It’s Raining Cats or Dogs? Adversarial Rain Attack on DNN Perception

Liming Zhai¹, Felix Juefei-Xu², Qing Guo³*, Xiaofei Xie³, Lei Ma⁴, Wei Feng⁵, Shengchao Qin⁶, Yang Liu³

¹Wuhan University, China ²Alibaba Group, USA ³Nanyang Technological University, Singapore ⁴Kyushu University, Japan ⁵Tianjin University, China ⁶Teesside University, UK

Abstract

Rain is a common phenomenon in nature and an essential factor for many deep neural network (DNN) based perception systems. Rain can often post inevitable threats that must be carefully addressed especially in the context of safety- and security-sensitive scenarios (e.g., autonomous driving). Therefore, a comprehensive investigation of the potential risks of the rain to a DNN is of great importance. Unfortunately, in practice, it is often rather difficult to collect or synthesize rainy images that can represent all raining situations that possibly occur in the real world. To this end, in this paper, we start from a new perspective and propose to combine two totally different studies, i.e., rainy image synthesis and adversarial attack. We present an adversarial rain attack, with which we could simulate various rainy situations with the guidance of deployed DNNs and reveal the potential threat factors that can be brought by rain, helping to develop more rain-robust DNNs. In particular, we propose a factor-aware rain generation that simulates rain steaks according to the camera exposure process and models the learnable rain factors for adversarial attack. With this generator, we further propose the adversarial rain attack against the image classification and object detection, where the rain factors are guided by the various DNNs. As a result, it enables to comprehensively study the impacts of the rain factors to DNNs. Our large-scale evaluation on three datasets, i.e., NeurIPS’17 DEV, MS COCO and KITTI, demonstrates that our synthesized rainy images can not only present visually realistic appearances, but also exhibit strong adversarial capability, which builds the foundation for further rain-robust perception studies.

1 Introduction

Rain is condensed aqueous vapor in the form of falling drops with high speeds and small sizes. As a common weather phenomenon, rain can bring massive impacts not only on our human society, but also significant influences on today’s intelligent era. Such impacts are most prominent in deep neural network (DNN) based perception systems, e.g., autonomous driving, video surveillance and unmanned aerial vehicle (UAV), which can be easily disturbed by the inevitable rain effects, suffering from severe safety and security issues (Zhang et al. 2018; Bahnsen and Moeslund 2018). Therefore, it is of great importance and pressing to comprehensively study how the rain affects the DNNs.

*Qing Guo is the corresponding author: tsingguo@gmail.com.

Figure 1: Illustration of clean images (left) and adversarial rain examples (right). First row: images from NeurIPS’17 DEV dataset with image classifier Inception v3 (Szegedy et al. 2016). Second row: images from MS COCO dataset with object detector Faster-RCNN (Ren et al. 2015).

The rain effect in images to DNNs has something in common with adversarial examples (Akhtar and Mian 2018; Yuan et al. 2019). In particular, rain can also mislead the DNNs as if by adding minor rain noises (i.e., some non-trivial perturbation) to benign samples under some constraints. For image samples, the adversarial perturbations are actually controllable artificial noises superimposed on images. However, rain often has various appearances determined by weather conditions and environment lighting, and further undergoes visual changes presented in rainy images due to varying camera parameters (Garg and Nayar 2007). While existing rain rendering methods can to some extent generate rainy images, they are mostly used for a specific scenario (Garg and Nayar 2006; Rousseau, Jolivet, and Ghazanfarpoor 2006; Cres and Patow 2013; Weber et al. 2015; Halder, Lalonde, and Charette 2019). Therefore, it is challenging or even impossible to collect or synthesize rainy images in all kinds of rain situations that can potentially occur in the real world.

In this paper, we intend to tackle this problem from a new angle, by taking a combined perspective of two differ-
ent studies, *i.e.*, rainy image synthesis and adversarial attack, and present an adversarial rain attack, by which we could simulate various (potentially worst-case) rainy situations with the guidance of deployed DNNs, to reveal the potential common threat factors of rain. Our adversarial rain attack takes both adversarial capability and rain appearance quality into consideration. The adversarial rain is made intentionally visible, which is completely opposite to the imperceptible adversarial noises, thus the adversarial ability has to be achieved with stronger constraints. For this purpose, we design a factor-aware rain generation method that simulates rain streaks following the camera exposure process. In addition, we also model several types of controllable rain factors, including rain intensity, rain direction, and rain brightness, all of which can be learned and tuned for adversarial attack under the constraints of rain appearance. After generating the rain streaks, we perform a composition of the rainy images by using rain synthesis models.

We conduct a large-scale evaluation of our adversarial rain attack on the state-of-the-art DNN-based image classifiers and object detectors, to demonstrate the effectiveness of our technique. We find that, indeed, our technique can successfully generate adversarial rain images with high-quality (see some examples in Figure 1). Our learned rain factors can also be used to guide the generation of real adversarial rain examples, demonstrating the potential threats of natural rain that should be called attention in future studies.

Our main contributions are summarized as follows:

- We identify the essential problem of the inevitable rain factors to the impacts and risks of DNNs. With a comprehensive study on the relation of rain effect and DNN, we propose a new type of adversarial attack by using common rainy weather rather than adversarial noises. The proposed adversarial rain enriches the current family of adversarial examples, in providing an important family of physical effects that can occur in the real world.
- We design a novel rain generation method that can synthesize both photo-realistic rain images, which also exhibits strong DNN attack capability, helping to analyze the risks of rain effects to a DNN.
- Furthermore, we also propose techniques to synthesize the more rains, which is more close to the real-world rains, which is significantly different from the traditional rain rendering methods.
- We perform a large scale evaluation and demonstrate the effectiveness of our adversarial rain attack in terms of attack success rate and transferability even under the state-of-the-art de-raining methods.

## 2 Related Work

### 2.1 Adversarial Noise Attacks

Extensive studies have shown that state-of-the-art DNNs are still vulnerable to adversarial examples (Yuan et al. 2019). In their early study, (Szegedy et al. 2014) demonstrated that intentionally designed perturbations added to images can mislead classification results, and proposed to generate adversarial examples by using a box-constrained L-BFGS method. To overcome the high complexity in (Szegedy et al. 2014), (Goodfellow, Shlens, and Szegedy 2015) designed a fast gradient sign method (FGSM) to produce adversarial noises with the sign of loss gradient in a one-step process. Since then, numerous gradient-based adversarial noise attacks for image classification have been proposed, including the attacks by developing the FGSM in multiple iterations (Kurakin, Goodfellow, and Bengio 2017; Carlini and Wagner 2017; Madry et al. 2018), integrating momentum iteratively into FGSM (MI-FGSM) (Dong et al. 2018), employing diverse input method (DIM) in each iteration (Xie et al. 2019a) and optimizing perturbations with a translation-invariant method (TIM) (Dong et al. 2019).

Recently, adversarial examples are also applied to other computer vision tasks. (Xie et al. 2017) proposed a dense adversary generation (DAG) algorithm to attack object detection and semantic segmentation models, in which the loss of object proposals with assigned false labels was minimized during gradient-descent iterations. (Li et al. 2018) suggested attacking the region proposal network (RPN) widely adopted in object detectors, and designed a robust adversarial perturbation (RAP) method. (Wang et al. 2020b) generated adversarial examples by using the total loss of both object detector and RPN. Differently, (Wei et al. 2019) proposed a unified and efficient adversary (UEA) method for image and video object detection based on generative adversarial network (GAN), and achieved high transferability and low computation cost. Moreover, recent studies also explore physical adversarial attacks against real-world object detectors (Chen et al. 2018; Huang et al. 2020).

The current adversarial attacks are mainly performed under the disguise of noises and some other image manipulations (Wang et al. 2020a; Cheng et al. 2020a; Guo et al. 2020b), all of which are artificial fabrications and may be suspicious to human eyes. We instead use common rainy weather as a natural camouflage for launching adversarial attacks.

### 2.2 Rain Rendering

Rain rendering refers to the generation or synthesis of photo-realistic rain effects in computer graphics. Existing rain rendering methods can roughly fall into two categories. The first category simulates the texture patterns of rain streaks, and post-processes the rain streaks to fit the scene (Wang and Wade 2004; Wang et al. 2006; Tatarchuk 2006). Such methods are often simple and computationally efficient, but lack of realistic rain appearance. The second category leverages particle systems, in which the rain rate, rain distribution, and rain velocity can be flexibly controlled (Garg and Nayar 2006; Rousseau, Jolivet, and Ghazanfarpour 2006; Creus and Patow 2013; Weber et al. 2015; Halder, Lalonde, and Charette 2019). In particular, these methods consider the physical and optical properties of rain, being able to generate visually realistic rain effects, but at expense of high computation costs.

Different from the above texture-based and physics-based rain rendering methods, we generate the rain directly from random noises with the guidance of DNNs, ensuring both high rain realism and low computation complexity.
2.3 Rain Removal

Single image rain removal is the opposite of rain rendering, and the rain removal methods can also be largely divided into two categories (Yang et al. 2020). The first category is model-based methods, regarding the single image rain removal as an image decomposition or signal separation problem between the rain layer and background layer. Typical methods include dictionary-based sparse coding (Kang, Lin, and Fu 2011; Luo, Xu, and Ji 2015), nonlocal mean filtering (Kim et al. 2013), low rank representation model (Chen and Hsu 2013; Chang, Yan, and Zhong 2017), and patch-based Gaussian mixture model (Li et al. 2016).

The second category is the recently fast-growing deep learning-based method, which automatically learns a non-linear mapping between rainy images and clean images. Many DNN architectures have been proposed for de-raining. Some representative methods include deep detail network for predicting rain residues (Fu et al. 2017), multi-stream dense network for rain density estimation and rain removal (Zhang and Patel 2018), multi-task network for joint rain detection and removal (Yang et al. 2017), and scale-free network for two-stage de-raining (Yang et al. 2019). Furthermore, some recent attempts are also made by using GANs for the de-raining task (Zhang, Sindagi, and Patel 2019; Qian et al. 2018; Li, Cheong, and Tan 2019).

The existing de-raining methods are applied to real rain or artificially synthesized rain. In experimental section, we use rain removal to evaluate the survivability of our adversarial rain, whose generation is guided by DNNs.

3 Methodology

In this section, we first introduce the factor-aware rain generation, which synthesizes the rainy images to be prepared for adversarial attack. Next, we detail the implementation of adversarial rain attack on image classification and object detection tasks.

3.1 Factor-aware Rain Generation

Natural rain is liquid water that falls visibly in small drops. Due to the high speeds of drops and the exposure time limitation of a camera, the rain in an image can usually be motion-blurred and displayed as longer streaks (Garg and Nayar 2007). The latest rain rendering methods (Halder, Lalonde, and Charette 2019; Creus and Patow 2013) are mostly based on a ready-made rain streak database (Garg and Nayar 2006) and/or a particle simulator, requiring strict operating conditions. We instead propose a new mechanism, named factor-aware rain generation, to generate photo-realistic rain streaks (see Figure 2). We directly leverage random noises as a starting point to generate rain streaks, by taking the following consideration.

- A raindrop typically has a diameter of 1-2 mm, and usually does not exceed 10 mm (Garg and Nayar 2007; Van Boxel 1997). The tiny raindrops in a far distance captured by the camera look like image noises. Some de-raining methods also treat the raindrops or rain streaks in images as special high-frequency noises (Chang, Yan, and Zhong 2017; Eigen, Krishnan, and Fergus 2013). These are important physical rain properties and should be considered during method design for better photo-realisticity.

- Many synthesized rain databases commonly adopt random noises as the basis (Fu et al. 2017; Zhang and Patel 2018), which can be an effective starting point.

- Random noises can be easily modulated during gradient back-propagation, which provides the chance and efficient way for the combination investigation of worst-case rain effects to the decision of DNNs.

To generate realistic looking rain from random noises, some properties of noises, including noise distribution, noise strength and noise density, should be carefully considered. The random noises are generated from a uniform distribution $U(a, b)$ and form a noise image with the same size as the input clean image. Since the rain streak usually has larger intensities, the lower bound $a$ and upper bound $b$ of $U(a, b)$ are assigned large values (see experimental section). The noise image is then uniformly sampled using a sampling rate $\varepsilon_n$ by setting the non-selected elements zero, thus obtaining a sparse noise image. The sampling rate $\varepsilon_n$ is the ratio of the desired number of noise elements to the image size, and it determines the noise density of a noise image, and also affects the rain rate in the final rainy image. The reason behind the uniform sampling is that the distribution of drops is uniform over space and time (Garg and Nayar 2007). Inspired by the formation of rain streaks in camera sensing process, we simulate the motion-blur of the noise image as the rain effects. Since the motion-blur is an integration of the positions of moving objects over the period of exposure time, the motion-blur of the noise image can be simulated by continuous translation along a specific direction.

Let $N$ be a noise image, and $T_\theta$ the translation transformation with translation parameters $\theta$. The transformed noise
image is denoted as $\mathbb{N}^T_0$. The $T_0$ can be performed by using a spatial transformer network (Jaderberg et al. 2015), and thus $\theta$ represents affine matrix parameters. To transform the noises into continuous rain streaks, we further divide the translation transformation process into $N$ parts. To this end, we apply a $N$-step translation transformation to $\mathbb{N}$ using translation parameters $i \Delta \theta$, where $\Delta \theta = \theta / N$ and $i \in [0, N]$. After the $N$-step translation transformation, we obtain $N$ transformed noise images $\mathbb{N}^T_{i\Delta \theta}$. To smooth the rain streak formed by summing up $N$ transformed noise images, we convolve the $N$ transformed noise images with a mean filter kernel $K_\theta$ with size $(N + 1) \times C \times 1 \times 1$, where $C$ is the number of color channels. Finally, the $p$-th rain streak $R_p$ is generated by

$$R_p = g(N_p, T_\theta, K_\theta) = \sum_{i,q \in [0,N]} N_p^{T_{i\Delta \theta}k_{pq}} \quad (1)$$

where $N_p$ is the $p$-th noise element in $\mathbb{N}$, $K_\theta$ is the kernel corresponding to $N_p$, and $k_{pq}$ is the $q$-th weight of $K_\theta$. All the $R_p$ make up a rain layer $\mathbf{R}$.

Given a clean image $\mathbf{X}$ and a rain layer $\mathbf{R}$, the rainy image $\mathbf{X'}$ can be synthesized by various rain synthesis models. Take the additive composite model (Yang et al. 2020) as an example, the rainy image synthesis is formulated as

$$\mathbf{X'} = h(\mathbf{X}, \mathbf{R}) = \mathbf{X} + \mathbf{R} \quad (2)$$

where $h(\mathbf{X}, \mathbf{R})$ represents a synthesis function.

Note that, in this initial attempt, we have not yet considered the environmental lighting for rain generation, and we consider a simple scenario where the rain streaks are thin enough to neglect the reflection and refraction of light. More complicated and advanced can be integrated into our rain generation process (Garg and Nayar 2007; Halder, Lalonde, and Charette 2019) as future work.

The factor-aware rain generation method just provides initial rainy images, which still lack of adversarial ability. In the following two subsections, we use the DNNs deployed in image classification and object detection to further guide the rain factors for producing adversarial rainy images.

### 3.2 Adversarial Rain Attack on Image Classification

Given a classifier $f(\mathbf{X}) : \mathbf{X} \in \mathcal{X} \rightarrow y \in \mathcal{Y}$ that maps an input image $\mathbf{X}$ to a ground-truth class label $y$. The adversarial attack aims to generate a constrained example $\mathbf{X'}$ that can mislead $f(\mathbf{X'})$ to output a false label, usually by maximizing the loss $L(\mathbf{X'}, y)$.

In conventional adversarial noise attacks, the adversarial example $\mathbf{X'}$ is typically generated by adding adversarial noises to $\mathbf{X}$. For our adversarial rain attack, we generate $\mathbf{X'}$ by using the rain generation (Eq. (1)) and rain synthesis model (Eq. (2)). To enable successful adversarial attacks of rainy images for image classification, we need to optimize the rain factors under the constraint of rain appearance. There are three types of rain factors should be considered: noise $\mathbb{N}$, translation parameters $\theta$ and kernels $K = \{K_p | p \in [1, \|\mathbb{N}\|_0]\}$. The $\mathbb{N}$ affects the rain intensity, $\theta$ controls the direction of rain streaks, and $K$ determines the brightness and smoothness of rain streaks. The adversarial rainy images can be generated by solving the following constrained optimization problem:

$$\arg \max_{\mathbb{N}, \theta, K} L(h(\mathbf{X}_p, \sum_{i,q \in [0,N]} N_p^{T_{i\Delta \theta}k_{pq}}), y) \quad (3)$$

s.t. $\forall p, \|\mathbb{N}\|_0 \leq \varepsilon_n \mathbf{M}$, $\theta \leq \varepsilon_\theta$, $k_{pq} \leq \varepsilon_k$ \quad (4)

where $\mathbf{M}$, $\theta$ and $k_{pq}$ are used to curb the rain factors to ensure realistic looking rain. The values of $\varepsilon_n$, $\theta$ and $k_{pq}$ will be elaborated in the experimental section.

During the rain generation process, we can calculate the gradient of the loss function w.r.t. all rain factors, and tune the rain factors along a gradient ascent direction for adversarial attacking. Therefore, the adversarial rain attack can be integrated into current existing gradient based adversarial noise attack framework, such as FGSM (Goodfellow, Shlens, and Szegedy 2015), MI-FGSM (Dong et al. 2018), DIM (Xie et al. 2019a) and TIM (Dong et al. 2019), etc. Algorithm 1 presents the details of our adversarial rain attack for image classification tasks. In particular, each iteration generates a loss, a rain layer and a rainy image. Eventually, only the rainy image with the largest loss in $S$ iterations is selected as the final adversarial rainy image.

### 3.3 Adversarial Rain Attack on Object Detection

In this section, we further extend our adversarial rain attack to the object detection task. Given an object detector $f(\mathbf{X}) : \mathbf{X} \in \mathcal{X} \rightarrow t \in \mathcal{T}$ that maps an input image $\mathbf{X}$ to a recognition target $t$. The $t$ can be of any intermediate outputs or final outputs of a detector, not limited to the class labels (Xie et al. 2017), such as final bounding-boxes (Wang et al. 2020b), confidence scores and bounding-boxes of the object proposals in an RPN (Li et al. 2018). All these types of targets can be utilized to generate constrained adversarial examples $\mathbf{X'}$ to fool $f(\mathbf{X'})$, in producing wrong detection results. This process can be accomplished by minimizing the loss $L(\mathbf{X'}, \mathcal{T'})$, where $\mathcal{T'}$ is a set of one type of adversarial targets $t'$.

Similar to the image classification task, the adversarial rainy images for object detection can be generated by

$$\arg \min_{\mathbb{N}, \theta, K} \sum_j L_j(h(\mathbf{X}_p, \sum_{i,q \in [0,N]} N_p^{T_{i\Delta \theta}k_{pq}}), \mathcal{T}'_j) \quad (5)$$

s.t. $\forall p, \|\mathbb{N}\|_0 \leq \varepsilon_n \mathbf{M}$, $\theta \leq \varepsilon_\theta$, $k_{pq} \leq \varepsilon_k$ \quad (5)

where $j$ is the index of adversarial target types (e.g., class labels, bounding-boxes, or confidence scores). One or more types of adversarial target set $\mathcal{T}'_j$ can be considered for the total loss function, in which $L_j$ denotes the loss function corresponding to $\mathcal{T}'_j$.

Similarly, we can tune the rain factors via a gradient descent algorithm, and also apply the adversarial rain attack to current gradient based adversarial noise attacks for object
4 Experiments

4.1 Experimental Setup

Datasets. To demonstrate the effectiveness of our technique, we perform comprehensive evaluation of our adversarial rain attack on two different tasks: image classification and object detection. For image classification, we use NeurIPS’17 adversarial competition dataset DEV (Kurakin et al. 2018) for experiments, which consists of 1,000 images, and is compatible with ImageNet. For object detection, we perform experiments on MS COCO 2014 minival split (Lin et al. 2014) and KITTI object benchmark (Geiger, Lenz, and Urtasun 2012), which contain 5,000 images and 7,480 images, respectively.

Threat Models. To validate the effectiveness of the adversarial rain attack on image classification, we exploit four publicly-available pre-trained models, including Inception v3 (Inc-v3) (Szegedy et al. 2016), Inception v4 (Inc-v4), Inception ResNet v2 (IncRes-v2) (Szegedy et al. 2017), and Xception (Chollet 2017). The threat model for object detection is Faster RCNN (FR) (Ren et al. 2015) with different backbones, including VGG16 (v16) (Simonyan and Zisserman 2014), MobileNet (mn) (Howard et al. 2017), ResNet50 (rn50), ResNet101 (rn101) and ResNet152 (rn152) (He et al. 2016).

Baselines. We compare the adversarial rain attack with six adversarial noise attacks: FGSM (Goodfellow, Shlens, and Szegedy 2015), MI-FGSM (Dong et al. 2018), DIM (Xie et al. 2019a) and TIM’s three variants TI-FGSM, TI-MI-FGSM, and TI-DIM (Dong et al. 2019). The comparative methods used for objection detection are RAP (Li et al. 2018) and UEA (Wei et al. 2019), respectively.

detection, such as DAG (Xie et al. 2017) and RAP (Li et al. 2018), etc. The attacking algorithm for object detection is similar to that of image classification, and we omit it here due to the page limit.

Evaluation Metrics. We use attack success rate and mean average precision (mAP) as major metrics to measure the adversarial ability on image classification and object detection, respectively.

Implementation Details. To attack image classifiers and object detectors, we individually generate adversarial rain examples by integrating with MI-FGSM and RAP in 20 iterations, and adopt the default parameters suggested in MI-FGSM and RAP.

4.2 Ablation Study

In this subsection, we first conduct ablation experiments to study the impact of rain factors. When considering one rain factor, the other rain factors are set as default values and remain unchanged. For simplicity, all the ablation experiments are only performed on the image classification task.

As described in Section 3, the adversarial rain is generated from random noises that follow a normal distribution $\mathcal{U}(a, b)$, and further optimized under the constraints related to thresholds $\varepsilon_n$, $\varepsilon_\theta$, and $\varepsilon_k$. We first show in Figure 3 the effects of different combinations of lower bound $a$ and upper bound $b$ in $\mathcal{U}(a, b)$ on the adversarial ability. We can observe that the combinations of $a$ and $b$ with larger values contribute to a higher attack success rate due to larger noise strength. A larger distribution interval increases noise diversity, and thus enriches the texture pattern of rain appearance. Eventually, we choose $\mathcal{U}(0.7, 1.0)$ as a suitable compromise.

The $\varepsilon_n$ and $\varepsilon_\theta$ are scalar thresholds, while $\varepsilon_k$ is a $2 \times 1$ vector containing thresholds for translation parameters in affine transform. We report the effects of three types of thresholds $\varepsilon_n$, $\varepsilon_\theta$, and $\varepsilon_k$ on the adversarial ability in Table 1. We can see that the adversarial rain attack achieves higher attack success rates at larger $\varepsilon_n$, $\varepsilon_\theta$, and $\varepsilon_k$, indicating that higher rain inten-

![Figure 3](image-url)
Further improved by employing lighting information, which we leave as future work.

User study. We also conduct a user study to further evaluate the rain appearance following (Halder, Lalonde, and Charette 2019). Specifically, we randomly select 10 images from each rain database (i.e., our adversarial rainy images and other five synthesized rain databases), in collecting a total of 60 rainy images. Then, we invite 30 participants (12 undergraduate students, 18 graduate students; 13 female, 17 male; aged 20 to 34) to rate the visual effect of rainy images using a 5-points Likert scale. The results are summarized in Figure 5. We can see that our adversarial rainy images and physics-based rainy images obtain similar high voting scores compared with other types of rainy images, confirming the visual realism of our rain generation method.
specific rainy image. Therefore, a natural question arises, i.e., determine the adversarial capability of a specific rainy image. Thus, a natural question arises, i.e., whether it is possible to use the obtained rain factors to adjust the appearance and dynamics of real rain so that to enable it to fool the DNNs. The resulting rain can be regarded as a type of real-world adversarial rain attack, which is performed in the following steps.

1) Fix the object to be classified or detected by DNNs, and capture an image with a camera.

2) Generate the corresponding rainy image with our adversarial rain attack, and also obtain the rain factor values.

3) Leverage the rain factors to adjust the rain to achieve adversarial capability.

For this experiment, we perform a physical behavior study in the real-world environment by our authors, using water drops from cans to emulate the real rain effects. In particular, we spray the water by a watering can so that the water drops are freely falling from a higher place. The desired rain intensity (N) can be achieved by adjusting the amount of water drops, and the rain directions ($\theta$) are controlled by the wind speed and wind direction of an electric fan. The kernels $\mathcal{K}$ cannot be obtained in the real world, so we do not consider adjusting the rain textures.

We use the Faster-RCNN to detect the clean image, adversarial rainy image and real-world adversarial rainy image, the results of which are summarized in Figure 7. We observe that the two types of adversarial rainy images can both fool the object detector, demonstrating that the rain factors can be used as guidance for launching real-world adversarial attacks. The appearances of two adversarial rainy images are not very similar, but this problem can be adjusted by using more sophisticated rain simulation equipment.

### 4.7 Real-world Adversarial Rain Attack

The adversarial rain attack produces adversarial examples by using the rain factors guided by DNNs. The rain factors, including rain noises $N_j$, translation parameters $\theta$, and kernels $\mathcal{K}$, determine the adversarial capability of a specific rainy image. Therefore, a natural question arises, i.e., whether it is possible to use the obtained rain factors to adjust the appearance and dynamics of real rain so that to enable it to fool the DNNs. The resulting rain can be regarded as a type of real-world adversarial rain attack, which is performed in the following steps.

1) Fix the object to be classified or detected by DNNs, and capture an image with a camera.

2) Generate the corresponding rainy image with our adversarial rain attack, and also obtain the rain factor values.

3) Leverage the rain factors to adjust the rain to achieve adversarial capability.

For this experiment, we perform a physical behavior study in the real-world environment by our authors, using water drops from cans to emulate the real rain effects. In particular, we spray the water by a watering can so that the water drops are freely falling from a higher place. The desired rain intensity (N) can be achieved by adjusting the amount of water drops, and the rain directions ($\theta$) are controlled by the wind speed and wind direction of an electric fan. The kernels $\mathcal{K}$ cannot be obtained in the real world, so we do not consider adjusting the rain textures.

We use the Faster-RCNN to detect the clean image, adversarial rainy image and real-world adversarial rainy image, the results of which are summarized in Figure 7. We observe that the two types of adversarial rainy images can both fool the object detector, demonstrating that the rain factors can be used as guidance for launching real-world adversarial attacks. The appearances of two adversarial rainy images are not very similar, but this problem can be adjusted by using more sophisticated rain simulation equipment.

### 4.8 Resistance to De-raining Methods

De-raining methods can remove the rain streaks in images, which can potentially be used for reducing the rain impact on DNNs. In this section, we further evaluate the effects of our adversarial rain under a rain removal scenario. Specifically, we use two types of de-raining methods, i.e.,
and then compare the adversarial ability before and after de-raining. The results are summarized in Table 5, where AdvRain-GMM and AdvRain-DID-MDN denote de-rained adversarial rainy images. We can see that the AdvRain-GMM and AdvRain-DID-MDN obtain lower success rate or higher mAP scores than the original AdvRain, indicating that the AdvRain is indeed affected by de-raining methods. However, the AdvRain still preserves the adversarial capability to a different extent, which calls for more advanced design of de-rain methods to reduce the potential impacts to DNN decisions.

5 Conclusions

In this paper, we perform a comprehensive study of the potential risks of rain effect on DNN perception systems. We propose a new type of adversarial rain attack, which simu-
lates natural rain situations with the guidance of DNNs and thus synthesizes visually realistic rainy images to mislead image classification and object detection. The adversarial rain attack reveals the essential and inevitable threat factors of rain, which commonly exist in the real world. We conducted extensive experiments to validate the effectiveness of our adversarial rain attack on different datasets and tasks. Our results show that the current state-of-the-art DNNs can be vulnerable to the inevitable rain effects in the real-world, which calls the attention to take rain effects into consideration for more advanced design of real-world DNN-based perception systems.

In future work, we would like to further optimize our adversarial rain by considering lighting conditions and fog-like rains, and also extend the adversarial rain attack to a more broad range of applications, such as semantic segmentation (Arnab, Miksik, and Torr 2018; Guo et al. 2018, 2017c) and visual object tracking (Guo et al. 2020a,c, 2017a,b). Moreover, we will use our adversarial rain as a new kind of mutation for DNN testing (Xie et al. 2019b; Ma et al. 2018; Du et al. 2019; Xie et al. 2019c). The formulation of the adversarial rain attack lies somewhere between traditional additive noise attacks and purely non-additive-noise attacks such as (Gao et al. 2020; Cheng et al. 2020b; Tian et al. 2020; Guo et al. 2020b). It will be worthwhile to further explore the interplay between the adversarial rain and other mentioned adversarial attack modalities.

References

Akhtar, N.; and Mian, A. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access 6: 14410–14430.

Arnab, A.; Miksik, O.; and Torr, P. H. 2018. On the robustness of semantic segmentation models to adversarial attacks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 888–897.

Bahnsen, C. H.; and Moeslund, T. B. 2018. Rain removal in traffic surveillance: Does it matter? IEEE Transactions on Intelligent Transportation Systems 20(8): 2802–2819.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In 2017 IEEE symposium on security and privacy (sp), 39–57. IEEE.

Chang, Y.; Yan, L.; and Zhong, S. 2017. Transformed low-rank model for line pattern noise removal. In Proceedings of the IEEE International Conference on Computer Vision, 1726–1734.

Chen, S.-T.; Cornelius, C.; Martin, J.; and Chau, D. H. P. 2018. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 52–68.

Chen, Y.-L.; and Hsu, C.-T. 2013. A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In Proceedings of the IEEE International Conference on Computer Vision, 1968–1975.

Cheng, Y.; Guo, Q.; Juefei-Xu, F.; Xie, X.; Lin, S.-W.; Lin, W.; Feng, W.; and Liu, Y. 2020a. Perceptually Aware and Stealthy Adversarial Denoise Attack. arXiv preprint arXiv:2007.07097.

Cheng, Y.; Juefei-Xu, F.; Guo, Q.; Fu, H.; Xie, X.; Lin, S.-W.; Lin, W.; and Liu, Y. 2020b. Adversarial Exposure Attack on Diabetic Retinopathy Imagery. arXiv preprint arXiv:1902.00755.

Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, 1251–1258.

Creus, C.; and Patow, G. A. 2013. R²: Realistic rain rendering in realtime. Computers & graphics 37(1-2): 33–40.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In Proceedings of the IEEE conference on computer vision and pattern recognition, 9185–9193.

Dong, Y.; Pang, T.; Su, H.; and Zhu, J. 2019. Evading defenses to transferable adversarial examples by translation-invariant attacks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4312–4321.

Du, X.; Xie, X.; Li, Y.; Ma, L.; Liu, Y.; and Zhao, J. 2019. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. In Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 477–487.

Eigen, D.; Krishnan, D.; and Fergus, R. 2013. Restoring an image taken through a window covered with dirt or rain. In Proceedings of the IEEE international conference on computer vision, 633–640.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. International journal of computer vision 88(2): 303–338.

Fu, X.; Huang, J.; Zeng, D.; Huang, Y.; Ding, X.; and Paisley, J. 2017. Removing rain from single images via a deep detail network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3855–3863.

Gao, R.; Guo, Q.; Juefei-Xu, F.; Yu, H.; Ren, X.; Feng, W.; and Wang, S. 2020. Making Images Undiscoverable from Co-Saliency Detection. arXiv preprint arXiv:2003.01612.

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

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

Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In 2012 IEEE Computer Vision Winter Workshop.

Ghodsi, A.; Chollet, F.; and Mansour, T. 2019. Shapley value meets deep learning: Explainable neural networks. In International Conference on Learning Representations, ICLR, 1–11.

Guo, Q.; Feng, W.; Zhou, C.; Huang, R.; Wan, L.; and Wang, S. 2017a. Learning dynamic siamese network for visual object tracking. In Proceedings of the IEEE international conference on computer vision, 1763–1771.

Guo, Q.; Feng, W.; Zhou, C.; Pun, C.-M.; and Wu, B. 2017b. Structure-regularized compressive tracking with online data-driven sampling. IEEE Transactions on Image Processing 26(12): 5692–5705.

Guo, Q.; Han, R.; Feng, W.; Chen, Z.; and Wan, L. 2020a. Selective spatial regularization by reinforcement learned decision making for object tracking. IEEE Transactions on Image Processing 29: 2999–3013.
Guo, Q.; Juefei-Xu, F.; Xie, X.; Ma, L.; Wang, J.; Feng, W.; and Liu, Y. 2020b. ABBA: Saliency-Regularized Motion-Based Adversarial Blur Attack. arXiv preprint arXiv:2002.03500.

Guo, Q.; Sun, S.; Dong, F.; Feng, W.; Gao, B. Z.; and Ma, S. 2017c. Frequency-tuned ACM for biomedical image segmentation. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 821–825. IEEE.

Guo, Q.; Sun, S.; Ren, X.; Dong, F.; Gao, B. Z.; and Feng, W. 2018. Frequency-tuned active contour model. Neurocomputing 275: 2307–2316.

Guo, Q.; Xie, X.; Juefei-Xu, F.; Ma, L.; Li, Z.; Xue, W.; Feng, W.; and Liu, Y. 2020c. SPARK: Spatial-aware online incremental attack against visual tracking. In Proceedings of the European Conference on Computer Vision (ECCV).

Halder, S. S.; Lalonde, J.-F.; and Charette, R. d. 2019. Physics-based rendering for improving robustness to rain. In Proceedings of the IEEE International Conference on Computer Vision, 10203–10212.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, 770–778.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Huang, L.; Gao, C.; Zhou, Y.; Xie, C.; Yuille, A. L.; Zou, C.; and Liu, N. 2020. Universal Physical Camouflage Attacks on Object Detectors. In Proceedings of the IEEE International Conference on Computer Vision, 720–729.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In Advances in neural information processing systems, 2017–2025.

Kang, L.-W.; Lin, C.-W.; and Fu, Y.-H. 2011. Automatic single-image-based rain streaks removal via image decomposition. IEEE transactions on image processing 21(4): 1742–1755.

Kim, J.-H.; Lee, C.; Sim, J.-Y.; and Kim, C.-S. 2013. Single-image deraining using an adaptive nonlocal means filter. In IEEE International Conference on Image Processing, 914–917.

Kurakin, A.; Goodfellow, I.; Bengio, S.; Dong, Y.; Liao, F.; Liang, M.; Pang, T.; Zhu, J.; Hu, X.; Xie, C.; et al. 2018. Adversarial Attacks and Defences Competition. arXiv preprint arXiv:1804.00097.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017. Adversarial Machine Learning at Scale. In International Conference on Learning Representations, ICLR, 1–17.

Li, R.; Cheong, L.-F.; and Tan, R. T. 2019. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1633–1642.

Li, Y.; Tan, R. T.; Guo, X.; Lu, J.; and Brown, M. S. 2016. Rain streak removal using layer priors. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2736–2744.

Li, Y.; Tian, D.; Bian, X.; and Lyu, S. 2018. Robust adversarial perturbation on deep proposal-based models. In The British Machine Vision Conference.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In European conference on computer vision, 740–755. Springer.

Luo, Y.; Xu, Y.; and Ji, H. 2015. Removing rain from a single image via discriminative sparse coding. In Proceedings of the IEEE International Conference on Computer Vision, 3397–3405.

Ma, L.; Zhang, F.; Sun, J.; Xue, M.; Li, B.; Juefei-Xu, F.; Xie, C.; Li, L.; Liu, Y.; Zhao, J.; and Wang, Y. 2018. DeepMutation: Mutation Testing of Deep Learning Systems. In The 29th IEEE International Symposium on Software Reliability Engineering (ISSRE).

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In International Conference on Learning Representations.

Qian, R.; Tan, R. T.; Yang, W.; Su, J.; and Liu, J. 2018. Attentive generative adversarial network for raindrop removal from a single image. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2482–2491.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems, 91–99.

Rousseau, P.; Jolivet, V.; and Ghazanfarpour, D. 2006. Realistic real-time rain rendering. Computers & Graphics 30(4): 507–518.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations, ICLR.

Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. A. 2017. Inception-v4, inception-ResNet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4278–4284.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2818–2826.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In International Conference on Learning Representations, ICLR, 1–9.

Tatarchuk, N. 2006. Artist-directable real-time rain rendering in city environments. In ACM SIGGRAPH 2006 Courses, 23–64. ACM.

Tian, B.; Guo, Q.; Juefei-Xu, F.; Chan, W.; Cheng, Y.; Li, X.; Xie, X.; and Qin, S. 2020. Bias Field Poses a Threat to DNN-based X-Ray Recognition. arXiv preprint arXiv.

Van Boxel, J. H. 1997. Numerical model for the fall speed of raindrops in a rain fall simulator. In Workshop on wind and water erosion, 77–85.

Wang, L.; Lin, Z.; Fang, T.; Yang, X.; Yu, X.; and Kang, S. B. 2006. Real-time rendering of realistic rain. In ACM SIGGRAPH 2006 Sketches, 156–es. ACM.

Wang, N.; and Wade, B. 2004. Rendering falling rain and snow. In ACM SIGGRAPH 2004 Sketches, 14. ACM.

Wang, R.; Juefei-Xu, F.; Xie, X.; Ma, L.; Huang, Y.; and Liu, Y. 2020a. Amora: Black-box adversarial morphing attack. In Proceedings of the 28th ACM International Conference on Multimedia.

Wang, Y.; Wang, K.; Zhu, Z.; and Wang, F.-Y. 2020b. Adversarial attacks on Faster R-CNN object detector. Neurocomputing 382: 87–95.

Weber, Y.; Jolivet, V.; Gilet, G.; and Ghazanfarpour, D. 2015. A multiscale model for rain rendering in real-time. Computers & Graphics 50: 61–70.
Wei, X.; Liang, S.; Chen, N.; and Cao, X. 2019. Transferable adversarial attacks for image and video object detection. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, 954–960.

Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; and Yuille, A. 2017. Adversarial examples for semantic segmentation and object detection. In Proceedings of the IEEE International Conference on Computer Vision, 1369–1378.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019a. Improving transferability of adversarial examples with input diversity. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2730–2739.

Xie, X.; Ma, L.; Juefei-Xu, F.; Xue, M.; Chen, H.; Liu, Y.; Zhao, J.; Li, B.; Yin, J.; and See, S. 2019b. DeepHunter: A Coverage-Guided Fuzz Testing Framework for Deep Neural Networks. In ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA).

Xie, X.; Ma, L.; Wang, H.; Li, Y.; Liu, Y.; and Li, X. 2019c. DiffChaser: Detecting Disagreements for Deep Neural Networks. In IJCAI, 5772–5778.

Yang, W.; Liu, J.; Yang, S.; and Guo, Z. 2019. Scale-free single image deraining via visibility-enhanced recurrent wavelet learning. IEEE Transactions on Image Processing 28(6): 2948–2961.

Yang, W.; Tan, R. T.; Feng, J.; Liu, J.; Guo, Z.; and Yan, S. 2017. Deep joint rain detection and removal from a single image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1357–1366.

Yang, W.; Tan, R. T.; Wang, S.; Fang, Y.; and Liu, J. 2020. Single image deraining: From model-based to data-driven and beyond. IEEE Transactions on Pattern Analysis and Machine Intelligence.

Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial examples: Attacks and defenses for deep learning. IEEE transactions on neural networks and learning systems 30(9): 2805–2824.

Zhang, H.; and Patel, V. M. 2018. Density-aware single image deraining using a multi-stream dense network. In Proceedings of the IEEE conference on computer vision and pattern recognition, 695–704.

Zhang, H.; Sindagi, V.; and Patel, V. M. 2019. Image de-raining using a conditional generative adversarial network. IEEE transactions on circuits and systems for video technology.

Zhang, M.; Zhang, Y.; Zhang, L.; Liu, C.; and Khurshid, S. 2018. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), 132–142. IEEE.