LEARNING TO GENERATE NOISE FOR ROBUSTNESS AGAINST MULTIPLE PERTURBATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial learning has emerged as one of the successful techniques to circumvent the susceptibility of existing methods against adversarial perturbations. However, the majority of existing defense methods are tailored to defend against a single category of adversarial perturbation (e.g. ℓ_∞-attack). In safety-critical applications, this makes these methods extraneous as the attacker can adopt diverse adversaries to deceive the system. To tackle this challenge, we propose a novel meta-learning framework that explicitly learns to generate noise to improve the model's robustness against multiple types of attacks. However, training on multiple perturbations simultaneously significantly increases the computational overhead during training. To address this issue, we train our MNG while randomly sampling an attack at each epoch, which incurs negligible overhead over standard adversarial training. We validate the robustness of our framework on various datasets and against a wide variety of unseen perturbations, demonstrating that it significantly outperforms the relevant baselines across multiple perturbations with marginal computational cost compared to the multiple perturbations training.

1 INTRODUCTION

Deep neural networks have demonstrated enormous success on multiple benchmark applications (Amodei et al., 2016; Devlin et al., 2018), by achieving super-human performance on certain tasks. However, to deploy them to safety-critical applications (Shen et al., 2017; Chen et al., 2015; Mao et al., 2019), we need to ensure that the model is *robust* as well as *accurate*, since incorrect predictions may lead to severe consequences. Particularly, it is well-known that the existing neural networks are highly susceptible to carefully crafted image perturbations which are imperceptible to humans but derail the predictions of these otherwise accurate networks.

The emergence of adversarial examples has received significant attention in the research community and several defense mechanisms have been proposed (Ross & Doshi-Velez, 2018; Madry et al., 2017; Tramèr & Boneh, 2019; Zhang et al., 2019). However, despite such a large literature to improve upon the adversarial robustness of neural networks, most of the existing defenses leverage the knowledge of the adversaries and are based on the assumption of only a single perturbation. Consequently, many of the proposed defenses were circumvented by stronger attacks (Athalye et al., 2018; Uesato et al., 2018; Athalye & Carlini, 2018). Several recent works have claimed the problem of over-fitting of the existing defenses based on this assumption of the adversary (Schott et al., 2018; Tramèr & Boneh, 2019) and have demonstrated the vulnerability of existing defense methods against multiple adversarial perturbations. Yet there are only a handful of techniques that improve robustness across multiple perturbations.

The multiple perturbations training (Tramèr & Boneh, 2019) comes at an additional cost; it increases the training cost by a factor of four over adversarial training, which is already ten times more costly than standard training. This slowdown factor hinders the research progress of robustness against multiple perturbations due to the large computation overhead incurred during training. Some recent works reduce this cost by reducing the complexity of generating adversarial examples (Shafahi et al., 2019; Wong et al., 2020), however, they are limited to ℓ_∞ adversarial training. To tackle the issue, we advocate *Stochastic Adversarial Training (SAT)* that samples a perturbation from the distribution of perturbations to produce robust classifiers against multiple perturbations with negligible additional training cost over the standard adversarial training. Furthermore, we complement all the

methods with efficient training techniques (Smith, 2017; Wong et al., 2020) to accelerate training for multiple perturbations by a factor of eight on a single machine with four GeForce RTX 2080Ti using Wid ResNet 28-10 architecture (Zagoruyko & Komodakis, 2016). To this end, utilizing our SAT, we further reduce up to 50% training time compared to the baseline strategies.

To tackle the critical challenge of robustness to multiple perturbations, we investigate the relationship between generalization to diverse perturbations and noise injection. In particular, we resort to meta-learning (Thrun, 1996; Finn et al., 2017), and propose a *Meta Noise Generator (MNG)* that learns to stochastically perturb a given sample in a meta-learning framework to improve the generalization across perturbations. Our algorithm is motivated by the popular phenomenon of noise regularization being a common technique to improve the generalization performance of deep neural networks (Huang et al., 2016; Srivastava et al., 2014; Noh et al., 2017; Lee et al., 2020). Additionally, we introduce a novel *augmentation loss* to improve the performance utilizing our generated samples, adversarial samples, and clean samples to push the decision boundary and enforce a smooth and robust network across multiple perturbations. To this end, we combine our proposed meta-noise generator with SAT to achieve state-of-the-art networks robust against multiple perturbations with a significant reduction in training time compared to the multiple perturbations training techniques.

In summary, the major contributions of this paper are as follows:

- We tackle the critical problems of robustness to multiple perturbations and computational overhead incurred by training with multiple adversarial perturbations. To this end, we propose *Stochastic Adversarial Training (SAT)* that samples from a distribution of perturbations and achieves significant improvement over standard adversarial training with a negligible increase in computation cost.
- We formulate a novel *Meta-Noise Generator (MNG)* that explicitly meta-learns an input-dependent noise generator, such that it outputs stochastic noise distribution to improve the model's robustness against multiple types of adversarial perturbations.
- We validate our multiple perturbations adversarial learning framework on various datasets with state-of-the-art adversarial attacks, on which it achieves state-of-the-art performance, highlighting its practical impact in real-world systems.

2 Related Work

Robustness against single adversarial perturbation. In the past few years, multiple defenses have been proposed to defend against a single adversarial perturbation (Madry et al., 2017; Xiao et al., 2020; Wang et al., 2020; Carmon et al., 2019; Zhang et al., 2019) and have been consequently circumvented by subsequent attacks (Athalye et al., 2018; Schott et al., 2018; Brendel et al., 2018; Tramer et al., 2020). Adversarial-training based defenses (Zhang et al., 2019; Madry et al., 2017; Carmon et al., 2019) have been the only exceptions that have withstood the intense scrutiny and have provided empirical gains in adversarial robustness. There have also been various attempts to increase robustness using models that leverage the representative power of generative models to project onto the manifold of standard input images (Samangouei et al., 2018; Song et al., 2018). Consequently. these models have shown to be ineffective by stronger attacks (Carlini & Wagner, 2017; Athalye et al., 2018). Recently, the meta-learning framework has emerged as a novel technique to construct adversarial examples (Jang et al., 2019b; Chen et al., 2020). However, our proposed method is fundamentally orthogonal to all these methods; in contrast to learning the generative model to model the adversarial examples, we leverage the generator to explicitly learn an input-dependent optimal noise distribution to lower adversarial error across multiple perturbations.

Robustness against multiple adversarial perturbations. Schott et al. (2018) demonstrated that ℓ_∞ adversarial training is highly susceptible to ℓ_0/ℓ_2-norm adversarial perturbations and used multiple VAEs to defend against multiple perturbations on the MNIST dataset. However, it was not scalable and limited to the MNIST dataset. Tramèr & Boneh (2019) investigated the theoretical/empirical trade-offs between multiple perturbations and introduced adversarial training with worst/average perturbations to defend against multiple perturbations. However, computing all the perturbations is impractical for multiple perturbations and large scale datasets. In contrary, our proposed stochastic adversarial training in Section 4 overcomes this limitation, with improved performance over their method and has a negligible increase in training cost over standard adversarial training.

3 ROBUSTNESS AGAINST MULTIPLE PERTURBATIONS

Let us consider a classifier f_θ with parameters θ, inputs $\mathcal{X}$, and labels $\mathcal{Y}$. We consider the ℓ_p norm distance under the additive threat model (Laidlaw & Feizi, 2019) and adopt the projected-gradient descent (PGD) for crafting ℓ_p perturbations. In particular, for loss-function $\mathcal{L}$, norm-ball $\mathcal{B}(x, \varepsilon)$, and step-size α, PGD attack (Madry et al., 2017) optimizes for the following objective:

$$x^{t+1} = \underset{\mathcal{B}(x,\varepsilon)}{\text{proj}} \left(x^t + \underset{||v|| \leq \alpha}{\arg\max}\, v^T \nabla_x \mathcal{L}\left(f_\theta\left(x\right), y\right) \right). \tag{1}$$

where proj is the projection operator projecting the input onto the norm ball $\mathcal{B}$. Our objective is to attain robustness against attack procedures $\mathcal{A}_k, \forall \mathcal{A}_k \in p(\mathcal{A})$, where $p(\mathcal{A})$ denotes the attacks distribution, such that $\max_{\delta_k \in \mathcal{B}(x,\varepsilon)} L\left(f\left(x + \delta_k\right), y\right) \approx L\left(f\left(\mathcal{A}_k\left(x\right)\right), y\right)$.

3.1 ADVERSARIAL TRAINING

In the standard machine learning, the model iteratively optimizes the parameters θ for a network to generalize to unseen standard test examples. On the other hand, adversarial training (Kurakin et al., 2016; Madry et al., 2017) is an empirical optimization technique that optimizes the network using a min-max formulation:

$$\min_\theta \; \mathbb{E}_{(x,y)\sim\mathcal{D}}\, \mathcal{L}\left(f\left(\mathcal{A}\left(x\right)\right), y\right). \tag{2}$$

However, adversarial training is vulnerable to perturbations that are unseen during training (Schott et al., 2018; Tramèr & Boneh, 2019), which has been referred to as *overfitting* on the particular type of training perturbation. Recently, Tramèr & Boneh (2019) extended the adversarial training to multiple attacks. Their proposed strategies can more formally be defined as follows:

1. **The maximum over all perturbations**: It optimizes the outer objective in Eq. (2) on the strongest adversarial perturbation for each input example x from the whole set of n additive perturbations.

$$\min_\theta \; \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\arg\max_k \mathcal{L}\left(f\left(\mathcal{A}_k\left(x\right)\right), y\right) \right]. \tag{3}$$

2. **The average over all perturbations:** It optimizes the outer objective in Eq. (2) on the whole set of n additive perturbations.

$$\min_\theta \; \mathbb{E}_{(x,y)\sim\mathcal{D})} \frac{1}{n} \sum_{k=1}^{k=n} \mathcal{L}\left(f\left(\mathcal{A}_k\left(x\right), y\right)\right). \tag{4}$$

However, the practicality of these methods is limited due to an increased computational overhead for training To overcome this limitation, we introduce our framework in the next section, which has comparatively marginal overhead over adversarial training and achieves state-of-the-art robustness to multiple adversarial perturbations.

4 LEARNING NOISE GENERATOR FOR ROBUSTNESS

4.1 STOCHASTIC ADVERSARIAL TRAINING (SAT)

To eliminate the computation overhead in the aforementioned strategies, we propose a novel strategy against multiple adversarial perturbations. Specifically, we conjecture that it is essential to cover the threat model during training, not utilizing all the perturbations simultaneously. We formulate the threat model as a random attack $\mathcal{A}_k$ sampled from a distribution of attacks $p(A)$ during each episode (or batch) of training which prevents overfitting on a particular adversarial perturbation. More formally, the training objective of SAT can be defined as follows:

$$\min_\theta \; \mathbb{E}_{\substack{(x,y)\sim\mathcal{D} \\ \mathcal{A}_k\sim p(\mathcal{A})}} \mathcal{L}\left(f\left(\mathcal{A}_k(x), y\right)\right). \tag{5}$$

Our proposed objective is a drastic simplification of the average one in Eq. (4), which makes it highly efficient for multiple perturbations. We conjecture that optimizing via SAT promotes generalization (due to its stochasticity) by preventing over-fitting on a single adversary and helps convergence, as like mini-batch training for empirical risk minimization helps to generalize neural networks (Hardt et al., 2016; Wu et al., 2018; Zhou et al., 2019). Our experimental results confirm our hypothesis, demonstrating that it outperforms the baselines with significantly lower training costs.

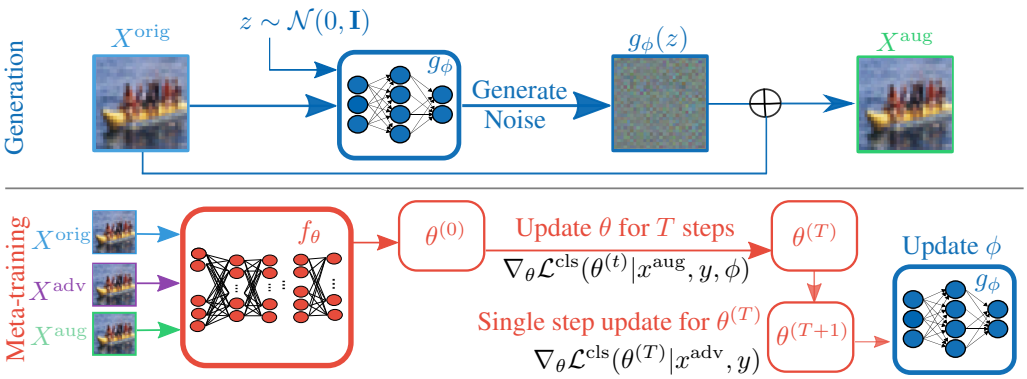

Figure 1: Our classifier is parametrized by θ and the generator is parametrized by ϕ. **Top: Generation:** The generator takes stochastic noise and input X^{orig} to generate the noise-augmented sample X^{aug}. **Bottom: Meta-training of generator:** The meta-noise generator parameters are learned via meta-learning to minimize the adversarial classification loss after $\theta^{(T+1)}$ updates of adversarial classifier model.

4.2 META-NOISE GENERATOR (MNG)

Our MNG framework learns to stochastically perturb adversarial samples for generalization across multiple perturbations. It explicitly meta-learns (Thrun & Pratt, 1998; Finn et al., 2017) the parameters g_ϕ of a noise generator to generate an input-dependent noise distribution to alleviate the issue of generalization across multiple adversaries. To train our adversarial classifier jointly with MNG, we alternatively update the parameters θ of the classifier with the parameters ϕ of MNG (see Bottom Figure 1). In particular, we generate x^{adv} for a mini-batch of size m using Eq. (5) and compute $\mathcal{L}^{\text{cls}}(\theta \mid x^{\text{adv}}, y)$, where $\mathcal{L}^{\text{cls}}$ is corresponding cross-entropy classification loss. Then, we sample $z \sim \mathcal{N}(0, \mathbf{I})$ for input to our generator jointly with the clean example x which generates a noise-augmented sample x^{aug} (see Top Figure 1). Our final training objective to train our model can be formulated as:

$$\mathcal{L}^{\text{total}}(\theta \mid x, y) = \min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_i^{\text{cls}}(\theta \mid x^{\text{adv}}, y) + \mathcal{L}_i^{\text{aug}}(\theta \mid x^{\text{aug}}, y, \phi). \tag{6}$$

Augmentation loss. It has been prominent in the research community that smoother networks lead to robustness of deep networks (Cohen et al., 2019; Engstrom et al., 2018; Madaan et al., 2020). Recently, KL Divergence between $\hat{p}(y \mid x)$ and $\hat{p}(y \mid x^{\text{adv}})$ has been incorporated into the adversarial loss (Zhang et al., 2019; Carmon et al., 2019). Based on this motivation, we minimize the Jensen-Shannon Divergence (JSD) among the posterior distributions of $\hat{p}(y \mid x)$ and $\hat{p}(y \mid x^{\text{adv}})$, and $\hat{p}(y \mid x^{\text{aug}})$. Specifically, our augmentation loss in Eq. (6) can be defined as follows:

$$\mathcal{L}^{\text{aug}}(\theta \mid x, y) = \min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \beta \cdot \mathcal{L}_i^{\text{cls}}(\theta \mid x^{\text{aug}}, y, \phi) + \gamma \cdot \text{JSD}_i(\hat{p}(y \mid x); \hat{p}(y \mid x^{\text{adv}}); \hat{p}(y \mid x^{\text{aug}})). \tag{7}$$

The computation of the optimal ϕ for MNG requires bi-level optimization. However, bi-level optimization for adversarial training would be computationally expensive. To tackle this challenge, we adopt an online approximation (Ren et al., 2018; Jang et al., 2019a) to update θ and ϕ using a single-optimization loop. Specifically, given current parameters θ of our adversarial classifier, we update MNG parameters ϕ using the following training scheme:

1. **Update model parameters for T steps.** First, we update θ to minimize $\mathcal{L}^{\text{cls}}(\theta \mid x^{\text{aug}}, y, \phi)$ for T steps (we chose $T = 2$ in our experiments) which ensures the learning of the adversarial classifier using the knowledge from the noise-augmented sample constructed by MNG. It explicitly increases the influence of the noise-augmented samples on the classifier in the inner loop. More specifically, $\theta^{(t)}$ moves along the following descent direction on a mini-batch of training data, i.e.,

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \mathcal{L}_i^{\text{cls}}(\theta^{(t)} \mid x^{\text{aug}}, y, \phi), \quad \text{where,} \quad x^{\text{aug}} = \underset{\mathcal{B}(x,\varepsilon)}{\text{proj}} \left(x + g_\phi(z, x)\right). \tag{8}$$

2. **Adapt model parameters on single step.** Second, perform one-step update to update $\theta^{(T)}$ to $\theta^{(T+1)}$ to minimize adversarial classification loss from Eq. (5). This step explicitly models the adaptation of adversarial model parameters in the presence of the noise-augmented data using a single step of update:

$$\theta^{(T+1)} = \theta^{(T)} - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \mathcal{L}_i^{\text{cls}}(\theta^{(T)} \mid x^{\text{adv}}, y). \tag{9}$$

3. **Update generator parameters.** In the last step, after receiving feedback from the classifier, we measure the adversarial loss from Eq. (5) and adapt ϕ to minimize this loss. In particular, ϕ performs the following update step to ameliorate the classifier parameters θ in the next step:

$$\phi = \phi - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla_\phi \mathcal{L}_i^{\text{cls}}(\theta^{(T+1)} \mid x^{\text{adv}}, y). \tag{10}$$

We list the detailed algorithm in Algorithm 1. Formally, the overall objective for a mini-batch of size m and learning rate α can be defined as follows:

$$\begin{aligned} &\min_{\phi} \ \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_i^{\text{cls}}(\theta^{(T+1)} \mid x^{\text{adv}}, y) \\ \text{subject to} \quad &\theta^{(T+1)} = \theta^{(T)} - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \mathcal{L}_i^{\text{cls}}(\theta^{(T)} \mid x^{\text{adv}}, y), \\ &\theta^{(t+1)} = \theta^{(t)} - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \mathcal{L}_i^{\text{cls}}(\theta^{(t)} \mid x^{\text{aug}}, y, \phi), \\ &t = 0, \ldots T-1. \end{aligned} \tag{11}$$

To summarize, SAT consists of perturbation sampling to generate adversarial examples and MNG stochastically perturbs the clean examples in a meta-learning framework to explicitly lower the adversarial classification loss on the sampled perturbation. Lastly, the adversarial classifier utilizes the generated samples, adversarial samples and clean samples to optimize the classification and augmentation loss. Overall, our proposed framework then optimizes the classification loss coupled with our augmentation loss which encourages generalization across diverse perturbations.

5 Experiments

5.1 Experimental setup

Baselines and our model. We compare the performance of our proposes method against various relevant baselines trained with single and multiple perturbations.

1. **Nat.** Standard convolution neural network.
2. **Adversarial Training with ℓ_p norm attack (Adv$_p$)** (Madry et al., 2017). Adversarial trained network with ℓ_p norm ball $\mathcal{B}(x, \varepsilon)$ using PGD attack. We use ℓ_∞, ℓ_1, and ℓ_2 norm-balls for adversarial training.
3. **TRADES with ℓ_∞ norm attack (TRADES$_\infty$)** (Zhang et al., 2019). TRADES adversarial trained network with ℓ_∞ norm ball $\mathcal{B}(x, \varepsilon)$ using PGD attack.
4. **Adversarial Training with robust self-training with ℓ_∞ norm (RST$_\infty$)** (Carmon et al., 2019). Adversarial training with robust self-training using ℓ_∞ PGD attack.
5. **Adversarial Training with maximum/average strategies (Adv$_{\text{max}}$ and Adv$_{\text{avg}}$)** (Tramèr & Boneh, 2019). Adversarial training with the maximum (see Eq. (3)) and average (see Eq. (4)) over all perturbations.
6. **Meta Noise Generator with our stochastic adversarial training (MNG$_{\text{SAT}}$).** Our proposed meta noise generator coupled with stochastic adversarial training.

Datasets. We evaluate our method on popular benchmark datasets for evaluation of adversarial robustness of neural networks including CIFAR-10 (Krizhevsky, 2012) and SVHN (Netzer et al., 2011) dataset. See the appendix for more detailed information for the datasets.

Table 1: **Evaluation on CIFAR-10 and SVHN datasets.** For ℓ_∞ perturbations, we use PGD (Madry et al., 2017), Brendel and Bethge attack (Brendel et al., 2019), and AutoAttack (cro, 2020). For ℓ_2 perturbations, we use CarliniWagner attack (Carlini & Wagner, 2017), PGD (Madry et al., 2017), Brendel and Bethge attack (Brendel et al., 2019), and AutoAttack (cro, 2020). For ℓ_1 perturbations, we use SLIDE (Tramèr & Boneh, 2019), Salt and pepper (Rauber et al., 2017), and EAD attack (Chen et al., 2018). We report the union and average of all the attacks for each ℓ_p norm averaged across 3 runs, the best and second-best results are highlighted in bold and underline respectively. Time denotes the training time in hours.

	Model	$\text{Acc}_{\text{clean}}$	ℓ_∞	ℓ_1	ℓ_2	$\text{Acc}^{\text{union}}_{\text{adv}}$	$\text{Acc}^{\text{avg}}_{\text{adv}}$	Time (h)
CIFAR-10	Adv_∞	86.8± 0.4	44.9± 0.5	12.8± 0.4	69.3± 0.1	12.9± 0.4	58.1± 0.2	<u>4.5</u>
	Adv_1	**93.3± 0.4**	0.0± 0.0	**78.1± 0.8**	0.00± 0.00	0.0± 0.00	35.9± 0.2	8.1
	Adv_2	<u>91.7± 0.0</u>	20.7± 0.1	27.7± 0.5	**76.8± 0.2**	18.3± 0.3	58.9± 0.1	**3.7**
	Trades_∞	84.7± 0.1	48.9± 0.5	17.9± 0.4	69,4± 0.0	17.9± 0.4	59.7± 0.0	5.2
	RST_∞	88.9± 0.0	**54.9± 3.1**	22.0± 0.3	73.6± 0.0	22.1± 0.3	64.3± 0.0	58.8
	Adv_{avg}	87.1± 0.1	33.8± 0.4	49.0± 0.1	<u>74.9± 0.2</u>	33.0± 0.4	<u>64.5± 0.1</u>	16.9
	Adv_{max}	85.4± 0.1	<u>39.9± 0.8</u>	44.6± 0.0	73.2± 0.0	<u>37.3± 0.2</u>	64.1± 0.1	16.3
	MNG_{SAT}	84.9± 0.0	38.7± 0.3	<u>52.6± 0.3</u>	73.5± 0.6	**38.6± 0.3**	**65.2± 0.1**	11.2
SVHN	Adv_∞	92.8± 0.1	46.2± 0.4	3.0± 0.1	59.2± 0.4	3.0± 0.1	58.1± 0.0	<u>6.2</u>
	Adv_1	92.4± 0.0	0.0± 0.0	**77.9± 8.2**	0.0± 0.0	0.0± 0.0	28.0± 3.1	11.8
	Adv_2	<u>94.9± 0.0</u>	18.7± 0.3	30.3± 0.3	**79.3± 0.0**	16.4± 0.5	63.0± 0.0	**6.1**
	Trades_∞	93.9± 0.0	<u>49.9± 2.9</u>	1.6± 0.1	56.0± 1.8	1.6± 0.1	57.5± 0.2	7.9
	RST_∞	**95.6± 0.0**	**60.9± 3.6**	0.7± 0.4	60.6± 0.4	0.7± 0.4	62.4± 0.1	112.5
	Adv_{avg}	92.6± 0.1	17.4± 0.2	<u>54.2± 6.9</u>	74.7± 3.5	<u>16.6± 0.2</u>	<u>63.4± 0.1</u>	24.1
	Adv_{max}	88.2± 1.6	5.9± 1.2	48.3± 4.1	31.0± 6.0	5.8± 1.3	55.7± 0.9	22.7
	MNG_{SAT}	93.4± 0.1	34.2± 2.4	36.6± 0.2	<u>77.7± 1.0</u>	**27.2± 0.3**	**67.2± 0.1**	11.9

Evaluation setup. We perform an exhaustive evaluation of our model on state-of-the art adversarial attacks (Carlini & Wagner, 2017; Madry et al., 2017; Chen et al., 2018; Schott et al., 2018; Brendel et al., 2019; Tramèr & Boneh, 2019; cro, 2020) with $\varepsilon = \{0.03, 8, 0.31\}$ for ℓ_∞, ℓ_1, and ℓ_2 attacks respectively. We use the Foolbox (Rauber et al., 2017) library for all the attacks and all our algorithms are implemented in Pytorch (Paszke et al., 2019). We list the hyper-parameters in the appendix.

5.2 Comparison of robustness against multiple perturbations

Results with CIFAR-10 dataset. Table 1 shows the experimental results for the CIFAR-10 dataset. It is evident from the results that MNG_{SAT} outperforms Adv_{avg} for all the attacks, except ℓ_2-norm attacks. Moreover, MNG_{SAT} achieves $\sim 33\%$ reduction in training time compared to the multiple perturbations training baselines. Interestingly, MNG_{SAT} also shows an improvement over Adv_{max}, which is fundamentally designed to address the worst perturbation. This demonstrates the effectiveness of MNG in preventing overfitting over a single perturbation and it's generalization ability to diverse perturbations.

Results with SVHN dataset. As shown in Table 1, MNG_{SAT} significantly outperforms Adv_{avg}, Adv_{max} on $\text{Acc}^{\text{max}}_{\text{adv}}$ and $\text{Acc}^{\text{max}}_{\text{adv}}$ metric with a $\sim 50\%$ reduction in training time. Additionally, MNG_{SAT} achieves significant better performance over ℓ_1 adversarial training with comparable training time which illustrates the utility of our method over standard adversarial training. This implies that our method also leverages the knowledge across multiple perturbations and improves the performance over single perturbation adversarial training.

Results with common corruptions. We further evaluate our model on corrupted datasets (16 different types with 5 levels of intensity each). Note that, the scope of this paper is on robustness against adversarial perturbations. Thus, we mainly compare MNG_{SAT} to the baselines trained with multiple perturbations. As shown in Figure 2, MNG_{SAT} outperforms both the baselines for the noise and blur corruptions for CIFAR-10-C dataset. Moreover, it obtains comparable performance on other types of corruptions.

Table 2: Ablation study analyzing the significance of Stochastic Adversarial Training (SAT) and Meta noise generator with SAT (MNG_{SAT}). The best results are highlighted in bold.

	Model	Clean Acc.	ℓ_∞	ℓ_1	ℓ_2	$\text{Acc}^{\text{union}}_{\text{adv}}$	$\text{Acc}^{\text{avg}}_{\text{adv}}$	Time (h)
CIFAR-10	Adv_{avg}	87.1± 0.1	33.8± 0.4	49.0± 0.1	74.9± 0.2	33.0± 0.4	64.5± 0.1	16.9
	Adv_{max}	85.4± 0.1	39.9± 0.8	44.6± 0.0	73.2± 0.0	37.3± 0.2	64.1± 0.1	16.3
	ANT	**94.7± 0.0**	0.1± 0.0	0.1± 0.0	2.93± 1.1	0.1± 0.0	23.4± 1.2	**0.7**
	Adv_{rand}	87.2± 0.1	32.4± 0.1	50.3± 0.9	74.7± 0.8	32.1± 0.3	64.1± 0.1	5.2
	SAT	87.4± 0.0	34.6± 0.4	49.3± 1.0	**75.5± 0.0**	33.9± 0.3	64.8± 0.1	5.5
	MNG_{SAT}	84.9± 0.0	**38.7± 0.3**	**52.6± 0.3**	73.5± 0.6	**38.6± 0.3**	**65.2± 0.1**	11.2
SVHN	Adv_{avg}	92.6± 0.1	17.4± 0.2	54.2± 6.9	74.7± 3.5	16.6± 0.2	63.4± 0.1	24.1
	Adv_{max}	88.2± 1.6	5.9± 1.2	48.3± 4.1	31.0± 6.0	5.8± 1.3	55.7± 0.9	22.7
	ANT	**96.8± 0.0**	0.2± 0.0	7.5± 0.5	33.9± 4.2	0.2± 0.0	40.6± 0.4	**1.1**
	Adv_{rand}	94.6± 0.1	23.5± 3.6	32.0± 1.2	69.8± 0.9	20.7± 0.7	63.4± 0.1	8.0
	SAT	92.8± 0.2	23.4± 0.0	**41.3± 0.1**	71.0± 0.6	22.8± 1.5	64.0± 0.5	7.6
	MNG_{SAT}	93.4± 0.1	**34.2± 2.4**	36.6± 0.2	**77.7± 1.0**	**27.2± 0.3**	**67.2± 0.1**	11.9

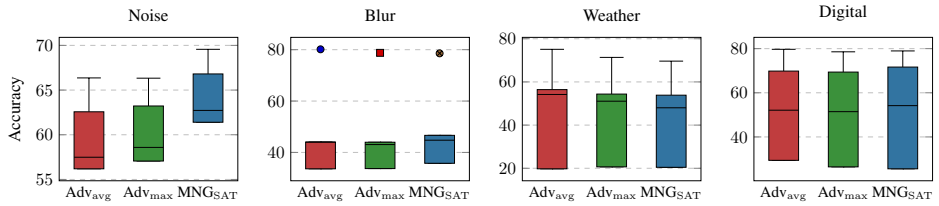

Figure 2: **Evaluation on CIFAR-10-C dataset**: Boxplots showing a comparison of average accuracy over 5 severities of common corruptions. Each box shows the quartiles summarizing the results across all types of skew while the error bars indicate the min and max across different skew types.

5.3 Ablation Studies

To investigate how our proposed framework facilitates robustness and computational efficiency, we dissect the effectiveness of various components.

Contribution of SAT. Our analyses comparing SAT and MNG_{SAT} are summarized in Table 2. First, observe that SAT leads to a $\sim 68\%$ and $\sim 30\%$ reduction in training time over the baselines trained with multiple perturbations and MNG_{SAT} for both the datasets. Interestingly, SAT is competitive to Adv_{avg} and Adv_{max} on both the datasets. To this end, MNG_{SAT} compliments SAT and improves the performance over all the methods across multiple attacks with significantly lower cost compared to the baselines trained with multiple perturbations.

Contribution of meta-learning framework. We further analyze the utility of our meta-learning framework in Table 2. We first compare with a naive baseline Adv_{rand} that utilizes SAT and injects random noise $z \sim \mathcal{N}(0, \mathbf{I})$ to the inputs. Interestingly, although Adv_{rand} utilizes SAT, injecting noise hurts the performance on both the datasets. We then compare MNG_{SAT} with Adversarial Noise Training (ANT) (Rusak et al., 2020) that learns a noise distribution to circumvent the classification network. We can observe in Table 2 that ANT leads to a significant drop in performance across all the datasets. Overall, we can infer that naive learning of noise is not able to defend against the state-of-the-art adversarial attacks highlighting the efficacy of our meta-learning training scheme to defend against multiple perturbations.

5.4 Further Analysis of Our Defense

Visualization of learned noise. To demonstrate the learning ability of our meta-noise generator, we visualize the learned noise by our generator during training. We present representative samples for various ℓ_p norms and datasets in Figure 3 where each sample is projected to their respected norm-ball $\mathcal{B}(x, \varepsilon)$ around x with radius ε. From the figure, we can observe that our meta-noise generator

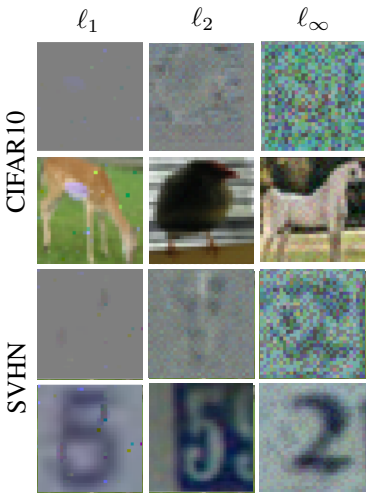

Figure 3: Visualization of the generated noise by MNG along with the perturbed samples on ℓ_1, ℓ_2, and ℓ_∞-norm attacks for CIFAR-10 and SVHN dataset.

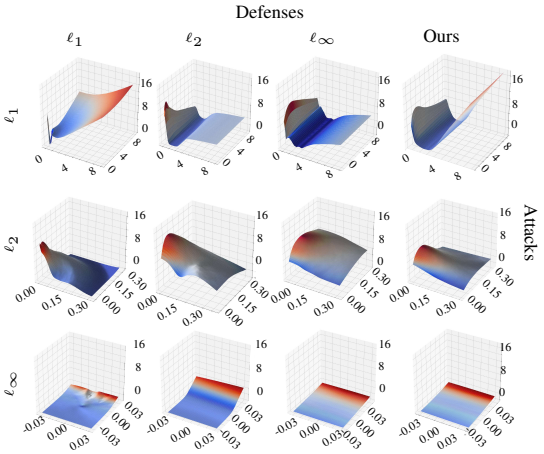

Figure 4: Visualization of the loss landscapes for the ℓ_1, ℓ_2, and ℓ_∞-norm perturbations on the CIFAR-10 dataset. The rows represent the perturbations and columns represent different defenses. We can observe that ours MNG_{SAT} obtains smooth loss surface across all ℓ_p-norm attacks.

incorporates the features by different attacks and learns local correlations. In particular, note that MNG learns diverse input-dependent noise distributions across multiple adversarial perturbations by explicitly minimizing the adversarial loss across multiple perturbations during meta-training. Overall, it combines two approaches that are complementary to each other and leads to a novel input-dependent learner for generalization across diverse attacks.

Visuaization of loss landscape We compare the loss surface of various methods against ℓ_∞, ℓ_1, and ℓ_2 norm attack in Figure 4. We vary the input along a linear space defined by the sign of gradient where x and y-axes represent the perturbation added in each direction, and the z-axis represents the loss. We can observe that in most of the instances when trained with a single adversary, the adversary can find a direction orthogonal to that explored during training; for example, ℓ_1 attack results in a non-smooth loss surface for both ℓ_∞ and ℓ_2 adversarial training. In the contrary, MNG_{SAT} achieves smoother loss surface across all types of attacks which suggests that the gradients modeled by our model are closer to the optimum global landscape.

6 Conclusion

We tackled the problem of robustness against multiple adversarial perturbations. Existing defense methods are tailored to defend against single adversarial perturbation which is an artificial setting to evaluate in real-life scenarios where the adversary will attack the system in any way possible. To this end, we proposed a novel *Meta-Noise Generator (MNG)* that learns to stochastically perturb adversarial examples by generating output noise across diverse perturbations. Then we train the model using a loss that accounts for both noise-augmented and adversarial samples. Additionally, to resolve the problem of computation overhead with conventional adversarial training methods for multiple perturbations, we introduce a *Stochastic Adversarial Training (SAT)* which samples a perturbation from the distribution of perturbations. The experimental validation of our methods against various benchmark datasets showed that it significantly outperforms the relevant baselines on a wide variety of perturbations. We believe that our method which allows efficient training as well as generalization across multiple perturbations will open the door for further research in this direction.

REFERENCES

Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *ICML*, 2016.

Anish Athalye and Nicholas Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018.

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *ICLR*, 2018.

Wieland Brendel, Jonas Rauber, Matthias Kümmerer, Ivan Ustyuzhaninov, and Matthias Bethge. Accurate, reliable and fast robustness evaluation. In *NeurIPS*, 2019.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 2017.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019.

Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *ICCV*, 2015.

Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: Elastic-net attacks to deep neural networks via adversarial examples. In *AAAI*, 2018.

Zhehui Chen, Haoming Jiang, Yuyang Shi, Bo Dai, and Tuo Zhao. Learning to defense by learning to attack, 2020.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

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

Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *ICML*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Learning what and where to transfer. In *ICML*, 2019a.

Yunseok Jang, Tianchen Zhao, Seunghoon Hong, and Honglak Lee. Adversarial defense via learning to generate diverse attacks. In *ICCV*, 2019b.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In *NeurIPS*, 2019.

Hae Beom Lee, Taewook Nam, Eunho Yang, and Sung Ju Hwang. Meta dropout: Learning to perturb latent features for generalization. In *ICLR*, 2020.

Divyam Madaan, Jinwoo Shin, and Sung Ju Hwang. Adversarial neural pruning with latent vulnerability suppression. In *ICML*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2017.

Chengzhi Mao, Ziyuan Zhong, Junfeng Yang, Carl Vondrick, and Baishakhi Ray. Metric learning for adversarial robustness. In *AAAI*, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *Workshop on Deep Learning and Unsupervised Feature Learning, NeurIPS*, 2011.

Hyeonwoo Noh, Tackgeun You, Jonghwan Mun, and Bohyung Han. Regularizing deep neural networks by noise: Its interpretation and optimization. In *NeurIPS*, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, ICML*, 2017.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, 2018.

Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI*, 2018.

Evgenia Rusak, Lukas Schott, Roland Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In *ECCV*, 2020.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *ICLR*, 2018.

Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on mnist. In *ICLR*, 2018.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019.

Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 2017.

Leslie N Smith. Cyclical learning rates for training neural networks. In *WACV*, 2017.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *ICLR*, 2018.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In *NeurIPS*, 1996.

Sebastian Thrun and Lorien Pratt. Learning to learn, 1998.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. In *NeurIPS*, 2019.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666*, 2018.

Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

Lei Wu, Chao Ma, and E Weinan. How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective. In *NeurIPS*, 2018.

Chang Xiao, Peilin Zhong, and Changxi Zheng. Enhancing adversarial defense by k-winners-take-all. In *ICLR*, 2020.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019.

Yi Zhou, Junjie Yang, Huishuai Zhang, Yingbin Liang, and Vahid Tarokh. SGD converges to global minimum in deep learning via star-convex path. In *ICLR*, 2019.

Algorithm 1 Learning to generate noise for robustness against multiple perturbations

Require: $\mathcal{D}$, T inner gradient steps, N training epochs.
input Dataset $\mathcal{D}$, training iterations T, batch size m, distribution of perturbations $p(\mathcal{A})$.
output Final model paramaters θ

for $n = \{1, \ldots, N\}$ **do**
  Sample mini-batch of size m, $B = \{x_1, \ldots, x_m\} \subset \mathcal{D}$.
  Generate the adversarial examples using Eq. (5).
  Sample $z \sim \mathcal{N}(0, \mathbf{I})$ and generate $x^{\text{aug}} = \underset{\mathcal{B}(x,\varepsilon)}{\text{proj}}\,(x + g_\phi(z, x))$ using MNG.
  Update θ to minimize Eq. (6).
  Initialize $\theta^{(0)} = \theta$
  for $t = \{1, \ldots, T\}$ **do**
    Update $\theta^{(t)}$ using Eq. (8).
  end for
  Descent a single step to update $\theta^{(T)}$ to $\theta^{(T+1)}$ by Eq. (9).
  Update the parameters ϕ of the generator by Eq. (10).
end for

A Experimental Setup

A.1 Datasets

1. **CIFAR-10.** This dataset (Krizhevsky, 2012) contains 60,000 images with 5,000 images for training and 1,000 images for test for each class. Each image is sized 32×32, we use the pre-activation version of the ResNet18 architecture (He et al., 2016) as a base network for this dataset.
2. **SVHN.** This dataset (Netzer et al., 2011) contains 73257 training and 26032 testing images of digits and numbers in natural scene images containing 10 digit classes. Each image is sized 32×32, we use the ResNet18 architecture similar to the CIFAR-10 dataset as the base network.

A.2 Training setup

We use the SGD optimizer with momentum 0.9 and weight decay $5 \cdot 10^{-4}$ to train all our models with cyclic learning rate with a maximum learning rate λ that increases linearly from 0 to λ over first $N/2$ epochs and then decreases linearly from $N/2$ to 0 in the remainder epochs, as recommended by Wong et al. (2020) for fast convergence of adversarial training. We train all the models for 30 epochs on a single machine with four GeForce RTX 2080Ti using WideResNet 28-10 architecture (Zagoruyko & Komodakis, 2016). We use the maximum learning rate $\lambda = 0.21$ for all our experiments. We use $\beta = 1.0$ and $\gamma = 16$ for all the experiments with our meta noise generator. The generator is formulated as a convolutional network with four $3{\times}3$ convolutional layers with LeakyReLU activations and one residual connection from input to output. We use the Foolbox (Rauber et al., 2017) library for all the attacks and all our algorithms are implemented in Pytorch (Paszke et al., 2019). We use the weight for the KL divergence ($\beta = 6.0$) for TRADES and RST in all our experiments.

A.3 Evaluation setup

For ℓ_∞ perturbations, we use PGD (Madry et al., 2017), Brendel and Bethge attack (Brendel et al., 2019), and AutoAttack (cro, 2020). For ℓ_2 perturbations, we use CarliniWagner attack (Carlini & Wagner, 2017), PGD (Madry et al., 2017), Brendel and Bethge attack (Brendel et al., 2019), and AutoAttack (cro, 2020). For ℓ_1 perturbations, we use SLIDE (Tramèr & Boneh, 2019), Salt and pepper (Rauber et al., 2017), and EAD attack (Chen et al., 2018). For all our experiments and evaluation, we use $\varepsilon = \{0.03, 8, 0.31\}$ and $\alpha = \{0.004, 1.0, 0.1\}$ for ℓ_∞, ℓ_1, and ℓ_2 attacks respectively. We use 10 steps of PGD attack for ℓ_∞, ℓ_2 during training and 50 steps during evaluation. For ℓ_1 adversarial training, we use 20 steps during training and 100 steps during evaluation.

Table 3: Summary of adversarial accuracy results for CIFAR-10

	Adv_∞	Adv_1	Adv_2	Trades_∞	RST_∞	Adv_{avg}	Adv_{max}	MNG_{SAT}
Clean Accuracy	86.8± 0.0	93.3± 0.4	91.7± 0.0	84.7± 0.1	88.9± 0.0	87.1± 0.1	85.4± 0.1	84.9± 0.0
PGD-ℓ_∞	46.9± 0.3	0.40± 0.5	23.6± 0.0	52.0± 0.3	56.9± 0.0	35.2± 0.6	42.2± 1.2	40.3± 0.9
PGD-Foolbox	54.7± 0.1	0.33± 0.3	35.3± 0.2	57.8± 0.2	62.9± 0.1	45.0± 0.2	50.4± 0.2	48.7± 0.2
AutoAttack	44.9± 0.5	0.0± 0.0	20.7± 0.1	48.8± 0.4	53.9± 0.0	33.8± 0.4	39.9± 0.8	38.7± 0.3
Brendel & Bethge	49.9± 1.2	0.0± 0.0	26.8± 0.1	52.1± 1.2	56.5± 0.1	39.6± 0.3	45.8± 0.1	43.9± 0.4
All ℓ_∞ attacks	44.9± 0.5	0.0± 0.0	20.7± 0.1	48.9± 0.5	54.9± 3.1	33.8± 0.4	39.9± 0.8	38.7± 0.3
PGD-ℓ_2	78.7± 0.1	47.6± 0.2	84.6± 0.1	77.0± 0.7	82.2± 0.2	81.5± 0.0	79.1± 0.1	79.0± 0.0
PGD-Foolbox	74.6± 0.0	5.1± 3.1	79.8± 0.0	73.3± 0.4	78.3± 0.0	77.6± 0.0	75.8± 0.1	76.0± 0.0
Gaussian Noise	85.2± 0.2	88.5± 0.3	90.5± 0.4	83.2± 0.0	87.8± 0.2	86.2± 0.1	83.3±	82.6± 0.3
AutoAttack	69.9± 0.1	0.0± 0.0	76.8± 0.2	69.4± 0.1	73.7± 0.0	74.9± 0.2	73.2± 0.0	73.5± 0.5
Brendel & Bethge	71.8± 0.6	0.0± 0.0	78.1± 0.3	70.2± 0.0	75.0± 0.0	75.9± 0.1	74.1± 0.1	74.2± 0.6
CWL2	70.5± 0.1	0.1± 0.0	77.2± 0.2	69.7± 0.1	74.2± 0.0	74.6± 0.8	73.5± 0.0	73.7± 0.5
All ℓ_2 attacks	69.9± 0.1	0.0± 0.0	76.8± 0.2	69.4± 0.1	73.6± 0.0	74.9± 0.2	73.2± 0.0	73.5± 0.6
PGD-ℓ_1	12.8± 0.4	91.6± 2.1	27.7± 0.4	17.9± 0.1	22.0± 0.3	49.0± 0.1	44.6± 0.0	52.6± 0.3
PGD-Foolbox	35.2± 0.5	92.3± 1.7	53.1± 0.2	40.3± 0.5	44.6± 0.1	64.5± 0.1	60.7± 0.2	64.8± 0.2
EAD	72.9±1.1	87.1± 10.8	75.9± 3.5	80.2± 0.5	84.5± 0.0	85.7± 0.0	83.3± 0.2	82.8± 0.3
SAPA	71.5± 0.0	80.2± 3.3	81.9± 0.2	71.4± 0.2	76.0± 0.5	82.7± 0.0	80.0± 0.1	79.7± 0.0
All ℓ_1 attacks	12.8± 0.4	75.3± 23.7	27.7± 0.5	17.9± 0.4	22.0± 0.3	49.0± 0.1	44.6± 0.0	52.6± 0.3
$\text{Acc}_{\text{adv}}^{\text{union}}$	12.9± 0.4	0.0± 0.0	18.3± 0.3	17.9± 0.5	22.1± 0.3	33.0± 0.1	37.8± 0.2	**38.2± 0.1**
$\text{Acc}_{\text{adv}}^{\text{avg}}$	58.1± 0.2	35.9± 0.2	58.9± 0.1	59.7± 0.0	64.3± 0.0	64.5± 0.1	64.2± 0.1	**65.2± 0.1**

Table 4: Summary of adversarial accuracy results for SVHN

	Adv_∞	Adv_1	Adv_2	Trades_∞	RST_∞	Adv_{avg}	Adv_{max}	MNG_{SAT}
Clean Accuracy	92.8± 0.1	92.4± 1.6	94.9± 0.0	93.9± 0.0	95.6± 0.0	92.6± 0.1	88.2± 1.6	93.4± 0.0
PGD-ℓ_∞	49.1± 0.0	3.2± 2.8	29.4± 0.0	55.5± 1.9	66.9± 0.6	22.4± 0.4	36.6± 2.0	39.0± 1.1
PGD-Foolbox	60.7± 0.1	2.5± 1.4	47.6± 0.4	66.4± 1.1	73.8± 0.1	32.5± 2.7	49.9± 0.0	52.6± 0.2
AutoAttack	46.2± 0.4	0.0± 0.0	18.9± 0.3	49.9± 3.1	61.0± 4.1	17.6± 0.4	17.5± 1.4	34.3± 2.9
Brendel & Bethge	51.6± 0.5	0.0± 0.0	22.9± 0.6	55.8± 2.2	65.6± 2.4	20.2± 1.2	6.3± 2.1	39.3± 2.6
All ℓ_∞ attacks	46.2± 0.4	0.0± 0.0	18.7± 0.3	49.9± 2.9	60.9± 3.8	17.4± 0.2	5.9± 1.2	34.2± 2.4
PGD-ℓ_2	81.6± 0.3	3.9± 2.4	87.8± 0.0	83.9± 0.7	85.3± 0.1	85.6± 0.3	84.3± 1.1	89.4± 1.3
PGD-Foolbox	73.2± 0.0	1.9± 1.2	82.8± 0.4	75.0± 0.4	76.0± 0.1	80.6± 0.1	60.1± 0.8	83.7± 0.1
Gaussian Noise	92.1± 0.1	16.5± 4.2	94.2± 0.0	93.3± 0.0	94.2± 0.1	92.2± 0.2	83.8± 1.8	92.8± 0.2
AutoAttack	59.0± 0.4	0.0± 0.0	79.3± 0.0	56.4± 1.6	60.7± 0.4	75.6± 1.1	40.0± 0.0	78.0± 0.9
Brendel & Bethge	68.2± 0.3	0.0± 0.0	81.0± 0.0	64.8± 0.8	68.1± 0.2	76.4± 3.5	32.7± 5.8	78.3± 1.1
CWL2	63.5± 0.6	0.1± 0.0	80.1± 0.0	61.4± 2.1	63.9± 0.1	76.8± 0.1	55.3± 5.2	79.5± 0.9
All ℓ_2 attacks	59.2± 0.4	0.0± 0.0	79.3± 0.0	56.0± 1.8	60.6± 0.4	74.7± 3.5	31.0± 5.0	77.7± 1.0
PGD-ℓ_1	3.1± 0.1	95.0± 3.3	30.5± 0.2	1.7± 0.1	0.7± 0.4	55.8± 2.1	48.4± 2.9	36.6± 0.2
PGD-Foolbox	19.9± 0.7	94.6± 0.2	57.5± 1.3	15.5± 0.0	11.3± 0.2	79.2± 0.4	85.4± 6.0	63.3± 1.8
EAD	65.7± 3.4	87.8± 3.6	82.3± 1.4	51.5± 6.5	60.4± 0.7	84.8± 1.7	84.5± 1.0	86.2± 0.2
SAPA	79.4± 0.7	77.3± 6.2	87.3± 0.0	73.5± 1.0	86.2± 0.3	88.5± 0.3	80.9± 7.2	90.2± 0.4
All ℓ_1 attacks	3.0± 0.1	77.9± 7.2	30.3± 0.1	1.6± 0.1	0.7± 0.4	54.2± 3.9	48.3± 4.1	36.6± 0.2
$\text{Acc}_{\text{adv}}^{\text{union}}$	3.0± 0.1	0.0± 0.0	16.4± 0.6	1.6± 0.1	0.7± 0.4	16.6± 0.2	5.8± 1.3	**27.2± 0.3**
$\text{Acc}_{\text{adv}}^{\text{avg}}$	58.1± 0.0	28.0± 3.1	63.0± 0.0	57.5± 0.2	62.4± 0.1	63.4± 0.1	55.7± 0.9	**67.2± 0.1**