SNN-RAT: Robustness-enhanced Spiking Neural Network through Regularized Adversarial Training

Anonymous Author(s)
Affiliation
Address
email

Abstract

Spiking neural networks (SNNs) are promising to be widely deployed in real-time and safety-critical applications with the advance of neuromorphic computing. Recent work has demonstrated the insensitivity of SNNs to small random perturbations due to the discrete internal information representation. The variety of training algorithms and the involvement of the temporal dimension pose more threats to the robustness of SNNs than that of typical neural networks. We account for the vulnerability of SNNs by constructing adversaries based on different differentiable approximation techniques. By deriving a Lipschitz constant specifically for the spike representation, we first theoretically answer the question of how much adversarial invulnerability is retained in SNNs. Hence, to defend against the broad attack methods, we propose a regularized adversarial training scheme with low computational overheads. SNNs can benefit from the constraint of the perturbed spike distance's amplification and the generalization on multiple adversarial ϵ-neighbourhoods. Our experiments on the image recognition benchmarks have proven that our training scheme can defend against powerful adversarial attacks crafted from strong differentiable approximations. To be specific, our approach makes the black-box attacks of the Projected Gradient Descent attack nearly ineffective. We believe that our work will facilitate the spread of SNNs for safety-critical applications and help understand the robustness of the human brain.

1 Introduction

Spiking Neural Networks (SNNs), unlike traditional Analog Neural Networks (ANNs), mimic the neuronal behaviours of a biological brain through spatio-temporal dynamics and spike representation [Gerstner et al., 2014], which represent the forefront of neural networks [Maass, 1997, Zenke et al., 2021]. The neurons evolve their membrane potentials as time goes by and transmit discrete information by 0 (nothing) and 1 (a spike). After the transmission, the membrane potentials are reset to the rest value and wait for the incoming input. Currently, the training algorithm of SNNs is a hot research topic due to the unique discrete spike activation. This results in a fundamental difference in training methods compared with ANNs. The rise of neuromorphic computing has allowed SNNs to run with more compatible hardware and a lower energy cost [Pei et al., 2019, DeBole et al., 2019, Davies et al., 2018, Nieves and Goodman, 2021, Fang et al., 2020]. The combination of SNN and neuromorphic hardware can enable numerous applications, such as spatio-temporal pattern recognition and high-speed detection [Wu et al., 2018a, Kim et al., 2020, Kheradpisheh and Masquelier, 2020, Zenke and Neftci, 2021].

For safety-critical applications such as autonomous driving, the reliability of the system becomes crucial, especially the robustness of the model against perturbations, such as additive Gaussian noise. Among all the perturbations, the adversarial attack is one of the most powerful categories [Szegedy

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

et al., 2014, Goodfellow et al., 2015]. It can generate subtle perturbations that are usually neglected by the human perception system. The perturbations, however, can deteriorate the capability of the system, that is, the model produces incorrect labels with a high probability. This could have serious impacts on those safety-related applications, where a single failure could have devastating results. Up to now, a wide variety of adversarial attack methods have been proposed [Madry et al., 2018]. The vulnerability holds even from a model that is trained for the same task but with a different architecture.

The operation mechanism and structure of SNN are similar to those of the biological brain, and studying its response to perturbation can help us understand how the human brain works. SNN is recognized as a new potential candidate with adversarial robustness due to its input coding and neuronal dynamics [Perez-Nieves et al., 2021, Leontev et al., 2021]. Among the coding schemes commonly used in SNN, constant input coding is considered to be more susceptible to disturbances than others, like Poisson coding [Sharmin et al., 2020]. Therefore, Kundu et al. [2021] suggested that careful training is required for constant input coding. In this setting, SNNs are now facing more challenges than typical ANNs. Because the key to constructing SNN gradient-based attacks is back-propagation, which is the same as that of ANNs. However, compared with ANNs, SNNs can learn through various gradient approximations. Therefore, combining various differentiable approximations and attack methods will pose a more severe threat to SNN [Liang et al., 2021].

The inter-layer communication of SNN is through spikes with a time dimension, which is very different from ANN. Therefore, one question can be raised naturally: whether and to what extent does spike communication detain adversarial invulnerability? And, are there training tools that can help SNNs defend against the threats described above? This paper aims to extend the Lipschitz analysis theory to spike representation and propose a more robust training algorithm on this basis. Our main contributions are summarized as follows:

- We design and summarize different differentiable approximations which can be deployed in gradient-based attacks to show the vulnerability of SNNs. Backward pass through time and rate are found to be capable of constructing stronger attacks.
- We theoretically analyze the l_2 perturbation distance on the representation of spikes and give a mathematical expression of the spiking Lipschitz constant.
- We propose a regularized adversarial training scheme for SNNs. It not only constrains the spiking Lipschitz constant but also exploits the mixture of the identified strong single-step adversarial attacks.
- Our experiments show that the proposed scheme can significantly improve the adversarial robustness in the image recognition tasks. The trained model exhibits better resistance under stronger Projected Gradient Descent attacks compared to the vanilla models.

2 Background and Related Work

2.1 Robustness of Spiking Neural Networks

The concept of spike representation is where SNNs are different from ANNs. Typically, the input is encoded as sequences ($T \times N_0$), with T denoting the total number of time-steps and N_0 denoting the number of the input nodes. The neurons in SNNs, like ANNs, receive a linear combination of the previous layer's output. The mechanism of leaky-integrate-fire (LIF) leads to the non-linearity nature of SNNs. Overall, the dynamic of the membrane potential $\boldsymbol{m}^l(t)$ of neurons in layer l ($l = 1, 2, \cdots, L$) at time-step t ($t = 1, 2, \cdots, T$) can be described by:

$$\boldsymbol{m}^l(t^-) = \boldsymbol{m}^l(t-1) + \boldsymbol{W}^l \boldsymbol{s}^{l-1}(t), \tag{1}$$

$$\boldsymbol{s}^l(t) = H(\boldsymbol{m}^l(t^-) - \theta), \tag{2}$$

$$\boldsymbol{m}^l(t) = \lambda \boldsymbol{m}^l(t^-)(1 - \boldsymbol{s}^l(t)), \tag{3}$$

where $\boldsymbol{s}^l(t)$ denotes the binary spikes of neurons in layer l at time t, which equals to 1 if there is spike. $\boldsymbol{m}^l(t^-)$ denotes the instantaneous state of membrane potential before triggering a spike, which accumulates the weighted input from presynaptic neurons in the last layer. When the potential exceeds the predefined threshold θ, a spike is generated at time t, and the membrane potential $\boldsymbol{m}^l(t)$

is reset to zero. Otherwise the membrane potential $\boldsymbol{m}^l(t)$ either leaks by λ for LIF model ($\lambda \in (0,1)$), or maintains for Integrate-and-Fire (IF) model ($\lambda = 1$).

The leaking potentials described above intuitively smooth the noise in the input current, which is an appealing characteristic for trustworthy neural networks. Indeed, SNNs can demonstrate robustness under certain conditions. Sharmin et al. [2020] presented their pioneering work on input discretization and leaky rate. They highlighted that specific input coding could improve the robustness of SNN. Up to now, Poisson coding, latency coding, and time-to-first-spike coding have been proven to have an effect on small datasets [Sharmin et al., 2020, Leontev et al., 2021, Nomura et al., 2022]. Sharmin et al. [2019] also found that SNNs trained by surrogate functions can boost the robustness of Poisson coding. Whereas, SNNs are not completely secure. Marchisio et al. [2020] pointed out that black-box attacks can also attack SNNs. El-Allami et al. [2021] searched for the structural parameters of SNNs to improve the robustness. Among rate coding, constant input coding (or direct coding) is shown to detain less robustness, which has been discussed in several papers [Kim et al., 2022, Kundu et al., 2021]. Based on careful observations, Kundu et al. [2021] asserted that fine training is essential for the robustness of direct-coding SNNs. Motivated by this, we propose a regularizer that can promote robust training schemes for SNN.

2.2 Gradient-based Adversarial Attacks

Most adversarial attacks consider a perturbation $\boldsymbol{\delta}$ in an l_p ball around clean data $\boldsymbol{x}$ and can be formulated as an optimization problem:

$$\arg\max_{\boldsymbol{\delta}} \mathcal{L}(f(\boldsymbol{x}+\boldsymbol{\delta};\boldsymbol{W}), y) \quad s.t. \ \|\boldsymbol{\delta}\|_p \leq \epsilon, \tag{4}$$

where y is the target, $\mathcal{L}$ is the loss function, f denotes the network with $\boldsymbol{W}$ as parameters, and ϵ is the parameter that guarantees the perturbation is imperceptible. Here, we introduce two widely adopted gradient-based adversarial attack algorithms: Fast Gradient Sign Method (FGSM) and Projected Gradient Descent method (PGD).

FGSM. As one of the simplest methods, the main idea of FGSM is to perturb the data along the sign of the gradient to increase the perturbed linear output, which can be expressed as follows [Goodfellow et al., 2015]:

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \epsilon \ \text{sign}(\nabla_{\boldsymbol{x}} \mathcal{L}(f(\boldsymbol{x}, y))). \tag{5}$$

PGD. PGD is an iterative version of FGSM, which offers a more powerful attack [Madry et al., 2018] and is convinced to reasonably approximate the optimal attack. The iteration can be summarized as:

$$\widetilde{\boldsymbol{x}}^k = \Pi_\epsilon\{\boldsymbol{x}^{k-1} + \alpha \ \text{sign}(\nabla_{\boldsymbol{x}} \mathcal{L}(f(\boldsymbol{x}^{k-1}, y)))\}, \tag{6}$$

where k denotes the number of the iteration step and α is the step size of each iteration. Π_ϵ denotes that the data in each iteration should be projected onto the space of the l_p ball around clean data $\boldsymbol{x}$ w.r.t. ϵ. Apart from these popular gradient-based attacks, RFGSM can be viewed as a randomized version of FGSM [Tramèr et al., 2018], and BIM is a kind of iterative attack similar to PGD [Kurakin et al., 2017]. All these methods are exploited to verify the vulnerability of SNN in this work. By applying differentiable approximations in back-propagation, the gradient-based attacks can also threaten SNN. In this work, we execute white-box (WB) and black-box (BB) attacks for scenarios where attackers have knowledge of or no knowledge of the model.

Without specific instructions, we set ϵ to $8/255$ for all methods for the purpose of testing. For iterative methods like PGD and BIM, the attack step $\alpha = 0.01$, and the step number is 7.

2.3 Defense Methods

The earlier research on ANN has discovered that adversarial attacks can cause the amplification of activation magnitude [Szegedy et al., 2014]. They measured the distance between clean and perturbed activation and suggested an analysis framework based on Lipschitz analysis as:

$$\|\boldsymbol{a}^{l+1} - \widetilde{\boldsymbol{a}}^{l+1}\|_2 \leq \text{Lip}_l \|\boldsymbol{a}^l - \widetilde{\boldsymbol{a}}^l\|_2, \tag{7}$$

where Lip_l is the Lipschitz constant for layer l. By penalizing the Lipschitz constant, the distortion of input is stabilized [Cisse et al., 2017]. Using exact estimation of Lipschitz bounds to certify the robustness of ANN has been popular [Arjovsky et al., 2017, Fazlyab et al., 2019, Weng et al., 2018, Miyato et al., 2018]. Yet there is no work currently working on the Lipschitz bound of SNN.

Apart from the analysis framework, adversarial training is the most powerful tool for defense, which is defined as a saddle point problem [Madry et al., 2018]:

$$\arg\min_{\boldsymbol{W}} \mathbb{E}\left[\max_{\boldsymbol{\delta}} \mathcal{L}(f(\boldsymbol{x}+\boldsymbol{\delta};\boldsymbol{W}), y)\right], \tag{8}$$

where the maximization process can be accomplished by applying different attack methods. By utilizing the adversarial input, the adversarial training learns to classify adversarial examples correctly. All these methods are based on deep ANNs, which are locally differentiable and have well-defined derivatives. In this paper, we aim to construct an analysis framework for non-differentiable SNNs and strengthen the adversarial training.

3 Vulnerability of Spiking Neural Networks

3.1 Spiking Neural Network under Attack

Although SNNs are more robust than ANNs under some conditions, most SNNs are still vulnerable when effective adversarial attack methods are applied. Previous works have experimentally demonstrated that gradient attack methods such as FGSM can be applied to SNNs [Sharmin et al., 2019, 2020]. However, because of the non-differentiable property of spiking neurons, the gradients obtained by backpropagation are not necessarily accurate, which may cause inefficient attacks. Thus, this defensive nature of SNNs can be thought of as obfuscated gradients [Athalye et al., 2018], which may create a false sense of security about SNNs.

Therefore, in this paper, we first reconsider the attacks for SNNs by utilizing a combination of attack methods and gradient approximations. We follow the Backward Pass Differentiable Approximation (BPDA) technique [Athalye et al., 2018] to overcome the obfuscated gradients. The key idea of the BPDA is to use a differentiable approximation in the backward pass while the forward pass unchanged. Similar differentiable approximation techniques have been used for training SNNs. Based on these, we design, compare, and summarize different differentiable approximations for SNNs.

3.2 Differentiable Approximation for Spiking Neural Networks

Conversion-based Approximation. The Conversion-based Approximation (CBA) for SNN was first proposed by Sharmin et al. [2019]. Since an SNN can be converted from an ANN Rueckauer et al. [2017], Han et al. [2020], the adversarial examples can be generated from an ANN with shared weights and bias from the source SNN. However, this method approximates the spiking neuron using the ReLU activation function on both forward pass and backward pass, which is contrary to the idea of the BPDA algorithm and is proved to be ineffective [Song et al., 2018].

Backward Pass Through Time. The most commonly used differentiable approximation is the Backward Pass Through Time (BPTT) with surrogate gradients. In this method, the non-differentiable neuron fire function is replaced by a differentiable function on the backward pass. By combining Eq. 1–Eq. 3, the backward pass can be described as follows:

$$\frac{\partial L}{\partial \boldsymbol{s}^l(t)} = \frac{\partial L}{\partial \boldsymbol{s}^{l+1}(t)} \frac{\partial \boldsymbol{s}^{l+1}(t)}{\partial \boldsymbol{m}^{l+1}(t^-)} \frac{\partial \boldsymbol{m}^{l+1}(t^-)}{\partial \boldsymbol{s}^l(t)} \tag{9}$$

$$+ \frac{\partial L}{\partial \boldsymbol{s}^l(t+1)} \frac{\partial \boldsymbol{s}^l(t+1)}{\partial \boldsymbol{m}^l(t+1^-)} \frac{\partial \boldsymbol{m}^l(t+1^-)}{\partial \boldsymbol{m}^l(t)} \frac{\partial \boldsymbol{m}^l(t)}{\partial \boldsymbol{s}^l(t)}. \tag{10}$$

The final result $\frac{\partial L}{\partial \boldsymbol{s}^0(t)}$ can be calculated by recursively calculating this equation across both layers and time-steps. And since we use the constant input coding for SNNs, the gradient of the image is exactly $\frac{\partial L}{\partial \boldsymbol{s}^0(t)}$. The non-differentiable part $\frac{\partial \boldsymbol{s}^{l+1}(t)}{\partial \boldsymbol{m}^{l+1}(t^-)}$ is replaced by a surrogate gradient function to get a smooth backward pass. The backward pass through time technique uses a similar gradient approximation as the Spatio-Temporal-Backward-Propagation training algorithm [Wu et al., 2018b]. Since the gradient generated from this method is useful for training, it is very likely to generate effective gradients.

Backward Pass Through Rate. Another differentiable approximation is the Backward Pass Through Rate (BPTR). In this method, the backward pass takes the derivative directly from the average firing

Table 1: Performance comparison between differentiable approximations (CBA/BPTR/BPTT).

Architecture	Dataset	CLEAN	FGSM	RFGSM	PGD
VGG-11	CIFAR-10	93.06	54.34/10.59/12.78	72.12/27.28/22.09	37.30/00.10/00.04
WRN-16	CIFAR-10	94.38	63.32/16.46/14.13	77.42/15.46/10.17	59.09/00.00/00.00
VGG-11	CIFAR-100	73.33	41.80/06.10/05.30	57.73/11.95/08.60	38.65/00.18/00.02
WRN-16	CIFAR-100	75.33	37.68/07.94/07.62	54.16/05.99/04.80	43.87/00.05/00.00

rate of the spiking neurons between layers. We consider neurons at each time-step equivalent, and the derivative is determined by the average firing rate.

$$\forall t \in \{1, 2, \cdots, T\}, \quad \frac{\partial L}{\partial \boldsymbol{s}^l(t)} = \frac{\partial L}{\partial \boldsymbol{s}^{l+1}(t)} \frac{\partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l+1}(i)}{\partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l}(i)}. \tag{11}$$

Since the relationship of the firing rates in adjacent layers for non-leaky IF neuron is nearly linear[Sengupta et al., 2019], the gradients of $\partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l+1}(i) / \partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l}(i)$ can be approximated using the straight-through estimator [Bengio et al., 2013]. Similar to Lee et al. [2020], here we use the constant $\frac{1}{T}$ to approximate the value of $\partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l+1}(i) / \partial \frac{1}{T}\sum_{i=0}^{T} \boldsymbol{s}^{l}(i)$ at Eq. 11. The backward pass of the complete neuronal dynamic is approximated by one single function, and the gradient will not accumulate through time-steps. As the forward pass still follows the rule of the spiking neurons, the obtained gradients will be more accurate than the conversion-based attack.

3.3 Effective Attack with Backward Pass through Time and Rate

To compare the effectiveness of the above three differentiable approximation techniques in constructing gradient-based attacks for SNNs, we applied combinations of three gradient-based methods (FGSM, RFGSM, PGD) and three differentiable approximation techniques (CBA, BPTR, BPTT) to two baseline models. The two models, VGG-11 and WideResNet-16, are trained on the CIFAR dataset using BPTT with no additional defenses. The number of time-steps is set to $T = 8$.

As shown in Tab. 1, the CBA is the most ineffective differentiable approximation technique to construct attacks for SNNs. The performance of all models under attack using CBA is significantly higher than that of models under attack using BPTR and BPTT. As we have discussed in Sec. 3.2 when using CBA, the changes in both forward pass and backward pass cause inaccuracy in the generated gradient. Both the BPTT and BPTR methods, combined with different attack methods, can significantly reduce the performance of the given model. Among all single-step attack combinations, FGSM(BPTR), RFGSM(BPTT), FGSM(BPTT), and RFGSM(BPTT) can generate efficient and effective adversarial examples for all the testing models. Among all multi-step attacks, the PGD(BPTT) always beats other attack combinations. These results suggest different properties between BPTT and BPTR. The BPTT approximation contains extra temporal information that is useful when using multi-step attacks, while the BPTR approximation saves more computation resources when back-propagating through firing rates and can get comparable results on single-step attacks.

4 Methods: Perturbation Analysis and Regularized Adversarial Training

The previous section identifies threats from effective differentiable approximations combined with gradient-based attacks. In this section, we will give a theoretical perturbation analysis under SNN attacks and propose a regularized adversarial training scheme.

4.1 Perturbation Analysis for Spike Representation

Current perturbation analysis of ANN considers the distance of continuous activation $\|\boldsymbol{a}^l - \widetilde{\boldsymbol{a}}^l\|$ and the rectified linear outputs $\|f(\boldsymbol{a}^l) - f(\widetilde{\boldsymbol{a}}^l)\|$. Kundu et al. [2021] used this rate-based distance to bridge the robustness of ANN and SNN. Spike trains in SNN not only contain rate information but also have a temporal structure. To evaluate the distance in the spike train space, various kernel methods are proposed for neuronal identification and encoding [Weng et al., 2018]. Inspired by

these works, we propose to model the distortion of the spike response using the spike train distance, which can bridge the robustness of SNN to the discovery of neuroscience and is also sensitive to the change of both firing rate and temporal information. Denote the output spike train of the lth layer as $\boldsymbol{S}^l = \{\boldsymbol{s}^l(t)|t = 1, 2, \cdots, T\} \in \chi^{T \times N_l}$ ($\chi \in \{0, 1\}$), where T is the number of time-steps and N_l is the number of neurons in layer l. Then the perturbation distance can be formulated as ($p \geq 1$):

$$D_p(\boldsymbol{S}^l, \widetilde{\boldsymbol{S}}^l) = \|\boldsymbol{S}^l - \widetilde{\boldsymbol{S}}^l\|_{p,p} = \left(\sum_{t=1}^{T} \|\boldsymbol{s}^l(t) - \widetilde{\boldsymbol{s}}^l(t)\|_p^p \right)^{1/p}, \tag{12}$$

where $\|\cdot\|_{p,p}$ denotes the l_p entry-wise matrix norm and $\|\cdot\|_p$ denotes the l_p vector norm. $\widetilde{\boldsymbol{S}}^l$ is the perturbed version of $\boldsymbol{S}^l$. Since the p power of spike activation values (0 and 1) are themselves, without loss of generality we set $p = 2$. In Theorem 1, we obtain a Lipschitz constant for the spike train distance.

Theorem 1. *Given an L-layered SNN intended to inference T time-steps with θ as threshold, suppose that there are N_l neurons in layer l for $l = 1, 2, \cdots, L$. $\boldsymbol{W}^l \in \mathbb{R}^{N_l \times N_{l-1}}$. For layer l, it satisfies:*

$$D_2(\boldsymbol{S}^l, \widetilde{\boldsymbol{S}}^l)^2 \leq \frac{1}{\theta^2} {\Lambda^l}^2 D_2(\boldsymbol{S}^{l-1}, \widetilde{\boldsymbol{S}}^{l-1})^2 + \Gamma^l, \tag{13}$$

where Λ^l is a Lipschitz constant and Γ^l is a constant for layer l, which can be expressed as:

$$\Lambda^l = \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \psi^{N_{l-1}}} \frac{\|\boldsymbol{W}^l \boldsymbol{s}\|_2}{\|\boldsymbol{s}\|_2}, \tag{14}$$

$$\Gamma^l = \frac{N_l T(T+1)}{\lambda} \left[\frac{\gamma^l}{\theta} + \left(\frac{\gamma^l}{\theta} \right)^2 \right], \tag{15}$$

where $\gamma^l = \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \chi^{N_{l-1}}} \|\boldsymbol{W}^l \boldsymbol{s}\|_\infty + \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \chi^{N_{l-1}}} \|-\boldsymbol{W}^l \boldsymbol{s}\|_\infty$. $\chi = \{0, 1\}, \psi = \{-1, 0, 1\}$.

In Eq. 14, Λ^l is referred to as the **spiking Lipschitz constant**. Since the inter-layer spike signals are not purely linear and the effect of the spiking generation mechanism cannot be neglected, the inequality relationship expressed by Eq. 13 is not exactly the same as the definition of the classical Lipschitz constant [O'Searcoid, 2006]. Nonetheless, this does not affect our understanding of the amplification effect of spike distance. On the right side of Eq. 13, there is an additional constant Γ^l related to the weight. The appearance of Γ^l is associated with the spike generation and the bounding of the current. The determination of Γ^l adopts a very loose constraint (see our proof in the Appendix), and its effect on the spike distance is additive, so we will still focus on how to constrain the spiking Lipschitz constant while not magnifying the additive item excessively.

4.2 Constraining Spiking Lipschitz Constant

Here we explain why SNNs are thought to be more robust than ANNs and how to constrain the spiking Lipschitz constant in Eq. 13. Szegedy et al. [2014] has concluded that the upper bound of the Lipschitz constant for ReLU-based layers is the largest singular value of the weight matrix, i.e., $\sup_{\boldsymbol{x} \neq 0} \frac{\|\boldsymbol{W}^l \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}$ for layer l. The difference between the classical Lipschitz constant and the spiking Lipschitz constant lies in the vector domain. The classical Lipschitz constant is the attained supremum for any vector $\boldsymbol{x} \neq 0$. In the case of the SNN layer, its Lipschitz constant is constrained in the space $\{\boldsymbol{x} \neq 0, \boldsymbol{x} \in \psi^{N_{l-1}}, \psi = \{-1, 0, 1\}\}$, a subspace of $\boldsymbol{x} \neq 0$. Inspired by this, we can derive an inequality between the classical Lipschitz constant and the spiking Lipschitz constant, which is presented in Proposition 1 (The detailed proof is in the Appendix).

Proposition 1. *Given a weight matrix with real values $\boldsymbol{W}^l \in \mathbb{R}^{N_l \times N_{l-1}}$. $\psi = \{-1, 0, 1\}$. It satisfies:*

$$\Lambda^l = \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \psi^{N_{l-1}}} \frac{\|\boldsymbol{W}^l \boldsymbol{s}\|_2}{\|\boldsymbol{s}\|_2} \leq \|\boldsymbol{W}^l\|_2 = \sigma^{\max}(\boldsymbol{W}^l), \tag{16}$$

where $\|\boldsymbol{W}^l\|_2$ is the induced l_2 matrix norm, and $\sigma^{\max}(\boldsymbol{W}^l)$ is the largest singular value of $\boldsymbol{W}^l$.

According to Eq. 16, the spiking Lipschitz constant Λ^l is less than $\sigma^{\max}(\boldsymbol{W}^l)$. The primary goal is to constrain the training through a weight regularizer. However, Λ^l is hard to estimate, whereas there

are numerous ways to modulate the spectral norm $\sigma^{\max}(\boldsymbol{W}^l)$ [Miyato et al., 2018, Cisse et al., 2017, Yoshida and Miyato, 2017]. Thus, we choose to adopt spectral norm regularization into our proposed training scheme. The target of regularization is to control the spectral norm to approach 1:

$$\text{Goal: } \Lambda^l \to 1 \quad \text{Implementation: } \sigma^{\max}(\boldsymbol{W}^l) \to 1. \tag{17}$$

Such that according to Eq. 16, we can also meet the criterion that Λ^l is less than 1 for each layer l. We note that the constraint to $\|\boldsymbol{W}^l\|_2$ can also contribute to limiting γ^l, which is presented in Proposition 2 (The detailed proof is in the Appendix).

Proposition 2. *Given a weight matrix with real values $\boldsymbol{W}^l \in \mathbb{R}^{N_l \times N_{l-1}}$. $\chi = \{0, 1\}$. It satisfies:*

$$\gamma^l = \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \chi^{N_{l-1}}} \|\boldsymbol{W}^l \boldsymbol{s}\|_\infty + \sup_{\boldsymbol{s} \neq 0, \boldsymbol{s} \in \chi^{N_{l-1}}} \| - \boldsymbol{W}^l \boldsymbol{s}\|_\infty \leq 2\|\boldsymbol{W}^l\|_\infty \leq 2\sqrt{N_{l-1}}\|\boldsymbol{W}^l\|_2, \tag{18}$$

where $\|\boldsymbol{W}^l\|_p$ is the induced l_p matrix norm.

In Proposition 2, the relationship between l_∞ norm and l_2 norm is based on the norm inequality theory. The item in γ^l defined in a constrained space, on the other hand, is strictly less than the l_∞ norm of $\boldsymbol{W}^l$. Thus, we can conclude γ^l has an upper bound determined by scaled $\sigma^{\max}(\boldsymbol{W}^l)$. Controlling $\sigma^{\max}(\boldsymbol{W}^l)$ is sufficient to limit Γ^l as Γ^l monotonously increases with the increase of γ^l.

4.3 Regularized Adversarial Training (RAT)

Application of regularization (REG). Once the target of regularization is set to control the matrix norm of weights, the next question is how to find a good regularizer to assist training. By setting the threshold of spiking neurons to 1, the amplification of spike distance is mainly related to Γ. Given that SNN frequently suffers from gradient problems during training [Wu et al., 2018b], we eventually settle on an orthogonal regularization method that can help prevent the network from vanishing gradients [Lin et al., 2021]. Singular values are naturally equal to 1 when the weight matrix row is orthogonal. Therefore, we propose to project the updating weights to the target of the orthogonal matrix at a rate defined by β [Cisse et al., 2017]:

$$\forall l = 1, 2, \cdots, L, \quad \boldsymbol{W}^l \leftarrow \Pi_{\boldsymbol{W}^T\boldsymbol{W}=\boldsymbol{I}}\left(\boldsymbol{W}^l\right) \tag{19}$$

$$\Pi_{\boldsymbol{W}^T\boldsymbol{W}=\boldsymbol{I}}(\boldsymbol{W}) = \boldsymbol{W} - \beta\left(\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{W} - \boldsymbol{W}\right) \tag{20}$$

For convolutional layers that have weight matrices of 4 dimensions $\boldsymbol{W} \in \mathbb{R}^{C_{out} \times C_{in} \times k \times k}$, these matrices should be first reorganized into 2-dimensional matrices in $\mathbb{R}^{C_{out} \times (C_{in} \times k \times k)}$ to accomplish the update, where k is the kernel size and C_{in}, C_{out} are numbers of input and output channels.

Generalizing strong SNN adversarial examples (MIX). As Sec. 3 suggested, SNNs are also vulnerable under different approximations of the backward pass. To equip the trained SNN with even better invulnerability, we argument the training dataset with stronger adversarial perturbations based on our observations. The analyses and results in Sec. 3 imply that the single-step FGSM and RFGSM can provide us with adequate adversaries at a relatively low computational cost. It is worth noting that RFGSM is recognized as an effective substitution for PGD in adversarial learning [Wong et al., 2019]. The BPTT and BPTR approximations further allow the dataset to extend to its l_∞ balls with a strong adversary compared with the CBA. Hence, our model is trained with adversarial examples randomly sampled from {FGSM(BPTT), RFGSM(BPTT), FGSM(BPTR), RFGSM(BPTR)}. Each mini-batch of images are perturbed by a single choice from the four methods with equal probability before being fed into the network. The network learns to generalize in the mixture of heterogeneous neighbourhoods defined by the four adversaries. For better network training performance, we choose the mainstream method where the network is updated by the gradients produced from BPTT with surrogate functions [Neftci et al., 2019, Zheng et al., 2021, Fang et al., 2021].

5 Experiments

5.1 Experimental Setup

We validate our proposed robust SNN training scheme on the image classification tasks, where the CIFAR-10 and CIFAR-100 datasets are used. We train the SNN version of VGG-11 and WideResNet-16 (with widening factor set as 4) for both datasets with $T = 8$. In addition, we set $\beta = 0.001$ and

0.004 for VGG-11 and WideResNet-16, respectively. The perturbation boundary ϵ is set to 2/255 when training models. Detailed implementation is referred to in Appendix.

The images are directly fed into SNNs. We include several gradient-based attack methods to evaluate the adversarial performance thoroughly: FGSM [Goodfellow et al., 2015], RFGSM [Tramèr et al., 2018], PGD [Madry et al., 2018], BIM [Kurakin et al., 2017]. Gaussian noise (GN) is also used to test the performance against random perturbation with the same ϵ.

5.2 Results

Performance under attacks. Tab. 2 reports the performance of our proposed RAT scheme. All gradient attacks are combined with powerful SNN attacks (BPTT, BPTR). The classification accuracy in brackets is the accuracy without the proposed training scheme. Black-box attacks are marked with "*" in the table. It is observed that for all the attack methods, our RAT can improve the model robustness, which is reflected in the improvement of accuracy. Compared with VGGs, Vanilla WideResNets are more vulnerable to RFGSM attacks. However, armed with RAT, it is even more robust than VGGs. Black-box attacks are almost ineffective against RAT-trained models. For stronger white-box iterative attacks, our RAT improves the robustness from being almost completely misclassified. For example, the VGG-11 model increases its accuracy by 34.76% under the PGD(BPTR) attack after being trained with RAT on the CIFAR-10 dataset.

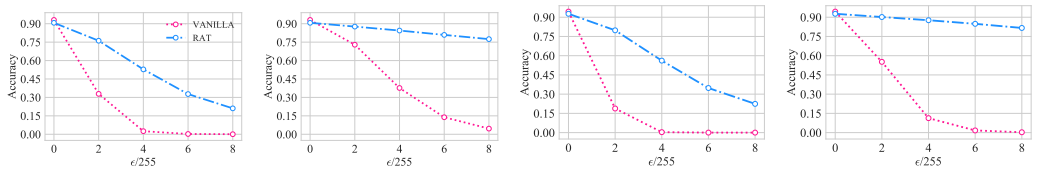
Performance with larger ϵ. We plot the accuracy of the white-box and black-box scenarios under the PGD(BPTT) attack in Fig. 1. For white-box attacks, both models converge to almost zero after $\epsilon = 4/255$, while RAT-trained models drop their accuracy with increasing ϵ. Black-box attacks are weaker than white-box ones. From Fig. 1(b) and (d), the accuracy of RAT-trained models decreases comparably slowly to that of vanilla models.

5.3 Ablation Studies

The proposed RAT scheme in Sec. 4.3 is composed of a regularizer to control the spiking Lipschitz constant and mixed adversarial neighbourhoods for adversarial training. We conduct ablation studies based on VGG-11 trained with the CIFAR-10 dataset. The results are shown in Tab. 3. The attack methods are all BPTT attacks. When trained without RAT, the performances under single-step FGSM and RFGSM are lower than 25%, and the accuracies of PGD and BIM are nearly zero. With only the application of the regularization, the performance of FGSM and RFGSM increase to 26.6% and

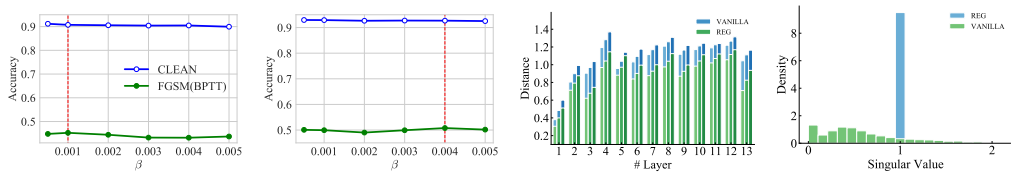
Table 2: Performance of the proposed RAT under different attacks.

Attack	VGG/CIFAR10	WRN/CIFAR10	VGG/CIFAR100	WRN/CIFAR100
Clean	90.74(93.06)	92.69(94.38)	70.89(73.33)	69.32(75.33)
GN	90.31(90.77)	**91.35**(83.08)	66.01(66.77)	**66.28**(49.86)
FGSM(BPTR)	**51.77**(10.59)	**55.70**(16.46)	**28.24**(6.10)	**31.75**(07.94)
FGSM(BPTT)	**45.23**(12.78)	**50.78**(14.13)	**25.86**(5.30)	**28.08**(07.62)
RFGSM(BPTR)	**70.69**(27.28)	**73.32**(15.46)	**43.50**(11.95)	**45.68**(05.99)
RFGSM(BPTT)	**64.61**(22.09)	**69.30**(10.17)	**38.72**(8.60)	**40.51**(04.80)
PGD(BPTR)	**34.86**(00.10)	**36.36**(00.00)	**18.15**(0.18)	**19.32**(00.05)
PGD(BPTT)	**21.16**(00.04)	**22.71**(00.00)	**10.38**(00.02)	**11.31**(00.00)
BIM(BPTR)	**33.29**(00.08)	**34.92**(00.00)	**17.81**(00.17)	**18.89**(00.05)
BIM(BPTT)	**18.64**(00.03)	**19.58**(00.00)	**09.50**(00.03)	**10.25**(00.00)
FGSM(BPTR)*	**75.71**(32.11)	**77.66**(25.25)	**48.54**(18.31)	**50.11**(13.95)
FGSM(BPTT)*	**73.76**(28.51)	**76.96**(24.49)	**47.25**(16.81)	**50.39**(13.50)
RFGSM(BPTR)*	**83.79**(56.82)	**86.45**(31.71)	**59.67**(32.66)	**60.73**(17.80)
RFGSM(BPTT)*	**83.35**(49.91)	**85.90**(26.68)	**58.71**(30.06)	**60.06**(15.71)
PGD(BPTR)*	**77.15**(12.02)	**82.81**(00.94)	**55.44**(10.54)	**59.23**(03.31)
PGD(BPTT)*	**77.43**(04.61)	**82.20**(00.31)	**54.11**(07.04)	**58.28**(02.53)
BIM(BPTR)*	**76.16**(10.57)	**81.16**(00.88)	**54.10**(09.53)	**56.52**(03.53)
BIM(BPTT)*	**75.89**(04.00)	**79.93**(00.29)	**52.60**(06.42)	**55.86**(02.49)



(a) VGG-11 white-box (b) VGG-11 black-box (c) WRN-16 white-box (d) WRN-16 black-box

Figure 1: Performance under PGD(BPTT) attacks.



(a) VGG-11 β Test (b) WRN-16 β Test (c) Perturbation Distance (d) Singular values

Figure 2: Effects of Regularized Adversarial Learning.

40.9%. The single deployment of mixed adversarial training significantly improves the robustness as the combination of heterogeneous neighbourhoods enhances data generalization. When both the regularizer and the adversarial training are used, the model gets the best robustness. These results indicate that each component in RAT alone can improve the model's adversarial performance, and the robustness of the model is greatly improved after they are combined.

Table 3: Ablation study with VGG-11 on CIFAR-10.

MIX	**REG**	Clean	FGSM	RFGSM	PGD	BIM
×	L2	93.06	12.78	22.09	0.04	0.03
×	✓	91.09	26.60	40.89	0.85	0.55
✓	L2	92.04	37.49	60.11	16.63	15.28
✓	✓	90.74	45.23	64.61	21.16	18.64

5.4 Effects of Regularized Adversarial Learning

We test the sensitivity of the orthogonal projection rate β on the CIFAR-10 dataset with VGG-11 and WideResNet-16 structures. As shown in Fig. 2(a) and (b), β has little effect on clean accuracy. For the two architectures, 0.001 and 0.004 give the best accuracy under the FGSM(BPTT) attack, respectively. To validate the effect of the regularization, the perturbation distance of each neuron layer in WideResNet-16 is visualized. In Fig. 2(c), the colors range from light to dark denote $\epsilon = 2, 4, 8$. It can be seen that the distances of the regularized model are consistently less than those of the vanilla model, which indicates that regularization inhibits the amplification of the spike distance. Fig. 2(d) presents the distribution of singular values of the WideResNet-16 model. The regularization makes singular values of weights to be aggregated at 1 as expected.

6 Conclusions

In this work, we are the first to give a theoretical analysis of perturbation for hardware-friendly SNNs. SNNs are more vulnerable due to their diverse and feasible gradient methods. Therefore, we propose a specialized regularization adversarial training scheme for SNNs. Our experiments demonstrate that models trained on this scheme can obtain much robustness, especially in black-box attacks. We believe this work will pave the way for SNNs in energy-efficient as well as safety-critical applications.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 38th International Conference on Machine Learning*, pages 274–283. PMLR, 2018.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 38th International Conference on Machine Learning*, pages 854–863. PMLR, 2017.

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.

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

Rida El-Allami, Alberto Marchisio, Muhammad Shafique, and Ihsen Alouani. Securing deep spiking neural networks against adversarial attacks through inherent structural parameters. In *Design, Automation & Test in Europe Conference & Exhibition*, pages 774–779. IEEE, 2021.

Biao Fang, Yuhao Zhang, Rui Yan, and Huajin Tang. Spike trains encoding optimization for spiking neural networks implementation in fpga. In *International Conference on Advanced Computational Intelligence*, pages 412–418. IEEE, 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.

Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations*, 2015.

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 International Conference on Computer Vision*, pages 13558–13567, 2020.

Saeed Reza Kheradpisheh and Timothée Masquelier. Temporal backpropagation for spiking neural networks with one spike per neuron. *International Journal of Neural Systems*, 30(06):2050027, 2020.

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*, pages 11270–11277, 2020.

Youngeun Kim, Hyoungseob Park, Abhishek Moitra, Abhiroop Bhattacharjee, Yeshwanth Venkatesha, and Priyadarshini Panda. Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks? In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 71–75. IEEE, 2022.

Souvik Kundu, Massoud Pedram, and Peter A Beerel. Hire-SNN: Harnessing the inherent robustness of energy-efficient deep spiking neural networks by training with crafted input noise. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5209–5218, 2021.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations, Workshop Track Proceedings*, 2017.

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*, page 119, 2020.

Mikhail Leontev, Dmitry Antonov, and Sergey Sukhov. Robustness of spiking neural networks against adversarial attacks. In *2021 International Conference on Information Technology and Nanotechnology (ITNT)*, pages 1–6. IEEE, 2021.

Ling Liang, Xing Hu, Lei Deng, Yujie Wu, Guoqi Li, Yufei Ding, Peng Li, and Yuan Xie. Exploring adversarial attack in spiking neural networks with spike-compatible gradient. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Zinan Lin, Vyas Sekar, and Giulia Fanti. Why spectral normalization stabilizes GANs: Analysis and improvements. *Advances in Neural Information Processing Systems*, 34:9625–9638, 2021.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the 38th International Conference on Machine Learning*, 2018.

Alberto Marchisio, Giorgio Nanfa, Faiq Khalid, Muhammad Abdullah Hanif, Maurizio Martina, and Muhammad Shafique. Is spiking secure? A comparative study on the security vulnerabilities of spiking and deep neural networks. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2020.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

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.

Nicolas Perez Nieves and Dan F. M. Goodman. Sparse spiking gradient descent. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, pages 11795–11808, 2021.

Osamu Nomura, Yusuke Sakemi, Takeo Hosomi, and Takashi Morie. Robustness of spiking neural networks based on time-to-first-spike encoding against adversarial attacks. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2022.

Mícheál O'Searcoid. *Metric spaces*. Springer Science & Business Media, 2006.

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, Vincent CH Leung, Pier Luigi Dragotti, and Dan FM Goodman. Neural heterogeneity promotes robust learning. *Nature communications*, 12(1):1–9, 2021.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017.

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 *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2019.

Saima Sharmin, Nitin Rathi, Priyadarshini Panda, and Kaushik Roy. Inherent adversarial robustness of deep spiking neural networks: Effects of discrete input encoding and non-linear activations. In *European Conference on Computer Vision*, pages 399–414. Springer, 2020.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. PixelDefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285. PMLR, 2018.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2019.

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, 2018a.

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, 2018b.

Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

Friedemann Zenke and Emre O Neftci. Brain-inspired learning on neuromorphic substrates. *Proceedings of the IEEE*, 109(5):935–950, 2021.

Friedemann Zenke, Sander M Bohté, Claudia Clopath, Iulia M Comşa, Julian Göltz, Wolfgang Maass, Timothée Masquelier, Richard Naud, Emre O Neftci, Mihai A Petrovici, et al. Visualizing a joint future of neuroscience and neuromorphic engineering. *Neuron*, 109(4):571–575, 2021.

Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11062–11070, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] A shared random seed is used for the reproducible purpose.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes]

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] Public datasets.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] Public datasets.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]