EVALUATING WORST CASE ADVERSARIAL WEATHER PERTURBATIONS ROBUSTNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Several algorithms are proposed to improve the robustness of deep neural networks against adversarial perturbations beyond ℓ_p cases, i.e. weather perturbations. However, evaluations of existing robust training algorithms are over-optimistic. This is in part due to the lack of a standardized evaluation protocol across various robust training algorithms, leading to ad-hoc methods that test robustness on either random perturbations or the adversarial samples from generative models that are used for robust training, which is either uninformative of the worst case, or is heavily biased. In this paper, we identify such evaluation bias in these existing works and propose the first standardized and fair evaluation that compares various robust training algorithms by using physics simulators for common adverse weather effects i.e. rain and snow. Additionally, our framework identified the lack of diversity in existing robust training algorithms. As a step to address this, we propose a light-weight generative adversarial network (GAN) with improved diverse weather effects controlled by latent codes that can be used in robust training. The proposed robust training algorithm is evaluated on two streetview classification datasets (BIC_GSV, Places365), where it outperforms other robust training approaches based on generative models for worst-case adversarial rain and snow attacks.

1 INTRODUCTION

Adversarial robustness of machine learning models has become an important topic in recent years. For safety-critical applications such as autonomous driving and healthcare systems, it is important to ensure the robustness of models before deploying them into the real world. Despite an overwhelming amount of studies on adversarial attacks and defenses from the past few years, most of them focus on the simplified ℓ_p norm threat model where robustness is defined as the worst-case perturbation within a small ℓ_p ball (Madry et al., 2018; Goodfellow et al., 2015). This simplified assumption ensures the perturbation is imperceptible and facilitates the development of defense algorithms, since under the ℓ_p norm perturbation model, both attacks and defenses can be easily designed and evaluated. However, in practice there are many other semantic-preserved or natural perturbations that are not ℓ_p norm bounded, such as adversarial shadows (Zhong et al., 2022), adversarial rains (Zhai et al., 2020), and physical adversarial T-shirts (Xu et al., 2020).

Since collecting real-world adversarial examples is infeasible, a natural solution to improve the natural adversarial robustness is through learning a perturbation set that introduces weather effects such as rain or snow into the original clean images, and then conduct adversarial training with the ℓ_p ball constraint replaced by the general perturbation set (Wong & Kolter, 2020; Robey et al., 2020). However, the lack of natural adversarial examples also makes evaluating the robustness of such a perturbation set with real images impossible. To bypass this difficulty, Wong & Kolter (2020) reuses the pretrained generative model to produce adversarial examples, which is also used in adversarial training. However, a generator trained on random perturbation set is not guaranteed to match the perturbation distribution when it is used to generate worst-case adversarial examples. Therefore, training and evaluating the models with the same generative model can lead to overly optimistic and unreliable evaluation of robustness. In addition, Robey et al. (2020) deploys out-of-distribution (OOD) perturbations in evaluation, which limits the exploration in the perturbation space. The ad-hoc nature of these evaluations makes it difficult to compare existing robust training algorithms.

This stems from a lack of a standardized and fair evaluation protocol for natural robust training algorithms, where current evaluation methods potentially lead to unreliable robustness metrics.

To address the above issue, we propose an evaluation framework for robust training algorithms based on physics-based weather simulators, where the robust training algorithm usually learns the perturbation set from datasets and train robust models under this perturbation set. The framework is designed in a differentiable manner to enable the generation of worst-case adversarial examples during evaluation. The adversarial examples generated by physical simulators can give a fair evaluation by using independent random simulators both in training and evaluation. We show that several existing models are robust to the perturbations from their own generative models as expected, but are not robust to our adversarial attacks based on physical simulators.

We further analyze the reason why these existing robust training algorithms perform worse under our evaluation. With extensive experiments, we identify general drawbacks of existing robust training algorithms, such as their lack of diversity. As an improvement, we propose a light-weight model based on generative adversarial networks to produce diverse weather perturbations controlled by latent codes. Our model supports both unpaired and paired datasets, which makes it more versatile for weather perturbation sets.

Our main contributions are summarized as follows:

- We propose the first standard protocol for evaluating robust training algorithms for machine learning models against weather perturbations. We do so by leveraging physics-based simulators to model adverse weather effects – yielding fair comparisons between algorithms for learning perturbation sets beyond ℓ_p norm and robust training.
- We demonstrate that existing robust training methods are over-optimistic in their claims due to their ad-hoc evaluations, which are either uninformative of worst case perturbations or favorable towards the particular training algorithm.
- We show that existing methods that learn from unpaired datasets failed to generate diverse perturbation sets; whereas those that rely on paired data are limited by the availability of paired datasets containing adverse weather conditions. To address this pervasive drawback, we improve the robust training algorithm for unpaired datasets with a light-weight GAN model that can generate more diverse weather perturbations.
- We leverage our GAN in adversarial training to yield robust classifiers. We demonstrate the effectiveness of our approach on two datasets, BIC_GSV (Kang et al., 2018) and Places365 (Zhou et al., 2017). Models trained with our approach achieve the best overall robustness for adversarial rain and snow perturbations compared to existing works.

2 Related Work

Adversarial robustness of beyond ℓ_p robustness: Neural Networks are shown to be vulnerable to perturbations that are imperceptible to humans. Several papers (Biggio et al., 2013; Carlini & Wagner, 2017; Goodfellow et al., 2015) have shown that neural networks can be attacked by small perturbations bounded by ℓ_1, ℓ_2, ℓ_p balls. For instance, Szegedy et al. (2014) shows that such perturbations can alter the output of a classification network. Goodfellow et al. (2015) introduces the fast gradient sign method (FGSM). Dong et al. (2018); Kurakin et al. (2016); Madry et al. (2018) extend FGSM to iterative optimization to boost its performance. Moosavi-Dezfooli et al. (2016) finds the minimal perturbation to alter the predicted class while Madry et al. (2018) proposed projected gradient descent (PGD) to find the worst-case perturbations.

Beyond ℓ_p robustness, some recent papers extend the perturbation sets to settings that can preserve semantic meaning. Some can be well-defined mathematically such as Wasserstein robustness (Wong et al., 2019), distributional shifts (Sinha et al., 2017; Sagawa et al., 2019) and word substitution (Jia et al., 2019) in texts. Others can not be well-defined but perturbed datasets can be generated or collected, like adversarial shadows (Zhong et al., 2022), adversarial rains (Zhai et al., 2020) and some physical adversarial perturbations (Duan et al., 2020; 2021; Li et al., 2019a; Xu et al., 2020).

Robust training algorithms for natural robustness: For general natural robustness settings, attack methods like PGD (Madry et al., 2018) and FGSM (Goodfellow et al., 2015) cannot be directly

applied whithin the perturbation set. Some works use generative adversarial networks (GANs) to learn the perturbation set and then generate adversarial examples during adversarial training (Xiao et al., 2018; Wong & Kolter, 2020; Robey et al., 2020). Another line of work uses random perturbations to improve the out-of-distribution robustness (Hendrycks et al., 2021; 2019; Calian et al., 2021). However, these solutions cannot bypass the lack of natural adversarial examples in evaluation. Wong & Kolter (2020) reuses the learned generators to generate adversarial examples, where the learned generators is not guaranteed to match the perturbation set under the worst-case perturbations, and evaluating and training the models with the same generative model can leads to fake robustness in evaluation. Robey et al. (2020) uses out-of-distribution (OOD) perturbations, which limits the exploration in perturbation space.

Weather simulation: The appearance of falling particles in rain and snow are highly complicated and can be affected by multiple factors, such as the particle properties, camera configurations, and environmental illumination (Garg & Nayar, 2007; Barnum et al., 2010). Copious amounts of research has been conducted to simulate the weather dynamics based on different principles, including raindrop oscillation models (Garg & Nayar, 2006; Li et al., 2016), frequency domain analysis (Barnum et al., 2010; Weber et al., 2015), and depth dependent formulations (Hu et al., 2019; Li et al., 2019b; Halder et al., 2019; Von Bernuth et al., 2019; Tremblay et al., 2021). More recently, several data-driven deep learning techniques have also been developed for weather effect simulation (Shen et al., 2019; Pizzati et al., 2020a;b; Wang et al., 2021; Wei et al., 2021). Other than these, the community also resorts to some existing image editing software (e.g. PhotoShop) for weather simulation (Liu et al., 2018; Zhang & Patel, 2018; Zhang et al., 2019; Chen et al., 2020; 2021).

3 Background and Definitions

In this section, we introduce the problem formulation and definitions used throughout the paper.

We consider the robust training problem where we are given a clean dataset and a perturbed dataset, which is normally the case when we want to train a model robust against natural perturbations where the perturbation set is difficult to be defined mathematically. For example, we can collect a dataset of sunny images $(\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_n)$ and a dataset of rainy i.e. perturbed images $(\mathbf{x}'_0, \mathbf{x}'_1, ..., \mathbf{x}'_m)$. Here the clean dataset is sampled from the clean distribution $\mathbf{x} \in \mathbb{R}^d \sim p(\mathbf{x})$. Each perturbed example $\mathbf{x}'$ in the perturbed dataset is corresponding to a clean image $\mathbf{x}$ and $\mathbf{x}' = g(\mathbf{x}, \mathbf{z})$ where $\mathbf{z}$ is the feature vector for this perturbation and g is the perturbation function. In this work, we focus on rain and snow perturbations where we can write the perturbation function as $\mathbf{x}' = g(\mathbf{x}, \mathbf{z}) = \mathbf{x} + \delta = \mathbf{x} + g'(\mathbf{x}, \mathbf{z})$. Here g' is the function that generates the perturbation mask δ from the perturbation feature $\mathbf{z}$. Here $\mathbf{z}$ can be limited within a pre-defined perturbation set $\mathbf{z} \in \Delta$ which limits the perturbation δ from being too strong to alter the semantic meaning of the clean image $\mathbf{x}$.

A classifier trained with a (robust) training algorithm $f_\theta(\mathbf{x}) : \mathbb{R}^d \to C$ parameterized by θ is considered as safe for an example $\mathbf{x}$ if we cannot find an adversarial example with an adversarial attack algorithm. In adversarial attack, we find the worst-case examples:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{z} \in \Delta} l(f_\theta(\mathbf{x}'), y). \tag{1}$$

Here $l(\cdot, y)$ is the standard cross entropy loss where y is the ground-truth label.

If $f_\theta(\hat{\mathbf{x}}) = y$, f_θ is safe for $\mathbf{x}$ under adversarial attack. In this work, we evaluate the robustness of a given model f_θ by evaluating the percentage of safe examples in the test set.

4 Evaluation Framework for Robust Training Algorithms

In this section, we describe our evaluation framework for robust training algorithms. Given a robust training algorithm, we run the algorithm with the same datasets generated by our framework and then evaluate the result robust model with the same environment. We illustrate the overview of this framework in Figure 2.

Differentiable physics-based weather simulator: The key module of this framework is the differentiable simulation engine which enables us to apply adversarial attacks directly to evaluate the

(a) Clean (b) Random (c) Adversarial (d) Difference

Figure 1: Examples for random simulation and simulation attack for rainy effects. (a) is the original clean image sampled from BIC_GSV Kang et al. (2018) dataset.; (b) is produced by random rain simulation; (c) is an adversarial example for a ResNet34 classifier pretrained with clean training; (d) is the difference between (b) and (c).

robustness of downstream models without the need for ad-hoc generative models used by previous works (Wong & Kolter, 2020; Robey et al., 2020).

The simulator generates weather perturbations in a two-stage process: (1) particles rendering, and (2) particles aggregation.

Falling particles like rain drops and snowflakes are rendered according to physical parameters such as particle size, scene illumination, and camera settings. For rain drops, we follow Garg & Nayar (2006). The appearance of each rain drop is controlled by several parameters including camera angle, rain drop size, light angle, and view angle. For snowflakes, we generate each snowflake from a Gaussian noise image and control the size, shape, and direction with threshold, standard deviation, and a motion blur, respectively. By altering these parameters, we are able to generate complex snow conditions, such as the snow streaks mentioned in Chen et al. (2021). These diverse rain drops and snowflakes are pre-generated with random parameters and stored in a database.

At the second stage, we randomly sample N particles from the particle database and aggregate them to form a weather mask. To make it differentiable, each particle image denoted by $\mathbf{s}_i$ is parameterized by a translation matrix $\mathbf{T}_i$ and a rotation matrix $\mathbf{R}_i$. We can generate a weather effect mask with N particles via:

$$\mathbf{o} = \sum_{i=0}^{N-1} \mathbf{s}_i \mathbf{R}_i \mathbf{T}_i. \tag{2}$$

Then the mask can be merged with the clean image to produce a perturbed image $\mathbf{x}' = \mathbf{o} + \mathbf{x}(1 - \mathbf{o})$ where $\mathbf{x}$ is the clean image. parameters in the translation matrix $\mathbf{T}_i$ and the rotation matrix $\mathbf{R}_i$ are differentiable in adversarial attack. This differentiable aggregation enables us to generate adversarial examples directly in the simulator:

$$\mathbf{R}_i^*, \mathbf{T}_i^* = \arg\max_{\mathbf{R}_i, \mathbf{T}_i} l(f(\mathbf{x}'), y),$$

where $\mathbf{R}_i$ and $\mathbf{T}_i$ are constrained in some given perturbation set. The constrained maximization problem can be solved with PGD attack (Madry et al., 2018). We call the adversarial attack with our differentiable simulator as simulation attack.

In Figure 1, we show examples generated by our simulator and simulation attack. We can see that the our adversarial examples are of the similar quality as the random ones.

Evaluation framework: We illustrate the overview of our evaluation framework in Figure 2. To evaluate robust training algorithms, we have a clean dataset which is the same for all training algorithms. Then we use the simulator described earlier to generate a random perturbed dataset with random transformation and rotation matrix. The clean and perturbed datasets are then fed into the training algorithm, where potentially the training algorithm includes a module to learn perturbation set and then conduct adversarial training within this learned perturbation set. After obtaining the trained robust model, we use the differentiable simulator to evaluate the adversarial natural robustness of this model with adversarial attack on the simulation parameters $\mathbf{R}_i$ and $\mathbf{T}_i$.

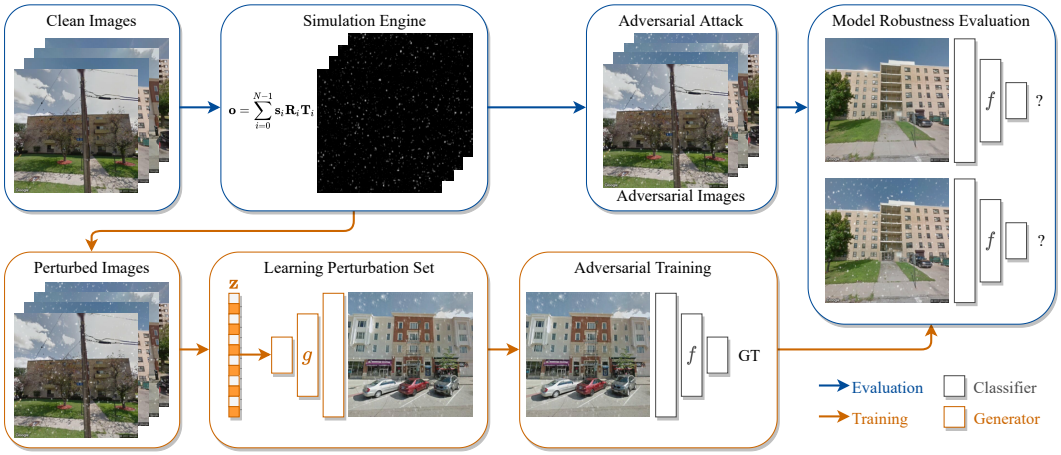

Figure 2: An overview of our proposed evaluation framework.

5 Evaluation Bias in Existing Evaluations

In this section, we evaluate some existing robust training algorithms with our proposed evaluation framework and compare the results with their own evaluations. We will show that evaluations in existing work have bias which prefers the models trained with its own generative models.

5.1 Settings

Datasets: Although there are some existing datasets for rain and snow effects like Snow100k (Liu et al., 2018), Snow Removal in Realistic Scenarios (Chen et al., 2020), and Rain100 (Zou et al., 2020), they are designed for deraining or desnowing tasks and do not have classification labels. In our experiments, we choose two streetview classification datasets which are suitable for weather effects. The BIC_GSV (Kang et al., 2018) dataset contains streetview figures extracted from Google StreetView labeled as 8 classes.

Another dataset is the commonly used Places365 dataset (Zhou et al., 2017) for scene recognition which includes 365 scene categories. We picked a subset with 7 outdoor scenes from Places365 as the dataset used in our experiments. We also picked another 4 classes for our unpaired dataset. A list of categories chosen in the subset can be found in the Appendix. When the Places dataset is used for training the unpaired generator models, we generate the perturbed dataset from the additional 4 classes. In all of our experiments, the input images are resized to 256×256 pixels. The number of particles in simulation is set to be 2,000.

Robust training algorithm baselines: We include three robust training algorithms in our evaluation experiments. VRGNet (Wang et al., 2021) proposes a Bayesian weather generation model to generate rain/snow masks with an input latent code which can be used in adversarial training by attacking the latent code. Robey et al. (2020) implements MUNIT (Huang et al., 2018) as the generator model which maps clean images to naturally perturbed images with a latent code which can also be used in adversarial training. For experiments related to MUNIT, we implemented a modified version of MUNIT for weather perturbations. The architecture of MUNIT used in our experiments can be found in the Appendix. Wong & Kolter (2020) learns the perturbation set with a CVAE model conditioned on a latent code. Both VRGNet and CVAE require paired datasets and MUNIT supports unpaired datasets.

Training settings: The number of dimensions for the latent code in VRGNet, MUNIT is set at 128. MUNIT is trained with a learning rate of 2e-4 for 5 epochs. For VRGNet, we ran the code provided by the authors for 100 epochs on our datasets with default hyper-parameters. For the CVAE model, we ran the code provided by the authors with the same configuration as the Multi-illumination dataset. For VRGNet and CVAE, we scaled the encoder and decoder to fit the image size in our datasets. Details of the architectures are listed in the Appendix. For all adversarial

Table 1: Ranks and robust error (R.E.) of different training methods under different evaluations for BIC_GSV (Kang et al., 2018) with rainy effects. Method with the lowest robust error is ranked as 1. "Random" is evaluated with randomly generated perturbed images, VRGNet (Wang et al., 2021), CVAE (Wong & Kolter, 2020) and MUNIT (Robey et al., 2020) are evaluated with adversarial attack based on different generators and "Attack" is evaluated with simulation attack.

Training methods	Random		VRGNet		CVAE		MUNIT		Attack	
	Rank	R.E.	Rank	R.E.	Rank	R.E.	Rank	R.E.	Rank	R.E.
Clean Training	4	0.6424	5	0.7634	5	0.4893	5	0.8027	5	0.6890
Augmented Training	3	0.4815	3	0.5656	2	0.4592	3	0.5928	3	0.5136
VRGNet AT (Wang et al., 2021)	1	0.4587	1	0.4767	3	0.4742	1	0.5063	1	0.4665
CVAE AT (Wong & Kolter, 2020)	5	0.6113	4	0.7498	3	0.4742	4	0.7430	4	0.6477
MUNIT AT (Robey et al., 2020)	2	0.4728	2	0.5228	1	0.4397	1	0.5063	2	0.4893

training, we set batch size to 8 and learning rate to 5e-5. After obtaining these trained generators, we use them to train a ResNet34 classifier with adversarial training. More details on the classifier training are introduced in Section 7.

5.2 Evaluation Results

We evaluate and compare the robust error of five models trained with different methods under the five evaluations for BIC_GSV (Kang et al., 2018) under rain perturbations. Clean training is trained with only the clean dataset. Augmented Training is augmented with randomly perturbed images from the simulator. For the five different evaluation methods, we use 10-step PGD attack in all methods that requires adversarial attack. In Table 1, we list the ranks of different training methods under different evaluations where the model with the lowest robust error is ranked as 1. We also report the exact numbers in evaluation.

As illustrated in Table 1, if we train and test a model using the same generator, the model can overfit the perturbation set defined by the generator during training. For example, when evaluating the model with adversarial attacks using VRGNet or MUNIT, the model trained with the corresponding AT has the best adversarial robust accuracy. This result validates our motivation to propose a standard evaluation protocol for robust training algorithms.

6 Generate diverse weathers with improved GAN

6.1 Lack of Diversity in existing methods

In this section, we illustrate why these existing methods, especially CVAE and MUNIT fail to be robust under our evaluation framework. We start the analysis by showing the adversarial examples generated by generative models used in these robust training algorithms. We show the clean images and corresponding generated adversarial images by different methods in Figure 3. For each method, we show 3 groups of generated adversarial examples. For each group, we randomly sample 3 latent codes and generate natural adversarial examples using its generative model and 10-step PGD attack starting from this latent code. The generative models are trained with the same setting as in Section 4 and we attack a clean trained ResNet34 model on BIC_GSV to generate these adversarial examples.

As shown in Figure 3, CVAE fails to capture the rain perturbation set with high diversity and simply generates adversarial examples with imperceptible perturbations. MUNIT is able to learn the perturbation set from given datasets, however, since the generator is conditioned on the original image to generate a perturbed image, the generator can make use of some lines on the clean image to generate rain streaks. The diversity is therefore limited and the distribution of strong streaks are similar for these generated examples.

6.2 Generate more diverse perturbations with improved GAN

Inspired by the observations in Section 6.1, we introduce a light-weight model based on Generative Adversarial Networks (GANs), which is an improved version over MUNIT. With GANs, we can model the perturbation set from unpaired clean and perturbed datasets which are more feasible in

Figure 3: Illustration of adversarial examples generated by PGD attack with CVAE and MUNIT. Column 1: Clean images. Column 2-4: Adversarial examples generated by PGD attack with CVAE starting from three different random latent codes. Column 5-7: Adversarial examples generated by PGD attack with MUNIT starting from three different random latent codes.

real-world scenarios. In short, we want to learn a model $g(\mathbf{x}, \mathbf{z})$ which can map a clean image $\mathbf{x}$ to a perturbed image $\hat{\mathbf{x}} = g(\mathbf{x}, \mathbf{z})$ where the diversity of perturbations can be captured by the latent code $\mathbf{z}$.

In MUNIT, the GAN model transforms the original clean image into a perturbed image given the latent code. The GAN model is conditioned on the clean image as well as the latent code to generate the perturbed image, which can weaken the connection between latent code and perturbations. For example, on Row 2 in Figure 3, the MUNIT generator makes use of window borders on buildings to generate rain streaks. To address this issue, we only use the GAN to model the rain/snow mask inspired by the simulation process for weather effects. We then combine the mask with the original clean image. Specifically, the model $g(\mathbf{x}, \mathbf{z})$ is decoupled into $g(\mathbf{x}, \mathbf{z}) = \mathbf{x} + g'(\mathbf{z})(1 - \mathbf{x})$ where $g'(\mathbf{z})$ is the generator for the rain/snow mask.

We show an overview of the generator architecture in Figure 4 which is highlighted by orange arrows. The model includes a decoder $g'(\mathbf{z})$ which maps a latent code $\mathbf{z}$ to a rain/snow mask. To stabilize the training, we also incorporate an inverse encoder $h(\mathbf{x})$ which encodes the perturbed image into a latent code and recovers the clean input image.

Loss construction: To encourage the generator to generate realistic rain/snow masks, we use a discriminator D_1 to distinguish generated perturbed images and perturbed images from the dataset. For the inverse encoder $h(\hat{\mathbf{x}})$, we include another discriminator D_2 to discriminate the real clean image and the reconstructed clean image. This constructs the standard GAN loss for forward generation where

$$L_{\text{GAN},1} = \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}),\mathbf{x}\sim p(\mathbf{x})}[\log(1 - D_1(g(\mathbf{x}, \mathbf{z})))] + \mathbb{E}_{\hat{\mathbf{x}}\sim p(\hat{\mathbf{x}})}[\log D_1(\hat{\mathbf{x}})].$$

The standard GAN loss for discriminator D_2 can be defined similarly.

In addition to the standard GAN loss, we also incorporate reconstruction loss $L^x_{\text{recon}}, L^z_{\text{recon}}$ for the input image and the latent code to encourage the generator to preserve the semantic meaning of input images and also generate diverse weather effects controlled by the latent code. A small identity loss L_{identity} is also added for stable training.

$$\begin{aligned} L^x_{\text{recon}} &= \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}),\mathbf{x}\sim p(\mathbf{x})}[\|h^x(g(\mathbf{x}, \mathbf{z})) - \mathbf{x}\|_1], \\ L^z_{\text{recon}} &= \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}),\mathbf{x}\sim p(\mathbf{x})}[\|h^z(g(\mathbf{x}, \mathbf{z})) - \mathbf{z}\|_1], \\ L_{\text{identity}} &= \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\|h^x(\mathbf{x}) - \mathbf{x}\|_1]. \end{aligned}$$

Total loss: Combining all the losses with weighted sum, we can get the total loss as

$$L_{\text{GAN}} = L_{\text{GAN},1} + L_{\text{GAN},2} + L^x_{\text{recon}} + L^z_{\text{recon}} + L_{\text{identity}}.$$

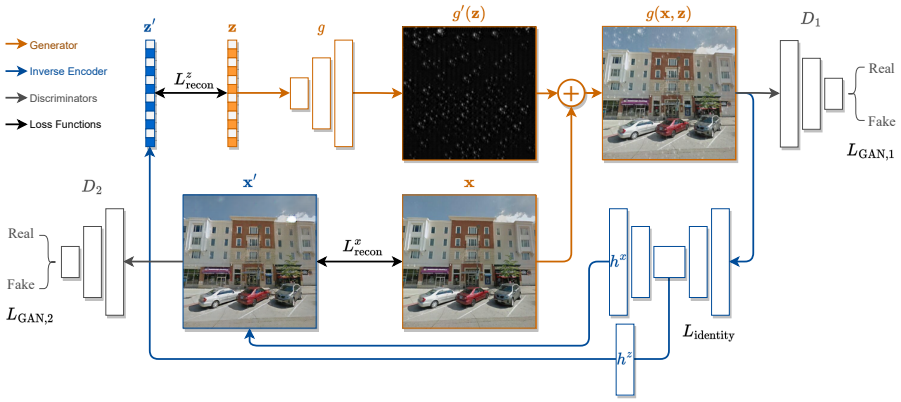

Figure 4: An overview of the GAN model introduced in Section 6.2. Architecture of the generator is highlighted by orange arrows.

Adversarial training: Given unpaired clean and perturbed datasets, we can train a generator $g(\mathbf{x}, \mathbf{z})$ as described before. We can then use this generator in adversarial training to improve the adversarial robustness of downstream classifiers under natural weather perturbations.

The optimization equation 1 can be converted to finding the latent code $\mathbf{z}$ that can produce the worst-case perturbed image. By applying adversarial attack on the latent code $\mathbf{z}$, we can use any existing adversarial training algorithm to boost the adversarial robustness of downstream models. We use $\hat{\mathbf{x}}_i$ to denote the adversarial example found for $\mathbf{x}_i$ given a downstream model $f(\mathbf{x})$, then we can construct the adversarial loss for adversarial training as

$$\begin{aligned} &\mathbf{z}_i \sim N(0, \mathbf{I}), \\ &\mathbf{z}'_i = \underset{\|\mathbf{z}'_i\| = \|\mathbf{z}_i\|}{\arg\max}\, l(f(g(\mathbf{x}_i, \mathbf{z}'_i)), y), , \mathbf{x}'_i = g(\mathbf{x}, \mathbf{z}'), \\ &L_{adv} = \lambda_{\text{clean}} \sum_{(\mathbf{x}_i, y_i) \in O} l(f(\mathbf{x}_i), y_i) + (1 - \lambda_{\text{clean}}) \sum_{\mathbf{x}'_i : (\mathbf{x}_i, y) \in O} l(f(\mathbf{x}'_i), y_i), \end{aligned} \tag{3}$$

where $\lambda_{\text{clean}} \in [0, 1]$ is the weight factor for clean training.

7 Robust Training with our Improved GAN

To boost the adversarial robustness under weather perturbations, we apply adversarial training (AT) by augmenting with adversarial examples generated by our proposed GAN model. We compare the performance of AT with our proposed GAN model with AT augmented by baseline generators. Besides generator-based AT, we also trained classifiers with clean training, augmented training with random perturbed images, and AT with PGD attack within the ℓ_∞ perturbation with perturbation radius $\epsilon = 1/255$. We use 5-step PGD in all training methods that requires adversarial training.

Datasets: For the clean datasets, we reuse the datasets BIC_GSV and Places365 used in Section 5. All the robust training algorithms are trained with the same clean dataset. The randomly perturbed dataset required by MUNIT, VRGNet, CVAE and our GAN model is generated by the simulator described in Section 4. For CVAE and VRGNet, the datasets are paired while for other methods the datasets are unpaired.

Classifier training setup: We use ResNet34 as our classifier architecture, which is one of the models used in Kang et al. (2018). We also follow the hyperparamter settings in Kang et al. (2018) where the learning rate is 5e-5, and the weight decay is 1e-5. We also add standard data augmentations, including random crops and horizontal flips in the training. For all the adversarial training and augmented training, we set the ratio of clean training to be $\lambda_{\text{clean}} = 0.1$. We trained the classifier for 10 epochs and we choose the epoch with the best adversarial robust error among the 10 epochs for each configuration and evaluate the clean error and perturbed error of this epoch. For each dataset, we also list the best clean accuracy from clean training among the 10 epochs.

Table 2: The clean error, adversarial robust error (Adv Err.) and error under random perturbations (Perturbed Err.) of models under rain and snow perturbations. Adv Err. is evaluated with simulation attack, and perturbed error is evaluated with random simulation.

Dataset	Training methods	Rain			Snow		
		Clean Err.	Adv Err.	Perturbed Err.	Clean Err.	Adv Err.	Perturbed Err.
BIC_GSV(Kang et al., 2018) Best clean err.: 0.4339	clean training	0.4344	0.6890	0.6467	0.4524	0.5330	0.5180
	Augmented Training	**0.4257**	0.5136	0.4825	0.4310	0.4801	0.4650
	ℓ_∞ AT ($\epsilon = 1/255$)	0.4402	0.6492	0.5986	0.4699	0.5389	0.5243
	VRGNet AT *(Wang et al., 2021)	0.4441	*0.4665*	*0.4587*	0.4286	*0.4495*	*0.4407*
	CVAE AT *(Wong & Kolter, 2020)	0.5005	0.6477	0.6161	0.4966	0.5831	0.5734
	MUNIT AT (Robey et al., 2020)	0.4388	0.4893	0.4772	*0.4286*	0.4563	0.4456
	Ours AT	*0.4281*	**0.4631**	**0.4553**	**0.4212**	**0.4470**	**0.4383**
Places(Zhou et al., 2017) Best clean err.: 0.2357	clean training	0.2443	0.3729	0.3514	0.2443	0.2957	0.2829
	Augmented Training	0.2542	0.2857	0.2700	0.2514	0.2700	0.2643
	ℓ_∞ AT ($\epsilon = 1/255$)	0.2786	0.3843	0.3686	0.2571	0.3086	0.3014
	VRGNet AT *(Wang et al., 2021)	0.2543	*0.2671*	**0.2543**	*0.2414*	*0.2500*	*0.2471*
	CVAE AT *(Wong & Kolter, 2020)	0.3029	0.4243	0.4143	0.2686	0.3171	0.3071
	MUNIT AT (Robey et al., 2020)	**0.2300**	0.2786	0.2729	0.2529	0.2671	0.2643
	Ours AT	*0.2343*	**0.2614**	*0.2557*	**0.2257**	**0.2443**	**0.2386**

* VRGNet and CVAE models are trained on paired dataset.

Table 3: Average variance of the generated perturbed images of different methods. We compare the average variance of our GAN model with two generative models for unpaired datasets.

	MUNIT	CVAE	Our GAN model
Avg. variance	14.94	0.03	97.65

Better natural adversarial robustness under simulation attack: In Table 2 we list the clean error and the adversarial robust error evaluated by simulation attacks. Note that in (Kang et al., 2018) they classify a building by averaging the predictions of images taken from different view angles, but the view angles are not provided in the open-sourced dataset. Therefore, they have a higher clean accuracy than our results. As shown in the table, adversarial training with our GAN model achieves the best adversarial robustness under simulation attack with unpaired dataset. Compared with CVAE and VRGNet which only support paired dataset, adversarial training with our model can achieve comparable robust accuracy with better clean accuracy. Besides, the adversarial augmentation can also help the model to generalize better in the test set with a higher clean accuracy.

More diverse perturbations: To show that our improved GAN model is able to generate more diverse perturbations with the same training datasets, we compare the diversity with average variance of the generated images. For each method, we sample 10 clean images from the BIC_GSV dataset and generate 10 perturbed images with random latent codes. Then we compute the diversity score as the average variance of all pixel among these 10 generated images. We report the average diversity score among 10 samples of each method in Table 3. We can see from the table that our GAN model can generate perturbations with the highest diversity.

8 DISCUSSION

In the paper, we propose the first standard evaluation protocol for robust training algorithms under perturbation sets beyond ℓ_p. This addresses a gap in the community as evaluations were previously performed using ad-hoc generative models that were also used by the robust training algorithms. Conscious of the limited diversity in weather perturbations generated by existing works, we show that our light-weight GAN can not only produce diverse perturbations, but also yields classifiers that achieve the best adversarial robustness among existing works. Although the evaluation results can provide some insights for other perturbation sets beyond ℓ_p, our work currently only focuses on weather perturbations i.e. rain and snow. We expect to extend to more perturbation sets in the future.

REFERENCES

Peter C Barnum, Srinivasa Narasimhan, and Takeo Kanade. Analysis of rain and snow in frequency space. *International journal of computer vision*, 86(2):256–274, 2010.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Dan A Calian, Florian Stimberg, Olivia Wiles, Sylvestre-Alvise Rebuffi, Andras Gyorgy, Timothy Mann, and Sven Gowal. Defending against image corruptions through adversarial augmentations. *arXiv preprint arXiv:2104.01086*, 2021.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57, 2017.

Wei-Ting Chen, Hao-Yu Fang, Jian-Jiun Ding, Cheng-Che Tsai, and Sy-Yen Kuo. Jstasr: Joint size and transparency-aware snow removal algorithm based on modified partial convolution and veiling effect removal. In *European Conference on Computer Vision*, pp. 754–770. Springer, 2020.

Wei-Ting Chen, Hao-Yu Fang, Cheng-Lin Hsieh, Cheng-Che Tsai, I Chen, Jian-Jiun Ding, Sy-Yen Kuo, et al. All snow removed: Single image desnowing algorithm using hierarchical dual-tree complex wavelet representation and contradict channel loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4196–4205, 2021.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9185–9193, 2018.

Ranjie Duan, Xingjun Ma, Yisen Wang, James Bailey, A Kai Qin, and Yun Yang. Adversarial camouflage: Hiding physical-world attacks with natural styles. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1000–1008, 2020.

Ranjie Duan, Xiaofeng Mao, A Kai Qin, Yuefeng Chen, Shaokai Ye, Yuan He, and Yun Yang. Adversarial laser beam: Effective physical-world attack to dnns in a blink. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16062–16071, 2021.

Kshitiz Garg and Shree K Nayar. Photorealistic rendering of rain streaks. *ACM Transactions on Graphics (TOG)*, 25(3):996–1002, 2006.

Kshitiz Garg and Shree K. Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1): 3–27, 2007.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL `http://arxiv.org/abs/1412.6572`.

Shirsendu Sukanta Halder, Jean-François Lalonde, and Raoul de Charette. Physics-based rendering for improving robustness to rain. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10203–10212, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90. URL `https://doi.org/10.1109/CVPR.2016.90`.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021.

Xiaowei Hu, Chi-Wing Fu, Lei Zhu, and Pheng-Ann Heng. Depth-attentional features for single-image rain removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8022–8031, 2019.

Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 172–189, 2018.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4129–4142, Hong Kong, China, 2019. doi: 10.18653/v1/D19-1423. URL https://aclanthology.org/D19-1423.

Jian Kang, Marco Körner, Yuanyuan Wang, Hannes Taubenböck, and Xiao Xiang Zhu. Building instance classification using street view images. *ISPRS journal of photogrammetry and remote sensing*, 145:44–59, 2018.

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

Juncheng Li, Frank Schmidt, and Zico Kolter. Adversarial camera stickers: A physical camera-based attack on deep learning systems. In *International Conference on Machine Learning*, pp. 3896–3904. PMLR, 2019a.

Ruoteng Li, Loong-Fah Cheong, and Robby T. Tan. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1633–1642, 2019b.

Yu Li, Robby T. Tan, Xiaojie Guo, Jiangbo Lu, and Michael S. Brown. Rain streak removal using layer priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2736–2744, 2016.

Yun-Fu Liu, Da-Wei Jaw, Shih-Chia Huang, and Jenq-Neng Hwang. Desnownet: Context-aware deep network for snow removal. *IEEE Transactions on Image Processing*, 27(6):3064–3073, 2018.

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. URL https://openreview.net/forum?id=rJzIBfZAb.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2574–2582, 2016. doi: 10.1109/CVPR.2016.282. URL https://doi.org/10.1109/CVPR.2016.282.

Fabio Pizzati, Pietro Cerri, and Raoul de Charette. Model-based occlusion disentanglement for image-to-image translation. In *European conference on computer vision*, pp. 447–463. Springer, 2020a.

Fabio Pizzati, Raoul de Charette, Michela Zaccaria, and Pietro Cerri. Domain bridge for unpaired image-to-image translation and unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2990–2998, 2020b.

Alexander Robey, Hamed Hassani, and George J Pappas. Model-based robust deep learning: Generalizing to natural, out-of-distribution data. *arXiv preprint arXiv:2005.10247*, 2020.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Zhiqiang Shen, Mingyang Huang, Jianping Shi, Xiangyang Xue, and Thomas S Huang. Towards instance-level image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3683–3692, 2019.

Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL `http://arxiv.org/abs/1312.6199`.

Maxime Tremblay, Shirsendu Sukanta Halder, Raoul De Charette, and Jean-François Lalonde. Rain rendering for evaluating and improving robustness to bad weather. *International Journal of Computer Vision*, 129(2):341–360, 2021.

Alexander Von Bernuth, Georg Volk, and Oliver Bringmann. Simulating photo-realistic snow and fog on existing images for enhanced cnn training and evaluation. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 41–46. IEEE, 2019.

Hong Wang, Zongsheng Yue, Qi Xie, Qian Zhao, Yefeng Zheng, and Deyu Meng. From rain generation to rain removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14791–14801, 2021.

Yoann Weber, Vincent Jolivet, Guillaume Gilet, and Djamchid Ghazanfarpour. A multiscale model for rain rendering in real-time. *Computers & Graphics*, 50:61–70, 2015.

Yanyan Wei, Zhao Zhang, Yang Wang, Mingliang Xu, Yi Yang, Shuicheng Yan, and Meng Wang. Deraincyclegan: Rain attentive cyclegan for single image deraining and rainmaking. *IEEE Transactions on Image Processing*, 30:4788–4801, 2021.

Eric Wong and J Zico Kolter. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020.

Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International Conference on Machine Learning*, pp. 6808–6817. PMLR, 2019.

Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

Kaidi Xu, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. Adversarial t-shirt! evading person detectors in a physical world. In *European conference on computer vision*, pp. 665–681. Springer, 2020.

Liming Zhai, Felix Juefei-Xu, Qing Guo, Xiaofei Xie, Lei Ma, Wei Feng, Shengchao Qin, and Yang Liu. It's raining cats or dogs? adversarial rain attack on dnn perception. *arXiv preprint arXiv:2009.09205*, 2020.

He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 695–704, 2018.

He Zhang, Vishwanath Sindagi, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE transactions on circuits and systems for video technology*, 30(11): 3943–3956, 2019.

Yiqi Zhong, Xianming Liu, Deming Zhai, Junjun Jiang, and Xiangyang Ji. Shadows can be dangerous: Stealthy and effective physical-world adversarial attack by natural phenomenon. *arXiv preprint arXiv:2203.03818*, 2022.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

Zhengxia Zou, Sen Lei, Tianyang Shi, Zhenwei Shi, and Jieping Ye. Deep adversarial decomposition: A unified framework for separating superimposed images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12806–12816, 2020.

A APPENDIX

A.1 EXPERIMENT DETAILS

Computational resources All the experiments are conducted on a single NVIDIA GeForce RTX 2080 Ti GPU. One epoch of adversarial training with our GAN model takes around 40 minutes on BIC_GSV dataset and 80 minutes on the Places dataset. The time cost of simulation attack with PGD-10 on Places dataset is around 10 minutes per epoch and on BIC_GSV dataset is around 25 minutes per epoch.

Datasets For Places dataset, we picked 7 categories from the 365 categories in Places365 Zhou et al. (2017) dataset as the training and test datasets for classifier training, including: apartment_building-outdoor, church-outdoor, garage-outdoor, general_store-outdoor, house, library-outdoor and office_building. For generator training which supports unpaired dataset, we picked 4 additional categories to generate the perturbed dataset, which includes art_gallery, entrance_hall, golf_course and yard.

Classifier architectures We use ResNet34 He et al. (2016) as our classifier architecture. In our classifier training, we initialize the Resnet34 classifier with pretrained weights on Imagenet provided by cnn_finetune[1] and the final linear layer is randomly initialized.

Generator architectures We implement MUNIT and our model based on pytorch-CycleGAN-and-pix2pix.[2] We denote a convolutional layer with padding 1, k 4×4 filters and stride s as Ck-s. Then the discriminators can be represented as C64-2 C128-2 C256-2 C512-2 C512-1 C1-1.

For the generators, we let x-Rk denote x ResNet blocks, each of which contains two 3×3 convolutional blocks with k filters, TCk denote a transpose convolutional block with k filters of size 3, stride 2, padding 1, output padding 1, SCk denote a convolutional block with k filters of size 3, stride 2, padding 1 and Ck denote a convolutional block with k filters of size 7 and padding 0. Then the mask generator can be represented as 9-R256 TC256 TC256 C3. The generator from perturbed image to clean image can be represented as C64 SC128 SC256 9-R256 TC256 TC256 C3. In our MUNIT implementation, we insert the latent code by concatenating into the hidden layer of the generator from clean images to perturbed images. The generator from clean image to perturbed image can be represented as C64 SC128 SC256 9-R257 TC257 TC257 C3 where the latent code which is transformed by a full-connected layer is concatenated with the output of SC256. For our model and MUNIT, we also have a simple convolutional neural network to reconstruct the latent code which takes the output of 9-R256 as the input. The architecture can be represented C16 C32 C64 L128 L128 where Ck is a convolutional layer with k 4×4 filters, stride 2 and padding 1 and Lk is a fully-connected layer with output dimension k.

For VRGNet and CVAE, we run the code provided by authors and we scale the architectures defined in the VRGNet code to fit the image size in our dataset where the input is randomly cropped to 224×224 in training. The weather mask generator architecture in VRGNet is L12544 TC128-4-2-1 TC64-4-2-1 TC32-4-2-1 TC3-8-4-2 and the encoder architecure is C32-8-4-2 C64-4-2-1 C128-4-2-1 C256-4-2-1 L256 where Lk is a fully-connected layer with output dimension k and TCk-f-s-p and Ck-f-s-p are a transposed convolutional and convolutional layers with k filters of size f, stride s and padding p.

License of the assets We include the licenses of codebases in the submitted code of our supplementary material. For Places and BIC_GSV datasets, they do not explicitly contain a license in their code repository, but are under MIT license.

A.2 ADDITIONAL RESULTS

Ablation study Here we did an ablation study to show the effectiveness of our proposed GAN model. There are two modules in the decoder of our GAN model including h^z and h^x. Including

[1] https://github.com/creafz/pytorch-cnn-finetune

[2] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

Figure 5: Examples of images used to compute the average variance score in Table 3 for our GAN model.

h^z in the GAN loss will encourage the model to use the information in z when generating the perturbations. And including h^x is a common practice used in GAN training which can stabilize the training Zhu et al. (2017). As an ablation study, we trained a GAN model without h^z and listed the clean accuracy and robust accuracy of the model trained with the GAN without h^z under simulation attack in Table 4.

Table 4: Ablation study: Performance comparison of adversarial training using GAN model with and without h^z.

AT with our GAN w/ h^z	0.4631
AT with our GAN w/o h^z	0.4757

Without h^z, the GAN model will generate weather masks without diversity and therefore decrease the robust accuracy.

Examples of images used to compute Table 3 We show some examples of images used to compute the variance score for MUNIT and CVAE in Figure 3. Here we show some examples of images used to compute the average variance of our GAN model in Figure 5. We can see that our model can generate rainy images with more diverse rain steak distribution.