

RESEARCH ARTICLE

A Comparative Study on the Performance and Security Evaluation of Spiking Neural Networks

YANJIE LI¹, XIAOXIN CUI², (Member, IEEE), YIHAO ZHOU¹, (Member, IEEE),
AND YING LI, JR.¹, (Member, IEEE)

¹Institute of Microelectronics, Chinese Academy of Sciences, Beijing 100029, China

²School of Integrated Circuits, Peking University, Beijing 100871, China

Corresponding author: Ying Li, Jr. (liyings1@ime.ac.cn)

This work was supported by the Science and Technology Innovation 2030-Major Project through the Brain Science and Brain-Like Intelligence Technology under Grant 2022ZD0208700.


ABSTRACT The brain-inspired Spiking neural networks (SNN) claim to present advantages for visual classification tasks in terms of energy efficiency and inherent robustness. In this work, we explore the impact on network inter-layer sparsity through neural coding schemes and the intrinsic structural parameters of Leaky Integrate-and-Fire (LIF) neurons, which can be a candidate metric for performance evaluation. Towards this, we perform a comparative study of four critical neural coding schemes: rate coding (poisson coding), latency coding, phase coding, and direct coding, as well as 6 LIF neuron intrinsic parameter options for a total of 24 combined parameter schemes. Specifically, the models were trained using a supervised training algorithm with a surrogate gradient, and two adversarial attacks, Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) were applied on a CIFAR10 dataset. We identified the sources of interlayer sparsity in SNN, and quantitatively analyzed the differences in sparsity caused by coding schemes, neuron leakage factors and thresholds. Various aspects of network performance were thoroughly considered in this paper, including inference accuracy, adversarial robustness, and energy efficiency. Our results show that latency coding is the optimum choice in achieving the highest adversarial robustness and energy efficient against low intensity attacks, while rate coding offers the best adversarial robustness against medium and high intensity attacks. The maximum deviations of robustness and efficiency between different coding schemes are 9.35% in VGG5 and 13.59% in VGG9. Increasing the sparsity of spike activity by improving the threshold can bring a short-lived adversarial robustness sweet spot, while excessive sparsity due to changes in threshold and leakage can instead reduce the adversarial robustness. The study reveals the advantages and disadvantages, and design space of SNN in various dimensions, allowing researchers to frame their neuromorphic systems in terms of the coding methods, neuron inherent structure, and model learning capabilities.

INDEX TERMS Spiking neural network, accuracy, energy efficiency, adversarial robustness, sparsity.

I. INTRODUCTION

Spiking neural networks (SNN), which have spatio-temporal spiking sparsity and biological-like properties, are increasingly used to investigate more energy-efficient neural computing circuits than artificial neural networks (ANN) [1], [2]. Numerous studies have begun to use biologically plausible

learning methods to implement neuromorphic circuits [3], [4], [5]. The SNN structure was discovered to have certain robust advantages in resisting sample noise and adversarial attacks due to its sparse inherent structure and discrete encoding of the input [6], [7], [8], [9]. The main challenge in optimizing SNN is the lack of a way to achieve reliable metrics for their inference accuracy. Current performance improving methods can be broadly classified into three types: ANN-converted SNN [10], [11], surrogate gradient training

The associate editor coordinating the review of this manuscript and approving it for publication was Mario Donato Marino .

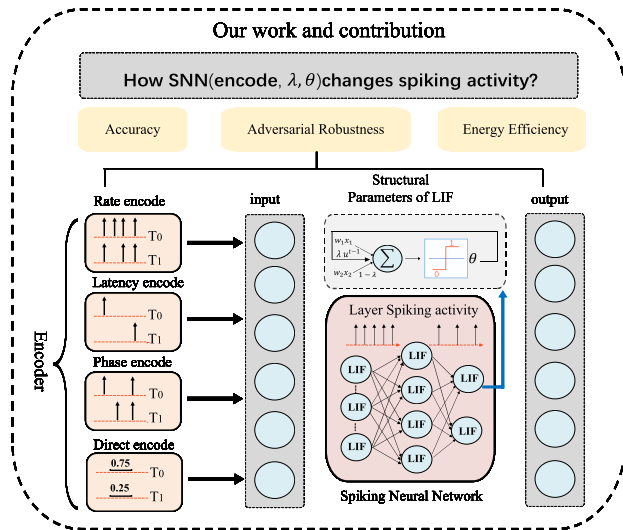


FIGURE 1. Overview of our work and contributions.

methods [12], and unsupervised learning training methods such as spike-timing dependent plasticity (STDP) or STDP variants [13]. Unlike ANN, SNN requires a larger time window to ensure accuracy, resulting in a longer inference delay. Unsupervised STDP produces good classification results for shallow networks and simple datasets but fails for deep networks and complex datasets. The other studies using back propagation (BP) and surrogate gradient fusion techniques to train SNN, represent promising results and provide guidance and references for neuromorphic hardware designers. Still, some key issues remain to be addressed as follows.

How to choose an appropriate neural coding at the algorithm level and determine the intrinsic parameters of neurons to achieve hardware design accuracy, robustness, and energy efficiency while reducing inference delay?

Since network structure and neuron parameters change the spike-sparsity distribution of spiking neural networks, what does this bring to the network’s accuracy, adversarial robustness, and energy efficiency?

Spike sparsification has shown potential impacts on energy reduction and improves robustness in previous studies but lacks systematic research and more comprehensive analysis. Thus, sparsity can be treated as a property to effectively explain and investigate SNN performances in different coding styles and key structural parameter settings, which was this work’s original intent and focus. The paper contributes in:

- 1) Re-model the LIF neuron dynamics equation to analyze the source of the sparsity of the SNN network, then determine the relationships among the network inner layer sparsity and the neuron leakage value λ , the encoding scheme, and the threshold voltage θ .
- 2) For the first time, we propose to use sparsity as a metric to quantitatively explain the effects of coding methods, neuron leakage factors, and thresholds of LIF neurons.
- 3) Comprehensively compare and evaluate the network performance and security of SNN from the view of

the sparseness of the network in three dimensions: accuracy, adversarial robustness, and energy efficiency.

Fig. 1 shows our main work. The task of our work is to evaluate spiking neural networks at three dimensions: inference accuracy, adversarial robustness and energy efficiency. The network architecture consists of four data encoders and spiking neural networks. The role of the encoders is to generate discrete data at each time step according to different encoding schemes, and the encoded data are received at different time steps by the LIF neurons in the input layer of the SNN. The SNN architecture structure mainly consists of neurons and neuron topological connections. In this paper, we use LIF neurons, whose structure is shown in the gray dashed box pointed by the blue arrow in Fig. 1, and the mathematical theory in Eq. 3 and Eq. 4. The neuron topology connection is adopted from the classical model of ANN neural network Visual Geometry Group(VGG). VGG network mainly includes: convolutional layers, pooling layers and fully connected layers, the specific connection is referred to [14]. During the training of the network, the input encoding method and the intrinsic structural parameters of LIF neurons are changed to regulate the spike sparsity, and the training is completed by generating adversarial samples to be injected into the network to calculate the accuracy loss and compare the inference accuracy, adversarial robustness and energy efficiency.

In the following section, we first review the related studies on the intrinsic structure of SNN inference accuracy, sparsity, and robustness (Section II). Next, we introduce the models used, including encoding methods, training methods, neuron models, and adversarial attack methods (Section III), followed by implementation and discussion (Section IV), and finally, a summary of the implementation and conclusions (Section V).

II. RELATED WORK

Sharmin et al. [6] reviewed SNN adversarial robustness research. They proposed a simple and practical framework to construct adversarial attacks and also pointed out that input discretization via poisson encoding and non-linear activations of LIF (or IF) neurons are sources of SNN robustness. Rida et al. [7] investigated the robustness of SNN against attacks with varying neuron firing voltage thresholds and time window boundary values. They listed the network’s robustness under various (V, T) combinations, where V refers to the voltage of LIF neuron threshold, and T refers to the training time step. Kundu et al. [15] assessed the robustness of the VGG5 and VGG11 network structures. They discovered that LIF can reduce input perturbation and that its impact should consider the effect of combined parameters such as weight, leakage, threshold, and time step. However, no quantitative analysis was performed. Guo et al. [16] extensively compared four neural coding schemes from the perspectives of algorithms and hardware to find the best coding scheme. Still, this comparison was limited to 2 Layer fully connected

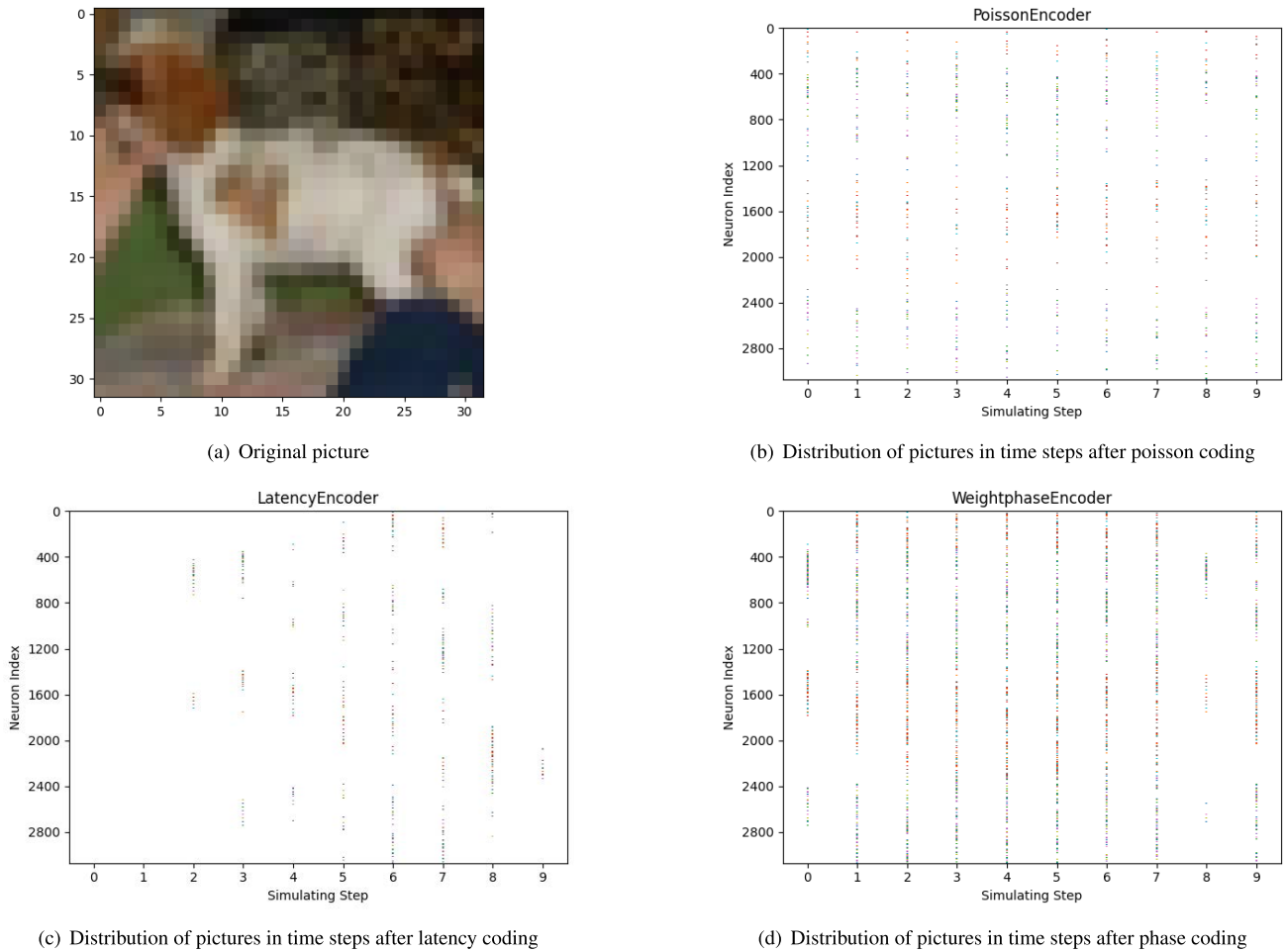


FIGURE 2. Sparse distribution of different coding methods.

network and STDP training, which cannot scale to deep SNN networks, based on vision datasets but without considering the critical metric of adversarial robustness. Kim et al. [17] compared the robustness, energy efficiency, accuracy, and other characteristics of the two encodings during low-latency training but did not compare them to other encoding methods.

III. MODEL DESCRIPTION

A. SNN CODING SCHEMES

1) RATE CODING

Rate coding [6], [18] is the most widely used coding scheme in neural network models, which treats each input pixel as a trigger frequency and converts the pixel into a sequence of poisson spiking with a firing frequency. A poisson encoder encodes the input data as a spiking sequence whose firing times distribution conforms to a poisson process. The spiking firing probability p is set to x of a timestep, where x needs to be normalized to a specific interval and compared with a random number.

2) LATENCY CODING

Latency encoders delay spiking according to the size of the input data. When the stimulus intensity is larger, the fir-

ing time is earlier, and there is a maximum spiking firing time. [19] Therefore, for each input data x , a spike sequence with a timestep of the maximum spike time can be obtained, and each sequence has one and only one spike. The spike emission time can be described as:

$$t_f = (T - 1)(1 - x) \tag{1}$$

where x is input data and $x \in [0, 1]$, after getting the spiking firing time, the spiking sequence is fed to the neural network's input layer in time timesteps.

3) PHASE CODING

Weighted phase [19] coding spreads the input data into binary bits, traversing the input from high to low for spiking coding. Compared with rate coding, each bit carries more information. When the number of encoding phases is K , the number in the interval $[0, 1 - 2^{-(k)}]$ can be encoded. In this paper, we set the timestep to 10 and the number of encoding phases K to 10.

4) DIRECT CODING

Direct coding [15], [17] uses floating-point input directly in the first layer of the spiking neural network. We normalize

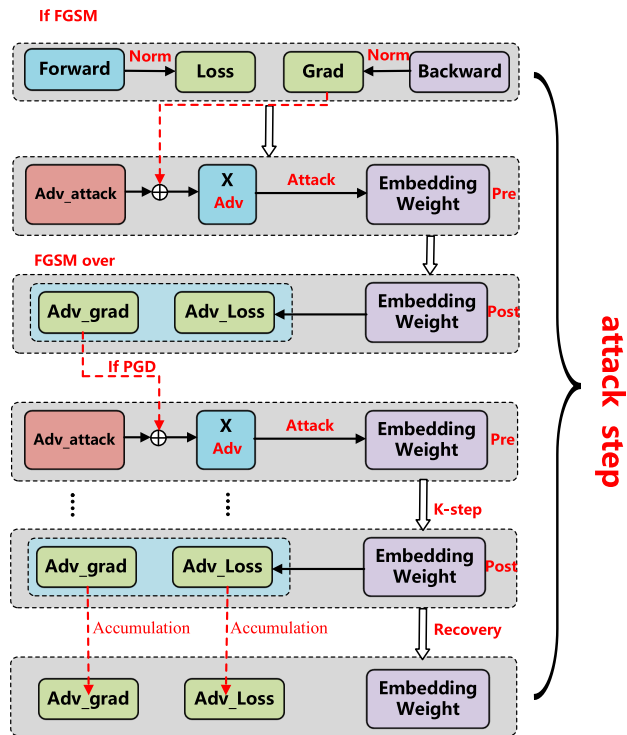


FIGURE 3. Overview of adversarial attack. where, norm in red represents the normal forward and backward propagation, and attack in red represents the attack process. Training: SNNs are trained according to the surrogate gradient formula. Adversarial input generation: adversarial input for X_{adv} is produced from the clean input perturbed by the sign of the gradient of the loss of norm process function.

the pixel value of the input image and send it directly to the first layer of the spiking neural network. The floating-point output value is processed by the SNN at T times repeat step by step.

We randomly sampled single image samples from a portion of the CIFAR10 dataset and plotted the distribution of spikes issued at each time step for the three coding schemes, rate, latency and weighted phase coding. We observe some interesting patterns as shown in Fig. 2: latency coding has fewer spikes issued at each time step and sparser distribution at each time step within the same size time window compared to rate coding and weighted phase coding. In order to gain more insight into how this sparse distribution affects network accuracy, robustness against and energy efficiency, we performed experiments for quantitative estimation, the details of which are shown in Section V.

B. TRAINING METHOD

Current SNN training methods can be divided into three categories, ANN-transformed SNN [10], STDP [13] and methods based on BP and surrogate gradients [11], [12]. STDP does not show good classification performance for deep neural networks and complex datasets, and the method based on ANN-to-SNN conversion requires a time window for reliable inference, which leads to a huge inference delay. So we train

the SNN with alternative gradients and spatio-temporal back-propagation(STBP).

C. LIF NEURON MODEL

The LIF neuron model not only can capture the essential characteristics of nervous system information but also takes into account high computational efficiency, so it has become the most popular neuron scheme for neuromorphic chips. The model consists of a first-order linear differential equation, which defines the dynamics of the membrane potential, and a threshold condition, which determines the peak generation. Synapses act as transmission media, passing signals from one neuron to the target neuron. We use a variety of thresholds and leakage parameters in our experiments to evaluate and employ a soft reset approach, which resets the membrane potential below the threshold after a trigger spiking after charging.

They are modeled as conductances with time-varying kinetics. The equation can be described as:

$$u_t^i = \lambda u_{t-1}^i + (1 - \lambda) \sum_j w_{ij} o_t^j - \theta(t - t_k) \tag{2}$$

$$o_t^j = \begin{cases} 0, & u_t^i < \theta \\ 1, & u_t^i \geq \theta \end{cases} \tag{3}$$

where $\lambda = 1 - 1/t_m$, t_m is the time constant of membrane potential decay, u_t^i and w_{ij} are output spikes and membrane potential at timestep t for layer l , and θ is the neuron membrane threshold voltage. When u exceeds the membrane threshold $\theta(t = t_k)$, LIF neurons accumulate potential and generate spiking output. Threshold θ and leakage factor λ directly affect SNN interlayer spike sparsity.

To take advantage of standard backpropagation based optimizers, we use the surrogate gradient technique. The input-output characteristic of LIF neuron is a step function whose gradient is discontinuous at the spiking point. In surrogate gradient techniques, gradients are approximated by pseudo-derivatives. Kim et al. [17] used a gradient approximation function that exploits the peak time information in the derivative. The gradient for timestep t is calculated as follows:

$$\frac{\partial o_t}{\partial u_t} = \max \left\{ 0, 1 - \left| \frac{u_t}{\theta} - 1 \right| \right\} \tag{4}$$

where u_t and o_t are the output peak value and membrane potential at timestep t . LIF neurons in the hidden layer only generate spiking output when the membrane potential exceeds the firing threshold.

D. ADVERSARIAL ATTACK

Determine a dataset (x, y_{true}) and a classification model h , where x is a clean image and y_{true} is the corresponding correct label. The concept of an adversarial attack is to find an input x_{adv} such that x and x_{adv} are indistinguishable to the human eye, but the model h misclassifies x_{adv} , i.e. produces a high probability output on the wrong label. In our work, we consider the following two methods for generating x_{adv} ,

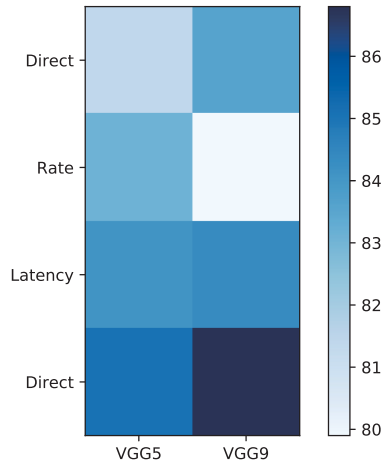


FIGURE 4. The accuracy of the four coding methods when the threshold parameter of VGG5, VGG9 network is $(\theta, \lambda) = ((0.4, 1), 0.5)$. The vertical axis represents the coding scheme, and the vertical axis represents (θ, λ) combination parameters. The scale on the color bar represents the inference accuracy under that color, e.g., 86 means the inference accuracy is 86%.

TABLE 1. model parameter.

Parameter	Quantity
neuron model	LIF
learning rate	0.001
training epoch	200
dataset	CIFAR10
time step	10
network architectures	VGG5,VGG9
VGG5 architectures	32×32-c-p-c-c-p-f-f-o
VGG9 architectures	32×32-c-c-p-c-c-p-c-c-p-f-f-o

FGSM and PGD. The attack generation steps are shown in Fig. 3.

1) FGSM ATTACK

This is the basic and broadest way to generate an adversarial attack, which takes the form:

$$x_{adv} = x + \varepsilon \text{sign}(\nabla_x J(x, y_{true})) \quad (5)$$

where ε refers to the amount of disturbance, usually ε is much smaller than x , and $\nabla_x J$ is the gradient of the loss function relative to the original clean data [20].

2) PGD ATTACK

The PGD attack [21] is a multi-step method for generating adversarial inputs with an iterative step size $\alpha \geq \epsilon/k$, where k is the number of iterations. In an untargeted PGD attack, the loss is calculated according to the true label y_{true} .

$$x_{adv}^0 = x \quad (6)$$

$$x_{adv}^N = \text{Clip}_{x,\varepsilon} \left\{ x_{adv}^N + \alpha \text{sign}(\nabla_x J(x_{adv}^N, y_{true})) \right\} \quad (7)$$

where x_{adv}^N is the adversarial example in the x_{adv} iteration, and α is the perturbation at each step. $\text{Clip}_{x,\varepsilon}$ means to clip

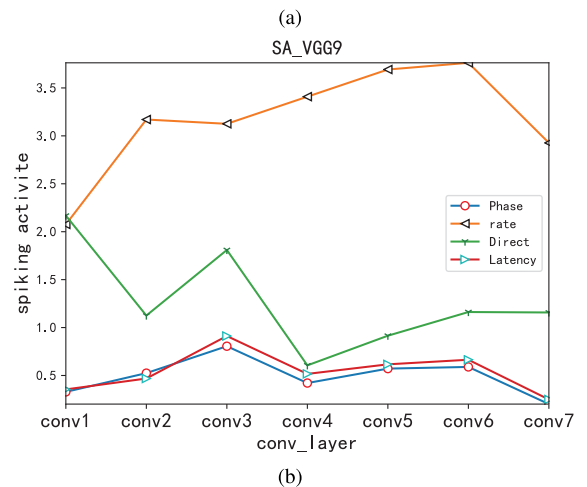
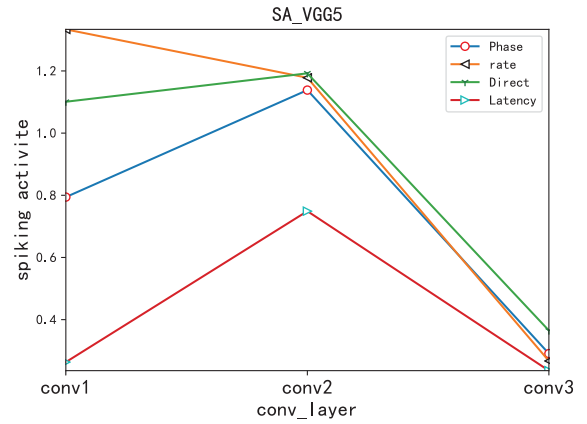


FIGURE 5. The spike activity of the four coding methods. The horizontal axis represents the different layers of the neural network (e.g., convolutional layer for conv1) The vertical axis represents the number of spike activities between layers. The different color lines in each figure represent different coding methods (a) when the combination parameters of VGG5 are $(\theta, \lambda) = ((0.4, 1), 0.5)$. (b) When the combination parameters of VGG9 are $(\theta, \lambda) = ((0.4, 1), 0.5)$.

the parameters element-wise to the range $[x - \varepsilon, x + \varepsilon]$, sign is the sign function.

IV. EXPERIMENTS AND DISCUSSIONS

A. EXPERIMENTAL PARAMETER DEVICE

We used the image classification dataset CIFAR10 in experiments and supplement the input with conventional data. Various peak sequences are generated by different encoding functions, which are then fed into the network’s input. The VGG5 architecture is 32×32-c-p-c-c-p-f-f-o, The VGG9 architecture is 32×32-c-c-p-c-c-p-c-c-p-f-f-o. where 32×32 represents the input feature map size, c = convolutional layer, p = pooling layer, fc = fully connected layer and o = output layer. VGG5 and VGG9 were trained for 200 epochs with an initial learning rate of 0.001. Other hyperparameters differ for different encoding settings in order to achieve reliable inference accuracy. All experiments were carried out on an NVIDIA A5000 Graphic Processing Unit (GPU) with 24 GB of memory, and the models were built with PyTorch. The model hyperparameter shows in Table 1.

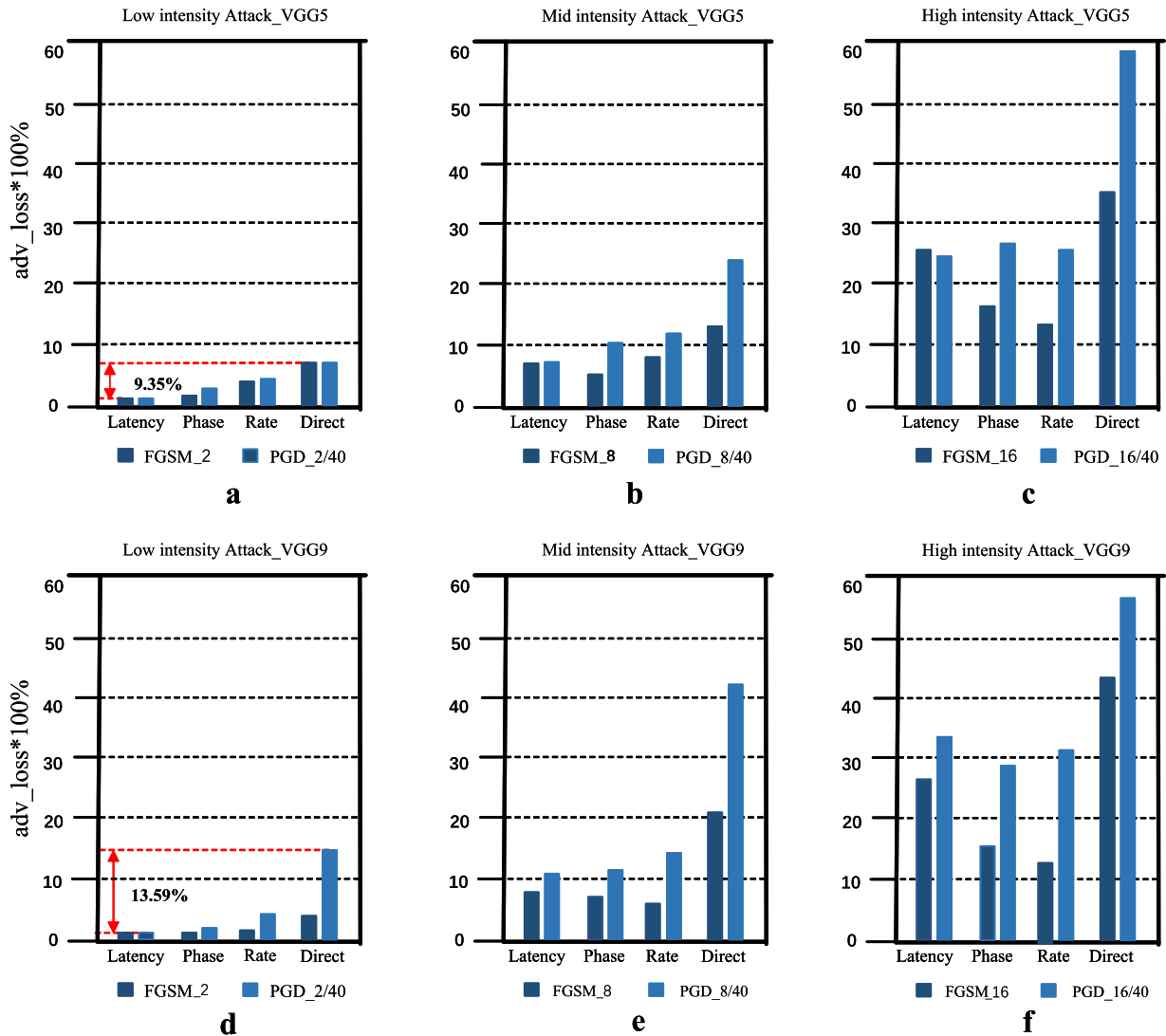


FIGURE 6. (a,b,c,d,e,f): A comparison of the loss in accuracy under adversarial attack of VGG5 and VGG9. we set different intensity Attack as following: (a) Low intensity Attack_VGG5. (b) Middle intensity Attack_VGG5. (c) High intensity Attack_VGG5. (d) Low intensity Attack_VGG9. (e)Middle intensity Attack_VGG9. (f)High intensity Attack_VGG9. each of these cases, the amount of perturbation ϵ has been varied from 2_255 to 16_255, the number of iterations k has been varied from 7 to 40. FGSM_2 presents perturbation is 2, PGD_2_40 presents perturbation is 2_255 and k is 40. Smaller adv_loss in accuracy implies more adversarial robustness. The difference between the red dashed lines in figures a and d represents the largest difference in adv_loss among the four coding methods under low-intensity attacks.

B. INFLUENCE OF CODING METHOD ON NETWORK ROBUSTNESS AND ACCURACY

We compare the accuracy and adversarial robustness of SNN adversarial attacks under the four coding methods on VGG5 and VGG9 networks. In attack model, ϵ refer to the amount of perturbation, is set to 2, 8, and 16, the perturbation step size α is set to 2, and the number of iterations k is set to 40, the threshold voltages of the convolutional layer and the fully connected layer are set to 0.4, 1, respectively, The leakage parameter λ is set to 0.5, the network time step T is set to 10, in order to generate adversarial examples for SNN using rate coding.

1) ACCURACY COMPARISON

Fig. 4 shows the variation of the accuracy of different network coding architectures. For the VGG5 and VGG9 architectures,

the inference accuracy of direct coding is better than other coding methods, followed by latency coding, rate coding accuracy is lower than phase coding on VGG5, but higher than phase coding on VGG9. Fig. 5 shows the spike activity of the four coding methods when the threshold parameters of VGG5 and VGG9 are $(\theta, \lambda) = ((0.4, 1), 0.5)$.

2) ROBUSTNESS COMPARISON

We performed FGSM, PGD attacks on the networks trained with four coding methods and defined three attack intensities: low intensity attack, medium intensity attack, and high intensity attack. The results of VGG5 and VGG9 are shown in Fig. 6(a,b,c) and Fig. 6(e,f,g). The comparison results of the adversarial robustness under low intensity attacks is: latency coding > phase coding > rate coding > direct coding, and the maximum deviation of robustness between latency coding

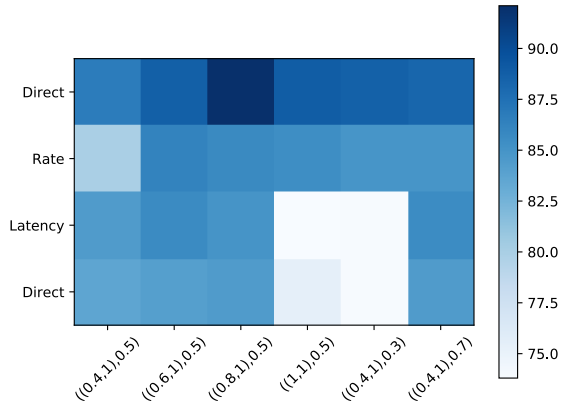


FIGURE 7. Heatmap showing the accuracy of VGG9, SNN trained on CIFAR10 dataset under different encoding schemes and (θ, λ) parameter combinations. The vertical axis represents the coding scheme, and the horizontal axis represents (θ, λ) combination parameters. The scale on the color bar represents the inference accuracy under that color, e.g., 85 means the inference accuracy is 85%.

and direct coding is 9.35% on VGG5 network and 13.59% on VGG9. Direct coding also shows the worst adversarial robustness under medium and high intensity attacks due to the greater network robustness for sparse coding inputs. However, the similar conclusion cannot be obtained for the other three coding schemes. For a deeper understanding, we counted the number of spike activities per layer of the VGG5 and VGG9 networks as shown in Fig. 5(a,b). It can be seen that latency coding has less spike activities per layer of the network compared to rate and phase coding, which should get the best adversarial robustness under each attack, but the phenomenon is not the case under high intensity attacks. This is because, according to Eq. 2, the parameters affecting the sparsity of the network via the coding method, training weights, LIF neuron threshold voltage θ and leakage factor λ , while according to Eq. 7, the generation of adversarial samples is closely related to the convolutional 1 layer weights and the hidden layer alternative gradient generation weights. The distribution of training weights is inversely related to the distribution of inter-layer spike activity. Therefore it is more meaningful to study the effect of combined parameters such as coding method, threshold, and leakage factor on robustness.

C. EFFECT OF COMBINED PARAMETERS ON ACCURACY, ADVERSARIAL ROBUSTNESS

Based on the above analysis, we observe that the sparsity of input coding methods affects obviously in VGG5 and VGG9 networks. In order to further investigate the effect of combined parameters on the network, accuracy, and robustness differences, we expand more experiments on the VGG9 network. We firstly compare the training accuracy for different combinations of encoding, threshold θ , and leakage factor λ of a total of 24 combined parameter schemes. To ensure that the experiments are within acceptable accuracy, we choose the parameter combinations as shown in Fig. 7.

1) ACCURACY COMPARISON

Fig. 7 shows the difference in inference accuracy of the four coding schemes under different combined parameters. Fig. 8 shows the spike activity of combined parameters of VGG9. From the experimental results: (1) Under the same threshold condition, direct coding has higher accuracy than the other three coding schemes. (2) Threshold changes have different effects on the accuracy of the four coding schemes. Both direct coding and rate coding can show high accuracy under the conditions of several thresholds and leakage values we selected. Latency coding and phase coding show high accuracy at low thresholds, and inference accuracy is low after training with high thresholds. This low accuracy is mainly caused by input coding schemes are too sparse and difficult to train, resulting in fewer spike activities (SA) in the convolutional layer, so the accuracy drops. (3) Latency coding with sparser input coding shows an overall rising and then decreasing trend of inter-layer spike activity, direct coding, phase coding and rate coding show an overall decreasing trend of inter-layer spike activity.

2) ADVERSARIAL ROBUSTNESS COMPARISON

We conducted PGD and FGSM attack experiments with different combinations of parameters. Where ϵ refers to the perturbation amount set to 2, 16, the perturbation step size α to 2, and the number of iterations k set to 7, 40, respectively. As shown in Fig. 9: (1) With the same (θ, λ) combination parameters, we obtain the consistent conclusion in 4.2 that direct coding has the worst robustness among the four coding methods. Latency coding, on the other hand, show the best robustness at low intensities, but does not stand out against robustness under high-intensity attacks. Latency coding input is relatively sparser, and the inter-layer weight values and interval distributions become larger after training in order to ensure that the number of post-layer spikes in the deep network is more likely to affect the adversarial samples. (2) Varying different combinations of (θ, λ) under the same coding method, direct coding shows a tendency of increased adversarial robustness as the threshold of the convolutional layer increases, which can be attributed to the fact that the succession of input coding methods makes it easier to find the appropriate gradient during the adversarial sample generation. (3) With the change of combination parameters, the maximum difference of adv_loss is 6.65% for direct coding at low intensity attacks and 18.11% at high intensity attacks. Latency coding differed by a maximum of 1.23% on adv_loss for low-intensity attacks and 12.08% on high-intensity attacks. Phase coding at low intensity attacks, adv_loss differs by 1.7% at maximum and 10.86% at high intensity attacks. The maximum difference in adv_loss for rate coding is 1.14% for low-intensity attacks and 7.9% for high-intensity attacks. Direct coding is more likely to change drastically in robustness due to changes in combination parameters, and rate coding is not sensitive to changes in combination parameters.

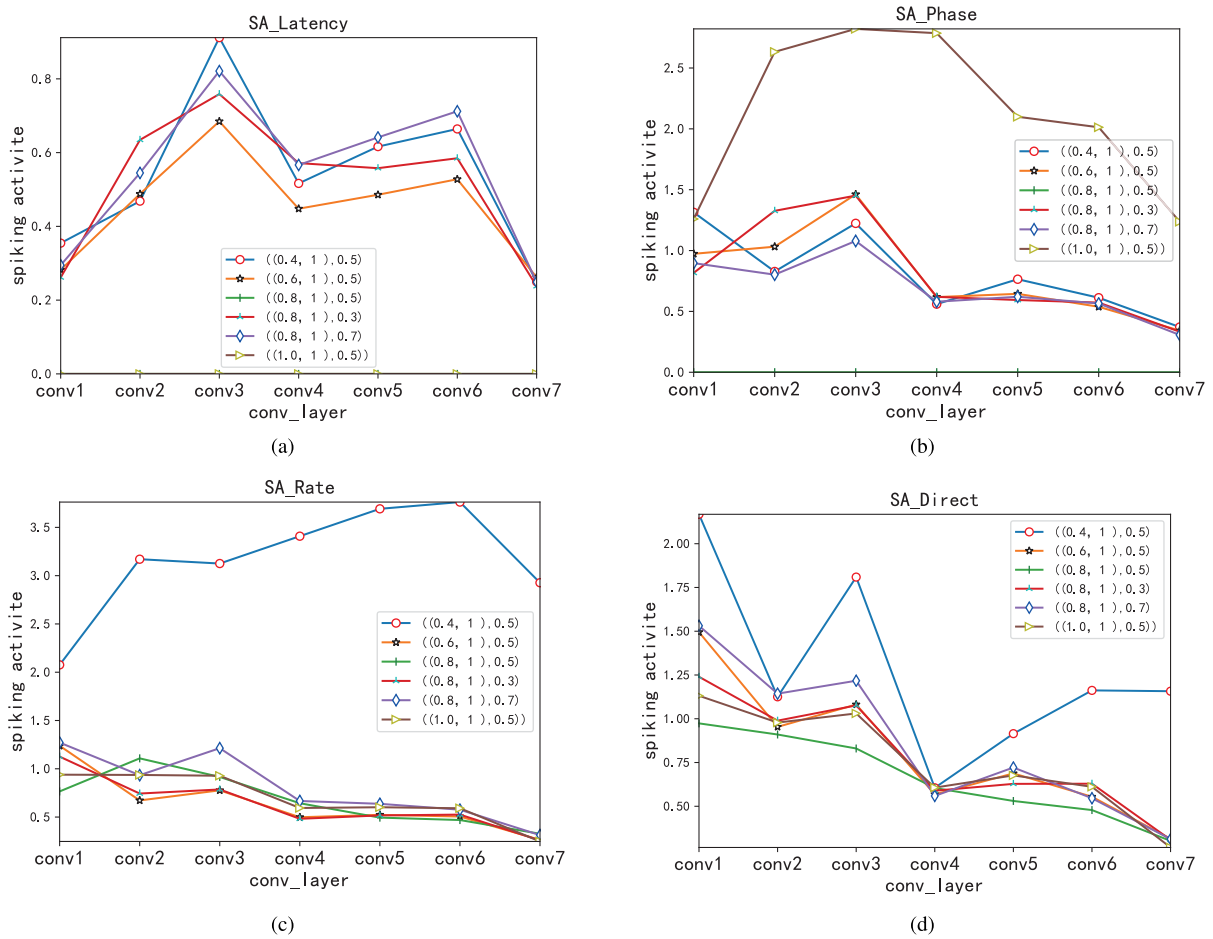


FIGURE 8. The spike activity of combined parameters of VGG9. The horizontal axis represents the different layers of the neural network (e.g., convolutional layer for conv1) The vertical axis represents the number of spike activities between layers. Different colored lines in each figure represent different combined parameters of (θ, λ) (a) Latency coding in VGG9. (b) Phase coding in VGG9. (c) Rate coding in VGG9. (d) Direct coding in VGG9.

3) ENERGY EFFICIENCY COMPARISON

Many different neuromorphic chips do not support sparse mapping topology, the weights have different widths, and the computational operation overhead is also different. Therefore, a single standard operating procedure (SOP) operation is used to evaluate the power consumption of the system for hardware evaluation of power consumption. In order to evaluate the energy efficiency of these four coding schemes more fairly at the algorithm level, we use the statistical spiking activity operand method for comparison. Table 2 shows that under different training parameters on VGG5 and VGG9, under the direct coding method $(\theta, \lambda) = ((0.4, 1), 0.5)$ spiking activity number as the benchmark, and the calculated energy after data normalization is compared. Table 2 shows the specific differences. In general, direct coding has the lowest energy efficiency, and Latency coding has the highest energy efficiency. The other two coding methods have slightly different energy efficiencies under different threshold combinations. Energy efficiency of latency coding and phase coding is 0 under the effect of some combined parameters, which is due to the fact that those combined parameter and coding scheme leads to excessive sparsity of

TABLE 2. Estimated energy costs for various operations in a 45 nm ComplementaryMetal-Oxide-Semiconductor (CMOS) process at 0.9 V [22].

Architecture	(θ, λ)	Direct	Phase	Rate	Latency
VGG5	$((0.4, 1), 0.5)$	1	0.855	0.875	0.593
VGG9	$((0.4, 1), 0.5)$	1	0.618	2.828	0.452
	$((0.6, 1), 0.5)$	0.612	0.611	0.446	0.392
	$((0.8, 1), 0.5)$	0.617	0.660	0.458	0.455
	$((1.0, 1), 0.5)$	0.604	0	0.536	0.568
	$((0.8, 1), 0.3)$	0.540	0	0.554	0

spike activity between network layers during the training process, bringing down both inference accuracy and adversarial robustness.

4) SUMMARY

Our results show that (1) on different architectures and with different combination parameters, direct coding shows the best inference accuracy, the worst robustness and high energy efficiency, the accuracy variation is stable and easy to train, but it is resistant to attacks has the highest intensity

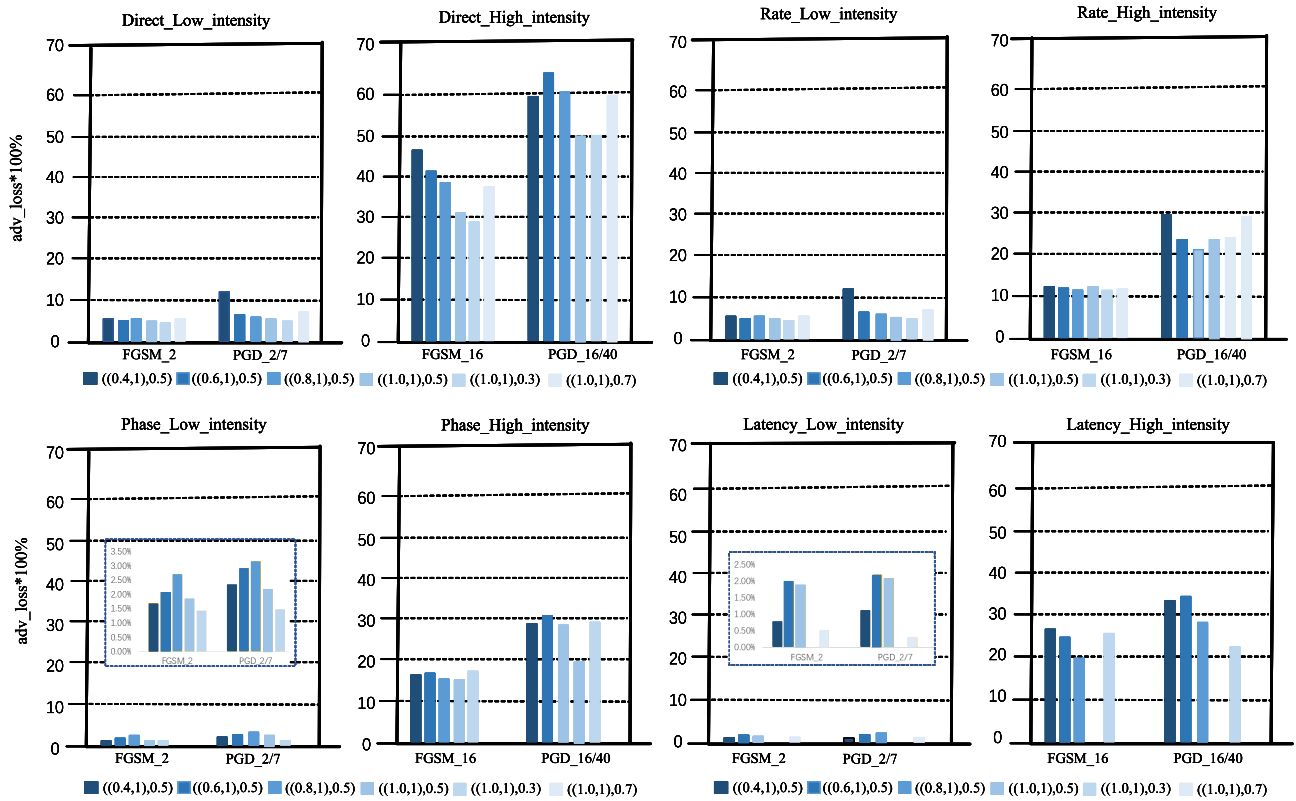


FIGURE 9. Adversarial robustness with different combination parameters.

sensitivity. (2) Latency coding shows the best energy efficiency, the best robustness under low-intensity attacks, is unstable to threshold and leakage accuracy changes and not easy to train, but is relatively insensitive to attack intensity. (3) Rate coding shows the best adversarial robustness under medium and high intensity attacks, moderate performance in inference accuracy and energy efficiency, and relatively low sensitivity to attack intensity. (4) Phase coding is less effective in every way. Each coding method has the optimal threshold and leakage value interval, and the network energy efficiency with better accuracy is close. (5) The input encoding and inter-layer data sparsity show a negative correlation with inference accuracy, a positive correlation with energy efficiency, and a more ambiguous correlation with adversarial robustness, where the most important influencing factor is in the distribution of weights. In order to achieve the same issuing condition with parameters that have lower input encoding and inter-layer data sparsity during training, larger weights of local neurons are needed. However, when creating adversarial attack samples, the process works the exact opposite way, and the result of too large weights and perturbations superimposed on the original sample under the same perturbation will reduce inference accuracy and exhibit poor adversarial performance.

V. CONCLUSION

In this work we use the spiking activity sparsity among SNN layers as a main clue to analyze the impact of differ-

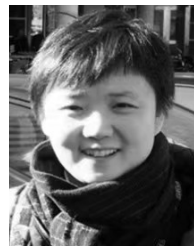
ent neural coding schemes, threshold voltage, leakage factor, and other combined structural parameters of the SNN based on STBP training through inference accuracy, adversarial robustness and energy efficiency. Four coding schemes are tested on two network architectures, VGG5 and VGG9. We also compared and analyzed the influence of SNN inherent structural parameters on the accuracy and robustness of the network from the perspective of sparsity. Based on the experimental results, while rate coding delivers the most adversarial robustness against medium and high intensity attacks, latency coding is the best option for achieving the highest adversarial robustness and energy efficiency. A comprehensive comparison among these schemes was provided, revealing that spike sparsification of the input data encoding scheme can bring improvements in adversarial robustness, but excessive sparsification of spike activity between network layers brought about by the combination of thresholds, leakage coefficients and combinatorial parameters can instead reduce the inference accuracy and adversarial robustness of the network. This study provides a reliable reference and insight for selecting and designing better neural coding schemes in neuromorphic systems and training optimal neural models. Neural network weight compression, quantization further affects inter-layer sparsity. The future research will focus on weight quantification for more accurate comparisons to natural hardware environments, giving designers of neuromorphic hardware better theoretical support.

REFERENCES

- [1] Y. Kim and P. Panda, "Optimizing deeper spiking neural networks for dynamic vision sensing," *Neural Netw.*, vol. 144, pp. 686–698, Dec. 2021.
- [2] D. V. Christensen, "2022 roadmap on neuromorphic computing and engineering," *Neuromorphic Comput. Eng.*, vol. 2, no. 2, May 2022, Art. no. 022501.
- [3] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, and, "TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, Oct. 2015.
- [4] M. Davies, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, Jan. 2018.
- [5] Y. Kuang, X. Cui, Y. Zhong, K. Liu, C. Zou, Z. Dai, Y. Wang, D. Yu, and R. Huang, "A 64K-neuron 64M-1b-synapse 2.64pJ/SOP neuromorphic chip with all memory on chip for spike-based models in 65nm CMOS," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 7, pp. 2655–2659, Jul. 2021.
- [6] S. Sharmin, N. Rathi, P. Panda, and K. Roy, "Inherent adversarial robustness of deep spiking neural networks: Effects of discrete input encoding and non-linear activations," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 12374. Cham, Switzerland: Springer, 2020, pp. 399–414.
- [7] R. El-Allami, A. Marchisio, M. Shafique, and I. Alouani, "Securing deep spiking neural networks against adversarial attacks through inherent structural parameters," 2020, *arXiv:2012.05321*.
- [8] A. Bagheri, O. Simeone, and B. Rajendran, "Adversarial training for probabilistic spiking neural networks," 2018, *arXiv:1802.08567*.
- [9] S. Sharmin, P. Panda, S. S. Sarwar, C. Lee, W. Ponghiran, and K. Roy, "A comprehensive analysis on adversarial robustness of spiking neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.
- [10] P. O'Connor and M. Welling, "Deep spiking networks," 2016, *arXiv:1602.08323*.
- [11] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: VGG and residual architectures," *Front Neurosci-Switz*, vol. 13, p. 95, Mar. 2019.
- [12] N. Rathi, G. Srinivasan, P. Panda, and K. Roy, "Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation," in *Proc. 8th Int. Conf. Learn. Represent. (ICLR)*. Addis Ababa, Ethiopia: OpenReview.net, Apr. 2020. [Online]. Available: <https://openreview.net/forum?id=B1xSperKvH>
- [13] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers Comput. Neurosci.*, vol. 9, p. 99, Aug. 2015.
- [14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
- [15] S. Kundu, M. Pedram, and P. A. Beerel, "HIRE-SNN: Harnessing the inherent robustness of energy-efficient deep spiking neural networks by training with crafted input noise," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5209–5218.
- [16] W. Guo, M. E. Fouda, and A. E. A. Eltawil, "Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems," *Front Neurosci-Switz*, pp. 1–21, 2021.
- [17] Y. Kim, H. Park, A. Moitra, A. Bhattacharjee, Y. Venkatesha, and P. Panda, "Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks?" in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 71–75.
- [18] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, "Training spiking neural networks using lessons from deep learning," 2021, *arXiv:2109.12894*.
- [19] W. Fang et al., "SpikingJelly," Multimedia Learn. Group, Inst. Digit. Media (NELVT), Peng Cheng Lab., Peking Univ., 2020. Accessed: Dec. 12, 2021. [Online]. Available: <https://github.com/fangwei123456/spikingjelly>
- [20] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.
- [21] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent. Vis. (ICLR)*, 2018, pp. 1–10.
- [22] M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 10–14.



YANJIE LI is currently pursuing the master's degree with the Institute of Microelectronics, Chinese Academy of Sciences. Her research interests include spiking neural networks algorithms and neuromorphic chip design.



XIAOXIN CUI (Member, IEEE) received the B.Sc. degree in cybernation from Beihang University, Beijing, China, in 2002, and the Ph.D. degree in microelectronic from Peking University, Beijing, in 2007. She is currently a Professor with the School of Integrated Circuits, Peking University. She is also a Research Fellow of the Peng Cheng Laboratory, Shenzhen, China. Her research interests include neuromorphic computing and neuro-inspired computing chips, energy-efficient AI accelerators, and hardware security.



YIHAO ZHOU (Member, IEEE) received the B.E. degree from the East China University of Science and Technology (ECUST), Shanghai, China, in 2003, and the M.S. degree from Florida International University (FIU), Miami, FL, USA, in 2005. He is currently a Senior Engineer with the Institute of Microelectronics of the Chinese Academy of Sciences (IMECAS). His research interests include SoC design and verification methods, hardware security architecture and verification methods, artificial intelligence and smart IoT, and coarse-grained reconfigurable architecture.



YING LI, JR. (Member, IEEE) received the bachelor's (B.S.) degree in microelectronics from Nankai University, China, in 2005, and the Ph.D. degree in circuit design and methodology from Peking University, China, in 2010. She is currently an Associate Professor with the Institute of Microelectronics, Chinese Academy of Sciences (IME CAS). During 2008 to 2010, she was a Visiting Scholar in computer science engineering with the University of California at San Diego, San Diego, USA. Her research interests include hardware security, IC design and methodology, and the Internet of Things. She is the Invited Reviewer of *Journal of Computer Science*, *Journal of Beijing University of Posts and Telecommunications*, IEEE PIMRC, and ATC.

...