AdvRain: Adversarial Raindrops to Attack Camera-based Smart Vision Systems

Amira Guesmi, Muhammad Abdullah Hanif, and Muhammad Shafique

eBrain Laboratory, NYU Abu Dhabi, UAE

***Abstract*—Vision-based perception modules are increasingly deployed in many applications, especially autonomous vehicles and intelligent robots. These modules are being used to acquire information about the surroundings and identify obstacles. Hence, accurate detection and classification are essential to reach appropriate decisions and take appropriate and safe actions at all times. Current studies have demonstrated that "printed adversarial attacks", known as physical adversarial attacks, can successfully mislead perception models such as object detectors and image classifiers. However, most of these physical attacks are based on noticeable and eye-catching patterns for generated perturbations making them identifiable/detectable by human eye or in test drives. In this paper, we propose a camera-based inconspicuous adversarial attack (AdvRain) capable of fooling camera-based perception systems over all objects of the same class. Unlike mask based fake-weather attacks that require access to the underlying computing hardware or image memory, our attack is based on emulating the effects of a natural weather condition (i.e., Raindrops) that can be printed on a translucent sticker, which is externally placed over the lens of a camera. Note, such perturbations are still inconspicuous in real-world deployments and their presence goes unnoticed due to their association with a natural phenomenon (as also advocated in [1]). To accomplish this, we provide an iterative process based on performing a random search aiming to identify critical positions to make sure that the performed transformation is adversarial for a target classifier. Our transformation is based on blurring predefined parts of the captured image corresponding to the areas covered by the raindrop. We achieve a drop in average model accuracy of more than 45% and 40% on VGG19 for ImageNet and Resnet34 for Caltech-101, respectively, using only 20 raindrops.**

***Index Terms*—Adversarial machine learning, physical adversarial attack, Security, efficiency, perturbations, physical attacks.**

I. Introduction

The emergence of deep learning (DL) is creating disruptive transformations in a wide range of sectors especially autonomous driving [2]. For instance, leading manufacturers such as Google, Audi, BMW, and Tesla are striving to create autonomous vehicles (AVs) by combining this cutting-edge technology with low-cost cameras forming the vision-based perception modules. AVs are being increasingly equipped with these modules to address high-pressure real-life scenarios, reach suitable decisions, and take appropriate and safe actions. In fact, their incorporation increased product demand and helped the market of autonomous vehicles grow. According to the Strategic Market Research (SMR), the market for autonomous vehicles will reach \$196.97 billion by 2030 [3], growing at a CAGR of 25.7%.

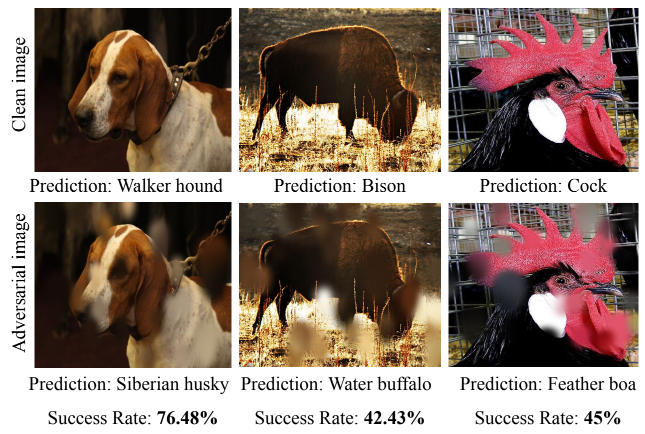

Fig. 1: Adversarial examples generated by **AdvRain** and their corresponding attack success rate when using only 10 raindrops.

However, these models suffer from vulnerabilities to adversarial attacks that threaten their integrity and trustworthiness. In particular, adversarial attacks that modify an input to a DL classifier with carefully crafted perturbations chosen by a malicious actor to force the classifier to a wrong output. If attackers are able to manipulate the decisions of an ML classifier to their advantage, they can jeopardize the security and integrity of the system, and even threaten the safety of people that it interacts with. For example, adding adversarial noise to a stop sign that leads an autonomous vehicle to wrongly classify it as a speed limit sign [4], [5] potentially leading to crashes and loss of life. In addition, adversarial examples have been shown effective in real-world conditions [6], that when printed out, an adversarially crafted image can remain adversarial to classifiers even under different lighting conditions and orientations. Therefore, understanding and mitigating these attacks is essential to developing safe and trustworthy intelligent systems.

We can distinguish two types of adversarial attacks: *Digital Attacks* [5], [7] and *Physical Attacks* [8] based on their different attack forms. In fact, in a digital attack, an attacker adds adversarial perturbation to the digital input image that is optimized to be imperceptible by a human observer. The noise imperceptibility in this case is constrained by a noise budget during the generation process. On the other hand, for physical attacks the attacker crafts adversarial perturbations that can be printed in the physical world and deployed in the scene that is captured by the victim model. Most of these adversarial perturbations are not generated under noise magnitude constraints but rather under location and printability constraints.

The main goal of an adversarial attack and its relevance to real-world scenarios is to go unnoticed, to be considered common or plausible rather than hostile. Most previous works developing adversarial patches for image classification mainly focus on the attack performance, and enhancing the strength of the adversarial noise. This leads to conspicuous patches which can be easily recognized by human observers. Another line of work tried to work on the stealthiness of the added perturbation by camouflaging into natural styles that appear legitimate to human observers such as shadows [9], color films [10], and laser beams [11], etc. In Figure 2, we present a visual comparison of AdvRain to existing physical attacks. While all the adversarial examples in Figure 2 successfully attack DNNs, we can see that AdvRain is able to generate adversarial perturbation with natural stains compared to a patch with conspicuous pattern generated by AdvPatch, or unrealistic patterns generated by FakeWeather [1].

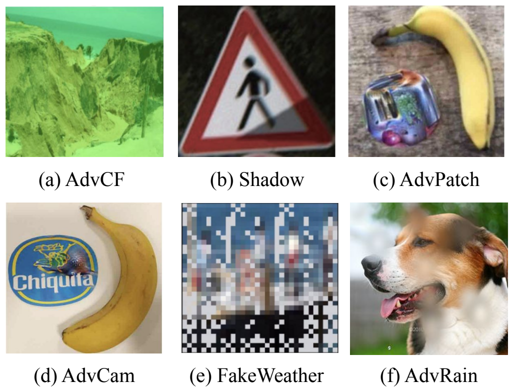

Fig. 2: AdvRain vs. existing physical-world attacks.

A. Proposed Technique and Concept Overview

In this paper, we present a technique for physically deceiving a deep learning model so that all objects belonging to a specific class are misclassified by introducing a subtle perturbation. To achieve this, we produce an adversarial camera sticker that can be attached to the camera's lens (See Figure 3). The sticker has a specially designed pattern of raindrops that misclassifies the images the camera perceives (See Figure 1). These patterns appear in the camera image as water drops, but they are not immediately apparent to someone looking at the image. In contrast to prior attacks, which operate at the pixel-level granularity of the images, the main challenge in developing this attack is that the space of feasible perturbations we can conceivably introduce with this model is very limited. In fact, the optics of the camera limit our ability to produce anything other than blurry dots, which lack the typical high frequency patterns of the traditional adversarial attacks.

In summary, the *contributions* of this work are:

- We propose a random search based optimization method to craft adversarial perturbations that will be introduced into the visual path between the camera and the object, without altering the object itself.
- We ensure an inconspicuous pattern associated with a natural phenomenon (i.e., raindrops) that can be printed on a translucent sticker affixed to the lens of the camera. In addition to being universal; the sticker will apply the same perturbation to all images from the same class.
- We evaluate the effectiveness of our proposed attack using two classifiers VGG-19 and Resnet34 trained with ImageNet and Caltech-101 datasets, respectively.
- We achieve a drop in accuracy of more than 45% and 40% on ImageNet and Caltech-101, respectively, using only 20 raindrops.
- We study the impact of blurring parts (i.e., adding low frequency patterns) of the image on model interpretability.

II. Background and Related Works

A. Camera-based Vision Systems

Environment perception is one of the main applications that drove tremendous effort in both industry and research community towards the development of promising DL-based solutions for applications like autonomous robots and intelligent transportation systems. One of the major challenges in building high-performance environment perception is designing reliable recognition systems. The automotive cameras and associated perception models, such as object detectors and image classifiers, serve as the base for the vision-based perception modules. These latter are employed to gather information about the environment and assist AVs in making decisions that are essential to safe driving. Designing dependable recognition systems is one of the biggest challenges in establishing such high-performance environment perception. However, it has been proven that these systems are susceptible to adversarial attacks.

Adversarial attacks can be categorized into digital and physical attacks.

B. Digital Adversarial Attacks

An attacker in a scenario involving a digital attack has the freedom to arbitrarily change the input image of a victim model at the pixel level. Therefore, these attacks assume implicitly that the attacker has access to the DNN's input system (e.g., a camera). The first adversarial example, which was based on injecting small, unnoticeable noise to attempt to shift the input image prediction in the direction of the wrong class, was put forth by [12]. The algorithms for creating adversarial examples have advanced as a result of a number

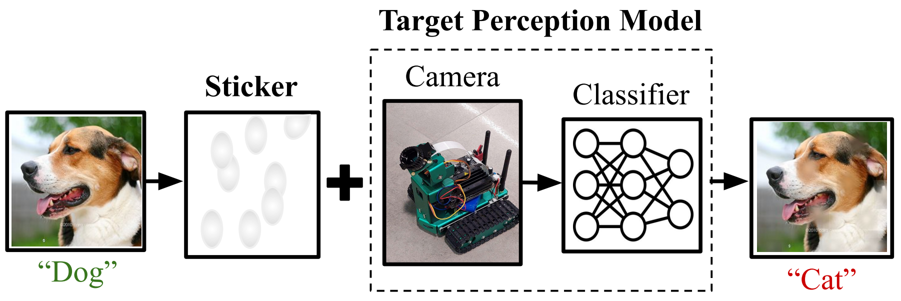

Fig. 3: **Attack threat model:** The generated pattern is printed on a translucent sticker placed over the lens of the camera. Hence, any captured image will contain the adversarial dots resulting in an inconspicuous natural looking adversarial image that fools the target model and the human eye.

of digital attacks that have been presented [5], [7], [13]–[15]. A realistic threat model would assume that the attacker is fully in control of the system's external environment or external objects, instead of the system's internal sensors and data pipelines. In what follows, we go through some state-of-the-art physical attacks on image classification.

C. Physical Adversarial Attacks

A physical attack is when the perturbations are added in the physical space. The process of crafting a physical perturbation can be summarised as follows: To ensure that the attack is still effective in a real-world scenario, the adversary first generates an adversarial perturbation in the digital space. Next, he attempts to produce the adversarial perturbation in the physical space. This adversarial perturbation is then perceived by sensors (such as cameras and radars) in the physical space, resulting in fooling the target model.

Various methods were proposed to add adversarial perturbations in different locations. We summarize existing methods into three categories: *attack by directly modifying the targeted object* (e.g., adversarial clothing [16]), *modifying the camera* (e.g., leveraging the Rolling Shutter Effect [17]) and *modifying the medium between the camera and the object* (e.g., light-based attacks [18]).

Another stream of work is based on creating adversarial perturbation that have natural styles to insure stealthiness and make the perturbation appear legitimate to human observers such as shadows [9], color films [10], and laser beams [11], etc.

III. Methodology

In this section, we present our methodology to design the adversarial camera sticker.

A. Threat Model for Physical Camera Sticker Attacks

Traditionally, the problem of generating an adversarial example is formulated as a constrained optimization (Eq.1), given an original input image x and a target classification model $f(.)$,:

$$\min_{\delta} \|\delta\|_p \, s.t. f(x+\delta) \neq f(x) \tag{1}$$

Algorithm 1 Random Search-based Optimization Method.

```
1:  Input: Number of drops: $n$, Radius of the drop: $r$, Number of iterations: $N$, Image hight: $imgH$, Image width: $imgW$, Dataset: $D$, Raindrop Generator: $G$, classifier $f$.
2:  Output: $Best$
3:  Initialize $Best \leftarrow 0$
4:  for $iter_k = 0 : N$ do
5:      $Acc(p_k) \leftarrow 0$
6:      for $x, y \in D$ do
7:          $p_k = RandomSolution(n, imgH, imgW)$
8:          $x_{adv} = G(x, p_k, n, r)$
9:          if $f(x_{adv})! = f(x)$ then
10:             $Acc(p_k) = Acc(p_k) + 1$
11:         end if
12:     end for
13:     if $Acc(p_k) < Acc(p_{k-1})$ then
14:         $Best \leftarrow p_k$
15:     end if
16: end for
```

Where the objective is to find a minimal inconspicuous universal perturbation, δ, such that when added to an arbitrary input from a target input domain D, it will cause the underlying DNN-based model $f(.)$ to misclassify. Note that one cannot find a closed form solution for this optimization problem since the DNN-based model $f(.)$ is a non-convex machine learning model, i.e., a deep neural network. Therefore, Eq.1 is formulated as follows to numerically solve the problem using empirical approximation techniques:

$$\arg\max_{\delta} \sum_{x \in \mathcal{D}} l(f(x+\delta), f(x)) \tag{2}$$

Where l is the DNN-based model loss function and $\mathcal{D} \subset D$ is the attacker's classifier training dataset. To solve this problem, existing optimization techniques (e.g., Adam [19]) can be used. In each iteration of the training the optimizer updates the adversarial noise δ.

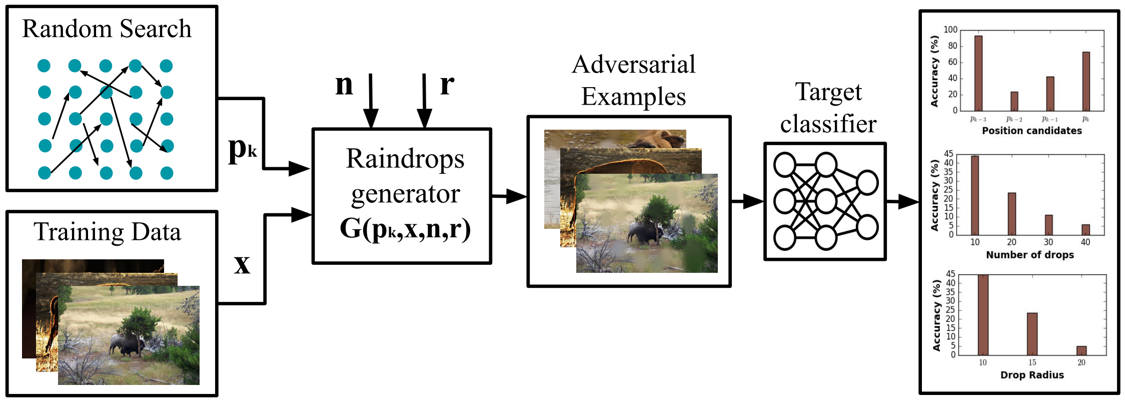

Fig. 4: Overview of the proposed approach; **Training phase:** we use random search based optimization method to determine the best raindrop positions that insure the highest attack success rate. The raindrop generator takes as inputs the number of drops n, the radius of the drop r, the positions p_k of the n drop for the k iteration, and the input image x. The output of the generator is the simulated raindrops added to the input image. This image is later on fed to the classifier to monitor its classification accuracy.

In contrast to these attacks, which operate at the pixel-level granularity of the images, the main challenge in our proposed threat model (i.e., physical camera sticker attacks) is that the space of feasible perturbations we can conceivably introduce is very constrained. In fact, the optics of the camera restrict our ability to produce anything other than blurry dots, which miss the high frequency patterns introduced by traditional adversarial attacks. In this work, we propose an approach to design the adversarial perturbation based on approximating the effect of placing small dots (raindrops) on a sticker. Owing to the optics of the camera lens, a small water drop placed upon the camera lens will create a small translucent patch on the image itself, representing the introduced low frequency perturbations. The adversarial example is crafted as follows:

$$x_{adv} = G(x, p, n, r) \tag{3}$$

Where G is the raindrop generator, p stands for the drops positions, n for number of drops, r for the drop radius. Technically, considering a 2D image x with $x(i,j)$ denoting the pixel at the (i,j) location. Our aim is to find the best position candidate $p(i,j)$ for the n raindrops, in a way that the model wrongly classifies the generated adversarial example.

$$p(i,j)?\ s.t.\ f(G(x, p(i,j), n, r)) \neq f(x) \tag{4}$$

B. Overview of the proposed approach

An overview of our proposed attack is shown in Figure 4. Our goal is to generate adversarial perturbations with patterns that resemble the effect of natural weather events. Hence, such patterns are crafted by faking that the camera lens has raindrops on its surface due to atmospheric conditions (i.e., Rain). For the training of the adversarial perturbation, we use a random search method to identify the best pattern of drops that achieves the highest success rate. The raindrop generator takes as inputs the number of drops n, the radius of the drop r, the positions p_k of the n drop for the k iteration, and the input image x. The output of the generator is the simulated raindrops applied to the input image.

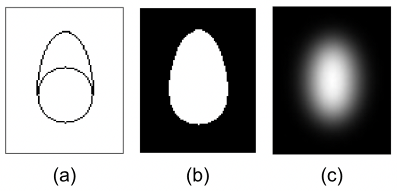

Fig. 5: Raindrop generation process: (a) The raindrop shape is formed using one circle and one oval. (b) The final shape. (c) We then create the effect of the water droplet surface through adding the blur effect.

C. Raindrop Generator

The drop generator applies a transformation to each pixel belonging to the area covered by the raindrop, this transformation corresponds to Gaussian blur, hence the transformation function is the following:

$$T_G(a,b) = \frac{1}{2\pi\sigma^2} \exp - \frac{a^2+b^2}{2\sigma^2} \tag{5}$$

where a is the distance from the origin in the horizontal axis, b is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution. To form our final perturbation model, we simply compose n of these dot perturbations.

To create the effect of the *raindrop surface*, in addition to the blur we add a fish-eye effect. The splashing of raindrops is simulated using collision detection, we check if the center of the drop is occupied or not, if yes, the raindrops are merged, otherwise, nothing is done. To simulate the *raindrop shape*,

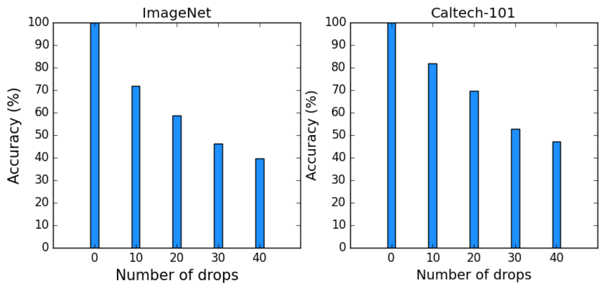

Fig. 6: Average Model Accuracy (radius = 10) for ImageNet and Caltech-101: The higher the number of introduced raindrops the higher the drop in classification accuracy.

we use one circle and one oval to make the shape (See Figure 5-a), the final shape is presented in Figure 5-b, we then create the effect of the water droplet surface through the blur (See Figure 5-c). The concept and the drop after blur are shown in Figure 5. The generated perturbation will be printed on a translucent sticker affixed to the lens of a camera.

D. *Identify critical positions*

The identification of critical positions is achieved using a random search-based optimization method (See Algorithm 1), where starting from random positions we try to select the best position candidate that results in minimal classification accuracy (i.e., the maximum attack success rate).

IV. Experiments

A. *Experimental Setup*

In this work, we simulated different shapes and sizes of raindrops and studied their impact on model classification accuracy. As victim classifiers, we use VGG-19 [20] and Resnet34 [21] with an input image resolution of 224×224. In our evaluations, we use the images of the ImageNet [22] and Caltech-101 [23] datasets. ImageNet is a large visual dataset formed by more than 14 million annotated images, containing more than 20K categories. Caltech-101 consists of images of objects belonging to 101 classes. The dataset is formed by around 9K images. Each class contains roughly 40 to 800 images. Images are of variable sizes, with typical edge lengths of 200-300 pixels. It's worth noting that for all the experiments the classification accuracy of the model when fed clean images was considered as the baseline classification accuracy.

B. *Evaluation of Attack Performance*

We use classification accuracy as our evaluation metric. First, we generate the adversarial perturbation for 10 different classes and we report the model accuracy for different numbers of drops. Our attack successfully dropped the classification accuracy. For instance, for only 10 raindrops, the accuracy of the model decreased by 30% and 20% for ImageNet and Caltech-101 datasets. For 20 raindrops, we had more than 40% and 30% for the two datasets, respectively, (See Figure 6).

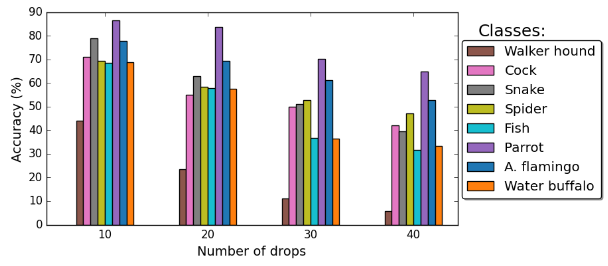

Fig. 7: Average Model Accuracy per class for different number of raindrops (radius = 10) for ImageNet and Caltech-101.

In Figure 7, we report the per-class classification accuracy for 8 different classes (i.e., Walker hound, Cock, Snake, Spider, Fish, Parrot, American flamingo, and Bison). The highest drop in accuracy was reported for the "Walker hound" class, this could be explained by the fact that ImageNet includes 120 categories of dog breeds, with close features, however for more distinguishable objects (i.e., incorporating distinctive features) such as the "Parrot" class and the "American flamingo", we notice a smaller drop, fooling these objects is more challenging.

C. *AdvRain Vs FakeWeather*

In this section, we compare AdvRain to the *fakeWeather* attack [1]. FakeWeather attack tries to emulate the rain, snow, and hail effects by designing three masks that fake the effect of such weather conditions on the camera lenses. The problem of this attack is that the generated noise is based on unrealistic and pixelated patterns. Also, the added perturbations cover most of the image (See Figure 8) and its effectiveness was only tested for small images of 32×32 pixels from the Cifar-10 dataset. In contrast, our proposed attack results in a realistic perturbation with better emulation of the rain effect, in addition to being proved effective on larger images with a size of 224×224 pixels.

D. *AdvRain Vs Natural rain*

We compare our adversarial rain (i.e., AdvRain) to natural rain. We simulated natural rain using randomly placed raindrops which didn't drastically degrade the performance of the victim model. For instance, different positions resulted in different accuracy drops. As shown in Figure 9, the left combination (totally random) resulted in 3% drop, however the carefully selected combination resulted in 60% drop.

V. Discussion

A. *Impact of drop radius*

We evaluate the impact of changing the radius of the drop (i.e., the blurred area) and we notice an increase in attack success rate, in fact the accuracy of the victim model drops an additional 25% when increasing the radius from 10 to 15 pixels for ImageNet dataset as shown in Figure 10.

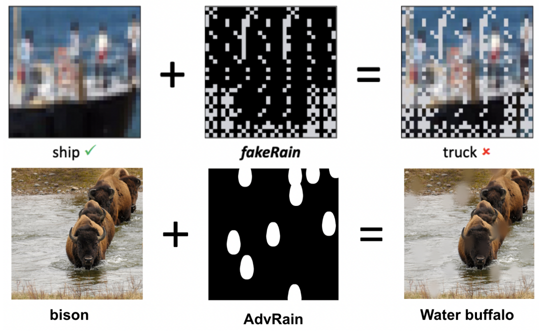

Fig. 8: AdvRain compared to FakeWeather attack. FakeWeather attack tries to emulate the rain effect by designing a mask that fakes the effect of such weather conditions on the camera lenses by changing the pixel values. However, the generated mask resulted in unrealistic and pixelated patterns. In contrast, AdvRain is based on generating more realistic raindrops simulation with a shape closer to that of a real raindrop.

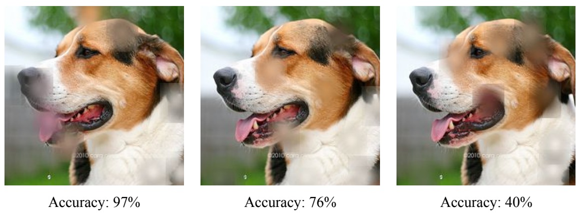

Fig. 9: Impact of AdvRain on classification accuracy compared to natural rain. Natural rain (Random positioning of the raindrops) resulted in 3% drop, however the carefully selected combination (AdvRain's output) resulted in 60% drop.

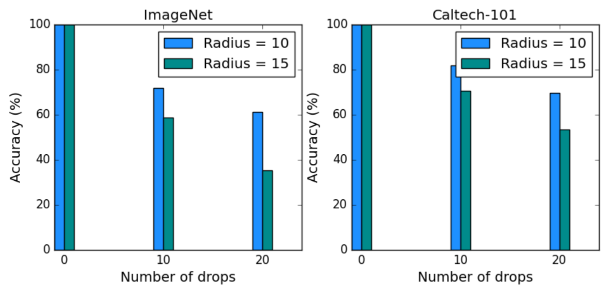

Fig. 10: Impact of drop radius on model accuracy for ImageNet and Caltech-101: The bigger the radius of the drop the higher drop in classification accuracy we achieve.

B. Impact on SSIM

In this section, we aim to assess the impact of the adversarial perturbation on the captured image using the structural similarity index measure (SSIM). Since our perturbation is based on blurring some regions of the image, we notice a relatively small impact on SSIM compared to traditional adversarial attacks (See Table I). For instance, the SSIM of the adversarial example compared to the clean image is 0.89 when using 10 raindrops.

TABLE I: SSIM of adversarial examples compared to the original images.

# of Drops	10	20	30	40
SSIM	0.89	0.78	0.73	0.69

C. Impact on Network Interpretation

Although it has been demonstrated that adversarial patches are quite powerful at causing misclassification, these patches are highlighted using standard network interpretation methods, thereby disclosing the identity of the adversary [24]. One of the most well-known network interpretation algorithms, Grad-CAM [25], outperforms other state-of-the-art interpretation algorithms on a sanity check. The Grad-CAM visualization results for traditional adversarial patch vs. AdvRain are evaluated using an ImageNet pretrained VGG-19 classifier (adding low vs high frequency patterns). Unlike patch-based attacks that shift the model's focus from the object to the location of the patch, making them detectable, our attack causes the model to overlook some important features that help the model make the decision, as shown in Figure 11.

VI. Conclusions

In this paper, we introduce a novel approach for producing realistic physical adversarial camera stickers for image classifiers. Our proposed attack (AdvRain) emulates the natural weather condition (Rain). In fact, we simulate the shape and the effect of the raindrop when placed on the camera lens. Our methodology is based on performing random search based optimization methods to identify the most effective raindrop positions. Our attack achieves a drop in average classification accuracy of more than 45% and 40% on VGG19 for ImageNet and Resnet34 for Caltech-101, respectively, using only 20 raindrops.

References

[1] A. Marchisio, G. Caramia, M. Martina, and M. Shafique, "fakeweather: Adversarial attacks for deep neural networks emulating weather conditions on the camera lens of autonomous systems," 2022. [Online]. Available: https://arxiv.org/abs/2205.13807

[2] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha, "Deep learning algorithm for autonomous driving using googlenet," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 89–96.

[3] "Autonomous car market size, industry growth — forcast 2020," https://www.strategicmarketresearch.com/market-report/autonomous-car-market, accessed: 2022-08.

[4] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 601–618. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[6] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *CoRR*, vol. abs/1607.02533, 2016. [Online]. Available: http://arxiv.org/abs/1607.02533

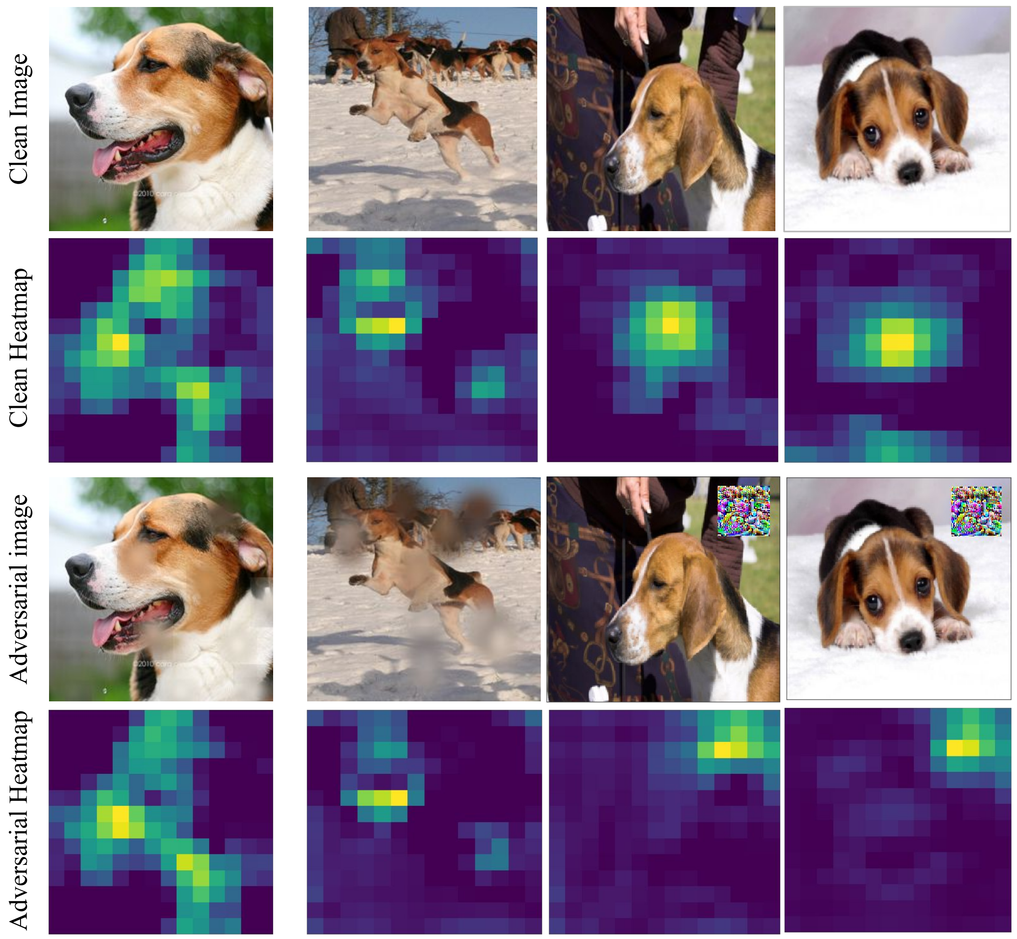

Fig. 11: Comparing the Grad-CAM visualization results for adversarial patch vs AdvRain.

[7] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," *CoRR*, vol. abs/1608.04644, 2016. [Online]. Available: http://arxiv.org/abs/1608.04644

[8] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *CoRR*, vol. abs/1607.02533, 2016. [Online]. Available: http://arxiv.org/abs/1607.02533

[9] C. Hu and W. Shi, "Adversarial color film: Effective physical-world attack to dnns," 2022. [Online]. Available: https://arxiv.org/abs/2209.02430

[10] Y. Zhong, X. Liu, D. Zhai, J. Jiang, and X. Ji, "Shadows can be dangerous: Stealthy and effective physical-world adversarial attack by natural phenomenon," 2022. [Online]. Available: https://arxiv.org/abs/2203.03818

[11] R. Duan, X. Ma, Y. Wang, J. Bailey, A. K. Qin, and Y. Yang, "Adversarial camouflage: Hiding physical-world attacks with natural styles," 2020. [Online]. Available: https://arxiv.org/abs/2003.08757

[12] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014. [Online]. Available: http://arxiv.org/abs/1312.6199

[13] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," 2017.

[14] N. Narodytska and S. P. Kasiviswanathan, "Simple black-box adversarial perturbations for deep networks," *CoRR*, vol. abs/1612.06299, 2016. [Online]. Available: http://arxiv.org/abs/1612.06299

[15] J. Chen and M. I. Jordan, "Boundary attack++: Query-efficient decision-based adversarial attack," *CoRR*, vol. abs/1904.02144, 2019. [Online]. Available: http://arxiv.org/abs/1904.02144

[16] Y.-C.-T. Hu, J.-C. Chen, B.-H. Kung, K.-L. Hua, and D. S. Tan, "Naturalistic physical adversarial patch for object detectors," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 7828–7837.

[17] A. Sayles, A. Hooda, M. Gupta, R. Chatterjee, and E. Fernandes, "Invisible perturbations: Physical adversarial examples exploiting the rolling shutter effect," *CoRR*, vol. abs/2011.13375, 2020. [Online]. Available: https://arxiv.org/abs/2011.13375

[18] A. Gnanasambandam, A. M. Sherman, and S. H. Chan, "Optical adversarial attack," *CoRR*, vol. abs/2108.06247, 2021. [Online]. Available: https://arxiv.org/abs/2108.06247

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition.* Ieee, 2009, pp. 248–255.

[23] F.-F. Li, M. Andreeto, M. Ranzato, and P. Perona, "Caltech 101," avr 2022.

[24] A. Subramanya, V. Pillai, and H. Pirsiavash, "Towards hiding adversarial examples from network interpretation," *CoRR*, vol. abs/1812.02843, 2018. [Online]. Available: http://arxiv.org/abs/1812.02843

[25] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell,

D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016. [Online]. Available: http://arxiv.org/abs/1610.02391