Learning to Generate Noise for Multi-Attack Robustness

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

The majority of existing adversarial defense methods are tailored to defend against a single category of adversarial perturbation (e.g. ℓ_∞-attack). However, this makes these methods extraneous as the attacker can adopt diverse adversaries to deceive the system. Moreover, training on multiple perturbations simultaneously significantly increases the computational overhead during training. To address these challenges, we propose a novel meta-learning framework that explicitly learns to generate noise to improve the model's robustness against multiple types of attacks. Its key component is *Meta Noise Generator (MNG)* that outputs optimal noise to stochastically perturb a given sample, such that it helps lower the error on diverse adversarial perturbations. By utilizing samples generated by MNG, we train a model by enforcing the label consistency across multiple perturbations. We validate the robustness of models trained by our scheme on various datasets and against a wide variety of perturbations, demonstrating that it significantly outperforms the baselines across multiple perturbations with a marginal computational cost.

1 Introduction

Deep neural networks have demonstrated enormous success on multiple benchmark applications [1, 2], by achieving super-human performance on certain tasks. However, to deploy them to safety-critical applications [3, 4, 5], we need to ensure that the model is *robust* as well as *accurate*, since incorrect predictions may lead to severe consequences. Notably, 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 [6, 7, 8]. However, despite a large literature to improve upon the 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 type of perturbation. Consequently, many of the proposed defenses were circumvented by stronger attacks [9, 10, 11].

Meanwhile, several recent works [12, 13] have demonstrated the vulnerability of existing defense methods against multiple perturbations. For the desired multi-attack robustness, various recent strategies [13, 14] have aggregated multiple perturbations during training. However, training with multiple perturbations comes at an additional cost; it increases the training cost by a factor of four over adversarial training, which is already an order of magnitude 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 [15, 16], however, they are limited to ℓ_∞ adversarial training.

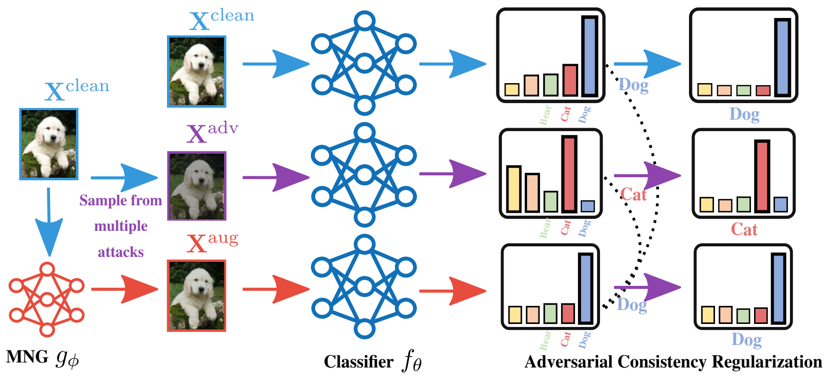

Figure 1: **Overview of Meta-Noise Generator with Adversarial Consistency (MNG-AC)**. First, we stochastically sample a perturbation to generate the adversarial examples X^{adv}. The generator g_ϕ takes stochastic noise and input X^{clean} to generate the noise-augmented sample X^{aug}. The classifier f_θ then minimizes the stochastic adversarial classification loss and the adversarial consistency loss. MNG is learned via meta-learning to explicitly minimize the adversarial classification loss.

To address the drawbacks of existing methods, we propose a novel training scheme, *Meta Noise Generator with Adversarial Consistency (MNG-AC)*, which learns instance-dependent noise to minimize the adversarial loss across multiple perturbations while enforcing label consistency between them, as illustrated in Figure 1 and explained in details below.

First, we tackle the heavy computational overhead incurred by multi-perturbation training by proposing *Stochastic Adversarial Training (SAT)*, that samples from a distribution of perturbations during training, which significantly accelerates training for multiple perturbations. Then, based on the assumption that the model should output the same predictions for different perturbations of the same image, we introduce *Adversarial Consistency (AC)* loss that enforces label consistency across multiple perturbations. Finally, motivated by the noise regularization techniques [17, 18, 19, 20] which target generalization, we formulate a *Meta Noise Generator (MNG)* that learns to stochastically perturb a given sample in a meta-learning framework to explicitly improve the generalization and label consistency across multiple attacks.

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

- We introduce *Adversarial Consistency (AC)* loss that enforces label consistency across multiple perturbations to enforce smooth and robust networks.
- We formulate *Meta-Noise Generator* that explicitly meta-learns an input-dependent noise generator, such that it outputs stochastic noise distribution to improve the model's robustness and adversarial consistency across multiple types of adversarial perturbations.
- We validate our proposed method on various datasets against diverse benchmark adversarial attacks, on which it achieves state-of-the-art performance, highlighting its practical impact.

2 Robustness against multiple perturbations

We first briefly review single/multi-perturbation adversarial training and introduce *Stochastic Adversarial Training (SAT)* to reduce the computational cost incurred by training with multiple perturbations. We consider a dataset $\mathcal{D}$ over observations $x \in \mathbb{R}^d$ and labels $y \in \mathbb{R}^C$ with C classes. Let $f_\theta : \mathbb{R}^d \to \mathbb{R}^C$ be a L-layered classifier with parameters θ and classification loss $\mathcal{L}_{\text{cls}}$. Given an attack procedure $\mathcal{A}(x)$ which introduces a perturbation δ, we let $x^{\text{adv}} = x + \delta$ denote the corresponding adversarial examples. More formally, for a single perturbation with norm-ball $\mathcal{B}(x, \varepsilon)$, we approximate the maximum loss by an attack procedure $\mathcal{A}(x)$, such that $\max_{\delta \in \mathcal{B}(x,\varepsilon)} \mathcal{L}_{\text{cls}}\left(f_\theta\left(x+\delta\right), y\right) \approx \mathcal{L}_{\text{cls}}\left(f_\theta\left(\mathcal{A}\left(x\right)\right), y\right)$.

Single-perturbation adversarial training. In the standard single-perturbation adversarial training [21, 6], the model optimizes the network using a min-max formulation. More formally, the inner maximization generates the adversarial perturbation by maximizing the loss, while the outer minimization minimizes the loss on the generated examples.

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}} \mathcal{L}_{\text{cls}}\left(f_\theta\left(\mathcal{A}\left(x\right)\right), y\right). \tag{1}$$

The majority of existing single-perturbation defenses are primarily able to defend against a single category of adversarial perturbation. However, this limits the generalization of these methods to perturbations that are unseen during training [12, 13], which has been referred to as *overfitting* on the particular type of training perturbation.

Multi-perturbation adversarial training. Tramer et al. [13] extended the adversarial training to multiple perturbations by optimizing the outer objective in Eq. (1) on the strongest/union of adversarial perturbations for each input example. Their proposed strategies can more formally be defined as follows:

1. **The maximum over all perturbations**: It optimizes the outer objective in Eq. (1) on the strongest adversarial perturbation from the whole set of additive adversarial perturbations

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\arg\max_k \mathcal{L}_{\text{cls}}\left(f_\theta\left(\mathcal{A}_k\left(x\right)\right),y\right)\right]. \tag{2}$$

2. **The average over all perturbations:** It optimizes the outer objective in Eq. (1) on the whole set of n additive perturbations.

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\frac{1}{n}\sum_{k=1}^{k=n}\mathcal{L}_{\text{cls}}\left(f_\theta\left(\mathcal{A}_k\left(x\right),y\right)\right). \tag{3}$$

Recently, Maini et al. [14] proposed "Multi Steepest Descent" (MSD) by incorporating the different perturbations into the direction of steepest descent. However, the practicality of all these methods is limited due to an increased computational overhead for training.

Stochastic Adversarial Training (SAT). To overcome this limitation, we propose Stochastic Adversarial Training to defend 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 $p(\mathcal{A})$ 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:

$$\mathbb{E}_{\substack{(x,y)\sim\mathcal{D}\\ \mathcal{A}_k\sim p(\mathcal{A})}}\mathcal{L}_{\text{cls}}\left(f\left(\mathcal{A}_k(x),y\right)\right). \tag{4}$$

Our proposed objective is a drastic simplification of the average one in Eq. (3), which makes it highly efficient for multiple perturbations. It promotes generalization and convergence (due to its stochasticity) by preventing over-fitting on a single type of perturbation.

3 Learning to generate noise for multi-attack robustness

In this section, we introduce our framework MNG-AC, which leverages an *adversarial consistency loss (AC)* and a *meta-noise generator (MNG)* to help the model generalize to multiple perturbations. Let $g_\phi : \mathbb{R}^d \to \mathbb{R}^d$ denote the generator with parameters ϕ and x^{adv} be the adversarial examples generated by SAT. We sample $z \sim \mathcal{N}(0, \mathbf{I})$ for input to our generator jointly with the clean examples x to generate the noise-augmented sample x^{aug}. The total loss function $\mathcal{L}_{\text{total}}$ for the classifier consists exclusively of two terms: SAT classification loss and an adversarial consistency loss:

$$\mathcal{L}_{\text{total}} = \frac{1}{B}\sum_{i=1}^{B}\underbrace{\mathcal{L}_{\text{cls}}(\theta \mid x_i^{\text{adv}}, y_i)}_{\text{SAT classification loss}} + \underbrace{\beta\cdot\mathcal{L}_{\text{ac}}\left(p_i^{\text{clean}}; p_i^{\text{adv}}; p_i^{\text{aug}}\right)}_{\text{adversarial consistency loss}}. \tag{5}$$

where B is the batch-size, β is the hyper-parameter determining the strength of the adversarial consistency (AC) loss denoted by $\mathcal{L}_{\text{ac}}$ and $p^{\text{clean}}, p^{\text{adv}}, p^{\text{aug}}$ represent the posterior distributions $p(y \mid x^{\text{clean}}), p(y \mid x^{\text{adv}}), p(y \mid x^{\text{aug}})$ respectively. Specifically, $\mathcal{L}_{\text{ac}}$ represents the Jensen-Shannon Divergence (JSD) among the posterior distributions. Consequently, $\mathcal{L}_{\text{ac}}$ enforces stability and insensitivity across a diverse range of inputs based on the assumption that the classifier should output similar predictions when fed perturbed versions of the same image.

To generate the augmented samples for our purpose, we explicitly perturb the input examples for generalization across multiple perturbations. In particular, MNG meta-learns [22, 23] the parameters ϕ of a noise generator g_ϕ to generate an input-dependent noise distribution to alleviate the issue of generalization across multiple adversaries. The standard approach to train our adversarial classifier jointly with MNG is to use bi-level optimization [23]. However, bi-level optimization for adversarial training would be computationally expensive.

To tackle this challenge, we adopt an online approximation [24, 25] to update θ and ϕ using a single-optimization loop. We alternatively update the parameters θ of the classifier with the parameters ϕ of MNG. 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 which ensures the learning of the classifier using the knowledge from the generated samples constructed by MNG. It explicitly increases the influence of the noise-augmented samples on the classifier in the inner loop. More specifically, for a learning rate α, projection operator proj, $\theta^{(t)}$ moves along the following descent direction on a mini-batch of training data:
$$\begin{aligned} \theta^{(t+1)} &= \theta^{(t)} - \alpha \cdot \frac{1}{B}\sum_{i=1}^{B} \nabla_\theta \mathcal{L}_{\text{cls}}(\theta^{(t)} \mid x_i^{\text{aug}}, y_i, \phi), \\ \text{where,} \quad x^{\text{aug}} &= \underset{\mathcal{B}(x,\varepsilon)}{\text{proj}} \left(x + g_\phi(z, x)\right). \end{aligned} \tag{6}$$

2. **Adapt model parameters on a single step.** Second, perform one-step update to update $\theta^{(T)}$ to $\theta^{(T+1)}$ to minimize SAT loss from Eq. (4). This step explicitly models the adaptation of adversarial model parameters in the presence of the noise-augmented data using a single step of update:
$$\theta^{(T+1)} = \theta^{(T)} - \alpha \cdot \frac{1}{B}\sum_{i=1}^{B} \nabla_\theta \mathcal{L}_{\text{cls}}(\theta^{(T)} \mid x_i^{\text{adv}}, y_i). \tag{7}$$

3. **Update generator parameters.** In the last step, after receiving feedback from the classifier, we measure the SAT loss from Eq. (4) and adapt ϕ to minimize this loss. In particular, ϕ performs the following update step to facilitate the classifier parameters θ in the next step:
$$\phi = \phi - \alpha \cdot \frac{1}{B}\sum_{i=1}^{B} \nabla_\phi \mathcal{L}_{\text{cls}}(\theta^{(T+1)} \mid x_i^{\text{adv}}, y_i). \tag{8}$$

Overall, MNG-AC consists of perturbation sampling to generate adversarial examples. Then, it 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 adversarial consistency loss.

4 Experiments

4.1 Experimental setup

We compare our method MNG-AC with the standard network (Nat) and state-of-the-art single-perturbation baselines including Madry et al. [6] (Adv_{p}) for ℓ_∞, ℓ_1, and ℓ_2 norm, [7] (TRADES_∞), and [8] (RST_∞) for ℓ_∞ norm. We also consider state-of-the-art multi-perturbation baselines: namely, we consider Adversarial training with the maximum (see Eq. (2)) (Adv_{max}), average (Adv_{avg}) [13] (see Eq. (3)) over all perturbations, and Multiple steepest descent (MSD) [14]. We evaluate our method on multiple benchmark datasets including CIFAR-10 [28], SVHN [29] on Wide ResNet 28-10 [26] and Tiny-ImageNet [1] on ResNet-50 [27] architecture.

We have evaluated the proposed defense scheme and baselines against perturbations generated by state-of-the-art attack methods. We use the same attack parameters as Tramer et al. [13] for training and evaluation. We validate the clean accuracy ($\text{Acc}_{\text{clean}}$), the worst ($\text{Acc}_{\text{adv}}^{\text{union}}$) and average ($\text{Acc}_{\text{adv}}^{\text{avg}}$) adversarial accuracy across all the perturbation sets for all the models. For ℓ_∞ attacks, we use PGD [6], Brendel and Bethge [30], and AutoAttack [31]. For ℓ_2 attacks, we use CarliniWagner [32], PGD [6], Brendel and Bethge [30], and AutoAttack [31]. For ℓ_1 attacks, we use SLIDE [13], Salt and pepper [33], and EAD attack [34]. We provide a detailed description of the experimental setup in the Appendix. The code is available at `https://github.com/anonycodes/NeurIPS_MNG`.

[1]`https://tiny-imagenet.herokuapp.com/`

Table 1: Comparison of robustness against multiple types of perturbations. All the values are measured by computing mean, and standard deviation across three trials upon randomly chosen seeds, the best and second-best results are highlighted in bold and underline respectively. Time denotes the training time in hours. We report the worst-case accuracy for all the attacks and defer the breakdown of all attacks to the Appendix.

	Model	$\text{Acc}_{\text{clean}}$	ℓ_∞	ℓ_1	ℓ_2	$\text{Acc}_{\text{adv}}^{\text{union}}$	$\text{Acc}_{\text{adv}}^{\text{avg}}$	Time (h)
CIFAR-10	Nat [26]	94.7± 0.1	0.0± 0.0	4.4± 0.8	19.4± 1.4	0.0± 0.0	7.9± 0.3	**0.4**
	Adv_∞ [6]	86.8± 0.1	44.9± 0.7	12.8± 0.6	69.3± 0.4	12.9± 0.5	42.6± 0.4	4.5
	Adv_1	**93.3± 0.4**	0.0± 0.0	**78.1± 1.8**	0.0± 0.0	0.0± 0.00	25.1± 1.6	8.1
	Adv_2	91.7± 0.2	20.7± 0.3	27.7± 0.7	**76.8± 0.4**	17.9± 0.8	47.6± 0.4	3.7
	TRADES_∞ [7]	84.7± 0.3	48.9± 0.7	17.9± 0.6	69.4± 0.3	17.2± 0.6	45.4± 0.3	5.2
	RST_∞ [8]	88.9± 0.2	**54.9± 1.8**	22.0± 0.5	73.6± 0.1	21.1± 1.0	50.2± 0.5	58.8
	Adv_{avg} [13]	87.1± 0.2	33.8± 0.7	49.0± 0.3	74.9± 0.4	31.0± 1.4	52.6± 0.5	16.9
	Adv_{max} [13]	85.4± 0.3	39.9± 0.9	44.6± 0.2	73.2± 0.2	35.7± 0.3	52.5± 0.3	16.3
	MSD [14]	82.6± 0.0	43.7± 0.2	41.6± 0.2	70.6± 1.1	35.8± 0.1	52.0± 0.4	16.7
	MNG-AC (Ours)	81.5± 0.3	42.2± 0.9	55.0± 1.2	71.5± 0.1	**41.6± 0.8**	**56.2± 0.2**	11.2
SVHN	Nat [26]	**96.8± 0.1**	0.0± 0.0	4.4± 0.8	19.4± 1.4	0.0± 0.0	7.9± 0.3	**0.6**
	Adv_∞ [6]	92.8± 0.2	46.2± 0.6	3.0± 0.3	59.2± 0.7	3.0± 0.3	36.2± 0.3	6.2
	Adv_1	92.4± 0.9	0.0± 0.0	**77.9± 6.3**	0.0± 0.0	0.0± 0.0	23.9± 2.1	11.8
	Adv_2	94.9± 0.1	18.7± 0.6	30.3± 0.3	**79.3± 0.1**	16.4± 0.7	42.8± 0.2	6.1
	TRADES_∞ [7]	93.9± 0.1	49.9± 1.7	1.6± 0.3	56.0± 1.4	1.6± 0.3	35.8± 0.6	7.9
	RST_∞ [8]	95.6± 0.0	**60.9± 2.0**	0.7± 0.6	60.6± 0.6	0.7± 0.6	40.7± 0.8	112.5
	Adv_{avg} [13]	92.6± 0.3	17.4± 2.3	54.2± 2.9	74.7± 0.1	16.6± 1.3	43.0± 1.0	24.1
	Adv_{max} [13]	88.2± 1.3	5.9± 1.2	48.3± 4.1	31.0± 5.0	5.8± 1.7	26.7± 2.5	22.7
	MNG-AC (Ours)	93.7± 0.1	35.1± 1.9	47.4± 2.2	77.6 ± 1.0	**30.3± 1.8**	**52.6± 0.5**	11.9
Tiny-ImageNet	Nat [27]	**62.8± 0.4**	0.0± 0.0	2.7± 0.3	12.6± 0.8	0.0± 0.0	5.1± 0.4	**0.9**
	Adv_∞ [6]	54.2± 0.4	29.6± 0.1	31.8± 1.0	42.5± 0.6	19.8± 1.1	33.8± 0.1	4.3
	Adv_1	57.8± 0.2	10.5± 0.7	39.3± 1.0	41.9± 0.0	10.1± 0.7	30.4± 0.1	12.9
	Adv_2	59.5± 0.1	5.2± 0.6	37.2± 0.4	**44.9± 0.1**	5.2± 0.6	29.1± 0.0	3.7
	TRADES_∞ [7]	48.2± 0.2	28.7± 0.9	30.9± 0.2	35.8± 0.7	26.1± 0.9	32.8± 0.1	5.8
	Adv_{avg} [13]	56.0± 0.0	23.7± 0.2	38.0± 0.2	44.6± 1.8	23.6± 0.3	35.4± 0.7	26.8
	Adv_{max} [13]	53.5± 0.0	**29.8± 0.1**	33.4± 0.3	42.4± 1.0	**29.0± 0.3**	35.3± 0.4	20.8
	MNG-AC (Ours)	53.1± 0.3	27.4± 0.7	**39.6± 0.7**	44.8± 0.1	27.4± 0.8	**37.2± 0.6**	10.4

4.2 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-AC achieves a relative improvement of $\sim 6\%$ and $\sim 4\%$ on the $\text{Acc}_{\text{adv}}^{\text{union}}$ and $\text{Acc}_{\text{adv}}^{\text{avg}}$ metric over the state-of-the-art methods trained on multiple perturbations. Moreover, MNG-AC achieves $\sim 33\%$ reduction in training time compared to the multi-perturbations training baselines. It is also worth mentioning that, MNG-AC also shows an improvement over Adv_{max}, which is fundamentally designed to address the worst perturbation.

Results with SVHN dataset. The results for the SVHN dataset are shown in Table 1. We make the following observations from the results: (1) Firstly, MNG-AC significantly outperforms Adv_{avg}, Adv_{max} by $\sim 14\%$ and $\sim 25\%$ on $\text{Acc}_{\text{adv}}^{\text{union}}$ metric. Furthermore, it achieves an improvement of $\sim 7.2\%$ and $\sim 26\%$ on $\text{Acc}_{\text{adv}}^{\text{avg}}$ metric over Adv_{avg}, Adv_{max} respectively. (2) Secondly, MNG-AC leads to a $\sim 50\%$ reduction in training time compared to the multi-perturbation training baselines. Interestingly, MNG-AC achieves significant better performance over ℓ_1 adversarial training with comparable training time which illustrates the utility of our method over standard adversarial training.

Results with Tiny-ImageNet. We also evaluate our method on Tiny-ImageNet to verify that it performs well on complex datasets. In Table 1 we observe that MNG-AC outperforms the multi-perturbation training baselines and achieves comparable performance to the single-perturbation baselines. Only against ℓ_∞ perturbations, we notice that Adv_{max} achieves better performance. We

Table 2: Ablation study analyzing the significance of SAT, Adversarial Consistency loss (AC) and Meta Noise Generator (MNG). The best results are highlighted in bold.

	SAT	AC	MNG	$\text{Acc}_{\text{clean}}$	ℓ_∞	ℓ_1	ℓ_2	$\text{Acc}_{\text{adv}}^{\text{union}}$	$\text{Acc}_{\text{adv}}^{\text{avg}}$	Time (h)
CIFAR-10	✓	-	-	**87.4± 0.0**	34.6± 0.7	49.3± 1.0	**75.5± 0.1**	33.9± 0.6	53.1± 0.1	**5.5**
	✓	✓	-	81.4± 0.0	40.4± 0.1	53.2± 0.9	70.2± 0.1	40.1± 0.2	54.6± 0.4	6.8
	✓	✓	✓	81.5± 0.3	**42.2± 0.9**	**55.0± 1.2**	71.5± 0.1	**41.6± 0.8**	**56.2± 0.2**	11.2
SVHN	✓	-	-	92.8± 0.5	23.4± 2.4	41.3± 4.3	71.0± 3.6	22.8± 1.5	44.9± 1.2	**7.6**
	✓	✓	-	92.1± 0.2	32.9± 1.8	35.4± 1.5	77.1± 1.3	28.3± 0.1	49.6± 0.5	9.6
	✓	✓	✓	**93.7± 0.1**	**35.1± 1.9**	**47.4± 2.2**	**77.6 ± 1.0**	**30.3± 1.8**	**52.6± 0.5**	11.9

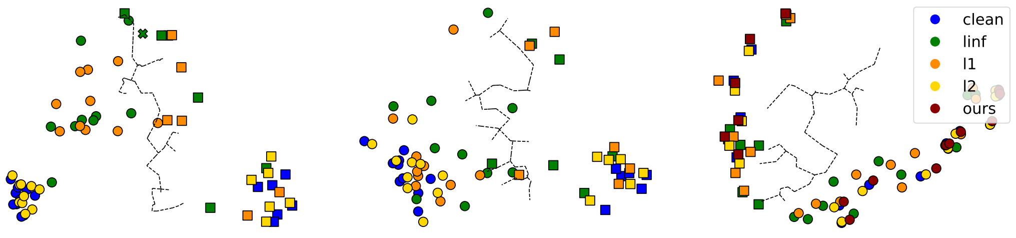

Figure 2: Visualization of decision boundary in the penultimate latent-feature space for Adv_{avg} in the left, Adv_{max} in the middle, MNG-AC in the right for SVHN dataset on Wide ResNet 28-10 architecture. The two shapes represent different classes in a binary classification task.

believe this is an artefact of the inherent trade-off across multiple perturbations [13, 12]. Interestingly, MNG-AC even achieves comparable performance to the single perturbation baselines trained on ℓ_1 and ℓ_2 norm. This demonstrates the effectiveness of MNG in preventing overfitting over a single attack, and it's generalization ability to diverse types of attacks.

4.3 Further analysis of our defense

Component analysis. To further investigate our training scheme, we dissect the effectiveness of various components in Table 2. First, we examine that SAT leads to a $\sim 68\%$ and $\sim 30\%$ reduction in training time over multiple perturbations baselines and MNG-AC for both the datasets, however, it does not improve the adversarial robustness. Then, we analyze the impact of our meta-noise generator by injecting random noise $z \sim \mathcal{N}(0, \mathbf{I})$ to the inputs for the generation of augmented samples. We observe that it significantly improves the performance over the SAT with a marginal increase in the training time. Furthermore, leveraging MNG our combined framework MNG-AC achieves consistent improvements, outperforming all the baselines, demonstrating the efficacy of our meta-learning scheme to defend against multiple perturbations.

Visualization of decision boundary. Finally, we visualize the learned decision boundary on binary-classification task across multiple attacks in Figure 2. We can observe that MNG-AC obtains the least error against all the attacks compared to the baselines trained on multiple perturbations. Furthermore, the consistency regularization embeds multiple perturbations onto the same latent space, which pushes them away from the decision boundary that in turn improves the overall robustness.

5 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 propose 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 *Adversarial Consistency loss* that accounts for label consistency across clean, adversarial, and augmented 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. We believe that our method can be a strong guideline when other researchers pursue similar tasks in the future.

References

[1] 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.

[2] 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.

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

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

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

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

[7] 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.

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

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

[10] 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.

[11] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

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

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

[14] Pratyush Maini, Eric Wong, and J Zico Kolter. Adversarial robustness against the union of multiple perturbation models. In *ICML*, 2020.

[15] 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.

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

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

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

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

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

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

[22] Sebastian Thrun and Lorien Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, 1998.

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

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

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

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

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

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

[29] 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.

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

[31] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

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

[33] 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.

[34] 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.