ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447287

SCALE-FREE AND TASK-GENERIC ATTACK: GENERATING PHOTO-REALISTIC ADVERSARIAL PATTERNS WITH PATCH QUILTING GENERATOR

Xiangbo Gao[1], Qinliang Lin[2], Cheng Luo[2], Weicheng Xie[2†], Linlin Shen[2], Keerthy Kusumam[3], Siyang Song[3†]

[1]University of California, Irvine, US. [2] Shenzhen University, China. [3] University of Leicester, UK.
† Corresponding authors: Siyang Song, ss1535@leicester.ac.uk, Weicheng Xie, wcxie@szu.edu.cn

ABSTRACT

Recent CNN generator-based attack approaches can synthesize unrestricted and semantically meaningful entities to the image, which are able to improve the transferability and robustness. However, such methods attack images by either synthesizing local adversarial entities, which are only suitable for attacking specific contents, or performing global attacks, which are only applicable to a specific image scale. In this paper, we propose a novel Patch Quilting Generative Adversarial Networks (PQ-GAN) to learn the first scale-free CNN generator that can be applied to attack images with arbitrary scales for various computer vision tasks. The principal investigation on transferability of the generated adversarial examples, robustness to defense frameworks, and visual quality assessment show that the proposed PQG-based attack framework outperforms the other nine state-of-the-art adversarial attack approaches when attacking the neural networks trained on two standard evaluation datasets (i.e., ImageNet and CityScapes). Our code is made available at `https://github.com/XiangboGaoBarry/PQAttack`.

***Index Terms*—** Generative adversarial attack, Scale-free generator, Task-generic attack, Adversarial patterns

1. INTRODUCTION

Deep Neural Networks (DNNs) are vulnerable to adversarial examples [1]. To develop strong adversarial defense algorithms [2–4], investigating robust adversarial attack algorithms to disrupt these defenses [5] is also crucial. Traditional image attack approaches [6–8] focus on generating perturbations at the pixel-level with L_p-norm restrictions, whose attack strengths are not easily transferred to unseen networks [2]. Subsequently, many studies devote their efforts to formulating methods that deliver more robust attack patterns against the defense algorithms, including adding noisy perturbation with weaker restriction [9,10] or even without restriction [11]. Despite that larger perturbation boosts up the transferability and robustness of attack algorithms, the perturbation is perceptible to human eyes and the adversarial examples are not photo-realistic enough.

To solve this problem, some studies employ generative adversarial networks (GAN) [12] to generate semantically meaningful local entities and synthesize them to the image [5, 13, 14] or to change the texture of a particular area of the image [15]. Such GAN-style methods can generate robust and transferable adversarial examples with high image quality. However, such methods are designed for a specific task, such as face recognition [13], and vehicle motion prediction [14] (**Problem 1**). Alternatively, instead of generating a local entity, others generate adversarial examples in an end-to-end manner where a global adversarial perturbation is carefully hidden in the target image [16, 17]. However, these methods can only generate adversarial examples of one particular or highly limited scale (**Problem 2**). The whole generative network must be re-trained when the target image scale is changed. Although some works can generate adversarial examples with global semantic patterns without using GAN [18–20], these methods can only generate a specific attack pattern defined by manually designed math formulations, which cannot be extended for generic usage.

In this paper, we propose a Patch Quilting Generative Adversarial Network (PQ-GAN) that can synthesize images of **any scale** without any distortion or discontinuity. It learns three cascaded generators that can synthesize photo-realistic, scale-free patterns to attack target images of any scale on the whole-image level (globally) (**addressing problem 2**). Importantly, the synthesized realistic and semantically meaningful pattern ensures the visual quality of the adversarial examples. Task generic property allows our approach to be applied to generate various photo-realistic patterns, e.g., rain streaks, snow flakes, and camera lens dirt, for various computer vision tasks such as image classification, object detection, instance segmentation, etc. (**addressing problem 1**). Moreover, our approach generates patterns with unrestricted pixel value, ensuring transferability and robustness.

The work was supported by the Natural Science Foundation of China under grants no. 62276170, 82261138629, the Science and Technology Project of Guangdong Province under grants no. 2023A1515011549, 2023A1515010688, the Science and Technology Innovation Commission of Shenzhen under grant no. JCYJ20220531101412030.

2. METHODOLOGY

2.1. Patch Quilting Generator-based image Attack

Given a target network of generic image analysis task with model parameters ϕ and a loss function $\mathcal{L}(\phi, I, y)$ used for model training, where y is the label of the benign image I, the goal is to find an adversarial example I^{adv} that maximizes $\mathcal{L}(\phi, I^{\text{adv}}, y)$ under the restriction, i.e. I^{adv} is perceptually natural. In particular, Fig. 1 illustrates our scale-generic generative model (whose pre-trained model parameters are represented as ψ'), namely Patch Quilting Generator (PQG). The PQG takes a set of latent embeddings $\mathcal{Z}$ initialized with Gaussian distribution of mean 0 and standard deviation 1 as the input, which controls the characteristic of the pattern and outputs a photo-realistic pattern P^I. Then, P^I is synthesized to the target image I to produce an adversarial example I^{adv} that is perceptually natural. Now the problem is transformed to find a latent embeddings $\mathcal{Z}$ that maximize $\mathcal{L}(\phi, I^{\text{adv}}, y)$ as:

$$\begin{aligned} &\text{MAX}_{\mathcal{Z}}\mathcal{L}(\phi, I^{\text{adv}}, y) \\ \textbf{Subject to}\quad &I^{\text{adv}} = \text{Syn}(I, P^I) \qquad P^I = \text{PQG}(\mathcal{Z}, \psi') \end{aligned} \tag{1}$$

Notice that PQG is a pre-trained model that does not need to be retrained in this attack stage. We explain the latent embeddings $\mathcal{Z}$ and how the PQG is designed to be scale-generic in detail in Sec. 2.2. $\text{Syn}(\cdot)$ is a customized synthesis function depending on the target pattern, (*i.e.* pixel-wise addition, or depth-aware synthesis [21]).

To achieve the adversarial objective, we simply apply gradient ascent to the loss function, to update the latent embeddings $\mathcal{Z}$ through an iterative process as:

$$\begin{aligned} \tilde{\mathcal{Z}}^{t+1} &\leftarrow \mathcal{Z}^t + \alpha \nabla_{\mathcal{Z}^t}\mathcal{L}(\phi, I^{\text{adv}}, y) \\ \mathcal{Z}^{t+1} &= \text{NORM}(\tilde{\mathcal{Z}^{t+1}}) \end{aligned} \tag{2}$$

where t is the time-stamp and α denotes the learning rate. NORM stands for the normalization of every vector in $\mathcal{Z}$ with mapping $\text{NORM}(\cdot) = (\cdot - \min)/(\max - \min)$. We put $\mathcal{Z}$ of the last iteration into PQG to generate the adversarial example.

2.2. Patch Quilting GAN

2.2.1 Scale-free Image Generation via PQG. The proposed Patch Quilting Generator (PQG) consists of three cascaded generators G_{PAT}, G_{HS}, and G_{VS} with the learnable weights ψ_{PAT}, ψ_{HS} and ψ_{VS}, respectively. Each generator can generate image patches of the scale $h \times w$. PQG takes three sets of latent embeddings $\mathcal{Z} = \{\mathcal{Z}_{\text{PAT}}, \mathcal{Z}_{\text{HS}}, \mathcal{Z}_{\text{VS}}\}$ as the input, and outputs a set of patches with desired attack pattern at the scale of $h \times w$. These patches can then be combined as a smooth and photo-realistic global attack pattern P^I whose scale can be customized based on the target image. As shown in Fig. 1, given a target image I with the scale of $H \times W$, our PQG first initializes an attack pattern $P^{\text{raw}} \in \mathbb{R}^{\hat{H} \times \hat{W}}$ consisting of a integral number of empty patches of size $h \times w$ as:

$$\hat{H} = N_h \times h, \quad \hat{W} = N_w \times w \tag{3}$$

where $N_h = \text{ceil}(\frac{H}{h})$, $N_w = \text{ceil}(\frac{W}{w})$ denote the minimum number of patches that are required to fill up each row and column, $\text{ceil}(\cdot)$ means rounding up to an integer. The three generators then generate a set of attack patches as follows.

Firstly, G_{PAT} takes a set of latent embeddings $\mathcal{Z}_{\text{PAT}}$ to generate a set of attack patches P_{PAT} to fill up non-adjacent odd rows and columns in the P^{raw}. By letting $N_h^{\text{PAT}} = \text{ceil}(\frac{N_h}{2})$ and $N_w^{\text{PAT}} = \text{ceil}(\frac{N_w}{2})$, this step can be formulated as:

$$\begin{aligned} &p^{\text{raw}}_{2a-1,2b-1} \in P_{\text{PAT}} = \text{G}_{\text{PAT}}(\mathcal{Z}_{\text{PAT}}, \psi_{\text{PAT}}) \\ \textbf{Subject to}\quad &\mathcal{Z}_{\text{PAT}} = \{z_{2a-1,2b-1} \in \mathcal{N}(0,1)^k\} \\ &a \in \{1, 2, \cdots, N_h^{\text{PAT}}\},\ b \in \{1, 2, \cdots, N_w^{\text{PAT}}\} \end{aligned} \tag{4}$$

where $p^{\text{raw}}_{2a-1,2b-1}$ denotes the image patch located at the $2a - 1_{\text{th}}$ row and $2b - 1_{\text{th}}$ column, while $z_{2a-1,2b-1} \in \mathcal{N}(0,1)^k$ denotes the latent embedding of dimension k being used to generate $p^{\text{raw}}_{2a-1,2b-1}$.

Secondly, G_{HS} generates a set of horizontal context-aware realistic adversarial attack patches P_{HS}, where each patch fills up a horizontal gap in P^{raw} based on not only Z_{HS} but also its horizontal neighbors in P^{raw}, which are generated from G_{PAT}. By letting $N_h^{\text{HS}} = \text{ceil}(\frac{N_h}{2})$ and $N_w^{\text{HS}} = \text{ceil}(\frac{N_w}{2}) - 1$, this process is formulated as:

$$\begin{aligned} &p^{\text{raw}}_{2a-1,2b} \in P_{\text{HS}} = \text{G}_{\text{HS}}(\mathcal{Z}_{\text{HS}}, p^{\text{raw}}_{2a-1,2b\pm1}, \psi_{\text{HS}}) \\ \textbf{Subject to}\quad &\mathcal{Z}_{\text{HS}} = \{z_{2a-1,2b} \in \mathcal{N}(0,1)^k\} \\ &a \in \{1, 2, \cdots, N_h^{\text{HS}}\},\ b \in \{1, 2, \cdots, N_w^{\text{HS}}\} \end{aligned} \tag{5}$$

Notice that $p^{\text{raw}}_{2a-1,2b\pm1} \in P_{\text{PAT}}$.

Finally, G_{VS} generates a set of vertical context-aware realistic adversarial attack patches P_{VS}, targeting filling up the rest of regions (all vertical gaps) in P^{raw}. Specifically, each patch generated by G_{VS} fills up a gap based on not only $\mathcal{Z}_{\text{VS}}$ but also its vertical neighbors in P^{raw}. By letting $N_h^{\text{VS}} = \text{ceil}(\frac{N_h}{2}) - 1$ and $N_w^{\text{VS}} = N_w$, this step is formulated as:

$$\begin{aligned} &p^{\text{raw}}_{2a,b} \in P_{\text{VS}} = \text{G}_{\text{VS}}(\mathcal{Z}_{\text{VS}}, p^{\text{raw}}_{2a\pm1,b}, \psi_{\text{VS}}) \\ \textbf{Subject to}\quad &\mathcal{Z}_{\text{VS}} = \{z_{2a,b} \in \mathcal{N}(0,1)^k\} \\ &a \in \{1, 2, \cdots, N_h^{\text{VS}}\},\ b \in \{1, 2, \cdots, N_w^{\text{VS}}\} \end{aligned} \tag{6}$$

Notice that $p^{\text{raw}}_{2a\pm1,b} \in P_{\text{PAT}} \bigcup P_{\text{HS}}$.

Consequently, a global pattern $\hat{P}^{\text{raw}}$ is obtained by filling all patches of the P^{raw}, where attack patches produced by three generators are concatenated. We then remove the extra pixels of the $\hat{P}^{\text{raw}} \in \mathbb{R}^{\hat{H} \times \hat{W}}$ to make it have the same size $H \times W$ to the target image I, which is denoted as the final P^I. In summary, the PQG can not only synthesize global image attack patterns of any required scale without requiring re-training networks but also allow the final produced pattern

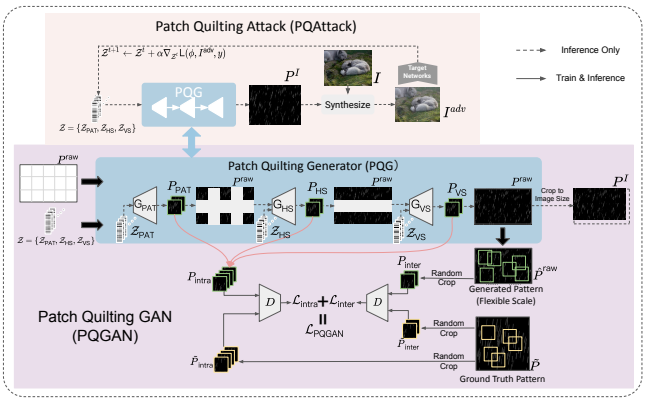

Fig. 1. Illustration of the Patch Quilting Attack pipeline (Top) and the training strategy of the Patch Quilting Generator (Bottom).

to be smooth, continuous, and semantically meaningful.

2.2.2 PQ-GAN Optimization: To learn three generators of the POG, we propose a GAN-style training strategy (called PQ-GAN). To produce a globally smooth and continuous attack pattern, we propose a **Intra-Patch Smoothness Loss** to ensure each generated attack patch is smooth and photo-realistic, and a **Inter-Patch Smoothness Loss** to enforce the smoothness between neighboring patches.

Intra-Patch Smoothness Loss: As displayed at the bottom of the Fig. 1, we collect all the patches $P_{\text{intra}} = P_{\text{PAT}} \bigcup P_{\text{HS}} \bigcup P_{\text{VS}}$ generated by the generators of POG, treating them as negative samples, while a set of positive samples $\tilde{P}_{\text{intra}} = \{\tilde{p}^m_{\text{intra}} | \ m = 1, 2, \cdots, M_{\text{intra}}\}$ are obtained by randomly cropping a set of ground truth patches of size $h \times w$ from the a ground truth Pattern $\tilde{P}$, where M_{intra} equals the number of patches in P_{intra}. All negative and positive samples are then fed into a discriminator D to calculate the generator and discriminator losses, respectively, based on the standard formulation of the Wasserstein GAN with gradient penalty [22]. This process is formulated as:

$$\begin{aligned}\mathcal{L}_{\text{intra}} =& \mathcal{L}_{\text{G}_{\text{PAT}}}(P_{\text{intra}}, \psi_{\text{PAT}}) + \mathcal{L}_{\text{G}_{\text{HS}}}(P_{\text{intra}}, \psi_{\text{HS}}) + \mathcal{L}_{\text{G}_{\text{VS}}}(P_{\text{intra}}, \psi_{\text{VS}}) + \\ & \mathcal{L}_{\text{D}}(P_{\text{intra}}, \psi_{\text{D}}) + \mathcal{L}_{\text{D}}(\tilde{P}_{\text{intra}}, \psi_{\text{D}}) \end{aligned} \tag{7}$$

where $\mathcal{L}_{\text{G}_{\text{PAT}}}(P_{\text{intra}}, \psi_{\text{PAT}})$, $\mathcal{L}_{\text{G}_{\text{HS}}}(P_{\text{intra}}, \psi_{\text{HS}})$, and $\mathcal{L}_{\text{G}_{\text{VS}}}(P_{\text{intra}}, \psi_{\text{VS}})$ denote the generator losses specific to the patches $P_{\text{PAT}}, P_{\text{HS}}$, and P_{VS}, respectively. $\mathcal{L}_{\text{D}}(P_{\text{intra}}, \psi_{\text{D}})$ denotes the discriminator loss of negative samples P_{intra}, and $\mathcal{L}_{\text{D}}(\tilde{P}_{\text{intra}}, \psi_{\text{D}})$ denotes the discriminator loss of positive samples $\tilde{P}_{\text{intra}}$.

Inter-Patch Smoothness Loss: To ensure the **smoothness and continuity** among neighboring patches, we additionally randomly crop M_{inter} patches $P_{\text{inter}} = \{p^m_{\text{inter}} | \ m = 1, 2, \cdots, M_{\text{inter}}\}$ of size $h \times w$ from the generated attack pattern $\hat{P}^{\text{raw}}$ and treat them as negative samples, where M_{inter} is a hyper-parameter. To balance between the positive and negative samples, we then randomly crop the same number of patches $\tilde{P}_{\text{inter}} = \{\tilde{p}^m_{\text{inter}} | \ m = 1, 2, \cdots, M_{\text{inter}}\}$ from the ground truth pattern $\tilde{P}$, and feed P^{inter} and $\tilde{P}^{\text{inter}}$ into D to compute the loss as:

$$\mathcal{L}_{\text{inter}} = \mathcal{L}_{\text{D}}(P_{\text{inter}}, \psi_{\text{D}}) + \mathcal{L}_{\text{D}}(\tilde{P}_{\text{inter}}, \psi_{\text{D}}) \tag{8}$$

where $\mathcal{L}_{\text{D}}(P_{\text{inter}}, \psi_{\text{D}})$ and $\mathcal{L}_{\text{D}}(\tilde{P}_{\text{inter}}, \psi_{\text{D}})$ denote the discriminator losses specific to the negative samples P^{inter} and positive samples $\tilde{P}_{\text{inter}}$, respectively. For the details of these losses, please refer to Arjovsky *et al*. [22].

Then, the final loss for training PQ-GAN is obtained by combining Inter- and intra-Patch Smoothness losses as:

$$\mathcal{L}_{\text{PQGAN}} = \mathcal{L}_{\text{intra}} + \mathcal{L}_{\text{inter}} \tag{9}$$

The combined loss would enforce three generators to be jointly learned for generating a smooth and continuous global attack pattern of any scale.

3. EXPERIMENTS AND RESULTS

Experimental setup: The image classification experiments are conducted on 5,000 randomly selected images from the validation set of the ImageNet, while the object detection and instance segmentation experiments are conducted on the entire validation set of the CityScapes dataset. Classification accuracy and mean average precision (mAP) are the main evaluation metrics, while lower classification accuracy or mAP indicates better attack performance.

Results of Cross-model Transferability. In our experiment, two classifiers including ResNet-18 and VGG-19 are employed. We additionally compare our method with two unrestricted local attack methods, i.e. ColorFool [18] and Shadow Attack [5], as well as two unrestricted global attack methods: Adversarial Vignetting Attack (AVA) [20] and Adversarial Haze [19]. As shown in Table 1, when our approach attacks ResNet-18 with the Rain Drops attack pattern and transfers to VGG-19, the classification accuracy is largely decreased from 68.3% to 49.1%, while the second best decreases the corresponding performance to 54.5%.

We also evaluate the cross-model transferability on the Object Detection and Instance Segmentation tasks. For object detection, we use Faster-RCNN [23] as the target network and transfer the adversarial examples to YOLOv3 [24] and Deformable DETR [25]. For Instance Segmentation, we choose Mask-RCNN [26] as the target network and transfer the adversarial examples to PointRend [27] and SOLO [28]. Tab. 2 shows that our method results in a larger decrease in mAP compared to other methods while maintaining its attack strength on unknown networks.

Results of Robustness. To evaluate the robustness of our method against defense models, three denoise-based defense algorithms, i.e. JPEG Compression, High-Frequency Suppression [2], HGD [3], and two additional generation-based defense algorithms, i.e. Ape-GAN [32], Defense-GAN [33] are used for the evaluation. Tab. 3 reports the experimental results on: attacking image classification. It shows that

Table 1. The classification accuracy of two well-trained models on both clean and adversarial examples, with * indicating the result of the white box attack. The "ATK Region" column indicates the area of the image that is attacked.

Model	ATK Region	Methods	ResNet-18	VGG-19
		None	67.3	68.3
ResNet-18	Local	Shadow ATK [5]	16.7*(50.6↓)	56.4 (11.9↓)
		ColorFool [18]	9.3*(58.0↓)	55.2 (13.1↓)
	Global	IadvHaze [19]	21.4*(55.9↓)	54.5 (13.8↓)
		RA-AVA [20]	4.2*(63.1↓)	55.5 (12.8↓)
		Rain Streaks (Ours)	3.6*(63.7↓)	52.2 (16.1↓)
		Lens Dirt (Ours)	4.3*(63.0↓)	50.0 (18.3↓)
		Snow Flakes (Ours)	2.5*(64.8↓)	51.0 (17.3↓)
		Rain Drops (Ours)	2.1*(65.2↓)	**49.1** (**19.2**↓)
VGG-19	Local	Shadow ATK [5]	54.6 (12.7↓)	14.8*(53.5↓)
		ColorFool [18]	53.4 (13.9↓)	10.1* (58.2↓)
	Global	IadvHaze [19]	55.1 (12.2↓)	25.1* (43.2↓)
		RA-AVA [20]	51.3 (16.0↓)	5.1* (63.2↓)
		Rain Streaks (Ours)	50.5 (16.8↓)	3.4* (64.9↓)
		Lens Dirt (Ours)	49.5 (17.8↓)	4.3* (64.0↓)
		Snow Flakes (Ours)	**49.3** (**18.0**↓)	2.1*(66.2↓)
		Rain Drops (Ours)	49.8 (17.5↓)	2.7* (65.6↓)

Table 2. The mAP values of cross-model transferability on the object detection and instance segmentation tasks. The white box attack results are marked with $\star$. **Left:** Object Detection; **Right:** Instance Segmentation

Methods	mAP %↓			mAP %↓		
	FR-RCNN	YOLO	DETR	Mk-RCNN	PtRend	SOLO
clean	40.3	32.8	46.6	36.4	37.1	34.9
DPatch	8.8*	12.3	20.2	-	-	-
AdvPatch	5.5*	12.5	15.4	-	-	-
UAP	12.1*	11.5	13.5	6.3*	9.7	8.5
Rain Streaks	**4.2***	7.8	**9.2**	**2.1***	7.8	**7.3**
Lens Dirt	5.3*	10.2	12.4	3.0*	10.1	9.4
Snow Flakes	4.9*	8.3	9.7	2.4*	**7.5**	8.3
Rain Drops	5.8*	**7.7**	11.0	2.5*	8.5	8.8

Table 3. Classification accuracy results on adversarial examples generated by attack methods under defense algorithms.

Attack \ Defense	NONE	JPEG	HFC	HGD	APE	DEF-GAN
IadvHaze [19]	21.4	45.5	46.3	35.7	41.4	35.7
RA-AVA [20]	4.2	40.2	41.6	43.9	48.9	42.2
ColorFool [18]	9.3	10.4	11.9	12.4	25.1	40.1
Shadow [5]	16.7	18.1	18.5	21.0	22.7	33.4
Rain Streaks	3.6	12.1	11.5	13.2	15.9	26.5
Lens Dirt	4.3	10.3	12.8	12.2	21.9	28.0
Snow Flakes	2.5	9.9	13.3	12.5	**14.5**	23.4
Rain Drops	2.3	**9.7**	**10.7**	**12.1**	13.6	**25.8**

the denoise-based defense algorithms are not effective against our unrestricted attacks. For example, our adversarial examples with raindrop patterns can reduce the classification accuracy by 8.4%, compared to the SOTA Shadow [5] under JPEG Compression.

Quantitative Evaluation of Image Quality. We employ three reference-free image quality assessment metrics: NIQE [29], BRISQUE [30], and PIQE [31] to quantify the image quality. We compare the average scores of the adversarial examples generated by different attack methods in Tab. 4. Tab. 4 demonstrates that the adversarial examples produced by our approach have the best image quality among all adversarial examples in terms of the employed three metrics.

Table 4. Classification accuracy of three non-reference image quality assessment metrics NIQE [29], BRISQUE [30], and PIQE [31] being evaluated on the adversarial examples generated by eight attack approaches, where the pre-selected 5,000 images from the ImageNet dataset are used.

ATK	BRISQUE ↓	NIQE ↓	PIQE ↓
Clean	22.4207	3.6792	32.1288
IadvHaze [19]	41.6842	12.7820	72.2287
RA-AVA [20]	33.2389	9.3145	32.8531
ColorFool [18]	29.4896	5.1896	31.3078
Shadow [5]	30.7310	5.1603	30.9756
RS (Ours)	30.3656	5.6763	**30.5022**
LD (Ours)	**27.3893**	6.2135	31.8128
SF (Ours)	32.6419	**4.8950**	33.2251
RD (Ours)	31.2009	6.5481	32.9371

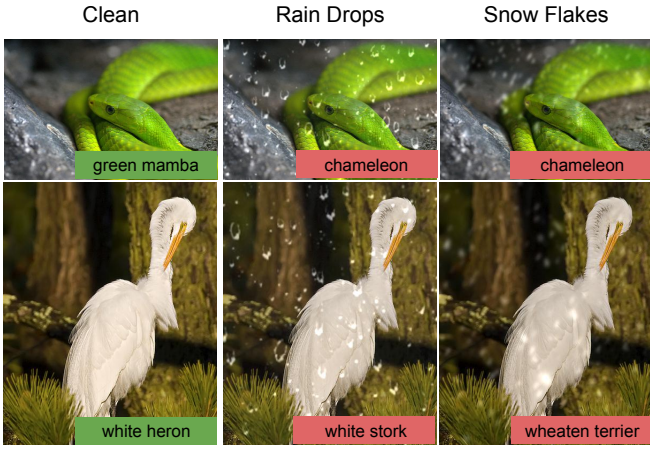

Fig. 2. The generated adversarial examples for the images in the ImageNet dataset, where the prediction label is displayed in the bottom right corner of each image.

Visual Evaluation of Image Quality. Fig. 2 provides a visualization of the adversarial examples generated by our method. Our PQG guarantees the photo-realistic quality of the generated patterns, while the PQAttack pipeline endows them with attack strength. These results demonstrate that our adversarial examples preserve a high level of photorealism, making them indistinguishable from the human eye.

4. CONCLUSION

This paper proposed a novel adversarial image attack approach, which can train a generator to produce photo-realistic and task-generic patterns to attack images of different scales. Results demonstrate that our PQG outperforms state-of-the-art methods in misleading both white-box and black-box computer vision models.

5. REFERENCES

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[2] Z. Zhang, C. Jung, and X. Liang, "Adversarial defense by suppressing high-frequency components," in *IJCAI*, 2019.

[3] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *CVPR*, 2018, pp. 1778–1787.

[4] Z. Li, H. Cheng, X. Cai, J. Zhao, and Q. Zhang, "Saes: Subspace activation evolution strategy for black-box adversarial attacks," *IEEE TETCI*, 2022.

[5] 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," in *CVPR*, 2022, pp. 15345–15354.

[6] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE S&P*. IEEE, 2017, pp. 39–57.

[7] R. Wiyatno and A. Xu, "Maximal jacobian-based saliency map attack," in *Montréal Artificial Intelligence Symposium*, 2018.

[8] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE TEC*, vol. 23, no. 5, pp. 828–841, 2019.

[9] K. R. Mopuri, A. Ganeshan, and R. V. Babu, "Generalizable data-free objective for crafting universal adversarial perturbations," *IEEE TPAMI*, vol. 41, no. 10, pp. 2452–2465, 2018.

[10] C. Sun, Y. Zhang, W. Chaoqun, Q. Wang, Y. Li, T. Liu, B. Han, and X. Tian, "Towards lightweight black-box attacks against deep neural networks," *arXiv preprint arXiv:2209.14826*, 2022.

[11] Y. Chen, X. Mao, Y. He, H. Xue, C. Li, Y. Dong, Q.-A. Fu, X. Yang, W. Xiang, T. Pang, et al., "Unrestricted adversarial attacks on imagenet competition," in *CVPR*, 2021.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *NIPS*, vol. 27, 2014.

[13] S. Komkov and A. Petiushko, "Advhat: Real-world adversarial attack on arcface face id system," in *ICPR*. IEEE, 2021, pp. 819–826.

[14] Z. Kong, J. Guo, A. Li, and C. Liu, "Physgan: Generating physical-world-resilient adversarial examples for autonomous driving," in *CVPR*, 2020, pp. 14254–14263.

[15] Z. Hu, S. Huang, X. Zhu, F. Sun, B. Zhang, and X. Hu, "Adversarial texture for fooling person detectors in the physical world," in *CVPR*, 2022, pp. 13307–13316.

[16] T. Bai, J. Zhao, J. Zhu, S. Han, J. Chen, B. Li, and A. Kot, "Ai-gan: Attack-inspired generation of adversarial examples," in *ICIP*. IEEE, 2021, pp. 2543–2547.

[17] M. Naseer, S. Khan, M. Hayat, F. S. Khan, and F. Porikli, "On generating transferable targeted perturbations," in *CVPR*, 2021, pp. 7708–7717.

[18] A. S. Shamsabadi, R. Sanchez-Matilla, and A. Cavallaro, "Colorfool: Semantic adversarial colorization," in *CVPR*, 2020, pp. 1151–1160.

[19] R. Gao, Q. Guo, F. Juefei-Xu, H. Yu, and W. Feng, "Advhaze: Adversarial haze attack," *arXiv preprint arXiv:2104.13673*, 2021.

[20] B. Tian, F. Juefei-Xu, Q. Guo, X. Xie, X. Li, and Y. Liu, "Ava: Adversarial vignetting attack against visual recognition," in *IJCAI*, 2021.

[21] X. Hu, C.-W. Fu, L. Zhu, and P.-A. Heng, "Depth-attentional features for single-image rain removal," in *CVPR*, 2019, pp. 8022–8031.

[22] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *ICML*. PMLR, 2017, pp. 214–223.

[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *NIPS*, vol. 28, 2015.

[24] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[25] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," in *ICLR*, 2021.

[26] K. He, G. Gkioxari, P. Doll'ar, and R. Girshick, "Mask r-cnn," in *ICCV*, 2017, pp. 2961–2969.

[27] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *CVPR*, 2020, pp. 9799–9808.

[28] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "Solo: Segmenting objects by locations," in *ECCV*. Springer, 2020, pp. 649–665.

[29] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal processing letters*, vol. 20, no. 3, pp. 209–212, 2012.

[30] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE TIP*, vol. 21, no. 12, pp. 4695–4708, 2012.

[31] N. Venkatanath, D. Praneeth, M. C. Bh, S. S. Channappayya, and S. S. Medasani, "Blind image quality evaluation using perception based features," in *NCC*. IEEE, 2015, pp. 1–6.

[32] S. Shen, G. Jin, K. Gao, and Y. Zhang, "Ape-gan: Adversarial perturbation elimination with gan," *arXiv preprint arXiv:1707.05474*, 2017.

[33] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," in *ICLR*, 2018.