IMPROVED CERTIFIED ROBUSTNESS AND ADVERSARIAL ATTACKS FOR RATE-ENCODED SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking Neural Networks (SNNs) are well known for their energy-efficient neuromorphic implementations and robustness to adversarial examples, especially with rate encoding, which encodes inputs as the rate of binary spikes from a Bernoulli distribution. Recently, rate encoded SNNs under Bernoulli smoothing were shown to be provably robust against any adversarial examples that lie within l_1 radius, $\|\delta\|_1 \leq \frac{p_a - p_b}{2T3^{T-1}}$, from the test input having top two class probabilities p_a and p_b, respectively. However, this radius decreases exponentially with the SNN latency T, notwithstanding experimental evidence. We provide an improved adversarial robustness guarantee for the rate-encoded SNNs under the same setting, showing that the certified radius of the smooth classifier decays only linearly with latency, i.e., $\|\delta\|_1 \leq \frac{p_a - p_b}{2(T+1)}$, offering a larger certified radius of robustness that reduces the gap between empirical and certified robustness. We extend the proof to the recently proposed signed rate encoding, which offers higher standard accuracy than rate encoding. Experimentally, we improve the existing l_0-norm-bounded sparse encoding attack (SEA) and propose SEA-PGD, which consistently outperforms SEA across three standard datasets and encoding techniques.

1 INTRODUCTION

Spiking neural networks (SNNs) offer energy-efficient neural computation by harnessing the spiking properties of biological neurons (Lemaire et al., 2023). Unlike conventional artificial neurons, which use continuous activation functions, spiking neurons produce binary spike outputs whenever their membrane potential exceeds a predetermined threshold. The computation occurs over a discretized temporal dimension, and at each step, the neuron responds with a binary spike. A spiking pattern that is sparse spatially (i.e., very few neurons emit spikes) and temporally (i.e., at a few time steps) greatly reduces the burden of synaptic operations on the network (Rueckauer et al., 2017). That is, at any particular time step, the computation of input to a spiking neuron amounts to accumulating the input edge weights associated with a pre-synaptic spike, bypassing the floating-point inner product between pre-synaptic neuronal outputs and the edge weights customary in a conventional artificial neural network (ANN). Such computational advantage of SNNs is harnessed by specialized neuromorphic hardware, which offers significant energy savings compared with conventional ANNs for the same task (Na et al., 2022).

As SNNs operate over a temporal domain, static inputs, such as images, must be encoded temporally. A popular encoding technique called direct/constant encoding supplies each input x_i to a spiking neuron for T steps, and the neuronal output over the same period is treated as the encoded input. Another bio-plausible technique is *rate encoding*, where an input pixel $x_i \in [0, 1]$ is encoded as $[z_{i,1}, \cdots, z_{i,T}]$, with each $z_{i,t} \sim \mathrm{Ber}(x_i)$ sampled independently from a Bernoulli distribution, so that the average number of spikes (or *rate*) $\frac{1}{T}\sum_{t=1}^{T} z_{i,t}$ approximates x_i. A recent extension *signed rate encoding* (Mukhoty et al., 2025), allows pixel $x_i \in [-1, 1]$ by generating signed Bernoulli samples, $\mathrm{sBer}(x_i) = \mathrm{sign}(x_i)\,\mathrm{Ber}(|x_i|)$. Signed rate encoding allows mean centering of the training data and, consequently, improves standard accuracy compared to rate encoding by reducing the randomness introduced by the input encoder.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Adversarial examples (Goodfellow et al., 2014) pose a security risk to neural networks. It introduces a small perturbation to the test input (e.g., changing a few pixels in the test image without altering its semantics), causing the model to misclassify the perturbed input. Adversarial training attempts to mitigate this risk by training models on adversarial examples with correct labels. Still, it significantly increases training costs because adversarial examples need to be found for each input after each training epoch with respect to the latest model parameters (Goodfellow et al., 2015). Rate encoded SNNs, in parallel to their energy benefits, exhibit natural robustness to adversarial perturbations (Sharmin et al., 2020), even without adversarial training, and more so with it. In contrast, direct-encoded SNNs achieve higher standard accuracy than rate or signed rate encoding but lack adversarial robustness (Kundu et al., 2021).

Inspired by randomized smoothing (Cohen et al., 2019), the empirical robustness offered by rate-encoded SNNs is complemented by a theoretical guarantee of certified robustness, which offers a probabilistic guarantee that no adversarial examples exist within a radius from the original test input under the l_1 norm (Mukhoty et al., 2024). The importance of such a result lies in the fact that, with high probability, the model is robust to all adversarial perturbations within the certified radius, regardless of the attack strategy employed. However, it can be the case that, given a perturbation radius, there is a significant gap between empirical and certified robustness, as the provable radius of robustness for a particular input can be much smaller than the radius at which a nearest adversarial example is found empirically. This is due to the incomplete nature of certified robustness proofs involving randomized smoothing, which show that there exists no adversarial example within the certified radius but are silent about the existence of adversarial examples outside the radius, allowing the possibility that the certified radius could have been bigger.

One of the main contributions of the present work is to show that the certified l_1 radius for smoothed rate-encoded SNNs is exponentially larger in latency T than it was previously proved, thereby effectively reducing the gap between certified and empirically observed robustness. Experimental evidence shows that increasing latency in rate encoding improves standard accuracy (Anumasa et al., 2024), but deteriorates adversarial robustness only linearly (Mukhoty et al., 2024). The present work proves that the certified radius deteriorates linearly (see Theorem 3). Further, we can show that a similar guarantee holds for signed rate encoding (see Lemma 4), which, to our knowledge, is the first proof of certified robustness for signed rate encoded SNNs.

A crucial assumption for the present proof of robustness is that the adversarial attack is performed in the input space $\mathbf{x} \in [0, 1]^d$, upon which randomness is introduced via the Bernoulli distribution in the encoding space. Thus, it is interesting to see whether, in the absence of adversarial training, the adversarial robustness of rate-encoded SNNs breaks down under an attack on the encoding space, where the randomized smoothing result does not apply. Towards this end, we improve existing Sparse Encoding Attack (SEA) (Mukhoty et al., 2025) in the encoding space and provide experimental evidence that the natural robustness of rate-encoded classifiers (without adversarial training) breaks down completely when only 20 pixels (out of 3072) of a CIFAR-10 frame are adversarially altered.

Contributions:

- We offer exponential improvement over certified radius for rate encoded SNN under Bernoulli smoothing by improved analysis of the Lipschitz continuity of the smooth classifier and extend the proof to signed rate encoded classifiers and compare certified and empirical robustness on standard datasets, such as, CIFAR-10, CIFAR-100, SVHN.

- We propose white-box adversarial attack SEA-PGD, an improved version of the existing Sparse Encoding Attack (SEA), that alters the encoding space under l_0-norm restriction, demonstrating stronger attack success rate across standard datasets compared to SEA.

- We compare the performance of rate and signed rate encoded SNNs under standard vs. adversarial training with SEA/ SEA-PGD, and establish that randomness introduced by the encoding layer is essential for robustness in absence of adversarial training.

2 Background and Related Work

The neuronal dynamic of a temporally discretized spiking neuron is governed by the following equations, where $u_i^l[t]$ denotes the membrane potential of the i-th neuron at the l-th layer, and at the time step $t \in [T]$ (Eshraghian et al., 2021).

$$u_i^{(l)}[t] = \beta(u_i^{(l)}[t-1] - s_i^{(l)}[t-1]u_{th}) + \sum_j w_{ij}^{(l)} s_j^{(l-1)}[t], \tag{1}$$

$$s_i^{(l)}[t] = H(u_i^{(l)}[t] - u_{th}) = \begin{cases} 1 & \text{if } u_i^{(l)}[t] > u_{th} \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

The neuron accumulates input spikes from pre-synaptic neurons weighted by $w_{ij}^{(l)}$ and retains the membrane potential from the previous step through a leaky factor $\beta \in (0, 1]$. It generates a binary output spike $s_i^l[t]$ whenever the membrane potential exceeds the fixed threshold u_{th}, as described by the Heaviside function H, which works as the activation function for the neuron. An output spike is followed by a soft reset in the membrane potential, which decreases the potential by u_{th}.

To supply a static image $\mathbf{x} = [x_1, x_2, \ldots, x_d] \in [0, 1]^d$ to the SNN as spikes, it requires to be encoded temporally for t steps. Let $Z_t \in \{0, 1\}^{d \times t}$ denote the encoded input, and let $\mathrm{Ber}(\mathbf{x})$ denote Bernoulli sampling applied pixel wise to $\mathbf{x}$, and $\mathrm{Ber}^{(t)}(\mathbf{x}) = \prod_{i=1}^{t} \mathrm{Ber}(\mathbf{x})$ denote the joint Bernoulli distribution involving t rounds of independent Bernoulli sampling using $\mathbf{x}$, so that, $Z_t \sim \mathrm{Ber}^{(t)}(\mathbf{x})$ gives rate encoding of $\mathbf{x}$ for t steps. Let the SNN classifier at t-th step is denoted by $f_t : \{0, 1\}^{d \times t} \to \mathbb{P}(\mathcal{Y})$, which takes Z_t as input and outputs $\mathbb{P}(\mathcal{Y})$, a probability simplex defined over the finite output label set $\mathcal{Y}$, i.e., $\sum_{y \in \mathcal{Y}} f_t(Z_t)_y = 1$, where $f_t(Z_t)_y$ denotes the probability of class y. The final soft SNN classifier $F_T : \{0, 1\}^{d \times T} \to \mathbb{P}(\mathcal{Y})$ takes the full encoding Z_T as input and averages the output of the intermediate, temporally dependent classifiers, $\{f_1, \cdots, f_T\}$, thus,

$$F_T(Z_T) = \frac{1}{T} \sum_{t=1}^{T} f_t(Z_t) \tag{3}$$

Since, F_T is a soft-classifier, the predicted label of the corresponding hard classifier can be obtained by $\arg\max_{y \in \mathcal{Y}} F_T(\cdot)_y$, i.e., the class label having the highest class probability in $F_T(\cdot)$.

We define the smooth classifier $g_t : [0, 1]^d \to \mathbb{P}(\mathcal{Y})$ corresponding to f_t, by taking expectation over randomness introduced by rate encoding, i.e.,

$$g_t(\mathbf{x}) = \mathbb{E}_{Z_t \sim \mathrm{Ber}^{(t)}(\mathbf{x})}[f_t(Z_t)] \tag{4}$$

And consequently the smooth classifier $G_T : [0, 1]^d \to \mathbb{P}(\mathcal{Y})$ corresponding to F_T can be given as:

$$G_T(\mathbf{x}) = \mathbb{E}_{Z_T \sim \mathrm{Ber}^{(T)}(\mathbf{x})}[F_T(Z_T)] = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{Z_t \sim \mathrm{Ber}^{(t)}(\mathbf{x})}[f_t(Z_t)] = \frac{1}{T} \sum_{t=1}^{T} g_t(\mathbf{x}) \tag{5}$$

Since it is difficult to evaluate $G_T(\mathbf{x})$ analytically due to the expectation involved, an empirical estimation of $G_T(\mathbf{x})$, using independent samples $Z_T^{(j)} \sim \mathrm{Ber}^T(\mathbf{x})$ given by:

$$\hat{G}_T(\mathbf{x}) = \frac{1}{m} \sum_{j=1}^{m} F_T(Z_T^{(j)}), \quad \text{with } Z_T^{(j)} \sim \mathrm{Ber}^T(\mathbf{x}) \tag{6}$$

perform reasonably in experiments to approximate G_T, and offer statistical guarantees, where, the failure probability $\mathbb{P}(\arg\max_y \hat{G}(\mathbf{x})_y \neq \arg\max_y G(\mathbf{x})_y)$ can be upper bounded by error rate α, which depends upon the number of samples m. See Algorithm 1 in (Mukhoty et al., 2024), and (Cohen et al., 2019) for details.

In contrast to rate encoding which allows input $\mathbf{x} \in [0, 1]^d$, signed rate encoding allows input $\mathbf{x} = [x_1, x_2, \cdots, x_d] \in [-1, 1]^d$, by essentially uses rate encoding, but retains the sign of the input

pixels, producing temporally encoded input $Z_t \in \{-1, 0, 1\}^{d\times t}$, via Signed Bernoulli Encoding (sBer). At a pixel level, sBer is defined as:

$$\mathrm{sBer}(x_i) = \begin{cases} \mathrm{Ber}(x_i) & \text{if, } x_i \geq 0 \\ -\mathrm{Ber}(x_i) & \text{if, } x_i < 0 \end{cases} = \begin{cases} 1 & \text{w.p, } x_i \mathbb{I}[x_i \geq 0] \\ -1 & \text{w.p., } -x_i \mathbb{I}[x_i < 0] \\ 0 & \text{w.p., } 1 - |x_i| \end{cases}$$

The advantage of signed rate encoding is that, due to centering of the input data, it delivers higher standard accuracy compared to rate encoding, while retaining the adversarial robustness properties, as evident from experimental results (see Table 1). Given that we redefine the support of the functions accordingly, i.e., $\tilde{f}_t : \{-1, 0, 1\}^{d\times t}$, $\tilde{F}_T : \{-1, 0, 1\}^{d\times T}$, and redefine the smoothed function w.r.t. sBer: $\tilde{g}_t(\mathbf{x}) = \mathbb{E}_{Z_t\sim \mathrm{sBer}^{(t)}(\mathbf{x})}[\tilde{f}_t(Z_t)]$, so that,

$$\tilde{G}_T(\mathbf{x}) = \mathbb{E}_{Z_T\sim \mathrm{sBer}^{(T)}(\mathbf{x})}[\tilde{F}_T(Z_T)] = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{Z_t\sim \mathrm{sBer}^{(t)}(\mathbf{x})}[\tilde{f}_t(Z_t)] = \frac{1}{T}\sum_{t=1}^{T} \tilde{g}_t(\mathbf{x}) \tag{7}$$

The *provable adversarial robustness* in this work is a consequence of Lipschitz continuity of the smooth classifier $G_T(\cdot)_y$ (also $\tilde{G}_T(\cdot)_y$), that gives probability of the class label $y \in \mathcal{Y}$.

Definition 1. *A function $h : \mathbb{R}^d \to \mathbb{R}$ is said to be L-Lipschitz continuous w.r.t. to the norm $\|\cdot\|$, if ,*

$$\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d : |h(\mathbf{x}_1) - h(\mathbf{x}_2)| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\| \tag{8}$$

Further, if h is differentiable, it is L-Lipschitz cont. iff $\forall \mathbf{x}, \|\nabla h(\mathbf{x})\|_* \leq L$ where $\|\cdot\|_* : \mathbb{R}^d \to \mathbb{R}$ is the dual norm of $\|\cdot\|$, defined as, $\|\mathbf{y}\|_* = \sup_{\|\mathbf{x}\|=1} \mathbf{x}^T\mathbf{y}$, e.g., $\|\cdot\|_\infty$ is the dual norm of $\|\cdot\|_1$.

Lemma 1. *Let $g : \mathbb{R}^d \to \mathbb{P}(\mathcal{Y})$ be a soft classifier giving probabilities of the class labels and suppose $g(\mathbf{x})_y$, for all the classes $y \in \mathcal{Y}$, are L-Lipschitz cont. w.r.t the norm $\|\cdot\|$. For a given input $\mathbf{x}$, if we have top two probabilities $p_a = \max_{y\in\mathcal{Y}} g(\mathbf{x})_y$, $p_b = \max_{y\neq a} g(\mathbf{x})_y$, then,*

$$\forall \boldsymbol{\delta} : \|\boldsymbol{\delta}\| < \frac{p_a - p_b}{2L} \quad \arg\max_{y\in\mathcal{Y}} g(\mathbf{x} + \boldsymbol{\delta}) = a \tag{9}$$

It is a known result, a proof can be found in Lemma 2.1 (Li, 2019).

Intuitively, to alter the predicted label $a = \arg\max_{y\in\mathcal{Y}} g(\mathbf{x})_y$, the probability of class a, i.e., p_a, should not remain the highest following the input perturbation $\mathbf{x} + \boldsymbol{\delta}$. If the next highest probability p_b, there is only a margin of $\frac{p_a - p_b}{2}$, less than which a class probability can change without altering the predicted class label. Lipschitzness of $g(\cdot)_y$ translates this to a allowable change in the input $\mathbf{x}$, and gives a radius of perturbation within which changes to the input $\mathbf{x}$ will not translate into a change in the predicted class label.

Computing certified radius: Since $G_T(\mathbf{x})$ is not computed analytically, and only approximated by $\hat{G}_T(\mathbf{x})$, we will not know the quantities p_a and p_b exactly. Using $\hat{G}_T$ (see Algorithm 2 (Mukhoty et al., 2024)), one can find statistical estimates, $\underline{p_a}, \overline{p_b}$ such that $p_a \geq \underline{p_a}$, and $p_b \leq \overline{p_b}$ which holds with probability $1 - \alpha$, where α is the failure probability that can be small with m. This ensures, $\underline{p_a} - \overline{p_b} \leq p_a - p_b$ with probability $1 - \alpha$. One may further use fact that, $p_b \leq 1 - p_a \leq 1 - \underline{p_a}$, and use, $\overline{p_b} := 1 - \underline{p_a}$. To obtain a probabilistic guarantee,

$$\forall \boldsymbol{\delta} : \|\boldsymbol{\delta}\| < \frac{\underline{p_a} - \overline{p_b}}{2L} = \frac{2\underline{p_a} - 1}{2L} \quad \arg\max_{y\in\mathcal{Y}} g(\mathbf{x} + \boldsymbol{\delta}) = a \tag{10}$$

The quantity $r = \frac{\underline{p_a} - \overline{p_b}}{2L}$ is known as certified radius for an input $\mathbf{x}$. Fixing a radius ϵ and error rate α, **certified robust accuracy** computes the fraction of test points that have certified radius $r \geq \epsilon$, so that they are probabilistically guaranteed to be robust to perturbation of radius ϵ.

3 Improved Certified Robustness for Rate Encoding

Below we state the *key lemma* the paper, establishing Lipschitz continuity of the smooth classifier.

Lemma 2. *For all class label y, $g_t(\mathbf{x})_y$ is $2t$-Lipschitz under the $\|\cdot\|_1$-norm, and consequently, for all y, $G_T(\mathbf{x})_y$ is $T+1$-Lipschitz under the $\|\cdot\|_1$-norm.*

The proof of this lemma is differed to the appendix A.1. Compared to the Lipschitzness $T3^{T-1}$ of the smooth classifier proved in the existing literature (Lemma 4 in Mukhoty et al. (2024)), we prove an exponentially tighter Lipschitzness bound of $T+1$. Combining Lemma 1 and 2, in Theorem3 it is immediate that under perturbation radius $\|\boldsymbol{\delta}\|_1 \leq \frac{p_a - p_b}{2(T+1)}$ no adversarial examples can exists and that the radius of robustness decays only linearly with T.

Theorem 3. *For the smooth classifier G_T defined as above, and $p_a = \max_{y \in \mathcal{Y}} G_T(\mathbf{x})_y$, $p_b = \max_{y \neq a} G_T(\mathbf{x})_y$, we have:*

$$\forall \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_1 < \frac{p_a - p_b}{2(T+1)} \quad \arg\max_{y \in \mathcal{Y}} G_T(\mathbf{x} + \boldsymbol{\delta}) = a \tag{11}$$

3.1 Extension of proof to Signed Rate Encoding

Then we manage to extend Lemma 2 for signed rate encoding as:

Lemma 4. *For all y, $\tilde{g}_t(\mathbf{x})_y$ is $2t$-Lipschitz under the $\|\cdot\|_1$-norm, and consequently, for all y, $\tilde{G}_T(\mathbf{x})_y$ is $T+1$-Lipschitz under the $\|\cdot\|_1$-norm.*

The proof of this lemma is differed to the appendix A.2. As a consequence, Theorem 3 is proved for $\tilde{G}_T$. The consequence of the lemma on certified robustness can be observed in Figure 1 where we compare certified robustness accuracy due to Theorem 3 for both RE and SRE, and compare it with empirical robust accuracy.

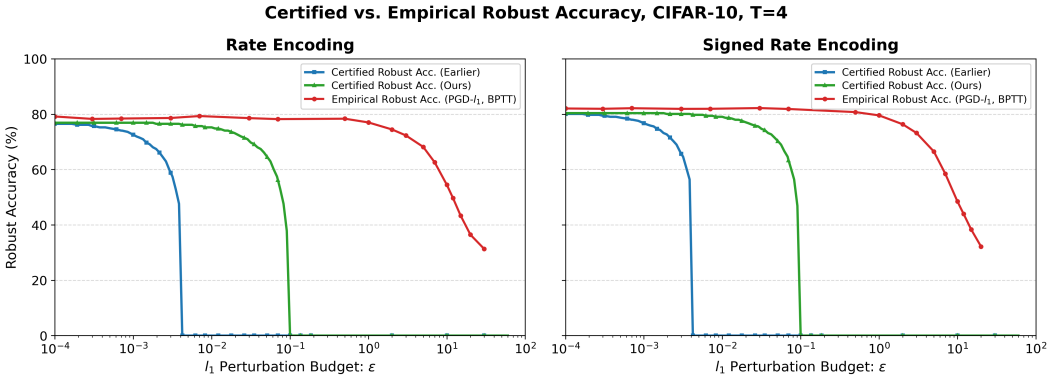

Figure 1: We compare the certified robust accuracy obtained from the present work with that established earlier (Mukhoty et al., 2024), under various l_1-norm-bounded perturbation budgets, along with an empirical robust accuracy against the PGD attack, on the CIFAR-10 dataset. The plot shows a significant improvement in certified robust accuracy due to the present work and highlights the remaining gap between certified and empirical robust accuracy.

3.2 Certified vs. Empirical Robust Accuracy:

We compare the certified robust accuracy computed using Theorem 3 with that of Theorem 5, established in (Mukhoty et al., 2024), along with an empirical robust accuracy obtained against the PGD attack under various l_1-norm-bounded perturbation budgets. Fig. 1 and 3 in Appendix B show certified accuracies computed on the CIFAR-10, CIFAR-100, and SVHN test datasets, using models obtained via standard training and PGD adversarial training with rate encoding and signed rate encoding. We use $m = 1$ to obtain the classifier prediction (corresponding to Table 1), and $m = 100$ for the class probability estimator $\underline{p_a}$ in Eq. 10. We observe a significant improvement in the certified accuracy due to the present result, e.g., in Fig.1 for CIFAR-10, rate encoding, at $\epsilon = 0.03$, we obtain a certified robust accuracy 70.35%, whereas the earlier result had offered 0%, in reference to the empirical robust accuracy of 78.56%. Table 3, 4 supplies the actual numbers corresponding to Fig.1.

4 ADVERSARIAL ATTACKS IN ENCODING SPACE

The certified robustness guarantee holds for the l_1-norm bounded perturbations in the input space $\mathbf{x} \in \mathbb{R}^d$. However, such a guarantee does not hold if the model is receiving an encoded input. For example, a rate-encoded SNN can receive an already encoded input $Z_T = [\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \cdots, \mathbf{z}^{(T)}] \in \{0,1\}^{d\times T}$, with $\mathbf{z}^{(t)} = [z_{1,t}, z_{2,t}, \cdots, z_{d,t}] \in \{0,1\}^d$ and the adversarial corruption can perturb in the encoded input Z_T instead of $\mathbf{x}$. To model such a perturbation Δ in the encoding space, we need to ensure that corrupted encoding $Z_T + \Delta \in \{0,1\}^{d\times T}$, where, $\Delta = [\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(T)}]$. To introduce sparsity of corruption at each time frame $\mathbf{z}^t$, we restrict $\|\boldsymbol{\delta}^{(t)}\|_0 \leq k$, making $\boldsymbol{\delta}^{(t)} \in \{-1,0,1\}^d$. The adversarial objective to incorrectly classify a input Z_T with the correct label y can be given as:

$$\Delta^\star = \arg \max_{\substack{\Delta:\, \forall t\in[T], \|\boldsymbol{\delta}^{(t)}\|_0 \leq k \\ \mathbf{z}^{(t)}+\boldsymbol{\delta}^{(t)} \in \{0,1\}^d}} l(F_T(Z_T + \Delta), y) \tag{12}$$

It is worth mentioning that Sparse Encoding Attack (SEA) (Mukhoty et al., 2025) simplifies this problem by assuming the first-order approximation of the loss, $l(Z_T + \Delta) \approx l(Z_T) + \langle \nabla l(Z_T), \Delta \rangle$. So instead, they optimize,

$$\Delta^\star = \arg \max_{\substack{\Delta:\, \forall t\in[T], \|\boldsymbol{\delta}^{(t)}\|_0 \leq k \\ \mathbf{z}^{(t)}+\boldsymbol{\delta}^{(t)} \in \{0,1\}^d}} \langle \nabla_{Z_T} l(F_T(Z_T), y), \Delta \rangle \tag{13}$$

which has a closed-form solution. To see this, let $\hat{\Delta} \in \{-1,0,1\}^{d\times T}$ be the solution of Eq.13 without the sparsity restriction $\|\boldsymbol{\delta}^{(t)}\|_0 \leq k$, then $\hat{\Delta}$ can be given co-ordinatewise:

$$[\hat{\Delta}]_{i,t} = \begin{cases} 1 & \text{if,} \quad [\nabla l]_{i,t} > 0, z_{i,t} = 0 \\ -1 & \text{if,} \quad [\nabla l]_{i,t} < 0, z_{i,t} = 1 \\ 0 & \text{otherwise} \end{cases}$$

One can then obtain $\Delta^\star$ from $\hat{\Delta}$ by introducing sparsity according to the top contributions to the inner product. Let $S \subseteq [d], |S| = k$, denote the set of indices top k elements of the set $\{[\nabla l]_{i,t}[\hat{\Delta}]_{i,t}\}_{i=1}^d$ and setting $[\Delta^\star]_{i,t} = [\hat{\Delta}]_{i,t}$ if $i \in S$ or $[\Delta^\star]_{i,t} = 0$ if $i \notin S$ ensures that $\Delta^\star$ adheres to the sparsity constraints while retaining only the top k contributions to the inner-product.

For signed rate encoding, the constraints in Eq. 13 changes to $Z_T, Z_T + \Delta \in \{-1,0,-1\}^{d\times T}$, and consequently, the closed form solution for $\hat{\Delta} \in \{-2,-1,0,1,2\}^{d\times T}$ becomes:

$$[\hat{\Delta}]_{i,t} = \operatorname{sign}(\nabla l)_{i,t} - z_{i,t} \operatorname{sign}(\nabla l)_{i,t} \tag{14}$$

$\Delta^\star$ can be obtained from $\hat{\Delta}$ in a similar fashion, by finding top k contributions to the inner product.

For additional clarity, it is helpful to highlight how the attack extends to $\hat{G}_T$, approximation of the smooth classifier G_T, given in Eq. 6. There we have, $Z_T^{(j)} \sim \text{Ber}^T(\mathbf{x})$, $j \in [m]$, such that, $Z_T^{(j)} = [\mathbf{z}^{(1,j)}, \mathbf{z}^{(2,j)}, \cdots, \mathbf{z}^{(T,j)}] \in \{0,1\}^{d\times T}$, and consequently, we find $\Delta^{(j)} = [\boldsymbol{\delta}^{(1,j)}, \boldsymbol{\delta}^{(2,j)}, \cdots, \boldsymbol{\delta}^{(T,j)}] \in \{-1,0,1\}^{d\times T}$ corresponding to $Z_T^{(j)}$ with the restriction $\|\boldsymbol{\delta}^{(t,j)}\|_0 \leq k$. And finally, the objective is to optimize, $D = [\Delta^{(1)}, \Delta^{(2)}, \cdots, \Delta^{(m)}]$, such that,

$$D^\star = \arg \max_{\substack{D:\, \forall t\in[T], \|\boldsymbol{\delta}^{(t,j)}\|_0 \leq k \\ \mathbf{z}^{(t,j)}+\boldsymbol{\delta}^{(t,j)} \in \{0,1\}^d}} l(\frac{1}{m}\sum_{i=1}^m F_T(Z_T^{(j)} + \Delta^{(j)}), y) \tag{15}$$

4.1 IMPROVED SPARSE ENCODING ATTACK VIA PGD:

We extend SEA using projected gradient descent to SEA-PGD. We observe that the objective of Eq. 12 can be solved over multiple gradient steps, where each update step consists of a projection step w.r.t. the latest gradient direction, ensuring at most k locations in each frame are altered w.r.t. the original input Z_T.

$$\Delta_{s+1} = \arg \max_{\substack{\Delta:\, \forall t\in[T], \|\boldsymbol{\delta}^{(t)}\|_0 \leq k \\ \mathbf{z}^{(t)}+\boldsymbol{\delta}^{(t)} \in \{0,1\}^d}} \langle \nabla_{Z_T+\Delta_s} l(F_T(Z_T + \Delta_s), y), \Delta \rangle \tag{16}$$

Note that each update step jointly optimizes the input space and sparsity restriction, similar to SEA, but uses gradient information computed at $Z_T + \Delta_s$, instead of Z_T. The update is performed for N iterations, as long as $\Delta_{s+1} \neq \Delta_s$, and $l(F_T(Z_T + \Delta_{s+1}), y) \geq l(F_T(Z_T + \Delta_s), y)$. We also implement SEA-PGD under signed rate encoding, using the closed-form solution in Eq. 14, followed by selection of top k co-ordinates. We use SEA-PGD for adversarial training, under both encoding techniques.

Table 1 shows that for CIFAR-10 under standard training (column 1, rate encoding) with sparsity budget $k = 10$, SEA gives robust accuracy 44.54%, while SEA-PGD offers robust accuracy 30.81%, demonstrating a significant improvement in attack success rate. Under standard training (column 1, signed rate encoding), SEA offers robust accuracy 39.43% while SEA-PGD offers accuracy 22.51%.

5 EXPERIMENTS:

Datasets and Model Architechture: We compare adversarial attacks and robustness on the image classification task within the supervised learning setting, using spiking neural networks with rate encoding and signed rate encoding. We use standard datasets CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009) and SVHN (Netzer et al., 2011). For comparability and reproducibility, we use temporal latency $T = 4$ and the VGGSNN architecture (Deng et al., 2022), originally based on VGG-11 and used in earlier works (Mukhoty et al., 2024; 2025). The classification accuracy generally improves with larger T, but needs higher computational bandwidth for training and inference (Anumasa et al., 2024).

(Adversarial) Training : We use standard training (STD), training with additive input Gaussian Noise (GN), adversarial training with l_∞-norm bounded Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015) and Projected Gradient Descent (PGD) (Madry et al., 2018) that attacks the input space $\mathbf{x} \in [0, 1]^d$, and adversarial training with l_0-norm bounded SEA (Mukhoty et al., 2025) and the proposed SEA-PGD which attack the encoding space for rate encoding and signed rate encoding.

Table 1 represent the models columns-wise, and, row-wise, it gives the empirical robust accuracy for clean data (CLEAN), input perturbed with Gaussian Noise (GN), and the mentioned adversarial attacks. At inference time, the models can use MC simulations to approximate the smooth classifier. Table 1 reports the results with MC simulations $m = 1$, while Table 2 reports them for $m = 10$. The hyperparameters for model training are reported in Appendix B, Table B.

Comparisons of Attacks and Robustnesses: Comparing sparse encoding space attacks from Table 1, 2, we observe that for both $k \in \{10, 20\}$, SEA-PGD outperforms SEA by yielding lower robust accuracy across encodings. This result is consistent across all the models for the three datasets and under both rate and signed-rate encoding techniques.

More interestingly, when compared with l_∞-norm bounded input space attacks, such as FGSM and PGD, which are allowed to alter all input pixel x_i by $\epsilon = \frac{8}{255}$, l_0- norm bounded encoding attacks appear significantly stronger under standard training (column 1, under both rate and signed rate encoding). This is expected, possibly because the layer of defense introduced by $\mathrm{Ber}\,/\,\mathrm{sBer}$ noise is taken away by directly attacking the encoding space. To handle encoding space attacks, one must employ adversarial training, in which SEA-PGD under signed rate encoding outperforms others, as evidenced by the column-wise average performance.

Fig. 2 (top) compares the robust accuracy on varying k for models obtained via standard training and adversarial training by SEA-PGD, which reveals (i) SEA-PGD to be stronger than SEA as an attack, (ii) under standard training, signed rate encoding to be more vulnerable than rate encoding, possibly because signed rate encoding by design reduces randomness by centering the inputs and (iii) under adversarial training with SEA-PGD, signed rate encoding outperforms rate encoded models. A similar comparison of STD vs. SEA vs. SEA-PGD for adversarial training on all three datasets is given in Fig. 4.

Fig. 2 (bottom) makes a similar comparison while varying m and fixing $k = 20$, where we can find noticeable improvement in robustness up to $m = 20$. The full experiments across three datasets are reported in Fig. 5.

$T=4, m=1$	CIFAR-10, Rate Encoding						CIFAR-10, Signed Rate Encoding					
MODELS / ATTACK	STD	GN	FGSM	PGD	SEA	SEA-PGD	STD	GN	FGSM	PGD	SEA	SEA-PGD
CLEAN	**79.38**	78.92	75.83	77.55	75.72	76.45	**82.37**	81.64	78.74	79.96	78.18	78.74
GN, $\epsilon = 8/255$	78.93	**79.53**	76.39	77.36	75.75	76.04	81.87	**82.16**	78.82	80.21	77.97	77.93
FGSM, $\epsilon = 8/255$	43.85	43.81	**53.94**	53.36	42.04	42.45	44.15	43.75	55.13	54.05	55.89	**58.04**
PGD, $\epsilon = 8/255$	38.63	38.07	**50.29**	49.24	37.08	37.66	36.50	36.72	50.34	48.41	51.09	**53.52**
SEA, $k = 10$	44.54	44.92	48.05	47.78	55.01	**55.93**	39.43	40.49	49.22	48.62	**82.42**	79.15
SEA-PGD, $k = 10$	30.81	30.88	32.68	32.31	43.64	**44.94**	22.51	23.05	32.40	31.27	63.29	**65.16**
SEA, $k = 20$	29.29	29.21	34.90	34.15	42.46	**43.26**	27.88	29.34	40.68	39.76	**79.50**	74.30
SEA-PGD, $k = 20$	14.86	14.69	19.05	17.58	29.30	**30.36**	10.46	10.25	19.53	18.63	53.04	**54.93**
Avg	45.04	45.00	48.89	48.67	50.13	**50.89**	43.15	43.43	50.61	50.11	67.67	**67.72**
$T=4, m=1$	SVHN, Rate Encoding						SVHN, Signed Rate Encoding					
CLEAN	85.71	**85.91**	85.90	85.45	84.91	84.83	**89.74**	89.56	89.55	89.59	84.35	86.25
GN, $\epsilon = 8/255$	**85.96**	85.74	85.89	85.46	84.90	84.86	89.04	**89.28**	89.02	89.18	81.92	84.40
FGSM, $\epsilon = 8/255$	44.37	44.13	**49.33**	48.70	42.48	43.10	42.65	42.60	48.76	48.69	**65.80**	64.84
PGD, $\epsilon = 8/255$	38.35	38.66	**44.29**	43.69	36.44	36.92	34.93	34.93	41.69	41.39	**63.82**	60.78
SEA, $k = 10$	37.82	37.96	37.70	37.85	51.99	**54.61**	34.60	33.24	34.66	36.52	**93.05**	89.35
SEA-PGD, $k = 10$	25.11	24.70	23.79	24.21	41.55	**43.67**	15.15	15.47	16.55	16.89	54.09	**76.26**
SEA, $k = 20$	21.92	21.94	22.18	22.42	30.66	**33.46**	23.05	22.03	24.69	25.65	**88.14**	84.40
SEA-PGD, $k = 20$	9.30	9.07	9.48	9.68	18.14	**19.59**	5.76	5.52	6.60	7.19	35.05	**62.91**
Avg	43.57	43.51	44.82	44.68	48.88	**50.13**	41.87	41.58	43.94	44.39	70.78	**76.15**
$T=4, m=1$	CIFAR-100, Rate Encoding						CIFAR-100, Signed Rate Encoding					
CLEAN	**50.24**	49.66	47.03	48.14	46.23	46.15	**54.54**	53.97	49.45	51.24	49.92	50.85
GN, $\epsilon = 8/255$	**50.66**	49.75	46.81	48.10	45.99	46.55	53.96	**54.28**	49.05	51.59	49.67	50.59
FGSM, $\epsilon = 8/255$	22.30	21.74	**28.15**	27.40	20.81	20.88	22.41	22.01	30.00	30.09	31.70	**33.64**
PGD, $\epsilon = 8/255$	18.74	18.50	**25.90**	24.79	18.52	18.17	18.68	18.44	27.91	27.18	28.61	**30.84**
SEA, $k = 10$	26.44	25.86	26.74	26.83	**30.65**	30.38	21.49	21.32	27.31	27.78	**57.76**	53.27
SEA-PGD, $k = 10$	17.31	17.13	18.00	17.99	**23.40**	23.30	11.56	11.43	17.42	17.80	**36.62**	36.49
SEA, $k = 20$	16.53	16.53	18.30	18.41	**22.23**	22.08	13.04	13.51	20.19	20.79	**53.75**	49.56
SEA-PGD, $k = 20$	7.86	8.29	9.90	9.70	14.91	**15.25**	4.73	4.65	9.40	9.63	27.30	**28.45**
Avg	26.26	25.93	27.60	27.67	27.84	**27.85**	25.05	24.95	28.84	29.51	**41.92**	41.71

Table 1: We compare standard training (STD), training with Gaussian Noise (GN), adversarially trained models with l_∞-norm bounded attack (FGSM, PGD), and adversarially trained models l_0-norm bounded attack (SEA, SEA-PGD) represented respectively under columns for rate encoding and Sign rate encoding, with temporal latency $T = 4$, and Monte Carlo inference $m = 1$ for approximation of smooth classifier. The bold elements highlight the highest accuracy obtained by a model under a specific attack, separately for each encoding. Comparing the column averages, it becomes evident that adv. training with SEA-PGD under signed rate encoding offers superior robustness.

$T=4, m=10$	CIFAR-10, Rate Encoding						CIFAR-10, Signed Rate Encoding					
MODELS / ATTACK	STD	GN	FGSM	PGD	SEA	SEA-PGD	STD	GN	FGSM	PGD	SEA	SEA-PGD
CLEAN	**83.92**	83.64	80.63	82.25	79.53	80.24	85.60	**85.74**	81.78	83.39	82.39	82.28
GN, $\epsilon = 8/255$	**83.58**	83.40	80.45	81.97	79.41	80.00	**85.68**	85.01	81.94	83.23	82.06	82.24
FGSM, $\epsilon = 8/255$	40.16	39.72	**50.32**	50.23	40.22	40.64	42.26	42.35	52.27	49.96	51.13	**54.66**
PGD, $\epsilon = 8/255$	33.31	32.83	**44.10**	43.38	33.13	34.65	34.30	34.73	45.38	43.65	45.01	**48.77**
SEA, $k = 10$	47.28	47.14	49.33	49.77	57.41	**58.01**	41.50	41.91	50.42	50.37	**81.33**	79.88
SEA-PGD, $k = 10$	38.32	38.17	40.62	40.99	52.41	**53.46**	31.00	31.20	41.82	41.09	77.55	**78.25**
SEA, $k = 20$	30.91	30.48	35.47	35.61	44.02	**45.00**	29.50	30.18	41.46	40.63	**79.37**	76.68
SEA-PGD, $k = 20$	20.56	20.83	25.24	24.43	36.48	**38.27**	16.12	16.52	28.47	27.64	70.54	**72.51**
Avg	47.26	47.03	50.77	51.08	52.83	**53.78**	45.75	45.96	52.94	52.50	71.17	**71.91**
$T=4, m=10$	SVHN, Rate Encoding						SVHN, Signed Rate Encoding					
CLEAN	91.98	92.11	**92.14**	92.06	90.85	90.87	**93.63**	93.50	93.35	93.58	91.32	91.78
GN, $\epsilon = 8/255$	91.88	**92.01**	91.97	91.84	90.61	90.74	**93.62**	93.41	93.25	93.48	90.34	91.34
FGSM, $\epsilon = 8/255$	44.55	44.58	**49.48**	48.09	41.66	41.88	44.15	43.73	48.20	48.25	**69.70**	65.83
PGD, $\epsilon = 8/255$	34.96	34.45	**39.72**	38.76	32.38	32.32	33.54	33.32	37.64	37.72	**63.96**	58.64
SEA, $k = 10$	44.13	44.35	43.22	43.73	59.58	**62.30**	38.96	37.73	38.31	40.07	**91.16**	90.85
SEA-PGD, $k = 10$	34.88	34.51	33.60	33.94	55.21	**57.74**	25.02	24.48	25.94	26.87	76.18	**90.09**
SEA, $k = 20$	24.55	24.64	24.95	24.88	35.11	**38.06**	25.53	23.70	26.36	28.13	**90.31**	89.01
SEA-PGD, $k = 20$	15.05	14.82	15.25	15.53	27.15	**28.45**	11.63	10.67	13.10	13.81	56.35	**85.92**
Avg	47.75	47.68	48.79	48.60	54.07	**55.30**	45.76	45.07	47.02	47.74	78.67	**82.93**
$T=4, m=10$	CIFAR-100, Rate Encoding						CIFAR-100, Signed Rate Encoding					
CLEAN	**54.71**	54.29	50.73	51.95	49.63	50.22	**58.43**	57.68	52.85	54.44	54.80	54.90
GN, $\epsilon = 8/255$	**54.95**	54.54	50.68	52.08	49.61	50.22	**58.15**	57.77	52.44	54.61	54.63	55.07
FGSM, $\epsilon = 8/255$	26.65	26.30	30.04	**30.19**	25.47	25.83	26.77	26.78	31.84	32.73	32.45	**35.32**
PGD, $\epsilon = 8/255$	18.87	18.92	**24.62**	24.25	19.22	19.05	19.19	19.10	26.88	26.35	26.30	**29.55**
SEA, $k = 10$	28.83	28.71	29.21	29.79	32.35	**33.08**	23.30	23.77	28.80	30.25	**53.28**	52.95
SEA-PGD, $k = 10$	23.00	22.90	24.10	24.42	29.37	**30.01**	17.19	16.63	23.29	24.05	49.01	**49.62**
SEA, $k = 20$	17.67	17.72	20.22	20.37	23.90	**23.98**	14.32	14.42	21.12	21.62	**51.69**	50.59
SEA-PGD, $k = 20$	11.68	11.76	13.74	13.71	19.99	**20.34**	7.67	7.45	13.87	13.87	41.40	**42.11**
Avg	29.55	29.39	30.42	30.85	31.19	**31.59**	28.13	27.95	31.39	32.24	45.45	**46.26**

Table 2: The comparisons are similar to Table 1, except that, $m = 10$, MC simulations were used to compute the model output, resulting in improved model accuracy under both encodings.

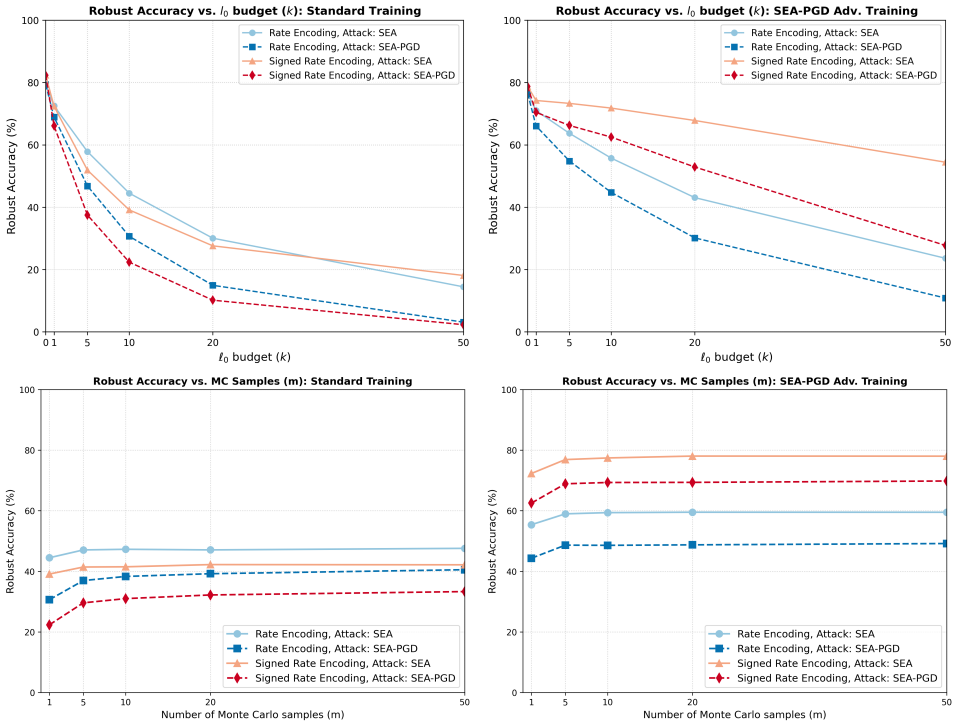

Figure 2: We compare adversarial attacks SEA vs. SEA-PGD on rate encoded vs. signed rate encoded models under standard training (left) on CIFAR-10. The same comparisons are carried out with adversarial training under SEA-PGD (right). We vary the l_0 sparsity budget (k) for the attack while fixing MC simulations, $m = 1$, for inference (top row), and varying MC simulations m while fixing attack sparsity $k = 20$. From the attacker's perspective, SEA-PGD consistently achieves lower robust accuracy compared to SEA, indicating a higher attack success rate, whereas from the model's perspective, signed rate encoded models outperform rate encoded models by achieving higher robust accuracy under adversarial training. In the bottom, We observe improvement in model performance with higher m. Results on other datasets are reported in appendix B in Fig.4 and 5.

6 Limitations and Future Work

We offer improved certified robustness for rate and sign rate encoded SNNs, which helps us reduce the gap between certified and empirical robustness significantly. However, as evident from Eq. 11, the certified l_1 radius offered by the present work is upper bounded by $\frac{1}{2(T+1)} = 0.1$ for $T = 4$, which does not corroborate with empirical evidence, and is a potential limitation of the present work. Our result does not offer a natural extension to l_∞ norm bounded input space attack, which is more popular than l_1-norm bounded attacks. Further, the present result offers a probabilistic guarantee due to randomized smoothing, but whether such a guarantee holds deterministically remains an open problem.

AI USE STATEMENT

In this work, we used generative AI tool (Grammarly.com) only for grammatical correction of english language, and to anonymize the code base for supplementary. We have not used generative AI tools for any other tasks. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We have included the codebase as part of the supplementary material. Table B gives all the training and testing hyperparameters in used in the paper. We will also maintain a GitHub repository containing the experiments reported in the paper, along with important details for reproducibility.

REFERENCES

Srinivas Anumasa, Bhaskar Mukhoty, Velibor Bojkovic, Giulia De Masi, Huan Xiong, and Bin Gu. Enhancing training of spiking neural network with stochastic latency. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 10900–10908, 2024.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320. PMLR, 2019.

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.

Jason K Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *arXiv preprint arXiv:2109.12894*, 2021.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *stat*, 1050:20, 2015.

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

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*, pp. 5209–5218, 2021.

Edgar Lemaire, Loïc Cordone, Andrea Castagnetti, Pierre-Emmanuel Novac, Jonathan Courtois, and Benoît Miramond. An analytical estimation of spiking neural networks energy efficiency. In Mohammad Tanveer, Sonali Agarwal, Seiichi Ozawa, Asif Ekbal, and Adam Jatowt (eds.), *Neural Information Processing*, pp. 574–587, Cham, 2023. Springer International Publishing. ISBN 978-3-031-30105-6.

Jerry Li. Lecture 14: Certified defenses iii: Randomized smoothing. `https://jerryzli.github.io/robust-ml-fall19/lec14.pdf`, 2019. Accessed: 21-08-2026.

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

Bhaskar Mukhoty, Hilal AlQuabeh, Giulia De Masi, Huan Xiong, and Bin Gu. Certified adversarial robustness for rate encoded spiking neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=5bNYf0CqxY`.

Bhaskar Mukhoty, Hilal AlQuabeh, and Bin Gu. Improving generalization and robustness in SNNs through signed rate encoding and sparse encoding attacks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=qLh6Ufvnuc`.

Byunggook Na, Jisoo Mok, Seongsik Park, Dongjin Lee, Hyeokjun Choe, and Sungroh Yoon. Autosnn: Towards energy-efficient spiking neural networks. In *International conference on machine learning*, pp. 16253–16269. PMLR, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

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*, Volume 11 - 2017, 2017. ISSN 1662-453X. doi: 10.3389/fnins.2017.00682. URL `https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2017.00682`.

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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pp. 399–414. Springer, 2020.

A Appendix: Robustness Proofs

A.1 Robustness of Rate Encoding

Rate encoding generates the temporal spike encoding $Z_T = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_d] \in \{0,1\}^{d\times T}$, where each $\mathbf{z}_i \in \{0,1\}^T$ is a temporal vector $\mathbf{z}_i = [z_{i,1}, z_{i,2}, \ldots, z_{i,T}]$, with spikes $z_{i,t} \sim \text{Ber}(x_i)$ is independently sampled using $\text{Ber}(x_i)$. We use the compact notation, $\mathbf{z}_{2:d} = [\mathbf{z}_2, \mathbf{z}_3, \cdots, \mathbf{z}_d]$ and $\mathbf{x}_{2:d} = [x_2, x_3, \cdots, x_d]$.

Lemma 2. *For all class label y, $g_t(\mathbf{x})_y$ is $2t$-Lipschitz under the $\|\cdot\|_1$-norm, and consequently, for all y, $G_T(\mathbf{x})_y$ is $T+1$-Lipschitz under the $\|\cdot\|_1$-norm.*

Proof.

$$
\begin{aligned}
g_t(\mathbf{x})_y &= \mathbb{E}_{\mathbf{z}_i\sim Ber^t(x_i)}[f_t([\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_d])_y] \\
&= \sum_{i=1}^{d} \sum_{\mathbf{z}_i\in\{0,1\}^t} f_t([\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_d])_y \prod_{i=1}^{d}\prod_{j=1}^{t} \mathbb{P}(z_{i,j}|x_i) \\
&= \sum_{i=1}^{d} \sum_{\mathbf{z}_i\in\{0,1\}^t} f_t([[z_{1,1}, \cdots, z_{1,t}], \mathbf{z}_2, \cdots, \mathbf{z}_d])_y \prod_{j=1}^{t} x_1^{z_{1,j}}(1-x_1)^{1-z_{1,j}} \prod_{i\neq 1}\prod_{j=1}^{t} \mathbb{P}(z_{i,j}|x_i) \\
&= \sum_{i=1}^{d} \sum_{\mathbf{z}_i\in\{0,1\}^t} f_t([[z_{1,1}, \cdots, z_{1,t}], \mathbf{z}_{2:d}])_y \prod_{j=1}^{t} x_1^{z_{1,j}}(1-x_1)^{1-z_{1,j}} \mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \qquad (17) \\
&= \sum_{i\neq 1} \sum_{\mathbf{z}_i\in\{0,1\}^t} \sum_{k=0}^{t} c_k(\mathbf{z}_{2:d}) x_1^k (1-x_1)^{t-k} \mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \qquad (18)
\end{aligned}
$$

where, we have used the compressed form $\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) := \prod_{i\neq 1}\prod_{j=1}^{t} \mathbb{P}(z_{i,j}|x_i)$. Note further that, if there are k spikes in the vector $\mathbf{z}_1$, i.e., $\|\mathbf{z}_1\|_1 = \sum_{j=1}^{t} z_{1,j} = k$, then, probability that $\mathbf{z}_1$ has exactly, k spikes is given as:

$$
\prod_{j=1}^{t} x_1^{z_{1,j}}(1-x_1)^{1-z_{1,j}} = x_1^k(1-x_1)^{t-k}
$$

Let $c_k(\mathbf{z}_{2:d}) := \sum_{\|\mathbf{z}_1\|_1=k} f_t([\mathbf{z}_1, \mathbf{z}_{2:d}])_y$ denotes the sum of $f_t(\cdot)_y$ over possible values that $\mathbf{z}_1$ can take with the restriction $\|\mathbf{z}_1\|_1 = k$, so that, each term in the sum has the probability $x_1^k(1-x_1)^{t-k}\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d})$. Since, $0 \le f_t(\cdot)_y \le 1$, we have:

$$
0 \le c_k(\mathbf{z}_{2:d}) \le \binom{t}{k} \qquad (19)
$$

For example,

$$
\begin{aligned}
&0 \le c_0(\mathbf{z}_{2:d}) := f_t([[0,0,\cdots 0], \mathbf{z}_2, \cdots, \mathbf{z}_d])_y \le \binom{t}{0} \\
&0 \le c_1(\mathbf{z}_{2:d}) := f_t([[1,0,\cdots 0], \mathbf{z}_2, \cdots, \mathbf{z}_d])_y + \cdots + f_t([[0,0,\cdots,1], \mathbf{z}_2, \cdots, \mathbf{z}_d])_y \le \binom{t}{1} \\
&0 \le c_t(\mathbf{z}_{2:d}) := f_t([[1,1,\cdots,1], \mathbf{z}_2, \cdots, \mathbf{z}_d])_y \le \binom{t}{t}
\end{aligned}
$$

Taking partial derivative w.r.t. x_1, from Eq.18, we have,

$$\frac{\partial g_t(\mathbf{x})_y}{dx_1} = \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\sum_{k=0}^{t} c_k(\mathbf{z}_{2:d})(kx_1^{k-1}(1-x_1)^{t-k} - (t-k)x_1^k(1-x_1)^{t-k-1})\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d})$$

$$\begin{aligned}
\left|\frac{\partial g_t(\mathbf{x})_y}{dx_1}\right| &= \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\sum_{k=0}^{t} c_k(\mathbf{z}_{2:d})\left|kx_1^{k-1}(1-x_1)^{t-k} - (t-k)x_1^k(1-x_1)^{t-k-1}\right|\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&\leq \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\sum_{k=0}^{t}\binom{t}{k}\left|kx_1^{k-1}(1-x_1)^{t-k} - (t-k)x_1^k(1-x_1)^{t-k-1}\right|\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&\leq \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\sum_{k=0}^{t}\binom{t}{k}\left(kx_1^{k-1}(1-x_1)^{t-k} + (t-k)x_1^k(1-x_1)^{t-k-1}\right)\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&= \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\left(\sum_{k=1}^{t}\binom{t}{k}kx_1^{k-1}(1-x_1)^{t-k} + \sum_{k=0}^{t-1}\binom{t}{k}(t-k)x_1^k(1-x_1)^{t-k-1}\right)\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&= \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t} t\left(\sum_{k=1}^{t}\binom{t-1}{k-1}x_1^{k-1}(1-x_1)^{t-k} + \sum_{k=0}^{t-1}\binom{t-1}{k}x_1^k(1-x_1)^{t-k-1}\right)\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&= \sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t} 2t\left(x_1+1-x_1\right)^{t-1}\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) \\
&= 2t\sum_{i\neq 1}\sum_{\mathbf{z}_i\in\{0,1\}^t}\mathbb{P}(\mathbf{z}_{2:d}|\mathbf{x}_{2:d}) = 2t
\end{aligned}$$

It can be extended from the above that, $\left|\frac{\partial g_t(\mathbf{x})_y}{dx_i}\right| \leq 2t$, for $\forall i \in [d]$, so that, $\|\nabla g_t(\mathbf{x})_y\|_\infty \leq 2t$, which implies $g_t(\cdot)_y$ is $2t$-Lipschitz under $\|\cdot\|_1$-norm. Also since, $G_T(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} g_t(\mathbf{x})$, we have,

$$\|\nabla G_T(\mathbf{x})_y\|_\infty = \|\frac{1}{T}\sum_{t=1}^{T}\nabla g_t(\mathbf{x})_y\|_\infty \leq \frac{1}{T}\sum_{t=1}^{T}\|\nabla g_t(\mathbf{x})_y\|_\infty \leq \frac{1}{T}\sum_{t=1}^{T} 2t = T+1 \tag{20}$$

So that $G_T(\cdot)_y$ is $T+1$-Lipschitz, under $\|\cdot\|_1$-norm.

□

A.2 Robustness of Signed Rate Encoding

Signed Rate encoding generates the temporal spike encoding $Z_T = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_d] \in \{-1, 0, 1\}^{d\times T}$, where each $\mathbf{z}_i \in \{-1, 0, 1\}^T$ is a temporal vector $\mathbf{z}_i = [z_{i,1}, z_{i,2}, \ldots, z_{i,t}]$, with spikes $z_{i,t} \sim \text{sBer}(x_i)$ is independently sampled using $\text{sBer}(x_i)$, and we use the compact notation, $\mathbf{z}_{2:d} = [\mathbf{z}_2, \mathbf{z}_3, \cdots, \mathbf{z}_d]$ and $\mathbf{x}_{2:d} = [x_2, x_3, \cdots, x_d]$.

Lemma 4. *For all y, $\tilde{g}_t(\mathbf{x})_y$ is $2t$-Lipschitz under the $\|\cdot\|_1$-norm, and consequently, for all y, $\tilde{G}_T(\mathbf{x})_y$ is $T+1$-Lipschitz under the $\|\cdot\|_1$-norm.*

Proof. The proof essentially follows the same techniques as Lemma 2, dividing an input pixel $x_i \in [-1, 1]$ into two regimes, $x_i \in [0, 1]$ and $x_i \in [-1, 0)$, and bounding the partial derivatives $\left|\frac{\partial \tilde{g}_t(\mathbf{x})_y}{\partial x_i}\right|$. For convenience of notational exposition, we would work with the partial derivative $\left|\frac{\partial \tilde{g}_t(\mathbf{x})_y}{\partial x_1}\right|$, by choosing $i = 1$, but it is without loss of generality, and holds for any $i \in [d]$.

Case 1: $x_1 \in [0, 1]$**.**

For $x_1 \geq 0$, there will be no negative spikes in the encoding corresponding to pixel x_1, i.e., $z_{1,j} \neq -1, \forall j \in [t]$, so that $\mathbf{z}_1 \in \{0, 1\}^t$. Since only x_1 is restricted to be non-negative, the rest of the encoded input $\mathbf{z}_{2:d}$ continue to retain the full support of sBer, i.e., $\mathbf{z}_{2:d} \in \{-1, 0, 1\}^{(d-1)\times t}$.

Expanding the expectation by marginalizing over $\mathbf{z}_1$ and $\mathbf{z}_{2:d}$:

$$\begin{aligned}
\tilde{g}_t(\mathbf{x})_y &= \mathbb{E}_{Z_t \sim \text{sBer}(\mathbf{x})}\left[\tilde{f}_t([\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_d])_y\right] \\
&= \sum_{\mathbf{z}_1 \in \{0,1\}^t} \sum_{\mathbf{z}_{2:d} \in \{-1,0,1\}^{(d-1)\times t}} \tilde{f}_t([\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_d])_y \prod_{i=1}^{d}\prod_{j=1}^{t} \mathbb{P}(z_{i,j} \mid x_i) \\
&= \sum_{\mathbf{z}_{2:d}} \sum_{\mathbf{z}_1 \in \{0,1\}^t} \tilde{f}_t([\mathbf{z}_1, \mathbf{z}_{2:d}])_y \left(\prod_{j=1}^{t} \mathbb{P}(z_{1,j} \mid x_1)\right) \prod_{i=2}^{d}\prod_{j=1}^{t} \mathbb{P}(z_{i,j} \mid x_i) \\
&= \sum_{\mathbf{z}_{2:d}} \sum_{\mathbf{z}_1 \in \{0,1\}^t} \tilde{f}_t(\mathbf{z}_1, \mathbf{z}_{2:d})_y \left(\prod_{j=1}^{t} x_1^{z_{1,j}}(1-x_1)^{1-z_{1,j}}\right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
&= \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} c_k^+(\mathbf{z}_{2:d}) x_1^k (1-x_1)^{t-k} \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d})
\end{aligned}$$

where we used the compact form $\mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) = \prod_{i=2}^{d}\prod_{j=1}^{t} \mathbb{P}(z_{i,j} \mid x_i)$, and further have:

$$0 \le c_k^+(\mathbf{z}_{2:d}) := \sum_{\substack{\mathbf{z}_1 \in \{0,1\}^t \\ \|\mathbf{z}_1\|_1 = k}} \tilde{f}_t(\mathbf{z}_1, \mathbf{z}_{2:d})_y \le \binom{t}{k}$$

where, $c_k^+(\mathbf{z}_{2:d})$ aggregates the classifier output over all possible realizations of $\mathbf{z}_1 \in \{0,1\}^t$ having exactly k positive spikes with the base classifier probabilities $\tilde{f}_t(\cdot)_y \in [0,1]$. Taking the partial derivative with respect to x_1 using the product rule:

$$\begin{aligned}
\frac{\partial \tilde{g}_t(\mathbf{x})_y}{\partial x_1} &= \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} c_k^+(\mathbf{z}_{2:d}) \left(k x_1^{k-1}(1-x_1)^{t-k} - (t-k)x_1^k(1-x_1)^{t-k-1}\right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
\left|\frac{\partial \tilde{g}_t(\mathbf{x})_y}{\partial x_1}\right| &\le \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} \binom{t}{k} \left(k x_1^{k-1}(1-x_1)^{t-k} + (t-k)x_1^k(1-x_1)^{t-k-1}\right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
&\le t \sum_{\mathbf{z}_{2:d}} \left(\sum_{k=1}^{t} \binom{t-1}{k-1} x_1^{k-1}(1-x_1)^{t-k} + \sum_{k=0}^{t-1} \binom{t-1}{k} x_1^k(1-x_1)^{t-k-1}\right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
&\le t \sum_{\mathbf{z}_{2:d}} \left(2(x_1 + 1 - x_1)^{t-1}\right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
&= 2t \sum_{\mathbf{z}_{2:d}} \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\
&= 2t
\end{aligned}$$

Case 2: $x_1 \in [-1, 0)$**.**

For $x_1 < 0$, there will be no negative spikes in the encoding corresponding to pixel x_1, i.e., $z_{1,j} \neq 1, \forall j \in [t]$, so that $\mathbf{z}_1 \in \{0, -1\}^t$. Hence, the support of $\mathbf{z}_1$ reduces to $\{-1, 0\}^t$ and $\mathbf{z}_{2:d} \in \{-1, 0, 1\}^{(d-1)\times t}$. Since, $x_1 < 0$, we have,

$$\mathbb{P}(z_{1,j} = -1) = -x_1, \qquad \mathbb{P}(z_{1,j} = 0) = 1 + x_1$$

Similar to case 1, the probability that $\mathbf{z}_1$ containing exactly k negative spikes is: $(-x_1)^k(1+x_1)^{t-k}$. We define the aggregate $c_k^-(\mathbf{z}_{2:d})$ to denote the sum of $\tilde{f}_t(\cdot)_y$ over all binary vectors $\mathbf{z}_1 \in \{-1, 0\}^t$ having exactly k negative spikes, with $\tilde{f}_t(\cdot)_y \in [0, 1]$

$$0 \le c_k^-(\mathbf{z}_{2:d}) := \sum_{\substack{\mathbf{z}_1 \in \{-1,0\}^t \\ \|\mathbf{z}_1\|_1 = k}} \tilde{f}_t(\mathbf{z}_1, \mathbf{z}_{2:d})_y \le \binom{t}{k}$$

Expanding the expectation and substituting our grouped terms, we get:

$$\begin{aligned}\tilde{g}_t(\mathbf{x})_y &= \sum_{\mathbf{z}_{2:d}} \sum_{\mathbf{z}_1 \in \{0,-1\}^t} \tilde{f}_t(\mathbf{z}_1, \mathbf{z}_{2:d})_y \left(\prod_{j=1}^{t} (-x_1)^{z_{1,j}} (1+x_1)^{1-z_{1,j}} \right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &= \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} c_k^-(\mathbf{z}_{2:d}) (-x_1)^k (1+x_1)^{t-k} \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d})\end{aligned}$$

Taking the partial derivative with respect to x_1:

$$\frac{\partial \tilde{g}_t(\mathbf{x})}{\partial x_1} = \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} c_k^- \left(-k(-x_1)^{k-1}(1+x_1)^{t-k} + (t-k)(-x_1)^k (1+x_1)^{t-k-1} \right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d})$$

To bound the absolute value, we apply the triangle inequality. Crucially, because $x_1 \in [-1, 0)$, we have $(-x_1) > 0$ and $(1 + x_1) > 0$. Thus, all polynomial terms inside the parentheses are strictly positive, allowing us to drop any absolute value signs on the terms and seamlessly substitute the upper bound $c_k^- \leq \binom{t}{k}$:

$$\begin{aligned}\left| \frac{\partial \tilde{g}_t(\mathbf{x})}{\partial x_1} \right| &\leq \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} \binom{t}{k} \left(k(-x_1)^{k-1}(1+x_1)^{t-k} + (t-k)(-x_1)^k (1+x_1)^{t-k-1} \right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &\leq t \sum_{\mathbf{z}_{2:d}} \sum_{k=0}^{t} \left(\binom{t-1}{k-1} (-x_1)^{k-1}(1+x_1)^{t-k} + \binom{t-1}{k} (-x_1)^k (1+x_1)^{t-k-1} \right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &\leq t \sum_{\mathbf{z}_{2:d}} \left((-x_1 + 1 + x_1)^{t-1} + (-x_1 + 1 + x_1)^{t-1} \right) \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &= 2t \sum_{\mathbf{z}_{2:d}} (1)^{t-1} \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &= 2t \sum_{\mathbf{z}_{2:d}} \mathbb{P}(\mathbf{z}_{2:d} \mid \mathbf{x}_{2:d}) \\ &= 2t\end{aligned}$$

Note that $\frac{\partial \tilde{g}_t(\mathbf{x})}{\partial x_1}$ exists for all $x_1 \in [-1, 1]$ and combining, case 1 and 2, we conclude, $\left| \frac{\partial \tilde{g}_t(\mathbf{x})}{\partial x_1} \right| \leq 2t$ for all $x_1 \in [-1, 1]$, which without loss of generality shows, $\left| \frac{\partial \tilde{g}_t(\mathbf{x})}{\partial x_i} \right| \leq 2t$ for all $i \in [d]$, giving, $\|\nabla \tilde{g}_t(\mathbf{x})_y\|_\infty \leq 2t$. So that,

$$\|\nabla \tilde{G}_T(\mathbf{x})_y\|_\infty = \|\frac{1}{T} \sum_{t=1}^{T} \nabla \tilde{g}_t(\mathbf{x})_y\|_\infty \leq \frac{1}{T} \sum_{t=1}^{T} \|\nabla \tilde{g}_t(\mathbf{x})_y\|_\infty \leq \frac{1}{T} \sum_{t=1}^{T} 2t = T + 1 \tag{21}$$

proving, $\tilde{G}_T(\cdot)_y$ is $T + 1$-Lipschitz, under $\|\cdot\|_1$-norm. □

B Additional Experimental Results

We provide the certified accuracy experiments for all three datasets, as well as, for standard and PGD adversarially trained model.

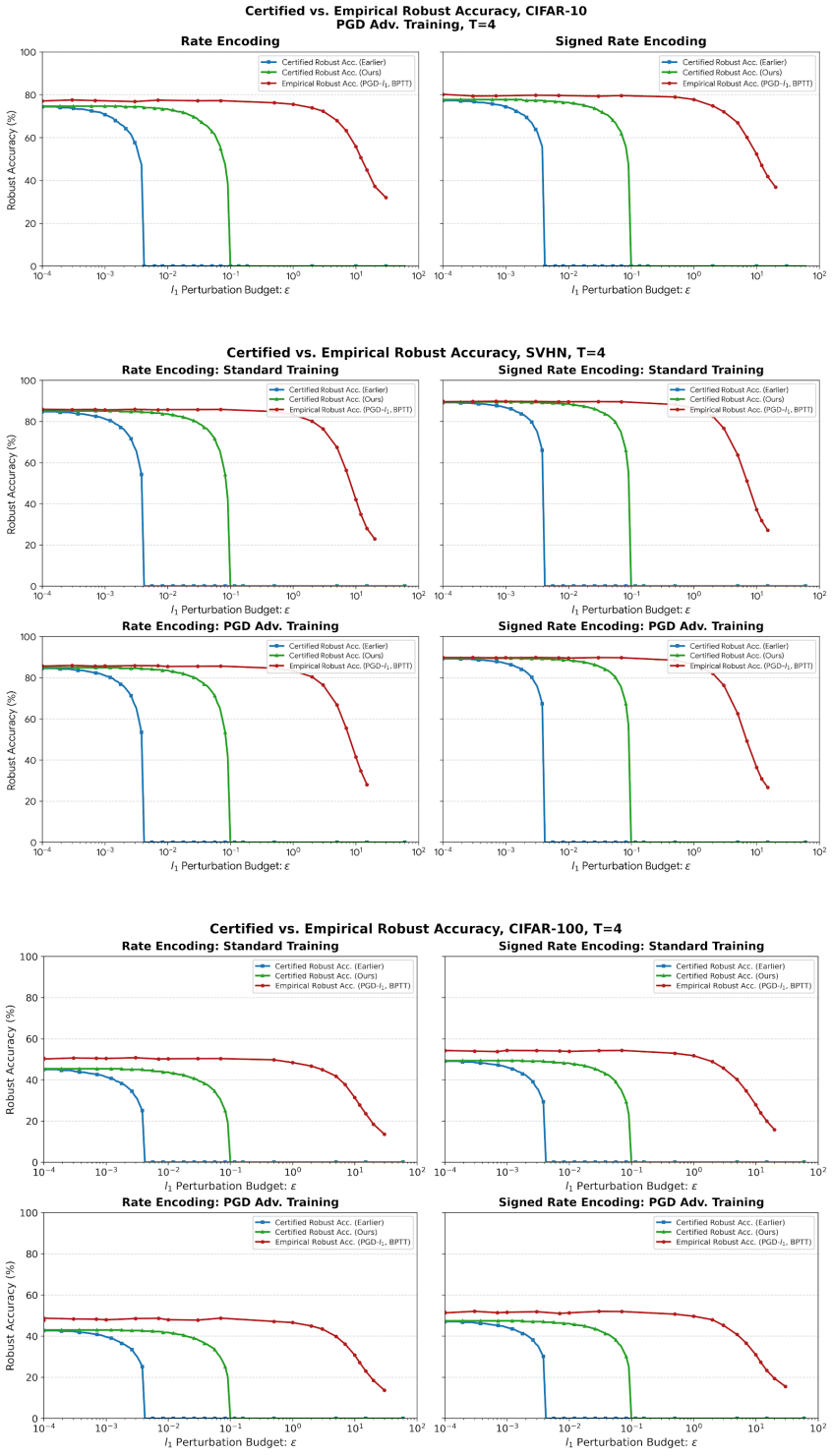

Figure 3: Additional results on certified vs. empirical robust accuracy.

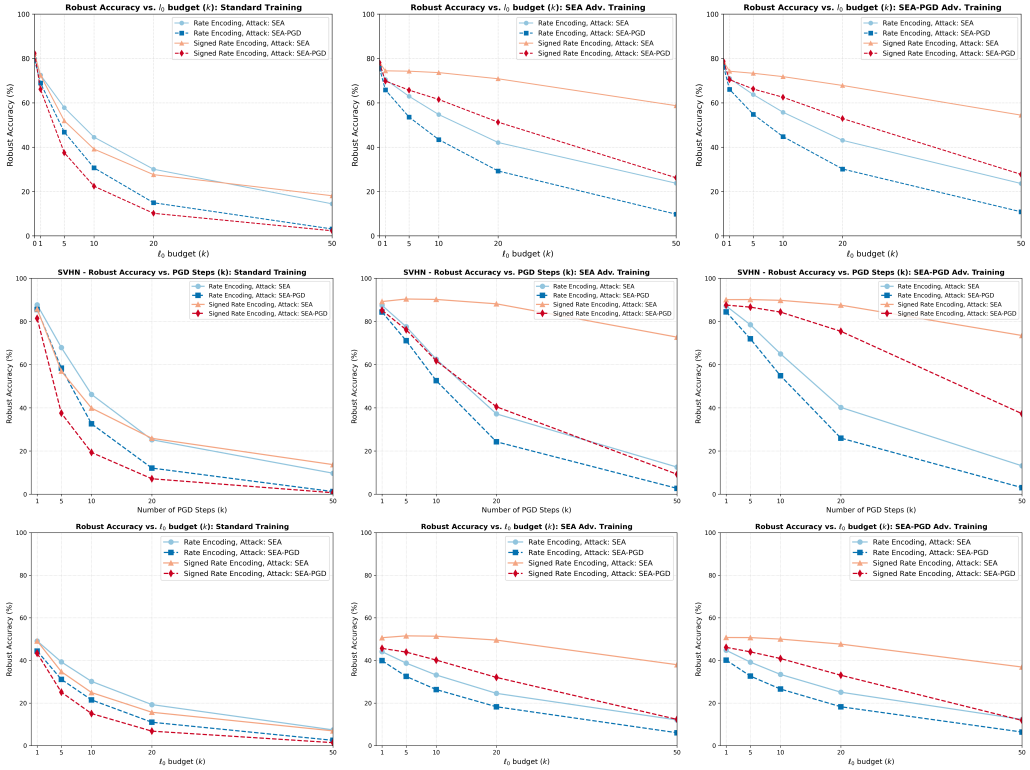

Figure 4: Accuracy vs. l_0 budget (k) on CIFAR-10 (top), SVHN (middle), and CIFAR-100 (bottom).

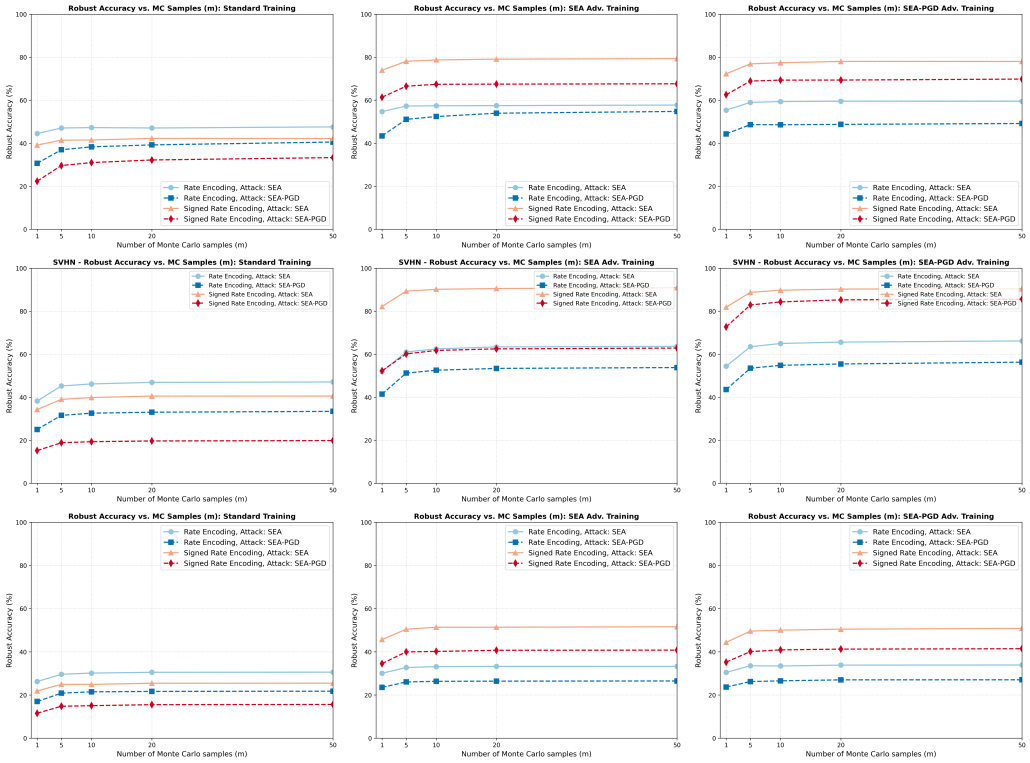

Figure 5: Accuracy vs. MC simulation (m) on CIFAR-10 (top), SVHN (middle), and CIFAR-100 (bottom).

Metric / ϵ	1.0×10^{-4}	7.0×10^{-4}	2.9×10^{-3}	9.1×10^{-3}	0.01	0.05	0.50	1.0	10.0	60.0
Certified Robust Acc.(Earlier)	76.50	74.05	60.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Certified Robust Acc.(ours)	76.89	76.89	76.50	75.22	75.22	64.56	0.00	0.00	0.00	0.00
Empirical Robust Acc.(PGD-l_1,BPTT)	79.15	78.41	78.41	79.31	79.31	78.56	78.35	77.00	54.47	33.17

Table 3: Certified vs. Empirical Robust Accuracy (Rate Encoding, Std. Training, CIFAR-10)

Metric / ϵ	1.0×10^{-4}	7.0×10^{-4}	2.9×10^{-3}	9.1×10^{-3}	0.01	0.05	0.50	1.0	10.0	60.0
Certified Robust Acc.(Earlier)	80.07	77.97	66.62	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Certified Robust Acc.(Ours)	80.37	80.37	80.07	79.04	79.04	70.30	0.00	0.00	0.00	0.00
Empirical Robust Acc.(PGD-l_1,BPTT)	82.04	82.14	82.14	81.93	81.93	82.19	80.74	79.55	48.48	33.52

Table 4: Certified vs. Empirical Robust Accuracy (Signed Rate Encoding, Std. Training, CIFAR-10)

Parameter	**CIFAR-10 & CIFAR-100**	**SVHN**
Number epochs	200	200
Mini batch size	128	128
T	4	4
LIF: β	1.0	1.0
LIF: u_0	0	0
LIF: u_{th}	1.0	1.0
Learning Rate	0.1	0.1
FGSM/PGD/GN (train): ϵ	8/255 ,8/255	2/255 ,8/255
PGD (train, test): η	2/255,2/255	1/255,2/255
PGD (train, test): Iteration	4,8	2,8
SEA (train, test): k	5, {10, 20}	5*,{10, 20}
SEA-PGD (train, test): k	5, {10, 20}	5*,{10, 20}
SEA-PGD (train, test): Iteration	5, {10, 20}	5,{10, 20}
Optimizer: SGD with momentum: 0.9, weight decay: 5×10^{-4}, Rate Scheduler: cosine annealing		

Table 5: Training (Adversarial) Hyperparameters for Models. * For SVHN, rate encoded models we use $k = 2$, as $k = 5$ does not converge.