PAR-AdvGAN: Improving Adversarial Attack Capability with Progressive Auto-Regression AdvGAN

Jiayu Zhang[1], Zhiyu Zhu[2], Xinyi Wang[3], Silin Liao[4], Zhibo Jin[2], Flora Salim[5], and Huaming Chen[6]✉

[1] Suzhou University of Technology, China
zjy@szut.edu.cn
[2] University of Technology Sydney, Australia
{zhiyu.zhu, zhibo.jin}@student.uts.edu.au
[3] University of Malaya, Malaysia
22103906@siswa.um.edu.my
[4] Nanning Normal University, China
anivie@email.nnnu.edu.cn
[5] University of New South Wales, Australia
flora.salim@unsw.edu.au
[6] The University of Sydney, Australia
huaming.chen@sydney.edu.au

Abstract. Deep neural networks have demonstrated remarkable performance across various domains. However, they are vulnerable to adversarial examples, which can lead to erroneous predictions. Generative Adversarial Networks (GANs) can leverage the generators and discriminators model to quickly produce high-quality adversarial examples. Since both modules train in a competitive and simultaneous manner, GAN-based algorithms like AdvGAN can generate adversarial examples with better transferability compared to traditional methods. However, the generation of perturbations is usually limited to a single iteration, preventing these examples from fully exploiting the potential of the methods. To tackle this issue, we introduce a novel approach named Progressive Auto-Regression AdvGAN (PAR-AdvGAN). It incorporates an auto-regressive iteration mechanism within a progressive generation network to craft adversarial examples with enhanced attack capability. We thoroughly evaluate our PAR-AdvGAN method with a large-scale experiment, demonstrating its superior performance over various state-of-the-art black-box adversarial attacks, as well as the original AdvGAN.Moreover, PAR-AdvGAN significantly accelerates the adversarial example generation, i.e., achieving the speeds of up to 335.5 frames per second on Inception-v3 model, outperforming the gradient-based transferable attack algorithms. Our code is available at: `https://github.com/LMBTough/PAR`

1 Introduction

Deep neural networks (DNNs) are widely used in different real-world applications, i.e., image classification [20], emotional analysis [29], and item recommendations [27]. DNNs demonstrate human-surpassed performance when properly trained. However, DNNs can be vulnerable to adversarial examples crafted by attackers [5,24,32], which is a concern in safety-critical scenarios. Thus, a practical approach is to develop effective attack algorithms that can assess the robustness of DNNs against adversarial attacks at an early stage, ultimately enhancing model safety.

Currently, both white-box and black-box attack algorithms, such as gradient-based methods like FGSM [10], NAA [38], SSA [23], and optimization-based approaches such as PGD [25] and C&W [1], require continuous computation of the model's gradient information throughout the attack process. However, they all require extensive running time. Generative Adversarial Networks (GANs) [9] have demonstrated promising results for realistic sample generation by leveraging both generator and discriminator for training [2,4,11,17]. While the generator constructs high-quality examples, a discriminator learn to distinguish the original and generated examples. Furthermore, once the generator is trained, there will be no additional gradient computation for input examples.

As an early GAN-based model, AdvGAN incorporates a perturbation, denoted as $G(x)$, into the original image instance x for attack [34]. AdvGAN aims to obtain the manipulated image $x + G(x)$ from the original instance x through the discriminator. To achieve high attack success rates in both white-box and black-box attacks, AdvGAN introduces an adversarial loss on top of GANs loss, ensuring the adversarial image is generated in a direction more effective for adversarial attacks. Additionally, it employs hinge loss to limit perturbation range, thereby preventing significant deviations between the adversarial and original images. Subsequently, AdvGAN++ [15] further enhances the attack success rate by utilizing latent features instead of input image instances x. It optimizes the latent features during adversarial examples generation.

However, GAN-based methods suffer from several challenges. Both AdvGAN and AdvGAN++ generate perturbations in a single iteration, which limits their control over these perturbations. We observe the reason may be the generator continuously increases the perturbations during the iterative process (as illustrated in the **Appendix 1.1**). This may not be effective against adversarial defenses and impacts the attack capability. It is critical since the goal is to maximize attack effectiveness with minimal perturbation. Furthermore, the transferability performance of such attacks is concerning, especially since internal model information is typically unavailable in real-world scenarios.

Inspired by recent works that utilise auto-regressive properties to generate realistic images or text [2, 26, 37], we propose a novel GAN-based algorithm, Progressive Auto-Regression AdvGAN (PAR-AdvGAN) to generate adversarial examples with enhanced transferability. PAR-AdvGAN employs a progressive, auto-regressive iterative method to effectively capture the specific structures of input examples. This process gradually generates more diverse and realistic adversarial examples. Specifically, at time step t, we combine the input examples x_{adv}^{t-1} from time step $t-1$ with the initial examples x_0 to generate the perturbation

$G(x_{adv}^{t-1}, x_0)$ at time step t. Consequently, the manipulated examples shifts from $x + G(x)$ in the original AdvGAN to $x_{adv}^{t-1} + G(x_{adv}^{t-1}, x_0)$ in PAR-AdvGAN.

To achieve optimal performance with minimal perturbation, we propose L_p loss to limit the perturbation range during iteration, thereby ensuring that the adversarial examples remain imperceptible to human. Furthermore, to enhance the quality of adversarial examples and mitigate potential distortions and significant noise during the generation process, we introduce L_d loss, imposing a stringent constraint between the final adversarial examples x_t and initial examples x_0. Finally, by independently optimising the generator and discriminator during training, we fine-tune the parameters of PAR-AdvGAN for more effective adversarial examples. Notably, owing to the high stealth and robust generalization capabilities, non-targeted adversarial attacks subject models to more rigorous evaluations and reveal more subtle vulnerabilities. Thus, we primarily focus on non-targeted adversarial attacks. We summarise the contributions as follows:

- We empirically study the limited transferability of adversarial examples generated by existing GAN-based algorithms. To address this, we explore the use of progressive generator network to enhance transferability.
- We propose an auto-regression iterative method and provide theoretical analysis on formulating L_p and L_d loss to ensure minimal distortions in the adversarial samples.
- Extensive experiments demonstrate that our PAR-AdvGAN significantly outperforms other methods, achieving highest attack success rates. Moreover, it outperforms traditional gradient-based transferable attack algorithms in both transferability and attack speed. We release the code of PAR-AdvGAN for future research development.

2 Related Work

2.1 Adversarial Attacks

While numerous adversarial algorithms are dedicated to generating high-quality and robust adversarial samples, gradient-based attack algorithms constitute a main type. FGSM [10] was the first to utilise the model's gradients, which adds a small perturbation to the input data in the direction of the gradient, thereby maximising the loss function through gradient ascent to achieve optimal attack performance. MI-FGSM [6] incorporates a momentum factor in each iteration to mitigate the impact of local optima on the attack success rate. TI-FGSM [7] employs shifted images to calculate the input gradient, a process that involves convolving the original image's input gradient with a kernel matrix.

Other adversarial attack algorithms, such as PGD [25], project samples onto suitable attack directions and limit the size of perturbations to generate robust adversarial examples. C&W method minimises the attack's objective function to optimise the generation process [1]. AdvGAN [34] employs an adversarial training process between the generator and discriminator. This process bolsters the generator's ability to produce adversarial samples, making them challenging for the discriminator to distinguish from genuine data. Besides attack purpose, we also note some other iterative training methods for GANs, such as the Progressive GAN [16] which divides the Generator into several layers, with each layer undergoing individual training. In our approach, we consider an auto-regression methods, where each subsequent generation is based on the results of previous step. Although auto-regression GAN has been improved for continuous generation tasks, all we need is the attack result of the last state, so we need to redesign it for this situation.

2.2 Adversarial Defenses

Adversarial defense represents an effective approach to mitigate the impact of attacks on DNNs. Commonly used adversarial defense techniques include denoising and adversarial training. The denoising technique employs preprocessing mechanisms to filter out adversarial examples, thereby preventing the poisoning of training data and reducing the likelihood of subsequent attacks on the model. Other notable works include HRGD [21], R&P [35] and so on [3, 8].

Adversarial training enhances model robustness by incorporating adversarial examples into the training process. Ensemble adversarial training [12] works by decoupling the target model from adversarial examples generated by other black-box models, thereby defending against transferable attacks. To enhance the robustness of our algorithm against adversarial defenses, we validated the attack effectiveness of PAR-AdvGAN on the target model subjected to ensemble adversarial training.

3 Methodology

In this section, we first provide the problem definition of adversarial attacks. Then, we discuss the issue of perturbation escalation in AdvGAN and propose three strategies to optimise the generator, aiming to generate highly transferable adversarial samples. Finally, we provide a detailed implementation for the proposed PAR-AdvGAN method.

3.1 Problem Definition of Adversarial Attacks

Consider a clean data distribution p_{data} in which benign samples are represented by $X \subseteq p_{data}$. In an untargeted attack, the network f is misled by the manipulated sample x_{adv}. For the original sample $x \in X$, with the original label denoted as m, the adversarial goal can be defined as:

$$f(x_{adv}) \neq m \tag{1}$$

$$\|x_{adv} - x\|_n \leq \epsilon \tag{2}$$

where $\|\cdot\|_n$ represents the n-order norm (e.g., L_2 norm), and ϵ denotes the maximum perturbation.

3.2 Perturbation escalation in AdvGAN

AdvGAN adopts a non-repetitive iterative approach to improve attack performance. However, as iterations progress, the perturbation magnitude of the adversarial example increases rapidly. This issue arises because it treats each iterated example as an independent instance, neglecting its relation to the initial sample x_0. Additionally, AdvGAN fails to impose constraints on the distance between the generated samples and the initial samples. This suggests that the perturbations generated in each iteration will have significant magnitudes. To address this issue, we introduce three propositions:

Proposition 1. *The generator should obtain information about the original sample x_0.*

Proposition 2. *To train the generative model, the inputs to the generator should include a significant number of non-initial samples, particularly those encountered during the adversarial process.*

Proposition 3. *The generator should enforce constraints on the distance between adversarial samples and the initial sample x_0 throughout the iterative progress.*

3.3 Progressive Auto-Regression AdvGAN

In this section, we first introduce the solutions for three propositions, namely progressive generator network, auto-regression iterative method, and generator constraints. Next, we explain the training processes for both the discriminator and the generator. Finally, as shown in Algorithm. 1, we provided the pseudo-code for the PAR-AdvGAN approach.

Progressive Generator Network For Proposition 1, we adjust the generator to include initial example x_0 as an input, resulting in a revised generator $G(x^t_{adv}, x_0)$. To do this, we expand the channel dimension of the generator's first layer, and employ a concat operator to merge x^t_{adv} and x_0 along the channel dimension. This design enables the generator to leverage information from both the current adversarial example x^t_{adv} and initial input x_0, thus facilitating the generation of incremental adversarial perturbations during the iterative process (refer to line 5 in Alg. 1).

Auto-Regression Iterative Method In Proposition 2, during each training iteration, we utilise a hyperparameter T to regulate the number of interactions for x^t_{adv} instances (refer to line 4). We iteratively generate x^t_{adv} by adding perturbations $G(x^{t-1}_{adv}, x_0)$ to the preceding x^{t-1}_{adv} (see line 6), then use the resulting gradient progression to update and train the generator.

$$\triangledown\theta = \frac{\partial L_{adv}}{\partial x + G(x)} \cdot \underbrace{\frac{\partial x + G(x)}{\partial G(x)}}_{1} \cdot \frac{\partial G(x)}{\partial \theta} \tag{3}$$

Algorithm 1 Progressive Auto-Regression AdvGAN (PAR-AdvGAN)

```
Input: iteration number $T$, batch size $n$, progressive generator $G$, discriminator $D$, target network $N$, corresponding label $m$, learning rate $\eta_1$, $\eta_2$, weight hyper-parameter $\lambda_1, \lambda_2, \lambda_3$
Output: $\theta_D$, $\theta_G$
 1: for $i$ in (range) epoch do
 2:   Sample a mini-batch of $n$ examples
      $x = \{x(1), ..., x(n)\}$;
 3:   $x_0 = x$
 4:   for $t = 1, ..., T$ do
 5:     $P_t = G(x_{adv}^{t-1}, x_0)$, here $x_{adv}^0 = x_0$
 6:     $x_{adv}^t = clip(x_{adv}^{t-1} + P_t)$
 7:     for $k = 1, ..., S_d$ do
 8:       $g_{\theta_D} = \nabla_{\theta_D} L_D$
 9:       Update the discriminator by descending its stochastic gradient $S_d$ times:
10:       $\theta_D = \theta_D - \eta_1 \cdot g_{\theta_D}$
11:     end for
12:     for $k = 1, ..., S_g$ do
13:       $L_{adv} = -Cross\ Entropy(x_{adv}^t, m)$
14:       $L_p = \|P_t\|_2$
15:       $L_d = \|x_{adv}^t - x_0\|_2$
16:       $L_G = (1 - D(x_{adv}^t))^2$
17:       $g_{\theta_G} = \nabla_{\theta_G}(L_G + \lambda_1 L_p + \lambda_2 L_d + \lambda_3 L_{adv})$
18:       Update the generator by descending its stochastic gradient $S_g$ times:
19:       $\theta_G = \theta_G - \eta_2 \cdot g_{\theta_G}$
20:     end for
21:   end for
22: end for
23: return $\theta_D$, $\theta_G$
```

To better understand the auto-regression iterative progress, we decompose $\bigtriangledown_\theta$ in Eq. 3. Here, $\frac{\partial x+G(x)}{\partial G(x)}$ equals 1, so it can be omitted (See **Appendix 1.3** for detailed proof). Following this, we further explore the relationship between $\frac{\partial G(x)}{\partial \theta}$ and $\frac{\partial L_{adv}}{\partial x+G(x)}$. Thus, $\frac{\partial G(x)}{\partial \theta}$ represents the degree of change in $G(x)$ with respect to changing in θ. $\frac{\partial L_{adv}}{\partial x+G(x)}$ can be interpreted as the degree of change in L_{adv} when changing $x + G(x)$. Therefore, we can interpret the gradient ascent process of the parameter θ as a modification of θ to drive $x + G(x)$ able to obtain a better adversarial effect. At this point, to enable G to iteratively generate perturbations, we will replace x with x_{adv}^t. This transforms the first part of Eq. 3 into $\frac{\partial L_{adv}}{\partial x_{adv}^t+G(x_{adv}^t)}$, indicating that G continues to generate perturbations based on x_{adv}^t.

Given a network N that accurately maps image x sampled from the distribution p_{data} to its corresponding label m. The adversarial sample x_{adv}^t at time t can be expressed as:

$$P_t = G(x_{adv}^{t-1}, x_0) \tag{4}$$

$$x_{adv}^{t} = clip(x_{adv}^{t-1} + P_t) \tag{5}$$

such that

$$N(x_{adv}^{t}) \neq m \tag{6}$$

Here, P_t is the perturbation at time t, thus we have $x_{adv}^{1} = x_0 + P_1$. And $G(\cdot, x_0)$ is the progressive generator network.

Training of the Discriminator We train the discriminator to accurately distinguish adversarial samples generated by the progressive generator and actual samples from the data distribution p_{data}. Specifically, we fix the parameters related to the progressive generator and trained the discriminator S_d times (line 7). The loss function L_D can be written as:

$$L_D = (1 - D(x))^2 + D(x_{adv}^{t})^2 \tag{7}$$

It is worth noting that we did not choose to calculate L_D in the form of $log(1 - D(x))$ as in AdvGAN. This is because we find that the gradient of $log(1 - D(x))$ for D will be very large and not smooth when $D(x)$ is close to 1, and gradient explosion will occur during iteration. We employ a squared form in the Eq. 10 to help mitigate this issue.

Hence, the gradient of the discriminator with respect to the parameters θ_D can be expressed using Eq. 8 (line 8):

$$g_{\theta_D} = \nabla_{\theta_D} L_D \tag{8}$$

By updating θ_D through gradient descent, we finally obtain the optimal parameters for the discriminator (line 9-10):

$$\theta_D = \theta_D - \eta_1 \cdot g_{\theta_D} \tag{9}$$

Here η_1 is the learning rate in discriminator training.

Constraints on the Generator We propose the use of L_{adv} to measure whether the adversarial samples are generated in a direction more conducive to the attack (refer to line 13).

$$L_{adv} = -Cross\ Entropy\ (x_{adv}^{t}, m) \tag{10}$$

It is worth noting that, in untargeted attacks, a larger value of *cross entropy* (x_{adv}^{t}, m) indicates a more effective adversarial example. Consequently, to enhance the adversarial nature of x_{adv}^{t} during gradient descent on L_{adv}, we prepend a negative sign to *cross entropy*(x_{adv}^{t}, m). It is also feasible to replace *cross entropy* with the loss function used in C&W [1].

To prevent the issue of perturbation explosion in the auto-regression iterative process, we introduce L_p to constrain the magnitude of perturbation (refer to line 14), where $\|\cdot\|_2$ stands for the l_2 norm:

$$L_p = \|P_t\|_2 = \left\|G(x_{adv}^{t-1}, x_0)\right\|_2 \tag{11}$$

To fulfill Proposition 3, we introduce an additional loss function L_d that enforces the generated adversarial examples to remain close to the initial example. This constraint ensures that the iterative progress of generating adversarial perturbations does not deviate significantly from the original input, thus maintaining the adversarial samples' proximity to the initial data (see line 15).

$$L_d = \left\| x_{adv}^t - x_0 \right\|_2 \tag{12}$$

Training of the Progressive Generator We train the progressive generator to generate adversarial samples with high transferability and low distortions from original samples while attacking the target neural network N. Specifically, we fixed the parameters related to the discriminator and trained the progressive generator S_g times (refer to line 12).

As shown in Eq. 14, we computed the gradient of the progressive generator with respect to the parameters θ_G (line 16):

$$L_G = (1 - D(x_{adv}^t))^2 \tag{13}$$

$$g_{\theta_G} = \nabla_{\theta_G}(L_G + \lambda_1 L_p + \lambda_2 L_d + \lambda_3 L_{adv}) \tag{14}$$

Note that L_G is the loss function to deceive the discriminator and λ_1, λ_2, λ_3 are the weight hyper-parameters that control the balance between loss functions. By updating θ_G through gradient descent, we ultimately obtain the optimal parameters for the progressive generator (line 17-18):

$$\theta_G = \theta_G - \eta_2 \cdot g_{\theta_G} \tag{15}$$

Here η_2 is the learning rate in discriminator training.

4 Experiments

In this section, we present the experiments conducted to evaluate the performance of our method. To guide the analysis, we address the following research questions.

- What is the attack success rate of PAR-AdvGAN compared to the baseline AdvGAN? (**RQ1**)
- How does PAR-AdvGAN's performance in attack transferability and attack speed compare to state-of-the-art methods in adversarial attacks? Is it effective? (**RQ2**)
- Why does PAR-AdvGAN work effectively? (**RQ3**)

4.1 Experiment Setup

Dataset and Models We conducted the experiments on the ImageNet-compatible dataset consisting of 1000 images with a resolution of 299×299×3 [28] [7]. The dataset generation process follows the literature [6, 7]

[7] `https://github.com/cleverhans-lab/cleverhans/tree/master/cleverhans_v3.1.0/examples/nips17_adversarial_competition/dataset`

Here we refer to the typical and state-of-the-art transferable adversarial attack methods [6, 7, 22, 34, 36, 38]. To ensure experiment fairness, we selected representative models from two types: normally-trained and defense-trained models. The normally trained models include Inceptionv3 (Inc-v3) [31], Inception-v4 (Inc-v4) [30], Inception-ResNet-v2 (IncRes-v2) [30], ResNet-v2-50 (Res-50) [13, 14], ResNet-v2-101 (Res-101) [13, 14], and ResNet-v2-152 (Res-152) [13, 14]. As for the defense-trained models through ensemble adversarial training, we selected Inc-v3ens3 [33], Inc-v3ens4 [33], and IncResv2ens [33].

Baseline Methods We employ the original AdvGAN [34] algorithm as our baseline to validate the transferability performance by incorporating self-regressive iteration in PAR-AdvGAN. Meanwhile, to evaluate our proposed PAR-AdvGAN, we selected seven state-of-the-art black-box adversarial attack methods as our competitive baselines, including FGSM [10], BIM [19], PGD [25], DI-FGSM [36], TI-FGSM [7], MI-FGSM [6], and SINI-FGSM [22].

Parameter Settings All experiments in this study are conducted using the Nvidia RTX 6000 Ada 48GB. In all experiments, we set the following fixed parameters for each algorithm according to the settings in [18]. For AdvGAN and PAR-AdvGAN, the training epochs are set to 60. The initial learning rate for both the Generator and Discriminator is set to 0.001, which is then reduced to 0.0001 at the 50th epoch. For DI-FGSM, we set the decay to 0, the resize_rate to 0.9, and the diversity_prob to 0.5. For TI-FGSM, decay is set to 0, kernel_name is set to "gaussian," len_kernel is set to 15, resize_rate is set to 0.9, and the diversity_prob is set to 0.5. For MI-FGSM, decay is set to 1. For SINI-FGSM, decay is set to 1, and m is set to 5.

Metrics Attack success rate (ASR) is a metric to evaluate the transferability of attacks. It quantifies the average proportion of mislabeled samples among all generated samples after the attack. Thus, a higher attack success rate signifies better transferability. Additionally, we use Frames Per Second (FPS) to assess the attack speed. Another crucial measure, the perturbation rate, is utilised to ensure that the adversarial images do not largely diverge from the original images in visual perception. A low value of this rate suggests that the adversarial examples maintain close visual fidelity to their originals. Detailed formulas are in the **Appendix 1.4**.

4.2 RQ1: Attacking Performance

As shown in Table. 1, we compare the attack success rates of the original AdvGAN and our proposed PAR-AdvGAN at three different perturbation rates of 8, 9, and 10. The comparisons are conducted using Inc-v3 and Inc-v4 as surrogate models and attacks are lunched on IncRes-v2. The results indicate that in most cases, our algorithm outperforms AdvGAN in terms of attack success rate.

Model	Attack	IncRes-v2
Inc-v3	AdvGAN	13.9/38.9/43.2
	PAR-AdvGAN	**27.1/35.2/41.4**
Inc-v4	AdvGAN	6.1/9.6/15.9
	PAR-AdvGAN	**21.4/30.8/34.7**

Table 1: ASR (%) of AdvGAN and PAR-AdvGAN on IncRes-v2. The adversarial examples are crafted on Inc-v3 and Inc-v4.

Table 2: The attack success rates (%) on four undefended models and three adversarial trained models by various transferable adversarial attacks. The adversarial examples are crafted on Inc-v3 with different perturbations. The best results are in bold.

Model	Attack	Perturbation	Inc-v4	Res-50	Res-101	Res-152	Inc-v3ens3	Inc-v3ens4	IncResv2ens	mASR
Inc-v3	AdvGAN	8.41/9.88/10.26	32.9/61.9/65.9	42.6/77.7/82.8	47.8/75.3/83.1	38.8/66.6/72.8	15.1/44.0/52.5	30.2/54.5/63.0	9.9/25.7/26.5	31.04/57.95/63.80
	PAR-AdvGAN	**8.29/9.27/9.95**	53.7/63.1/68.4	**74.8/85.0/89.8**	**79.3/86.2/89.5**	**68.2/78.0/82.5**	**39.4/53.5/63.8**	**45.7/60.8/69.4**	**28.0/39.0/46.5**	**55.58/66.51/72.84**
	FGSM	8.79/9.74/10.69	26.2/28.0/30.3	26.2/29.0/30.6	23.3/25.8/27.6	22.8/24.1/27.0	13.8/14.5/15.5	14.0/14.3/14.3	6.0/6.1/6.1	18.9/20.25/21.62
	BIM	8.46/9.50/9.96	47.6/52.2/56.6	42.1/45.8/48.5	36.7/40.6/42.9	35.6/39.2/39.3	14.4/16.0/15.8	14.1/14.6/14.5	8.5/8.1/8.4	28.42/30.92/32.28
	PGD	8.76/9.79/10.35	44.6/46.2/50.2	38.6/40.7/43.5	33.1/35.5/37.3	28.9/34.5/35.8	12.4/14.2/14.4	12.4/13.3/13.1	6.8/7.4/7.7	25.25/27.40/28.85
	DI-FGSM	8.51/9.56/10.02	**66.0/70.0/70.9**	54.0/60.0/61.4	50.7/55.1/55.2	49.3/53.7/52.7	19.1/19.6/20.0	18.4/20.4/19.1	10.5/11.0/11.0	38.28/41.40/41.47
	TI-FGSM	8.60/9.65/10.10	52.4/55.3/56.5	39.7/45.2/45.2	34.4/39.9/41.1	34.8/39.0/43.1	31.6/34.8/35.3	34.1/37.7/39.2	21.4/25.9/25.8	35.48/39.68/40.88
	MI-FGSM	8.98/9.77/10.55	44.5/47.9/51.4	39.9/41.7/45.1	36.5/38.8/41.0	34.3/37.8/38.9	16.3/17.1/16.8	15.6/14.9/16.2	6.5/7.7/7.4	27.65/29.41/30.97
	SINI-FGSM	8.99/9.77/10.55	56.0/59.7/64.2	53.8/58.0/62.5	47.2/51.1/55.2	45.6/50.7/53.8	24.6/24.4/26.4	23.9/23.8/25.9	11.0/12.0/12.6	37.44/39.95/42.94

We can see that compared to the most representative AdvGAN algorithm, PAR-AdvGAN has made significant improvements for attacking performance at a low perturbation rate. Specifically, similar to AdvGAN, PAR-AdvGAN, as a generative model, does not require additional gradient calculations based on different input data after training the generator. Compared to traditional gradient-based black-box transferable attack methods, it possesses faster attack speed. Therefore, we consider the PAR-AdvGAN algorithm feasible and suitable for attack scenarios that demand high transferability and fast generation of adversarial samples.

4.3 Effectiveness Experiment for RQ2: Transferability and Attack Speed

To validate the transferability and attack speed of PAR-AdvGAN compared to other SOTA methods, we conduct the experiments using various attack methods on Inc-v3, Inc-v4, and IncRes-v2 as source models to generate adversarial samples. We then conduct transferable attacks on different target models and use ASR and FPS as the main metrics, to validate the effectiveness of our algorithm.

Experiments on Inc-v3 As shown in Table. 2, we conduct attacks using Inc-v3 as the source model with three different perturbation rates on target models of Inc-v4, Res-50, Res-101, Res-152, Inc-v3ens3, Inc-v3ens4, and IncRes-v2. We can see that our PAR-AdvGAN algorithm has achieved an average increase of 30.3% in attack success rate compared to other baselines. Moreover, despite DI-FGSM achieving better performance than PAR-AdvGAN on Inc-v4, which may be

Table 3: The attack success rates (%) on four undefended models and three adversarial trained models by various transferable adversarial attacks. The adversarial examples are crafted on Inc-v4 with different perturbations. The best results are in bold.

Model	Attack	Perturbation	Inc-v3	Res-50	Res-101	Res-152	Inc-v3ens3	Inc-v3ens4	IncResv2ens	mASR
	AdvGAN	9.64/11.67/12.11	55.6/75.9/59.1	48.6/77.5/70.6	54.4/74.8/69.5	40.4/67.0/58.4	13.4/26.5/24.4	20.3/54.6/43.0	16.0/25.7/21.1	35.52/57.42/49.44
	PAR-AdvGAN	**9.55/11.05/11.60**	**72.9/85.6/87.8**	**66.1/79.6/85.0**	**73.7/86.4/89.8**	**55.7/69.2/74.8**	13.0/17.4/20.2	**35.0/50.9/58.2**	15.7/25.6/30.2	**47.44/59.24/63.71**
	FGSM	9.74/11.64/12.58	28.4/31.9/32.9	23.7/26.3/28.4	21.1/24.2/25.4	20.6/24.3/25.6	13.1/13.2/13.4	10.9/11.7/12.3	5.9/6.6/6.8	17.67/19.74/20.68
	BIM	9.98/11.46/11.91	60.1/62.5/62.4	42.5/45.8/46.2	38.9/40.2/41.0	34.7/39.8/39.8	13.9/15.7/16.1	13.5/15.7/14.8	9.3/10.9/9.6	30.41/32.94/32.84
Inc-v4	PGD	9.78/11.35/11.88	49.8/55.6/58.8	35.7/38.8/40.7	28.2/36.4/36.2	28.6/32.7/34.1	12.4/14.9/14.5	12.7/13.8/14.1	7.8/7.8/8.3	25.02/28.57/29.52
	DI-FGSM	10.01/11.49/11.94	75.4/80.4/80.3	57.6/63.7/62.9	51.2/57.5/56.5	49.9/55.0/55.7	18.5/19.6/20.6	16.4/18.5/18.7	10.7/13.5/12.4	39.95/44.02/43.87
	TI-FGSM	9.61/11.08/12.00	61.5/67.5/68.4	41.6/50.0/52.4	37.0/44.0/48.0	38.6/44.7/49.3	**33.6/38.5/39.7**	34.8/39.3/40.3	**24.1/28.5/32.0**	38.74/44.64/47.15
	MI-FGSM	9.81/11.42/12.22	53.2/61.2/61.7	40.2/43.2/47.0	36.7/41.6/43.9	34.0/39.8/41.3	15.0/15.6/16.4	14.6/15.3/15.0	6.2/7.9/7.6	28.55/32.08/33.27
	SINI-FGSM	9.79/11.39/12.18	75.1/78.2/80.1	63.1/69.0/70.6	58.3/65.9/66.7	56.9/62.6/64.8	27.8/31.1/32.1	26.4/28.8/29.9	14.3/16.6/17.4	45.98/50.31/51.65

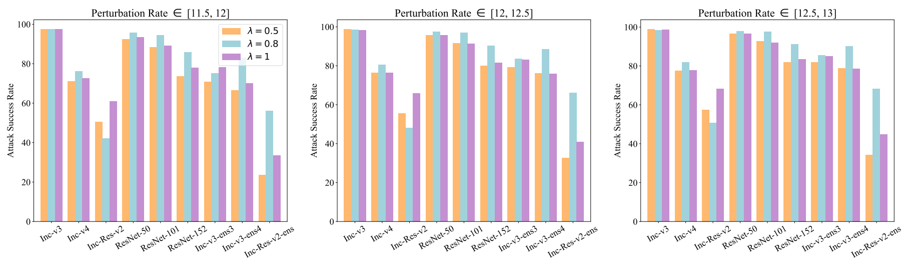

Fig. 1: The performance of PAR-AdvGAN at different high perturbation rate intervals

attributed to the randomness in model training, a comprehensive comparison across all models reveals that the attack success rate of PAR-AdvGAN is elevated by 24.6% compared to the best-performing competing baseline, DI-FGSM.

Experiments on Inc-v4 As shown in Table. 3, we conduct attacks using Inc-v4 as the source model with three different perturbation rates on target models of Inc-v3, Res-50, Res-101, Res-152, Inc-v3ens3, Inc-v3ens4, and IncRes-v2. We can see that our PAR-AdvGAN algorithm has achieved an average increase of 20.13% in attack success rate compared to other baselines. Furthermore, compared to the best performing SINI-FGSM among competitive baselines, PAR-AdvGAN achieved an increase of 7.48% in ASR.

Experiments on IncRes-v2 In this section, we conduct transferability tests on Inc-v3, Inc-v4, Res-50, Res-101, Res-152, Inc-v3ens3, Inc-v3ens4, and IncRes-v2 as target models with three different perturbation rates using IncRes-v2 as the source model. We have included the results in the Table 4. The results demonstrate that PAR-AdvGAN achieves an average increase of 14.96% in ASR compared to other baselines. We can see that although PAR-AdvGAN achieves a lower ASR of 0.02% than the best performing SINI-FGSM among competitive baselines, it outperforms AdvGAN by 6.31%.

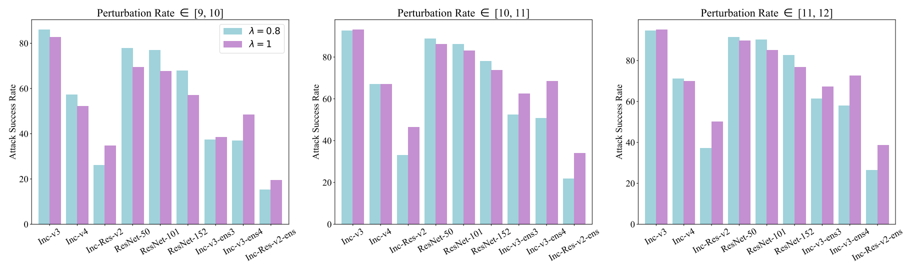

Fig. 2: The performance of PAR-AdvGAN at different low perturbation rate intervals

Table 4: The attack success rates (%) on four undefended models and three adversarial trained models by various transferable adversarial attacks. The adversarial examples are crafted on IncRes-v2. The best results are in bold.

Model	Attack	Inc-v3	Inc-v4	Res-50	Res-101	Res-152	Inc-v3ens3	Inc-v3ens4	IncResv2ens	mASR
IncRes-v2	AdvGAN	55.6/66.6/88.3	48.9/55.7/87.4	48.8/57.8/93.4	42.2/49.8/91.9	35.8/45.2/90.8	12.0/17.4/52.5	14.0/15.4/45.2	5.6/7.4/40.3	32.86/39.41/73.72
	PAR-AdvGAN	66.8/70.3/71.9	**71.2/75.1/76.4**	**77.6/80.3/82.0**	63.2/67.4/69.5	**66.0/70.8/73.0**	31.1/33.7/35.8	25.3/27.8/29.9	16.5/18.7/19.2	52.21/55.51/57.21
	FGSM	23.3/26.2/27.4	17.5/18.6/19.4	20.5/21.7/23.2	18.1/19.1/20.5	18.5/19.7/20.1	11.3/11.4/11.3	10.4/10.9/11.1	6.1/6.3/6.1	15.71/16.73/17.38
	BIM	58.4/62.0/63.8	47.1/51.8/40.7	41.9/46.4/47.3	36.6/39.4/42.2	35.6/38.7/41.8	14.9/14.7/15.2	12.8/13.8/14.6	9.9/10.8/11.2	32.15/34.70/35.85
	PGD	51.7/55.9/55.2	40.6/42.2/44.9	35.4/38.0/39.3	31.3/34.0/34.2	29.3/31.4/31.8	14.4/14.2/13.2	12.2/13.1/14.2	7.7/8.4/9.3	27.82/29.65/30.26
	DI-FGSM	**79.5/79.0/79.3**	69.7/72.0/74.8	61.8/61.7/67.2	58.3/59.6/62.5	57.2/56.9/59.9	20.9/20.5/22.2	18.9/20.3/20.9	14.8/15.1/14.7	47.63/48.13/50.18
	TI-FGSM	66.6/69.3/69.1	63.6/65.1/65.1	53.1/55.7/57.7	50.5/50.3/53.4	47.6/51.7/51.6	**43.3/45.2/46.5**	**44.0/44.8/48.8**	**39.5/41.1/40.9**	51.02/52.90/54.13
	MI-FGSM	58.2/59.8/61.7	49.8/50.1/55.7	43.7/46.1/48.4	39.2/44.7/43.3	37.6/39.5/42.6	15.0/16.1/17.4	14.8/15.4/15.3	8.9/9.4/9.8	33.40/35.13/36.77
	SINI-FGSM	76.7/79.3/80.6	70.1/73.8/75.7	66.1/70.4/73.0	**63.8/66.7/71.0**	60.9/63.9/66.6	34.7/35.2/37.0	29.5/29.9/31.4	20.6/20.7/22.4	**52.80/54.98/57.21**

Experiments on ResNet-50 As shown in Table 5, we conduct attacks using ResNet-50 as the source model with three different perturbation rates on target models of Inc-v3, Inc-v4, Res-101, Res-152, Inc-v3ens3, Inc-v3ens4, and IncRes-v2. The experimental results demonstrate that PAR-AdvGAN consistently achieves superior performance across all target models compared to other baseline methods. Specifically, PAR-AdvGAN achieves the highest mASR of 48.2%, 55.64%, and 63.81% for the three perturbation rates, outperforming the best-performing baseline DI-FGSM by 5.21%, 10.43%, and 15.45%, respectively. Furthermore, PAR-AdvGAN shows significant improvements over AdvGAN, with an average increase in attack success rate of 25.37%. Although DI-FGSM achieves competitive

Table 5: The attack success rates (%) on four undefended models and three adversarial trained models by various transferable adversarial attacks. The adversarial examples are crafted on ResNet-50 with different perturbations. The best results are in bold.

Model	Attack	Perturbation	Inc-v3	Inc-v4	Res-101	Res-152	Inc-v3ens3	Inc-v3ens4	IncResv2ens	mASR
ResNet-50	AdvGAN	8.32/9.15/10.40	28.2/38.5/27.5	31.7/33.5/21.6	30.3/34.5/26.9	29.4/32.4/19.7	14/23.9/7.3	16.7/19.9/12.8	9.5/10.9/5.3	22.83/27.66/17.3
	PAR-AdvGAN	**8.24/9.10/10.37**	**67.2/72.5/79.3**	43.7/48.8/55.1	**69.8/76.1/82.4**	52.7/61.9/69.6	**36.3/46.6/57.3**	**42.3/51.6/60.1**	**25.4/32/42.9**	**48.2/55.64/63.81**
	FGSM	8.79/9.74/10.69	27/29.6/31.6	22.4/24.5/26	24.9/27.3/29.4	24.7/27/28.6	11.9/12.9/13.6	12.3/12/12.2	5.7/5.9/6.3	18.41/19.89/21.1
	BIM	8.50/9.54/10.42	41.5/45.6/49.9	35.7/38.6/42.3	37/39.5/41.6	34/37.9/40.9	13.5/14/14.5	12/13.1/13.2	8.1/8/8.6	25.97/28.1/30.14
	PGD	8.34/9.37/10.46	30.8/35.6/39.2	26.4/30.3/33.1	26.9/29/32.5	24.5/27.9/31.5	11.6/11.6/13.1	10.4/11.8/12.6	6/6/7.5	19.51/21.74/24.21
	DI-FGSM	8.59/9.17/10.52	67.2/71.1/74.8	**65.5/68.8/73.1**	63.8/69.1/72.4	**62.2/64.9/70.6**	16.8/17/19	15/15/16.9	10.4/10.6/11.7	42.99/45.21/48.36
	TI-FGSM	8.45/9.46/10.38	51/54.9/60.1	48.9/54.9/58.2	44.9/47/52.2	42.7/45.8/51.1	34.4/36.5/38.2	35.4/39.1/41.2	24.7/28.1/31.3	40.29/43.76/47.47
	MI-FGSM	8.87/9.65/10.43	41.5/43.7/48.1	36.5/40/41.2	35.6/38.9/40.8	36.1/37.7/39.5	14.2/15/15.6	13.6/12.4/12.8	7.2/7.9/8	26.39/27.94/29.43
	SINI-FGSM	8.26/9.85/10.63	41.4/49.4/53.3	35.7/43.9/48.7	38.1/46.1/48.9	33.4/43.5/47.3	14.8/15.4/16.6	14.6/14/15.1	6.2/7.1/7.8	26.31/31.34/33.96

Table 6: The attack success rates (%) on four undefended models and three adversarial trained models by various transferable adversarial attacks. The adversarial examples are crafted on ViT-B/16 with different perturbations. The best results are in bold.

Model	Attack	Perturbation	Inc-v3	Inc-v4	Res-50	Res-101	Res-152	Inc-v3ens3	Inc-v3ens4	IncResv2ens	mASR
ViT-B/16	AdvGAN	10.59/11.30/12.25	75.6/72/80.3	70.3/60.5/70.2	72.5/75.7/84.3	70.8/75.2/84.9	68.9/71.8/77.1	61.2/74.9/83.4	61.8/69.5/79.1	53.8/50/60.5	66.86/68.7/77.46
	PAR-AdvGAN	**10.21/11.12/12.01**	**76.1/81.8/84.5**	63.6/67/70.9	**80.3/84.9/87.3**	**79.9/84.8/87.3**	**75.7/80.4/83**	**82.3/87.6/90.6**	**73.9/79.5/83.1**	**55.2/62.5/66.3**	**73.38/78.56/81.63**
	FGSM	10.76/11.71/12.65	35/36.3/38.6	31.9/34.6/36.4	33.6/35.1/37.6	32.2/34/36	31.9/34/36	27.9/30.1/31.7	29.9/30.3/31.5	23.8/25.3/26.5	30.78/32.46/34.29
	BIM	10.90/11.47/12.37	55.5/58.7/60.7	50.5/49.7/54.1	54.2/55.6/59.2	48.5/51.4/56.8	46.6/47.6/54.4	39.2/38.5/42.4	39.7/41/42.8	30.5/32.4/35.1	45.59/46.86/50.69
	PGD	10.73/11.23/12.24	46.6/48/53	40.7/42.3/44	44.2/47/49.6	40.3/42.7/45.7	37.9/39.6/42.4	28.2/29.7/31.7	31.8/31.5/33.9	21.8/22.9/25.2	36.44/37.96/40.69
	DI-FGSM	10.59/11.58/12.04	75.6/77.8/78.9	**70.3/74.1/75.1**	72.5/77/74.9	70.8/74.5/73.1	68.9/71.8/71.9	61.2/67.7/67.4	61.8/65.8/67.5	53.8/58.5/57.9	66.86/70.9/70.84
	TI-FGSM	10.77/11.21/12.23	60.5/61/65.4	54.1/56.2/59.9	53.1/55.4/60.2	52.6/54.6/58.3	50.7/54.8/59.3	56.4/58.6/61.2	59.4/62.4/63.6	52.1/53/57.7	54.86/57/60.7
	MI-FGSM	10.75/11.58/12.36	53.7/55.6/59.1	47.4/49/52.8	50.9/54.3/57.7	47.7/50.5/53.2	45.5/48.9/50.9	38.6/40.5/43.7	40.5/42.6/43.9	30.7/33.7/36.7	44.38/46.89/49.75
	SINI-FGSM	10.83/11.66/12.45	58.2/61.8/65.5	52.7/56.3/60.9	55.7/61.2/64.7	51.1/56.1/59.6	49.8/54.1/57.9	44.5/47.5/49.4	44.1/46.7/50.4	37.9/39.7/43.9	49.25/52.93/56.54

performance on certain models such as Inc-v4 and Res-101, the overall effectiveness of PAR-AdvGAN across all models underscores its robustness and transferability.

Experiments on ViT-B/16 In Table 6, we present the results of attacks using ViT-B/16 as the source model with three different perturbation rates on several target models, including Inc-v3, Inc-v4, Res-50, Res-101, Res-152, Inc-v3ens3, Inc-v3ens4, and IncRes-v2. Unlike traditional convolutional neural networks (CNNs), Vision Transformers (ViTs) adopt a fundamentally different architecture for image classification tasks. Our experimental findings show that the proposed PAR-AdvGAN algorithm performs exceptionally well when transferred to ViT-based models, achieving an average increase of 6.85% in attack success rate (ASR) compared to AdvGAN across all target models. Specifically, PAR-AdvGAN consistently delivers the highest mean ASR values of 73.38%, 78.56%, and 81.63% across the three perturbation rates, surpassing all baseline methods, including DI-FGSM, which was the best performer in certain cases. These results underscore the robustness and transferability of PAR-AdvGAN, demonstrating its ability to maintain high effectiveness not only with traditional CNNs but also with more recent transformer-based models like ViT, thus proving its versatility and reliability across different model architectures.

Attack Transferability Result Analysis With the results from Tables 2- 6, it can be observed that in most cases, our adversarial attack algorithm shows significantly improved transferability compared to the original AdvGAN, especially at lower perturbation rates. Additionally, compared to other competitive baselines, PAR-AdvGAN exhibits the best transferability. Notably, to ensure the fairness of the experiments, our algorithm was consistently compared with other methods for a lowest perturbation rate. In instances where the perturbation rates were higher, some algorithms did not exhibit a proportional increase in attack transferability. However, the transferability is overall improved.

Attack Speed Analysis As shown in Table. 7, we evaluated the computational efficiency of PAR-AdvGAN and seven competitive baselines using Inc-v3, Inc-v4, and IncRes-v2 as source models. We use FPS as the metric for measuring

Table 7: FPS comparison of PAR-AdvGAN with seven competitive baselines

Method	Inc-v3	Inc-v4	IncRes-v2
FGSM	176.5	116.7	76.1
BIM	10.6	6.5	4.1
PGD	5.5	3.3	2.1
DI-FGSM	10.8	6.6	4.1
TI-FGSM	10.6	6.5	4.2
MI-FGSM	42.7	26	16.4
SINI-FGSM	8.6	5.3	3.3
PAR-AdvGAN	**335.5**	**291.3**	**332.8**

attack speed, representing the number of images that can be processed by the attack per second. It can be observed that across Inc-v3, Inc-v4, and IncRes-v2, PAR-AdvGAN exhibits speed improvements of 61, 88.3, and 158.5 times over the slowest-performing PGD algorithm among the competitive baselines. Furthermore, in comparison to the fastest-performing FGSM algorithm among the competitive baselines, PAR-AdvGAN achieves speed enhancements of 1.9, 2.5, and 4.4 times, respectively. We assert that PAR-AdvGAN demonstrates significantly higher attack speed in comparison to traditional gradient-based transferable methods, while simultaneously achieving state-of-the-art transferability performance.

4.4 Ablation Experiment for RQ3

We investigate the effects of parameter λ on the attack transferability as it is an important parameter to control the perturbation range. Fig. 1 shows the performance of PAR-AdvGAN with Inc-v3 as the source model, with a fixed ϵ of 20, and λ set to 0.5, 0.8, and 1 for different target models. At λ of 0.5, the specific perturbation rates are 11.75, 12.56, and 12.89. At λ of 0.8, the specific perturbation rates are 11.55, 12.59, and 12.90. At λ of 1, the specific perturbation rates are 11.66, 12.45, and 12.81. We can see that at higher perturbation rate intervals, setting λ to 0.8 achieves best transferability performance.

Fig. 2 compares the results with fixed ϵ of 16 and λ set to 0.8 and 1. At λ of 0.8, the specific perturbation rates are 9.40, 10.60, and 11.20. At λ of 1, the corresponding perturbation rates are 8.95, 10.88, and 11.40. For lower perturbation rates, setting λ to 1 achieves the best transferability.

5 Conclusion

In this paper, we present a novel PAR-AdvGAN algorithm to boost adversarial attack capability through iterative perturbations. Specifically, to address the perturbation escalation issue in AdvGAN, we first adopt a progressive generator network to incorporate the initial sample x_0 in the perturbation generation process. An auto-regression iterative method is then proposed to include non-initial sample information in generator training. Furthermore, we constrain the distance between initial samples and subsequent samples. Our extensive experimental results exhibit the superior attack transferability of our method. Moreover, compared with the state-of-the-art gradient-based transferable attacks, our method achieves an accelerated attack efficiency.

References

1. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 ieee symposium on security and privacy (sp). pp. 39–57. Ieee (2017)
2. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11315–11325 (2022)
3. Cohen, J., Rosenfeld, E., Kolter, Z.: Certified adversarial robustness via randomized smoothing. In: international conference on machine learning. pp. 1310–1320. PMLR (2019)
4. Croce, D., Castellucci, G., Basili, R.: Gan-bert: Generative adversarial learning for robust text classification with a bunch of labeled examples (2020)
5. Deng, Y., Zheng, X., Zhang, T., Chen, C., Lou, G., Kim, M.: An analysis of adversarial attacks and defenses on autonomous driving models. In: 2020 IEEE international conference on pervasive computing and communications (PerCom). pp. 1–10. IEEE (2020)
6. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9185–9193 (2018)
7. Dong, Y., Pang, T., Su, H., Zhu, J.: Evading defenses to transferable adversarial examples by translation-invariant attacks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4312–4321 (2019)
8. Dziugaite, G.K., Ghahramani, Z., Roy, D.M.: A study of the effect of jpg compression on adversarial images. arXiv preprint arXiv:1608.00853 (2016)
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
10. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
11. Haidar, M.A., Rezagholizadeh, M.: Textkd-gan: Text generation using knowledge distillation and generative adversarial networks. In: Advances in Artificial Intelligence: 32nd Canadian Conference on Artificial Intelligence, Canadian AI 2019, Kingston, ON, Canada, May 28–31, 2019, Proceedings 32. pp. 107–118. Springer (2019)
12. Hang, J., Han, K., Chen, H., Li, Y.: Ensemble adversarial black-box attacks against deep learning systems. Pattern Recognition **101**, 107184 (2020)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
14. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14. pp. 630–645. Springer (2016)

15. Jandial, S., Mangla, P., Varshney, S., Balasubramanian, V.: Advgan++: Harnessing latent layers for adversary generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0 (2019)
16. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
17. Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T.: Alias-free generative adversarial networks. Advances in Neural Information Processing Systems **34**, 852–863 (2021)
18. Kim, H.: Torchattacks: A pytorch repository for adversarial attacks. arXiv preprint arXiv:2010.01950 (2020)
19. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: Artificial intelligence safety and security, pp. 99–112. Chapman and Hall/CRC (2018)
20. Li, X., Li, X., Pan, D., Zhu, D.: On the learning property of logistic and softmax losses for deep neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 4739–4746 (2020)
21. Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X., Zhu, J.: Defense against adversarial attacks using high-level representation guided denoiser. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1778–1787 (2018)
22. Lin, J., Song, C., He, K., Wang, L., Hopcroft, J.E.: Nesterov accelerated gradient and scale invariance for adversarial attacks. arXiv preprint arXiv:1908.06281 (2019)
23. Long, Y., Zhang, Q., Zeng, B., Gao, L., Liu, X., Zhang, J., Song, J.: Frequency domain model augmentation for adversarial attack. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV. pp. 549–566. Springer (2022)
24. Ma, X., Niu, Y., Gu, L., Wang, Y., Zhao, Y., Bailey, J., Lu, F.: Understanding adversarial attacks on deep learning based medical image analysis systems. Pattern Recognition **110**, 107332 (2021)
25. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
26. Ni, H., Szpruch, L., Wiese, M., Liao, S., Xiao, B.: Conditional sig-wasserstein gans for time series generation. arXiv preprint arXiv:2006.05421 (2020)
27. Pan, D., Li, X., Li, X., Zhu, D.: Explainable recommendation via interpretable feature mapping and evaluation of explainability. arXiv preprint arXiv:2007.06133 (2020)
28. Papernot, N., Faghri, F., Carlini, N., Goodfellow, I., Feinman, R., Kurakin, A., Xie, C., Sharma, Y., Brown, T., Roy, A., Matyasko, A., Behzadan, V., Hambardzumyan, K., Zhang, Z., Juang, Y.L., Li, Z., Sheatsley, R., Garg, A., Uesato, J., Gierke, W., Dong, Y., Berthelot, D., Hendricks, P., Rauber, J., Long, R.: Technical report on the cleverhans v2.1.0 adversarial examples library. arXiv preprint arXiv:1610.00768 (2018)
29. Qiang, Y., Li, X., Zhu, D.: Toward tag-free aspect based sentiment analysis: A multiple attention network approach. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2020)
30. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 31 (2017)
31. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)

32. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
33. Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., McDaniel, P.: Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204 (2017)
34. Xiao, C., Li, B., Zhu, J.Y., He, W., Liu, M., Song, D.: Generating adversarial examples with adversarial networks. arXiv preprint arXiv:1801.02610 (2018)
35. Xie, C., Wang, J., Zhang, Z., Ren, Z., Yuille, A.: Mitigating adversarial effects through randomization. arXiv preprint arXiv:1711.01991 (2017)
36. Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.L.: Improving transferability of adversarial examples with input diversity. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2730–2739 (2019)
37. Yoon, J., Jarrett, D., Van der Schaar, M.: Time-series generative adversarial networks. Advances in neural information processing systems **32** (2019)
38. Zhang, J., Wu, W., Huang, J.t., Huang, Y., Wang, W., Su, Y., Lyu, M.R.: Improving adversarial transferability via neuron attribution-based attacks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14993–15002 (2022)