping of spiking DS-ResNet and spiking ResNet are I^* and I respectively, then we have $P^* < P$ and $I^* > I$.

Proof. The proof of Theorem 1 is presented in Appendix C.
□

In this way, dormant units can be effectively suppressed and the shortcut connection can retain the ability of identity mapping, so as to further solve the degradation problem.

5 Experiments

The source code of our MLF and spiking DS-ResNet implementation[1] is available online.

5.1 Experimental Settings

The basic network architecture of our experiments is ResNet. The first layer is 3×3 convolutions as the encoding layer. Similar to ResNet for CIFAR10 in [He *et al.*, 2016], we start stacking residual block, which contains 2 convolution layers, to $6N$ layers with $2N$ layers for each feature map size. The number of channels will be doubled if the feature map is halved. We set three different initial channels (small, middle, large) for the first layer, the numbers of which are 16, 32, 64 respectively. ResNet ends with global average pooling and a 10/11 fully-connected classifier. The total layers of our ResNet are $6N + 2$. More detailed network architecture and other experimental settings are summarized in Appendix D.

5.2 Classification Accuracy

CIFAR10 For CIFAR10, we apply spiking DS-ResNet (20-layer, large) to evaluate the average performance of our model in 4 timesteps. The level of MLF is set to 3. The performance is averaged over 5 runs.

Table 1 shows the comparison of our results and existing state-of-the-art results on CIFAR10. We notice that [Yan *et al.*, 2021] reported a competitive accuracy on CIFAR10, whereas it is a conversion-based method and requires a large

---

[1]https://github.com/langfengQ/MLF-DSResNet.

Model	Method	CIFAR10		DVS-Gesture		CIFAR10-DVS	
		Acc.	Params	Acc.	Params	Acc.	Params
[Amir *et al.*, 2017]	TrueNorth	-	-	94.59	18.99M	-	-
[Sengupta *et al.*, 2019]	ANN-SNN	91.55	33.63M	-	-	-	-
[Wu *et al.*, 2019]	STBP	90.53	44.99M	-	-	60.50	26.82M
[He *et al.*, 2020]	STBP	-	-	93.40	2.32M	-	-
[Kugele *et al.*, 2020]	ANN-SNN	-	-	95.56	0.80M	65.61	0.50M
[Lee *et al.*, 2020]	Spike-based BP	90.95	18.20M	-	-	-	-
[Wu *et al.*, 2021]	ASF-BP	91.35	8.81M	-	-	62.50	26.78M
[Zheng *et al.*, 2021]	STBP-tdBN	93.16	15.10M	96.87	3.50M	67.80	16.27M
[Yan *et al.*, 2021]	ANN-SNN	94.16	9.33M	-	-	-	-
Our model	MLF ($K = 3$) + spiking DS-ResNet	94.25	4.32M	97.29	0.27M	70.36	0.69M

Table 1: Comparison of different methods on three datasets. The unit of Acc. is %.

number of timesteps to ensure encoding resolution. Our model achieves state-of-the-art performance (94.25%) with fewer timesteps and fewer trainable parameters.

DVS-Gesture DVS-Gesture [Amir *et al.*, 2017] is a neuromorphic vision dataset with more temporal information. We apply spiking DS-ResNet (20-layer, small) to evaluate the average performance of our model with MLF levels of 3. The performance is averaged over 5 runs.

As we can see from Table 1, [Zheng *et al.*, 2021] have better performance on DVS-Gesture with directly-trained SNNs compared with [He *et al.*, 2020]. The reason is that the former succeeded in the direct training of large-size and deep SNNs. However, they have to apply a large network structure to get a competitive accuracy, due to a large number of dormant units and the weak expression ability of neurons. We achieve more efficient training and get state-of-the-art performance (97.29%) with a smaller network structure, the initial channel number of which is only 16.

CIFAR10-DVS CIFAR10-DVS [Li *et al.*, 2017] is a more challenging and easy-overfitted neuromorphic vision dataset. We apply spiking DS-ResNet (14-layer, middle) to evaluate the average performance of our model with MLF levels of 3. The performance is averaged over 5 runs.

As indicated in Table 1, [Wu *et al.*, 2021] ignored the information in TD when training SNNs, resulting in their accuracy not being as good as [Zheng *et al.*, 2021], which demonstrates the advantages of the spike-based direct training method in dealing with spatio-temporal information. On this basis, our model further improves the performances of directly trained SNNs and gets state-of-the-art performance (70.36%) with much smaller network architecture.

5.3 Analysis of Our Model

Effects of Level *K* and Spiking DS-ResNet

The performance improvement of our model benefits from two aspects: MLF and spiking DS-ResNet. We design experiments to explicitly analyze the effects of the level of MLF and spiking DS-ResNet. We apply ResNet (20-layer, large) as basic architecture on CIFAR10 in 4 timesteps with various methods.

Method	Acc. (%)
Spiking ResNet($K = 1$)	92.55
ResNet-SNN($K = 1$)	93.04
Spiking DS-ResNet($K = 1$)	93.54
Spiking DS-ResNet($K = 2$)	94.13
Spiking DS-ResNet($K = 3$)	94.25

Table 2: Effects of Level K and Spiking DS-ResNet.

As shown in Table 2, ResNet-SNN [Zheng *et al.*, 2021] has better performance than spiking ResNet for the reason that ResNet-SNN can efficiently control the generation of the dormant units by adding tdBN to every shortcut connection. However, it cannot efficiently perform identity mapping and will increase the number of parameters. Our spiking DS-ResNet can take both dormant unit and identity mapping into account, therefore having better performance. Besides, MLF shows prominent advantages in the improvement of accuracies, and a larger K will benefit the performance progressively. While, the improvement from $K = 1$ to $K = 2$ is greater than that from $K = 2$ to $K = 3$ for the reason that the data falling into $h_2(u_2)$ is much more than the data falling into $h_3(u_3)$ considering that the input distribution satisfies $N(0, (V_{th_1})^2)$. In our experiments, MLF with level 3 is enough to cover most of the sharp features (see Appendix E).

MLF for Gradient Vanishing

In this part, we conduct experiments to demonstrate that MLF can effectively alleviate the blocking of gradient propagation in SD. We apply the spiking ResNet (20-layer, middle) on CIFAR10 with levels of 1, 2, 3 in 4 timesteps.

As indicated in Fig. 4(a), there is a large proportion of dormant units during the training process when $K = 1$ (without MLF), which will lead to the gradient vanishing and prevent further enhancement of sharp features. While, MLF ($K > 1$) has prominent advantages in reducing the proportion of dormant units, and the gradient vanishing can be alleviated. In this way, MLF can increase the gradient of trainable parameters (see Appendix F), accelerate the convergence speed of network training, and significantly improve the performance
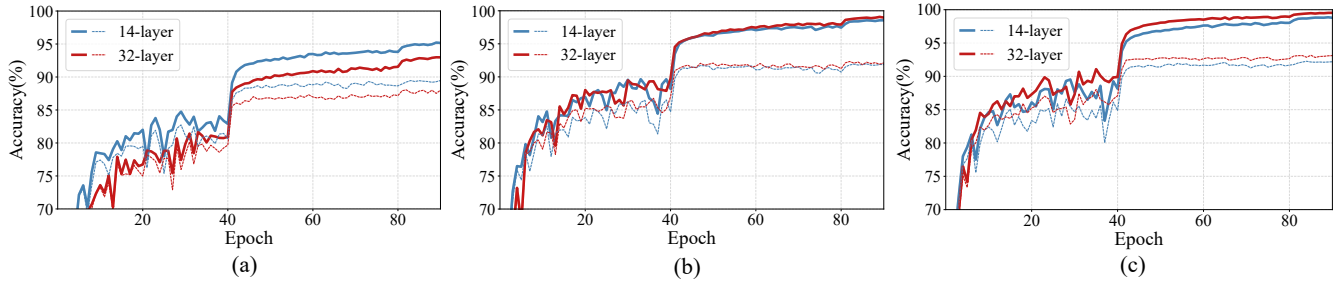
Figure 5: Training on CIFAR10. Thin dotted curves denote testing accuracy, and bold solid curves denote training accuracy. (a) ResNet-SNN
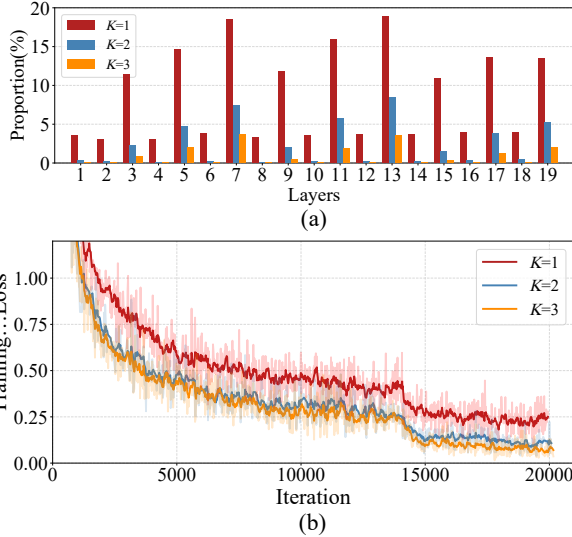

Figure 4: (a) The average proportion of dormant units in each layer. (b) The training loss during the whole training process.

Layer	14-layer	20-layer	32-layer	44-layer	68-layer
Acc.(%)	92.46	92.95	93.34	93.42	93.48

Table 3: Accuracy on CIFAR10 with spiking DS-ResNet and MLF.

Moreover, we explore deeper networks to test our model on CIFAR10, and we achieve very deep SNNs (68-layer) without degradation, which validates that our model can efficiently solve the degradation problem in deep SNNs. The testing accuracy is summarized in Table 3.

6 Conclusion

In this paper, we have proposed the MLF method based on STBP and spiking DS-ResNet for direct training of deep SNNs to combat the gradient vanishing and network degradation caused by the limitation of the binary and non-differentiable properties of spike activities. We prove that MLF can expand the non-zero area of the approximate derivatives by allocating the coverage of the approximate derivative of each level, so as to reduce dormant units and make the gradient propagation more efficient in deep SNNs. Besides, an MLF unit can generate spikes with different thresholds when activating the input, which can improve its expression ability. Spiking DS-ResNet can reduce the probability of dormant unit generation making it more suitable for gradient propagation and can efficiently perform identity mapping of discrete spikes in very deep SNNs. With MLF and spiking DS-ResNet, our model achieves state-of-the-art performances with fewer parameters on both non-neuromorphic and neuromorphic datasets compared with other SNN models and makes SNNs go very deep without degradation. In summary, this paper provides an efficient solution to gradient vanishing and network degradation in the field of the directly trained SNNs, enabling SNNs to go deeper with high performance.

of directly trained SNNs, as shown in Fig. 4(b).

More analysis of gradient vanishing is presented in Appendix G.

Going Deeper

In directly trained deep SNNs, the serious degradation problem greatly restricts SNNs to a shallow. In this part, we take ResNet (*x*-layer, middle) as basic architecture to conduct multiple experiments on very deep SNNs to demonstrate the ability of our model to solve the degradation problem. We train various models and take the reported deepest network architecture (ResNet-SNN) as the baseline. The level of MLF is set to 3. We record the training and testing accuracy of the whole training process.

As depicted in Fig. 5(a), ResNet-SNN without MLF have experienced serious network degradation problem only at 32-layer (the deeper 32-layer has lower training/testing accuracy than the shallow 14-layer). After introducing MLF, the degradation problem is alleviated, as shown in Fig. 5(b). At the same time, MLF effectively improves the training/testing accuracy of both 14-layer and 32-layer. Finally, with spiking DS-ResNet and MLF, the degradation problem is further solved, as shown in Fig. 5(c).

References

[Amir *et al.*, 2017] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra Modha. A low power, fully event-based gesture recognition system. In *CVPR*, pages 7243–7252, 2017.

[Diehl and Cook, 2015] Peter U. Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9:99, 2015.

[Fang *et al.*, 2020] Haowen Fang, Amar Shrestha, Ziyi Zhao, and Qinru Qiu. Exploiting neuron and synapse filter dynamics in spatial temporal learning of deep spiking neural network. In *IJCAI*, pages 2799–2806, 2020.

[Gu *et al.*, 2019] Pengjie Gu, Rong Xiao, Gang Pan, and Huajin Tang. STCA: Spatio-temporal credit assignment with delayed feedback in deep spiking neural networks. In *IJCAI*, pages 1366–1372, 2019.

[Han *et al.*, 2020] 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 *CVPR*, volume 1, pages 13555–13564, 2020.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[He *et al.*, 2020] Weihua He, YuJie Wu, Lei Deng, Guoqi Li, Haoyu Wang, Yang Tian, Wei Ding, Wenhui Wang, and Yuan Xie. Comparing SNNs and RNNs on neuromorphic vision datasets: similarities and differences. *Neural Networks*, 132:108–120, 2020.

[Hu *et al.*, 2021] Yangfan Hu, Huajin Tang, and Gang Pan. Spiking deep residual networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–6, 2021.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[Kugele *et al.*, 2020] Alexander Kugele, Thomas Pfeil, Michael Pfeiffer, and Elisabetta Chicca. Efficient processing of spatio-temporal data streams with spiking neural networks. *Frontiers in Neuroscience*, 14:439, 2020.

[Lee *et al.*, 2016] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10:508, 2016.

[Lee *et al.*, 2020] Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14:119, 2020.

[Li *et al.*, 2017] 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.

[Liu *et al.*, 2020] Qianhui Liu, Gang Pan, Haibo Ruan, Dong Xing, Qi Xu, and Huajin Tang. Unsupervised aer object recognition based on multiscale spatio-temporal features and spiking neurons. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12):5300–5311, 2020.

[Liu *et al.*, 2022] Qianhui Liu, Dong Xing, Lang Feng, Huajin Tang, and Gang Pan. Event-based multimodal spiking neural network with attention mechanism. In *ICASSP*, pages 8922–8926, 2022.

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

[Neftci *et al.*, 2019] 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.

[Qi *et al.*, 2018] Yu Qi, Jiangrong Shen, Yueming Wang, Huajin Tang, Hang Yu, Zhaohui Wu, and Gang Pan. Jointly learning network connections and link weights in spiking neural networks. In *IJCAI*, pages 1597–1603, 2018.

[Sengupta *et al.*, 2019] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: VGG and residual architectures. *Frontiers in Neuroscience*, 13:95, 2019.

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

[Wu *et al.*, 2019] Yujie Wu, Lei Deng, Lei Deng, Jun Zhu, Yuan Xie, and Yuan Xie. Direct training for spiking neural networks: Faster, larger, better. In *AAAI*, volume 33, pages 1311–1318, 2019.

[Wu *et al.*, 2021] Hao Wu, Yueyi Zhang, Wenming Weng, Yongting Zhang, Zhiwei Xiong, Zheng-Jun Zha, Xiaoyan Sun, and Feng Wu. Training spiking neural networks with accumulated spiking flow. In *AAAI*, volume 35, pages 10320–10328, 2021.

[Yan *et al.*, 2021] Zhanglu Yan, Jun Zhou, and Weng-Fai Wong. Near lossless transfer learning for spiking neural networks. In *AAAI*, volume 35, pages 10577–10584, 2021.

[Zheng *et al.*, 2021] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *AAAI*, volume 35, pages 11062–11070, 2021.