GOING DEEPER WITH SPIKING NEURONS: TOWARDS BINARY OUTPUTS OF DEEP LOGIC SPIKING NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

For the simulation of spikes in biological neurons, a natural fit is the spiking neural networks, which produce binary outputs from spiking neurons. SNN receives arising investigations for its high biological plausibility and efficient inference on neuromorphic chips. However, it is still a challenge to train SNNs with more than 50 layers due to the gradient vanishing problem caused by the spiking neuron layers, which greatly prevents SNNs from going deeper to obtain higher performance. In this paper, we first investigate the variants of spiking residual blocks and find that deep SNNs with binary outputs can not be constructed by simply replacing the activation function in the existing residual structure in ANN with the spiking neuron layer. We thus propose a logic spiking network (LSN) to benefit deep SNN training. Our LSN contains two functionally distinct branches, a structure inspired by excitatory and inhibitory pathways followed by a logical operation for binary spike outputs. We demonstrate both theoretically and experimentally that LSN can implement identity mapping as well as overcome the vanishing gradient problem. Our LSN can be expanded by more than 100 layers with binary outputs and performs favorably against existing spiking ResNet and its variants. Our proposed LSN achieved 94.68% accuracy on CIFAR10, 71.86% accuracy on ImageNet, and 75.1% accuracy on CIFAR10-DVS.

1 INTRODUCTION

Spiking neural networks (SNNs) are inspired by biological mechanisms and are considered as the third generation of artificial neural networks (Maass, 1997). Unlike using continuous floating-point activation values in Artificial Neural Networks (ANNs), the biological neural system transmits information through discrete spikes. SNNs employ spiking neurons with a firing threshold to transform the large activity as spikes, simulating the activity of biological neurons. As they incorporate both time and spikes, SNNs are more biologically plausible and energy efficient, which have been verified on neuromorphic chips, including TrueNorth, Loihi, and TianJic (Akopyan et al., 2015; Davies et al., 2018; Pei et al., 2019).

In parallel, the success of ANNs largely depends on deep learning (LeCun et al., 2015), since deeper networks can better extract the input features (Simonyan & Zisserman, 2014; Szegedy et al., 2015). But as the network depth increases, accuracy gets saturated and then degrades rapidly (He & Sun, 2015). Residual connections are used in ANN to solve the problem of network degradation (He et al., 2016a). In order to build deep SNNs, spiking ResNet simply replace ReLU in the ResNet of the ANN with spiking neurons (Zheng et al., 2021; Kim et al., 2018; Hwang et al., 2021). However, owing to the gradient vanishing problem caused by the spiking neuron layers, spiking ResNet is unable to address the issue of degradation, and the neurons in the deep layers hardly fire (Cox & Dean, 2014). To solve the dilemma of training deep SNNs, Zheng et al. (2021) extends batch normalization to the time dimension and successfully trains 34-layer and 50-layer spiking ResNet. Quite a few studies adopt the pre-activation residual structure to avoid the vanishing gradient caused by the spiking neuron layer and successfully expand the network to 100 layers, but make the output of the network contain positive integers or floating-point values, which loss the event-driven advantage of SNNs (Fang et al., 2021; Feng et al., 2022).

In this study, We review the evolution of the residual connections in ANNs and find that a deep SNN with binary output can not be constructed by simply replacing the activation function in the existing residual structure in ANNs with the spiking neuron layer. Specifically, we show that one branch with a skip connection is unable to construct a deep SNN with binary outputs. Inspired by the fact that the biological neural system contains two kinds of functionally distinct pathways, i.e., excitatory pathways and inhibitory pathways (Luo, 2021), we propose a logic spiking neural network (LSN), which contains two functionally distinct branches to mimic excitatory and inhibitory pathways. Then we explain the reciprocal transformational mechanism between 0 and 1 in the network and how to implement the required transformational mechanism through logical operations. Moreover, our theoretical derivation proves that it can both implement identity mapping of discrete spikes and address the issue of vanishing gradient. The experimental results are consistent with our analysis, showing that LSN can achieve better performance as its depth increase and outperform other networks such as spiking ResNet and its variants.

Our contribution can be summarized in the following three points:

- We design a novel structure that contains two branches inspired by excitatory pathways and inhibitory pathways in biological system. At the end of each block, the logic operation is used to replace the arithmetic operation commonly used before to obtain binary spikes, which makes better use of the information contained in whether the spike is fired or not, rather than only the numerical information contained in 0 and 1.

- To the best of our knowledge, we are the first to train an SNN with binary output and more than 100 layers, which relieves the degradation and silence problems of deep SNNs.

- The network we proposed is more biology-plausible and gets state-of-the-art (SOTA) accuracy, including 94.68% on CIFAR10, 71.86% on ImageNet, and 75.1% on CIFAR10-DVS with fewer simulation steps T.

2 Related work

2.1 Learning algorithm of deep SNNs

Generally, there are mainly two methods to obtain a high-performance SNN: ANN to SNN conversion method (ANN2SNN) and back-propagation based on surrogate gradient. Local self-supervised learning methods such as spike timing dependent plasticity (STDP) (Bi & Poo, 1998; Mozafari et al., 2018; Kheradpisheh et al., 2018) and Hebbian learning (Hebb, 2005) can only train shallow networks, usually no more than five layers. And the performance is far below the above two methods.

ANN2SNN This method first trains a corresponding ANN, and then replaces its activation function with spiking neurons (Deng & Gu, 2021; Han & Roy, 2020; Han et al., 2020; Kim et al., 2020b; Sengupta et al., 2019; Woźniak et al., 2020). Generally, scaling operations of network parameters and neuron firing thresholds are required. This method essentially uses the firing rates of neurons to approximate the floating-point outputs in ANN. Therefore, a very large number of simulation steps T is required to obtain high accuracy, which makes the SNN more energy-consuming. In addition, the converted SNN is restricted to rate-coding, which losses its temporal dynamics.

Surrogate Gradient The SNN obtained by direct training usually uses surrogate gradients during back-propagation (Deng et al., 2022; Perez-Nieves & Goodman, 2021; Guo et al., 2022; Kim et al., 2020a). The directly trained SNNs require fewer time steps than ANN2SNN. Although the accuracy of the directly trained SNN is still lower than ANN2SNN in large datasets like ImageNet, it has been significantly improved in recent years (Fang et al., 2021; Li et al., 2021). And Sharmin et al. (2019) have argued that the SNN obtained by ANN2SNN is inferior to the SNN obtained by direct training in adversarial robustness and biological similarity. The directly trained SNN is more suitable for application in neuromorphic datasets and is not limited to rate-coding, which makes the SNNs more brain-like.

2.2 GOING DEEPER WITH SNNS

Going deeper with residual connections in ANNs Residual connection is the most successful method to solve the degradation problem in ANNs. ResNet meets two requirements to address the degradation problem successfully: (i) Residual learning: He et al. (2016a) hypothesizes that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. What the branch structure has learned ought to be the discrepancy between the required features and the input features. (ii) Identity mapping: the output should be equal to the input when the branch output is zero.

Spiking ResNet suffers from vanishing gradient due to spiking neurons layers In SNNs, spiking ResNet (Zheng et al., 2021; Deng et al., 2022; Li et al., 2021) directly replaced the activation layer of ResNet in ANN with the spiking neuron layer but still suffers from the degradation problem. The ReLU function is commonly used in ANN, during the back-propagation process, if the input is greater than 0, the gradient is always 1, so the problem of gradient disappearance due to the activation function layer will not occur(shown in Fig. 1). For the spiking ResNet(shown in Fig. 3(a)), there is a neuron layer at the end of each residual block. For example, when the derivative function of the Sigmoid $S(x) = \frac{1}{1+e^{-\alpha x}}$, $\alpha = 1$ function is utilized as a surrogate gradient, during the back-propagation, each time the gradient passes through the last neuron layer in the residual block the gradient will decay to less than one-quarter of the previous thus there is a serious problem of gradient disappearance,

$$\text{Gradient}_{\text{after neuron layer}} \leqslant 0.25 \times \text{Gradient}_{\text{before neuron layer}} \tag{1}$$

Although the maximum gradient value of the neuron layer in the back-propagation process can be adjusted by tuning the parameters of the surrogate gradient function since the integral of the surrogate function is 1, Fig. 1 shows the adjustment of the parameters is only the adjustment of the gradient distribution and the too sharp surrogate gradient will lead the network to fail to converge. Consequently, there is no way to address the issue by adjusting the surrogate function. Spiking ResNet also has the problem that it cannot achieve identity mapping for complex neurons such as LIF (Fang et al., 2021). At present, when the number of layers of spiking ResNet is increased from 34 to 50, the accuracy is only improved by 1.16% (Zheng et al., 2021), while networks with more than 50 layers can not converge. Therefore, most current studies use spiking ResNet18 or ResNet34 as the backbone (Li et al., 2021; Guo et al., 2022), which greatly limits the performance of directly trained SNNs.

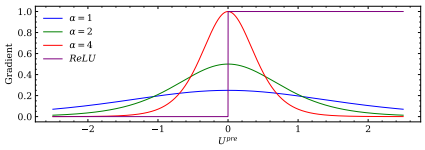


Figure 1: Derivative functions of ReLU and Sigmoid.

Spiking ResNet variants can not go deeper with binary outputs Some previous work (Zheng et al., 2021; Fang et al., 2021; Feng et al., 2022) attempts for residual structure in SNNs are based on the other residual structure in ANN. We firstly review the evolution of the residual connection in ANN, showing simply using the spiking neuron layer to replace the activation layer in ResNet or its variants can not obtain a deep SNN with binary output. Fig. 2 illustrates the evolution of residual structure in ANN. After the original ResNet (shown in Fig. 2(a)) was proposed, He et al. (2016b) attempted not to change its components but changed its order, resulting in a total of five kinds of residual connection structures. Fig. 2(b) adopts the structure that BN after ADD, but this structure destroys the idea of identity mapping. So ReLU before addition, ReLU-only pre-activation, and full pre-activation are proposed, which adopt the Pre-activation structure and do not use any extra operations in the shortcut skips. The Pre-activation structure is widely used in most deep networks and is not restricted to the convolutional network, which was used when the original residual structure was proposed, and it has also been extensively applied to Transformer (Vaswani et al., 2017) structures such as BERT (Devlin et al., 2018) and ViT (Dosovitskiy et al., 2020). Quite a few studies use the pre-activation structure to avoid the disappearance of the gradient disappearance caused by the neuron layer.

For ReLU before addition, the ReLU layer is placed last. Hence the output of the branch structure is always non-negative, which limits the fitting ability of the network and makes the output of the network infinitely increase with the depth increase. Fang et al. (2021) uses this structure, so it suffers the same question and the network output contains positive integers. For full pre-activation,

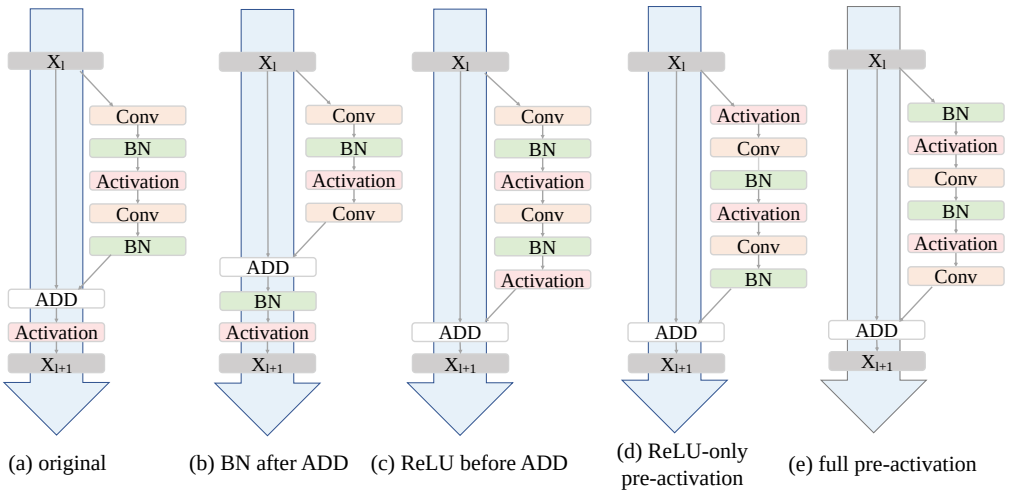


Figure 2: The evolution of residual connection, Activation in ANN refers to ReLU, and Activation in SNN refers to spiking neuron.

Feng et al. (2022) uses this uses this structure. Since the last layer in the branch structure is the convolutional layer, the output is the floating-point value and loses the event-driven advantage of SNN. Therefore, as shown in Table. 1, a deep SNN with binary output can not be constructed by simply replacing the activation function in the existing residual structure in ANN with the spiking neuron layer.

Table 1: Five Residual Structures Applied on ANN and SNN.

Case	**Drawbacks**	**Performance compared to original structure**	**Applied on SNN**	
			Gradient disappearse	**Output**
original	-	-	unsolved	binary
BN after ADD	Unable to implement identity	worse	unsolved	binary
ReLU before ADD	branch output is non-negative	worse	resolved	zero and positive integers
ReLU-only pre-activation	-	worse	resolved	float-point value
full pre-activation	-	better	resolved	float-point value

3 Proposed Method

3.1 Neuron model

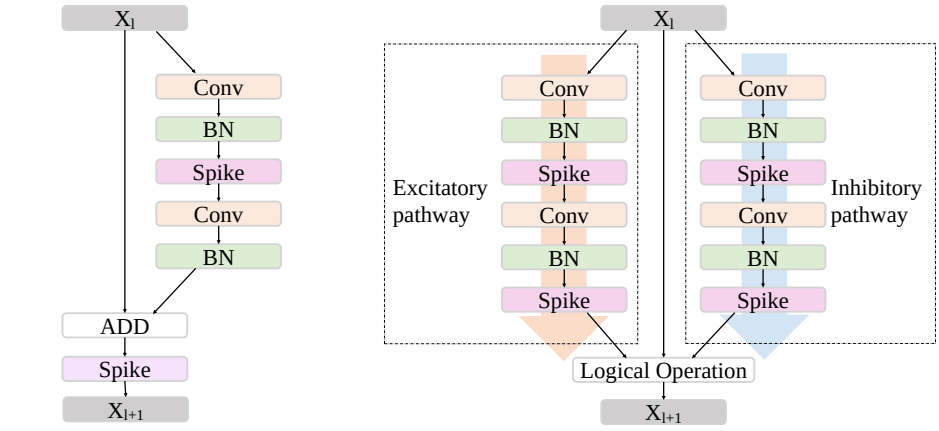
At present, the biggest difference between SNN and ANN lies in the neuron layer of SNN. We utilize Leaky Integrate-and-Fire (LIF) neurons here. U denotes the membrane potential, the post-synaptic potential is updated as,

$$\hat{U}^{t+1} = \alpha U^t + W * O^{t+1}, \tag{2}$$

where $\hat{U}^{t+1}$ denotes the membrane potential before the neuron fire mechanism at time-step $t+1$ and U^t denotes the membrane potential after the neuron fire mechanism at time-step t, α is a constant leaky factor of membrane potential between 0 and 1, W means the weight of inputs. $\hat{U}^{t+1}$ depends on the attenuated value of U^t in the time domain and on the spikes of neurons in the previous layer at the current time-step $t+1$ in the space domain, denoted as O^{t+1}. If $\hat{U}^{t+1}$ exceeds the firing threshold V_{th}, the neuron will fire a spike, and the output Y^{t+1} of neurons at time-step $t+1$ is 1, otherwise it is 0,

$$Y^{t+1} = \Theta(\hat{U}^{t+1} - V_{th}), \tag{3}$$

where U^{t+1} is the membrane voltage after the spike generation mechanism at time-step $t+1$. $\Theta(x)$ is the Heaviside step function. The membrane voltage after the neuron fires is reset to zero. The



(a) Traditional Spiking ResNet

(b) Logical Spiking Network

Figure 3: The basic block of traditional Spiking ResNet and Logical Spiking Network .

membrane potential after the firing mechanism U^{t+1} where is defined as below.

$$U^{t+1} = (1 - Y^{t+1}) \circ \hat{U}^{t+1}. \tag{4}$$

where $\circ$ means Hadamard product, i.e, bitwise multiplication. There is a membrane potential attenuation mechanism of LIF neurons, reflected in equation 2 as α , which makes the LIF neuron more biology-plausible than the IF neuron.

3.2 Logical Spiking Network

By reviewing the spiking ResNet and its variants, we found that the Pre-activation structure can avoid the problem of vanishing gradient in SNNs, but also introduce non-binary outputs, indicating one branch with a skip connection is unable to construct a deep SNN with binary outputs. To overcome the limitations, here we propose the Logical Spiking Network (LSN), a deep SNN with binary outputs, which contains two functionally different branches, and shows its transformation mechanism inspired by excitatory pathways and inhibitory pathways can both implement identity mapping and overcome the vanishing gradient problem caused by the spiking neuron layer.

Network structure inspired by excitatory and inhibitory pathways There are two kinds of pathways in the biological neural system. The first one is the excitatory pathway. There is a large number of this pathway, but their types are only a few. The second one is the inhibitory pathways. There is a limited number of pathways, but their types are diverse (Luo, 2021). The receptors activated by excitatory synaptic transmitters such as glutamate released by excitatory pathways can increase the firing rate of postsynaptic neurons, while receptors activated by inhibitory synaptic transmitters such as GABA released by inhibitory pathways can reduce the firing rate of postsynaptic neurons. Inspired by that, we construct the network according to the working principle of neural neurons. Fig. 3(b) shows the basic block of the network.

The left part corresponds to the excitatory pathways, and the right part corresponds to the inhibitory pathways. All conditions are shown in the Fig 4, where $\mathcal{E}$ and $\mathcal{I}$ denote the excitatory pathways output and the inhibitory pathways output, $\mathcal{X}_l$ and $\mathcal{X}_{l+1}$ denote the input and output of layer l.

Reciprocal transformation of 0 and 1 Excitatory pathways tend to increase output and inhibitory pathways tend to decrease output. When the neurons in the previous layer do not fire, i.e., the input is 0, and if the excitatory pathways fire spikes, the neurons that originally do not fire spikes are stimulated to fire spikes, and the output changes from 0 to 1. When the neurons in the previous layer fire spikes, i.e., when the input is 1 and the inhibitory pathways fire spikes, the neurons that originally fire spikes are inhibited and do not fire spikes, and the output changes from 1 to 0.

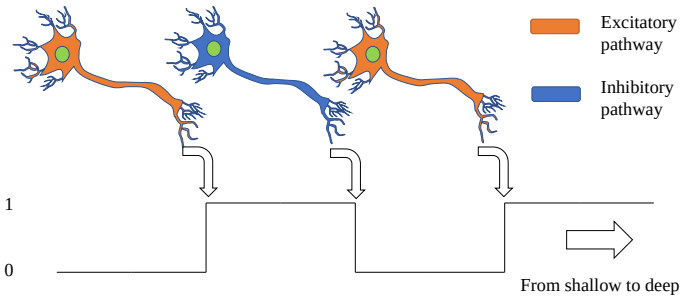


Figure 4: Excitatory and inhibitory pathways transform inputs in a more brain-like manner.

$\mathcal{X}_l$	$\mathcal{E}$	$\mathcal{I}$	$\mathcal{X}_{l+1}$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Table 2: Input and output of logical operation.

Identity mapping Excitatory and inhibitory pathways play opposite roles. When excitatory pathways and inhibitory pathways fire spikes simultaneously or stay silent simultaneously, the output remains unchanged, which is equivalent to identity mapping. For the previous layer, if the input has already been 0 when the excitatory pathways do not fire and the inhibitory pathways fire, the output is still 0. The nerve neurons rely on spikes to transmit information. When neurons that do not fire are received by inhibitory stimulation, they will still keep silent, so the output remains unchanged and remains 0, and vice versa. In this way, the network can realize the transformation from 0 to 1, from 1 to 0, and identity mapping.

Logical operation Then we use the logical operation to fulfill the required change process. The input and output of logical operation are 0 or 1, which can make better use of the characteristics of SNN and achieve the above transformation relationship. Different from the floating-point numbers output in ANN, the binary output of SNN makes logical operation possible, and the logical operation makes the SNN output remain binary. The specific implementation can be seen as the addition of the minimum term in the truth table where each output $\mathcal{X}_{l+1}$ is 1. As shown in equation 5, where $\circ$ means Hadamard product, i.e, bitwise multiplication.

$$\mathcal{X}_{l+1} = (1-\mathcal{X}_l)\circ\mathcal{E}\circ(1-\mathcal{I}) + \mathcal{X}_l\circ(1-\mathcal{E})\circ(1-\mathcal{I}) + \mathcal{X}_l\circ\mathcal{E}\circ(1-\mathcal{I}) + \mathcal{X}_l\circ\mathcal{E}\circ\mathcal{I} \tag{5}$$

After simplification we can get:

$$\mathcal{X}_{l+1} = (1-\mathcal{X}_l)\circ\mathcal{E}\circ(1-\mathcal{I}) + \mathcal{X}_l\circ(1-\mathcal{E})\circ(1-\mathcal{I}) + \mathcal{X}_l\circ\mathcal{E} \tag{6}$$

And it can be easily deployed on the hardware platform. Only three state gates are needed to achieve the required logic operations, as shown in equation 7, where $\neg, \wedge, \vee$ denotes logical operators NOT, AND, OR.

$$\mathcal{X}_{l+1} = (\neg\mathcal{X}_l \wedge \mathcal{E} \wedge \neg\mathcal{X}_l) \vee (\mathcal{X}_l \wedge \mathcal{E} \wedge \neg\mathcal{I}) \vee (\mathcal{X}_l \wedge \mathcal{E}) \tag{7}$$

The forward propagation process can also be viewed as the combination of differentiable math operations, which makes the backward can be easily implemented:

$$\mathcal{X}_{l+1} = (1-(\mathcal{E}+\mathcal{I}-2\mathcal{E}\circ\mathcal{I}))\circ\mathcal{X}_l + (\mathcal{E}-\mathcal{E}\circ\mathcal{I}) \tag{8}$$

We use $\mathrm{H}(\mathcal{X}_l)$ to denote $\mathcal{E}-\mathcal{E}\circ\mathcal{I}$ and use $\mathrm{T}(\mathcal{X}_l)$ to denote $\mathcal{E}+\mathcal{I}-2\mathcal{E}\circ\mathcal{I}$, then $\mathrm{H}(\mathcal{X}_l)$ and $\mathrm{T}(\mathcal{X}_l)$ are both nonlinear transformations of $\mathcal{X}_l$.

$$\mathcal{X}_{l+1} = (1-\mathrm{T}(\mathcal{X}_l))\circ\mathcal{X}_l + \mathrm{H}(\mathcal{X}_l) \tag{9}$$

The forward propagation process can be regarded as a propagation method using a gating mechanism (Srivastava et al., 2015). $\mathrm{T}(\mathcal{X}_l)$ determines how much of the $\mathcal{X}_l$ will be carried to the next layer, and $\mathrm{H}(\mathcal{X}_l)$ is the transformation of $\mathcal{X}_l$. When $\mathrm{T}(\mathcal{X}_l)$ is 0, i.e., $\mathcal{E}=\mathcal{I}=0$ or $\mathcal{E}=\mathcal{I}=1$, then $\mathrm{H}(\mathcal{X}_l)$ is 0 and $1-\mathrm{T}(\mathcal{X}_l)$ is 1, which means that $\mathcal{X}_l$ is completely passed to the next layer, there is no additional transformation, which is equivalent to identity mapping. When $\mathrm{T}(\mathcal{X}_l)$ is 1, i.e., $\mathcal{E}=0,\mathcal{I}=1$ or $\mathcal{E}=1,\mathcal{I}=0$, then $1-\mathrm{T}(\mathcal{X}_l)$ is 0, and the output is completely determined by

$\mathrm{H}(\mathcal{X}_l)$. The network can use the gate mechanism to realize identity mapping or transform the input (specifically, the mutual conversion between 0 and 1) as required.

$$\mathcal{X}_{l+1} = \begin{cases} \mathcal{X}_l, & \text{if} \quad \mathrm{T}(\mathcal{X}_l) = 0 \\ \mathrm{H}(\mathcal{X}_l), & \text{if} \quad \mathrm{T}(\mathcal{X}_l) = 1 \end{cases} \tag{10}$$

The back propagation is as follows:

$$\frac{d\mathcal{X}_{l+1}}{d\mathcal{X}_l} = \begin{cases} \mathbf{1}, & \text{if} \quad \mathrm{T}(\mathcal{X}_l) = 0 \\ \mathrm{H}'(\mathcal{X}_l), & \text{if} \quad \mathrm{T}(\mathcal{X}_l) = 1 \end{cases} \tag{11}$$

The network avoids the problem of gradient vanishing caused by the spiking neuron layer in the spiking ResNet during the back-propagation process. This means that the network can achieve higher performance by increasing the depth of the network.

4 EXPERIMENTS

We evaluated our proposed LSN model on the static datasets, including CIFAR10, ImageNet, and the neuromorphic dataset CIFAR10-DVS (Li et al., 2017) respectively. In order to compare with the spiking ResNet and its variants, we adopted the network with the same number of layers for comparison. The LSN includes two branches, so in order to maintain the same parameters as the spiking ResNet and its variants when the number of layers is the same, we reduced the number of channels of the LSN so that the number of channels was three-quarters of the number of standard channels. In this way, the parameters amount of the LSN is basically the same as that of other networks with the same number of layers. We also evaluate LSN with a standard number of channels to explore the impact of network depth and width on performance.

4.1 CIFAR10

The CIFAR10 dataset consists of 50000 training pictures and 10000 test pictures. There are a total of 10 categories. The number of training pictures and test pictures in each category is equal, and the picture resolution is 32*32. Compared with the previous spiking ResNet19, LSN19 improves the performance by 1.5% and uses fewer time steps, as illustrated in the Table. 3. We use LSN101 to prove that the proposed network has no degradation problem and achieves higher accuracy on CIFAR10. The data volume of the CIFAR10 dataset is small, and ResNet19 and LSN19 have already encountered the phenomenon of overfitting on CIFAR10. Therefore, as the network depth increases, the accuracy is not significantly improved. Thus we conduct a more comprehensive experiment on ImageNet.

4.2 IMAGENET

The ImageNet dataset includes 1.2 million training images and 50,000 testing images, with a total of 1,000 categories. We use standard data processing methods, including random cropping, random horizontal flipping, normalization, and no additional data augmentation methods are used. We first tested the performance of the LSN as the number of layers increased:

Going deeper with LSN As illustrated in Fig. 5 and Table. 3, as the depth of the network increases, the accuracy of the LSN on both the training set and the test set increases accordingly, indicating that LSN solves the problem of vanishing gradients. By increasing the number of channels in LSN34, the performance of the LSN34 (standard channel) is improved by 1.7%, and by deepening the network, the performance of LSN50 is improved by 2.3%. Compared with LSN34 (standard channel), LSN50 has fewer parameters and higher performance, which indicates that the depth of the network instead of the number of parameters contributes more to the shallow network's performance. In order to have the same number of parameters as spiking ResNet and its variants of the same layers, LSN reduces the number of channels, which makes the network relatively narrower while deepening, limiting the fitting ability of the network. To seek a balance between the depth and width of the network, we conduct a 71-layer LSN with a standard number of channels to obtain higher performance.

LSN vs. Spiking ResNet Compared with spiking ResNet, LSN has higher performance on 18-layer, 34-layer, and 50-layer networks with fewer time steps. Spiking ResNet is unable to train networks

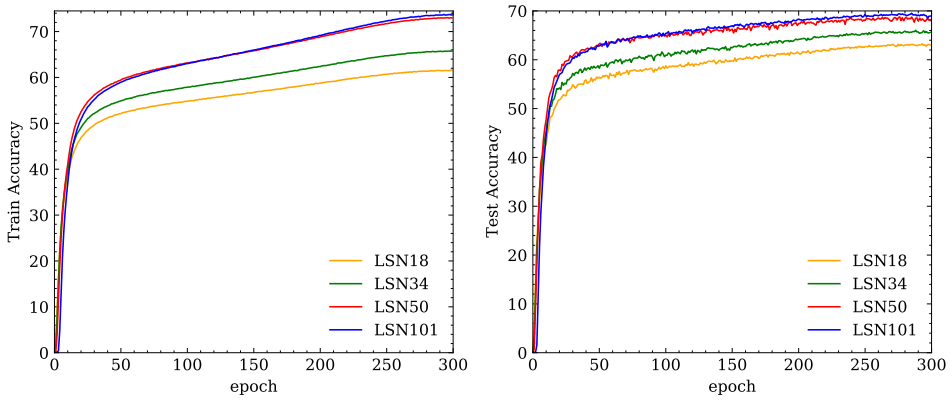


Figure 5: Train Accuracy and Test Accuracy on ImageNet.

Table 3: Comparison with Spiking ResNet and its variants on CIFAR10, ImageNet, and DVS-CIFAR10.

Datasets	Model		parameter	neuron	T	Accuracy
CIFAR10	(Zheng et al., 2021)	Spiking ResNet19	11.7M	IF	4	92.92%
		Spiking ResNet19	11.7M	IF	6	93.16%
	Ours	**LSN19**	**13.2M**	**LIF**	**4**	**94.47%**
		LSN101	**50.1M**	**LIF**	**4**	**94.68%**
ImageNet	(Zheng et al., 2021)	Spiking ResNet34	21.8M	IF	6	63.18%
		Spiking ResNet50	25.6M	IF	6	64.88%
		Spiking ResNet34(large)	87.2M	IF	6	67.05%
	(Fang et al., 2021)	SEW18	11.7M	IF	4	63.18%
		SEW34	21.8M	IF	4	67.04%
		SEW50	25.6M	IF	4	67.78%
		SEW101	44.6M	IF	4	68.76%
		SEW152	60.2M	IF	4	69.26%
	(Li et al., 2021)	Spiking ResNet34+Dspike	21.8M	LIF	6	68.19%
	(Deng et al., 2022)	Spiking ResNet34+TET	21.8M	LIF	6	64.79%
	(Guo et al., 2022)	Spiking ResNet34+MPD	21.8M	LIF	6	67.33%
	Ours	**LSN18**	**13.2M**	**LIF**	**4**	**63.27%**
		LSN34	**24.5M**	**LIF**	**4**	**66.02%**
		LSN50	**28.8M**	**LIF**	**4**	**68.79%**
		LSN101	**50.1M**	**LIF**	**4**	**69.53%**
		LSN34(standard channel)	**49.0M**	**LIF**	**4**	**68.09%**
		LSN71(standard channel)	**61.9M**	**LIF**	**4**	**71.86%**
DVS-CIFAR10	(Zheng et al., 2021)	Spiking ResNet19	11.7M	IF	10	67.80%
	Ours	**LSN15**	**8.9M**	**LIF**	**4**	**75.10%**

with more than 50 layers and suffers severe degradation problems, while LSN can train networks with more than 100 layers and has higher performance.

LSN vs. spike-element-wise (SEW) ResNet The SEW101 and the SEW152 are the only SNNs with more than 100 layers to compare, but their outputs contain positive integers. The performance of the 18-layer, 50-layer, and 101-layer LSNs is also higher than that of SEW. The 71-layer LSN with a standard number of channels, which has the same number of parameters as SEW152, outperforms SEW152 with 71.86% accuracy.

LSN vs. Networks with additional training Methods As shown in Table. 3, we compared LSN with previous training methods. Due to the gradient vanishing problem of spiking ResNet, most of the previous works are confined to shallow networks. Although these methods highly improve the performance of shallow networks, deep networks LSN50, LSN100, LSN71 (standard channel), SEW101, and SEW152 all perform better than applying additional training methods on shallow spiking ResNet. So this also verifies that limited network depth greatly limits the improvement of SNN performance.

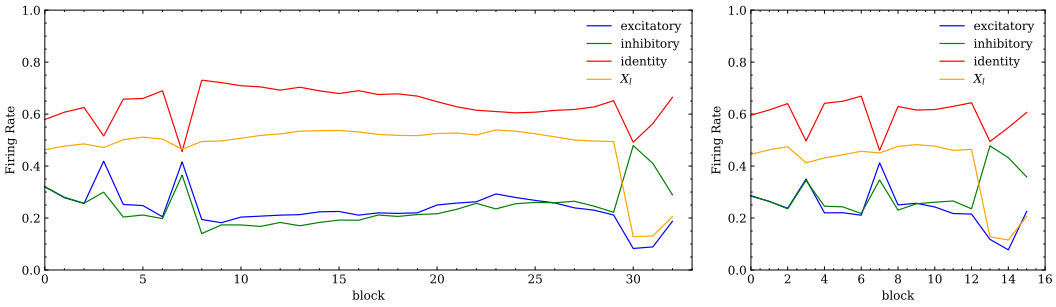


Figure 6: Firing rates on the ImageNet test set

Fig. 6 shows the firing rate in LSN101 and LSN50 on the ImageNet test set across blocks. The firing rate of excitatory pathways and inhibitory pathways in the network is around 0.2, consistent with the sparsity of the spiking neural network. Due to the unique structural design of LSN, it can be found that the firing rate of $\mathcal{X}_l$ of each block in the network remains basically unchanged with the increase of depth, indicating there is no silence problem of the deep network. About 60% of the input of each layer is transmitted to the next layer through identity mapping so that the network avoids the degradation problem in the process of the forward propagation, and the problem of gradient vanishing does not occur during the process of back-propagation(shown inequation 11). The deep network LSN101 has a higher proportion of identity mapping than the shallow network LSN50, which conforms to the characteristics of the deep network.

4.3 CIFAR10-DVS

The CIFAR10-DVS contains 10,000 event streams which were converted from 10,000 frame-based images across a dynamic vision sensor (DVS), in total of 10 different classes. Thus we use a smaller network LSN15 compared to the LSN19 used in the CIFAR10 experiments, and we use fewer time steps to achieve higher accuracy compared to the spiking ResNet.

5 Conclusion

We first explore spiking ResNet and its variants and demonstrate that it is impossible to directly replace the activation function layer of residual block in ANN with the spiking neuron layer to address the gradient vanishing problem in SNNs while keeping binary outputs. Inspired by the biological nervous system, we propose the LSN based on excitatory pathways and inhibitory pathways, which adopts logical operation after each branch to obtain binary outputs. It is theoretically proved that LSN will not suffer from the gradient vanishing/exploding during the back-propagation. Through experimental verification, for the first time, an SNN with the binary output and more than 100 layers is trained directly. As a more biology-plausible network, LSN outperforms spiking ResNet as well as its variants with the same number of layers. Also, we find that both the width and depth contribution to the performance of LSN, thus we optimize both network depth and network width for superior performance. LSN achieve the SOTA classification accuracy on CIFAR10, ImageNet, and neuromorphic dataset CIFAR10-DVS. Overall, we expanded the research of SNN to more than 50 layers through brain-inspired networks, which will greatly improve the future research of SNNs in both simulating the biological neural system and obtaining higher accuracy.

6 Reproducibility Statement

Our codes are based on SpikingJelly (Fang et al., 2020), which is an open-source SNN framework. Our codes are uploaded as supplementary material and will be available on GitHub after review. And the same random seed was used in the experiments to maximize reproducibility.

REFERENCES

Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, Gi-Joon Nam, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, 34(10):1537–1557, 2015.

Guo-qiang Bi and Mu-ming Poo. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience*, 18 (24):10464–10472, 1998.

David Daniel Cox and Thomas Dean. Neural networks and neuroscience-inspired computer vision. *Current Biology*, 24(18):R921–R929, 2014.

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.

Shikuang Deng and Shi Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. *arXiv preprint arXiv:2103.00476*, 2021.

Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. *arXiv preprint arXiv:2202.11946*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Wei Fang, Yanqi Chen, Jianhao Ding, Ding Chen, Zhaofei Yu, Huihui Zhou, and Yonghong Tian. Spikingjelly, 2020.

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.

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 Lud De Raedt (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 2471–2477. International Joint Conferences on Artificial Intelligence Organization, 7 2022. doi: 10.24963/ijcai.2022/343. URL https://doi.org/10.24963/ijcai.2022/343. Main Track.

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*, pp. 326–335, 2022.

Bing Han and Kaushik Roy. Deep spiking neural network: Energy efficiency through time based coding. In *European Conference on Computer Vision*, pp. 388–404. Springer, 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13558–13567, 2020.

Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5353–5360, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016b.

Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

Sungmin Hwang, Jeesoo Chang, Min-Hye Oh, Kyung Kyu Min, Taejin Jang, Kyungchul Park, Junsu Yu, Jong-Ho Lee, and Byung-Gook Park. Low-latency spiking neural networks using pre-charged membrane potential and delayed evaluation. *Frontiers in Neuroscience*, 15:629000, 2021.

Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J Thorpe, and Timothée Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018.

Jaehyun Kim, Heesu Kim, Subin Huh, Jinho Lee, and Kiyoung Choi. Deep neural networks with weighted spikes. *Neurocomputing*, 311:373–386, 2018.

Jinseok Kim, Kyungsu Kim, and Jae-Joon Kim. Unifying activation-and timing-based learning rules for spiking neural networks. *Advances in Neural Information Processing Systems*, 33:19534–19544, 2020a.

Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 11270–11277, 2020b.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in neuroscience*, 11:309, 2017.

Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34:23426–23439, 2021.

Liqun Luo. Architectures of neuronal circuits. *Science*, 373(6559):eabg7285, 2021.

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

Milad Mozafari, Saeed Reza Kheradpisheh, Timothée Masquelier, Abbas Nowzari-Dalini, and Mohammad Ganjtabesh. First-spike-based visual categorization using reward-modulated stdp. *IEEE transactions on neural networks and learning systems*, 29(12):6178–6190, 2018.

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

Nicolas Perez-Nieves and Dan Goodman. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems*, 34:11795–11808, 2021.

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.

Saima Sharmin, Priyadarshini Panda, Syed Shakib Sarwar, Chankyu Lee, Wachirawit Ponghiran, and Kaushik Roy. A comprehensive analysis on adversarial robustness of spiking neural networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

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

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Stanisław Woźniak, Angeliki Pantazi, Thomas Bohnstingl, and Evangelos Eleftheriou. Deep learning incorporating biologically inspired neural dynamics and in-memory computing. *Nature Machine Intelligence*, 2(6):325–336, 2020.

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, pp. 11062–11070, 2021.

A Appendix

A.1 Hyper-Parameters

surrogate gradient

We used the same triangular surrogate gradient in all experiments:

$$S(x) = \begin{cases} 1 - |x|, & \text{if} \quad |x| \leqslant 1 \\ 0, & \text{if} \quad |x| > 1 \end{cases} \tag{12}$$

neuron model

We adopt the LIF neuron model, we set $\alpha = 0.5$ and $V_{th} = 1$ for all neurons.

experiment

We adopt Adam as optimizer and Cosine Annealing as learning rate scheduler for all experiments, other hyper-parameters are set according to specific experiments, as shown in Table. 4:

Table 4: Hyper-parameters for different datasets and different network.

Dataset	Model	learning rate	weight decay	epoch	warm up epoch
CIFAR10	LSN19	5e-3	5e-3	300	none
	LSN101	5e-3	5e-3	300	none
ImageNet	LSN18	1e-3	5e-3	300	10
	LSN18	1e-3	5e-3	300	10
	LSN34	1e-3	5e-3	300	10
	LSN50	1e-3	5e-3	300	10
	LSN101	1e-3	5e-3	300	10
	LSN34(standard channel)	1e-3	5e-3	300	10
	LSN71(standard channel)	1e-3	15e-3	300	10
DVS-CIFAR10	LSN15	5e-3	5e-2	150	none

A.2 Gradient visualization

We make a further analysis of the gradient.

$$\begin{aligned} \frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{X}_l} = 1 - \mathcal{E} - \mathcal{I} + \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} - \mathcal{X}_l \circ \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} - \mathcal{X}_l \circ \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l} - \mathcal{I} \circ \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} \\ -\mathcal{E} \circ \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l} + 2\mathcal{E} \circ \mathcal{I} + 2\mathcal{X}_l \circ \mathcal{I} \circ \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} + 2\mathcal{X}_l \circ \mathcal{E} \circ \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l} \end{aligned} \tag{13}$$

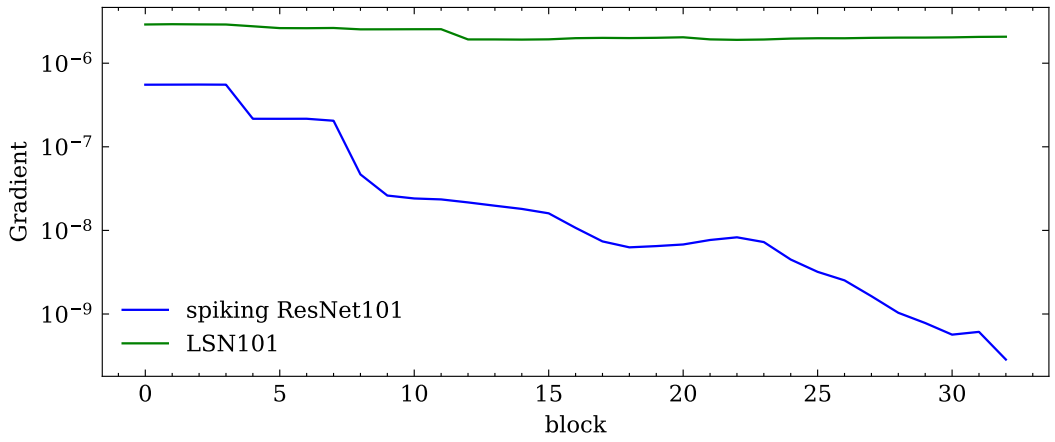


Figure 7: Gradients of epoch 5 of LSN101 and Spiking ResNet101 on CIFAR10. Deep LSN does not have the problem of gradient vanishing

$$\frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{E}} = 1 - \mathcal{X}_l - \mathcal{I} + 2\mathcal{X}_l \circ \mathcal{E} \circ \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l} \tag{14}$$

$$\frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{I}} = -\mathcal{X}_l - 2\mathcal{E} + 2\mathcal{X}_l \circ \mathcal{E} \circ \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l} \tag{15}$$

Table 5: Gradients in different situations.

$\mathcal{X}_l$	$\mathcal{E}$	$\mathcal{I}$	$\mathcal{X}_{l+1}$	$\frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{X}_l}$	$\frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{E}}$	$\frac{\partial \mathcal{X}_{l+1}}{\partial \mathcal{I}}$
0	0	0	0	$1 + \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l}$	1	0
0	0	1	0	0	0	0
0	1	0	1	$\frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} - \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l}$	1	1
0	1	1	0	$1 - \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l}$	0	1
1	0	0	1	$1 - \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l}$	1	1
1	0	1	0	$\frac{\partial \mathcal{E}}{\partial \mathcal{X}_l} - \frac{\partial \mathcal{I}}{\partial \mathcal{X}_l}$	0	1
1	1	0	1	0	0	0
1	1	1	1	$1 + \frac{\partial \mathcal{E}}{\partial \mathcal{X}_l}$	1	0