Enhancing transferability of adversarial examples with gradient shift

Anonymous Author(s)
Affiliation
Address
email

Abstract

The transferability of adversarial examples in the black-box attack setting has attracted extensive attention from the community. Among the previous research, input transformations are one of the most effective of all to improve the transferability of adversarial examples. However, we find that input transformations are short of theoretical analysis, which also means that they are an unexplainable methods and unfavorable to further improve the transferability of adversarial samples. This paper attempts to analyze input transformations from a gradient perspective, and is inspired by this to propose a gradient shift hypothesis. Specifically, for different models trained on the same dataset, there is a computable bias between their gradients with respect to the same input. Based on this, we propose a new method that can further enhance the transferability of adversarial examples. Experiments on the standard ImageNet dataset demonstrate the superiority of our proposed method. Code is available at https://...

1 Introduction

Deep neural networks (DNNs) have been shown superior performance in various tasks. However, adversarial examples [1] can fool the models by modifying normal examples with indistinguishable perturbations. Such undesirable vulnerability have raised great concern for applications based on DNNs, especially for security-sensitive realm, e.g., face verification [2], autonomous driving [3]. Therefore, we need for better techniques to evaluate the robustness of neural networks [4].

Many efforts have been devoted to crafting the adversarial examples. In general, according to the knowledge owned by attackers, adversarial attacks can be divided into two categories, i.e., white-box attacks where the adversary has full access to the model and black-box attacks where the adversary does not know or know limited information of the model. Obviously, black-box attacks are more challenging and damaging than white-box attacks and have attracted great attention.

Under the black-box attack setting, the transferability of adversarial examples is a central issue. Various techniques, such as [5, 6, 7, 8, 9], have been proposed to improve the transferability, of which input transformations are one of the most effective. DIM [10] is a classic input transformations method that randomly resizing and padding images to improve the transferability of adversarial examples. Other methods, e.g., TIM [8], SIM [6] and Admix [11], have also achieved some of success. Nevertheless, we found that these research works are short of theoretical analysis, that is unfavorable to further improve the transferability of adversarial samples.

We conduct a theoretical analysis of input transformations. And inspired by the analysis, we propose the gradient shift hypothesis which indicates the gradient relationship between different models. Specifically, the difference in model structure and randomness in traning process lead to the discrepancy in models themselves, i.e., the final functions. However, the gradient of various models with

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

respect to the same input exist similarity to some extend. Based on the hypothesis, a new method called Ropeway, which can significantly improve the transferability of adversarial samples, is proposed. The extend experiment not only shows the effectiveness of the Ropeway, but also proves the existence of gradient shift.

2 Background and Related work

There are many works aiming to improve the transferability of adversarial examples. According to the attacked layers, black-box attacks are also categorized into attacking internal features and disrupting output layers [12]. This study falls into the latter.

The methods attacking internal features, such as [13, 12, 14, 15, 16], seems like a good approach. However, the majority of these are dependent on the choice of the middle layer, which requires numerous experimentation. In the setting of disrupting output layers, the adversarial examples can be generated with high transferability without selecting the intermediate layer. In general, there are two methods to enhance the transferability of adversarial examples : (1) better optimization algorithm and (2) model augmentation [6].

MI-FGSM [5] is a typical better optimization algorithm. MI-FGSM integrate momentum into I-FGSM [17] to stabilize updates and avoid poor local optimum. Since it just like improving generalization ability of the trained models, better optimization algorithm can boost the transferability of adversarial examples. Other methods, e.g., NI-FGSM [6], AI-FGTM [18], EMI-FGSM [19], have a favorable performance as well. While the optimization method can be used to find better adversarial examples, it cannot further explain the connection between the models trained on the same dataset (which can only be attributed to generalization)

Model augmentation is another powerful method to enhance the transferability of adversarial examples and the most prominent one is input transformations. Common input transformations include randomly resizing and padding [10], translation [8], scaling [6], interpolation [11], and so on. When these transformations are applied to the input image, it is equivalent to deriving a enhanced model from the original model [6]. The adversarial examples generated on the new model are more transferable. However, there is the same problem with the "better optimization algorithm" that model augmentation cannot further explain the connection between the models trained on the same dataset. And even, they themselves are shortage of corresponding theoretical explanations and are unfavorable to further improve the transferability of adversarial samples.

The LI-FGSM proposed recently by Jang et al. [20] is similar to the Ropeway proposed by us, both of which use the gradients of next n steps to adjust the current gradient. But there are essentially 3 differences: (1) LI-FGSM is inspired by NI-FGSM [6], while Ropeway is mainly inspired by the theoretical analysis of input transformations; (2) LI-FGSM lacks the theoretical analysis of input transformations, while that is an important part of this paper; (3) For calculating gradients of next n steps, LI-FGSM only rely on VNI-FGSM [21] (which is a "better optimization algorithm" method), but Ropey can use any input transformations.

3 Gradient shift hypothesis

3.1 Preliminaries

Given a classfication model f and a classfication task $f(x) = y$, where x denotes the clean image and y denotes the output label which equals to the ground truth label. Our intention is to generate adversarial examples x^{adv} that can mislead the classifier, i.e., $f(x^{adv}) \neq y$. To align with previous works, we adopt the L_∞-norm to restrict pertubation. So, the process of generating adversarial examples is to solve the following optimization problems.

$$\arg\max_{x^{adv}} J(x^{adv}, y),\ s.t.\ \|x - x^{adv}\|_\infty \leq \epsilon,$$

where J is the loss function (i.e., cross-entropy) and ϵ is the upper bound of pertubations.

Fast Gradient Sign Method (FGSM) [1] exploits the gradient of the loss function for a one-step attack as follows:

$$x^{adv} = x + \epsilon \cdot sign(\nabla_x J(x, y)),$$

where $sign(\cdot)$ denotes the sign function and $\nabla_x J(x, y)$ denotes the gradient of the loss function w.r.t. x.

Iterative Fast Gradient Sign Method (I-FGSM) [17] can generate adversarial examples with high white-box attack success rate by iterative updating:

$$x_{t+1}^{adv} = x_t^{adv} + \alpha \cdot sign(\nabla_{x_t^{adv}} J(x_t^{adv}, y)),$$

where α is small pertubations and less than ϵ.

Momentum Iterative Fast Gradient Sign Method (MI-FGSM) [5] integrates the momentum term into I-FGSM [Admix11] to boost the transferability of adversarial examples, which can be expressed as:

$$g_t = \mu \cdot g_{t-1} + \frac{\nabla_{x_t^{adv}} J(x_t^{adv}, y)}{\|\nabla_{x_t^{adv}} J(x_t^{adv}, y)\|_1},$$
$$x_{t+1}^{adv} = x_t^{adv} + \alpha \cdot sign(g_t).$$

Diverse Inputs Method (DIM) [10] applies image transformations $T(\cdot)$ to the inputs with the probability p at each iteration of I-FGSM:

$$x_{t+1}^{adv} = x_t^{adv} + \alpha \cdot sign(\nabla_{x_t^{adv}} J(T(x_t^{adv}, p), y)),$$

where $T(\cdot)$ is firstly resizing the image with probability p and then padding it.

3.2 Gradient approximation

Goodfellow et al. [1] propose that the reason for adversarial examples is due to the linear nature of DNNs, so that making many infinitesimal changes to the input can add up to one large change to the output. In the setting of black-box attack, the transferability of adversarial examples crafted by FGSM depends on gradient approximation. In order to describe this gradient approximation, we simplify the FGSM algorithm and remove the $sign(\cdot)$ function:

$$x^{adv} = x + \epsilon \cdot \nabla_x J(x, y), \tag{1}$$

where y is the ground-truth label of the input x. Given two pretrained models f_1, f_2 and the loss function J. And f_1, f_2 is the source model, target model respectively. If we want high transferability of adversarial examples made by Eq. (1) between f_1 and f_2, we accutally expect the graident of f_1 can equal to the graident of f_2 w.r.t. the same inputs, i.e., $\nabla_x^1 J(x, y) \approx \nabla_x^2 J(x, y)$, where $\nabla_x^1 J(x, y)$ is the graident of f_1 and $\nabla_x^2 J(x, y)$ is the graident of f_2. Since that, no matter whether the value of the $J(x^{adv}, y)$ becomes larger or smaller than $J(x, y)$ of f_1, $J(x^{adv}, y)$ can always become larger than $J(x, y)$ of of f_2 and may lead to $f_2(x^{adv}) \neq y$.

In the example above, let $d_1 = \nabla_x^1 J(x, y) - \nabla_x^2 J(x, y)$ where d_1 denotes the discrepancy between graidents of f_1 and f_2. Obviously, d_1 is not necessarily a zero matrix, that means $\nabla_x^1 J(x, y)$ is not necessarily equal to $\nabla_x^2 J(x, y)$. Therefore, it is conceivable that the transferability of the adversarial samples made by Eq. (1) are poor. And now, we introduce the input transformations to improve the transferability of adversarial examples, that is $x' = T(x)$ where $T(\cdot)$ is some kind of transformations (e.g., transformations in DIM, refer to Section 3.1). After that, the Eq. (1) can be reformulated as:

$$x^{adv} = x + \epsilon \cdot \nabla_x J(T(x), y). \tag{2}$$

If we make adversarial examples by Eq. (2) on f_1, can we expect gradient approximation between the same x? The answer is yes. Firstly, we can rewrite $x' = T(x)$ as $x' = x+\delta$, that is, $T(x) = x+\delta$. What needs to be stated is that $T(x)$ and $x + \delta$ are only formally equal, which does not mean that the transformation is actually an addition operation. Secondly, we can rewrite $\nabla_x J(T(x), y)$, i.e., $\nabla_x J(T(x), y) = \nabla_x J(x + \delta, y)$, and rewrite d_1 as $d_1^{'} = \nabla_x^1 J(T(x), y) - \nabla_x^2 J(x, y)$ Suppose our model is 3 times differentiable in a closed ball $B = \{x'' : \|x'' - x\| \leq r\}$, for some $r \geq 0$. According to the first-order multivariate Taylor expansion

$$\nabla_x J(x + \delta, y) \approx \nabla_x J(x, y) + \delta \cdot D\nabla_x J(x, y),$$

where D is differential. Then, we apply it to $d_1^{'}$, and we get d_2:

$$\begin{aligned} d_2 &= \nabla_x^1 J((T(x), y) - \nabla_x^2 J(x, y) \\ &= \nabla_x^1 J(x + \delta, y) - \nabla_x^2 J(x, y) \\ &\approx \nabla_x^1 J(x, y) - \nabla_x^2 J(x, y) + \delta \cdot D \nabla_x^1 J(x, y). \end{aligned} \tag{3}$$

Due to the harmony of $\delta \cdot D\nabla_x^1 J(x, y)$, when models f_1, f_2 and input x are certain, d_2 is no longer fixed, but can vary with δ. So, if the transformation is ideal, d_2 is more likely to be close to the zero matrix than d_1. In theory, let $d_2 = 0$, we can find an optimal δ and even judge the "good or bad" of the transformation.

This paper does not aim to prove which transformation is more effective for improving the transferability of adversarial examples, so we will not go into details. And, previous studies [6, 8, 10, 11] have demonstrated the excellent performance of transformations, which also shows that the gradient approximation of transformations is relatively successful from the experimental perspective.

3.3 Gradient shift

However, We can see that the δ in Eq. (3) is uncertain, which is due to the uncontrollable randomness in transformations (e.g., random resizing). We don't want to control for these randomness, but we make a wild assumenation which inspired by the derivation in Section 3.2.

The start of our understanding of transformations is gradient approximation. Based on this, is there a possibility that the gradients of f_1 and f_2 w.r.t. the same inputs have some geometric relationship, such as translation? If there is a geometric relationship like translation, the problem that "how to obtain a more approximate gradient?" turns into "whether the bias b can be found so that $\nabla_x^1 J(x + b, y) \approx \nabla_x^2 J(x, y)$?".

Fortunately, after lots of experimental exploration, we finally found b. And b is not a simple translation, since that translation means the gradients will move the same distance and direction for different x. There is obviously a more complex relationship between the gradients of different models trained on the same dataset w.r.t. the same inputs. So, we call it "shift". For short, given pre-trained models f_1, f_2 and input x, the gradient of f_2 w.r.t. x can be obtained by regularly shifting the gradient of f_1 w.r.t. x. In contrast to the linear hypothesis proposed in FGSM, the Gradient shift hypothesis focuses on the relationship between the gradients of different models. In section 4, we show the discipline of the shifting and the calculation formula of b.

4 A new transferability enhancement method

4.1 Ropeway

Given the model f and its loss function J. The $\nabla_x J(x, y)$ denotes the gradient of f w.r.t. x. To compute the bias b (see the notion in section 3.3), we first need to go forward n steps along the direction of the gradient, and preserve these gradients on all steps:

$$\begin{aligned} &x_0^{'} = x, \\ &g_t^{'} = \nabla_{x_t^{'}} J(x_t^{'}, y), \\ &x_{t+1}^{'} = x_t^{'} + \beta \cdot g_t^{'}, \end{aligned} \tag{4}$$

where β is the step size. Then b can be calculated as follow :

$$b = \frac{1}{n+1} \sum_{t=0}^{n} g_t^{'}. \tag{5}$$

Unfortunately, this is obtained through experiments and summaries, rather than strict theoretical derivation. So, it is difficult to theoretically prove the gradient shift hypothesis and Eq. (5), and we have to choose other methods. However, as presentation in Section 3.3, if b could be found, we can use the gradient of the source model to approximate the gradient of the target model. Once approximate successfully, the transferability of adversarial examples will be easily improved, and the gradient shift hypothesis and Eq. (4) will be proved to a some extent as well.

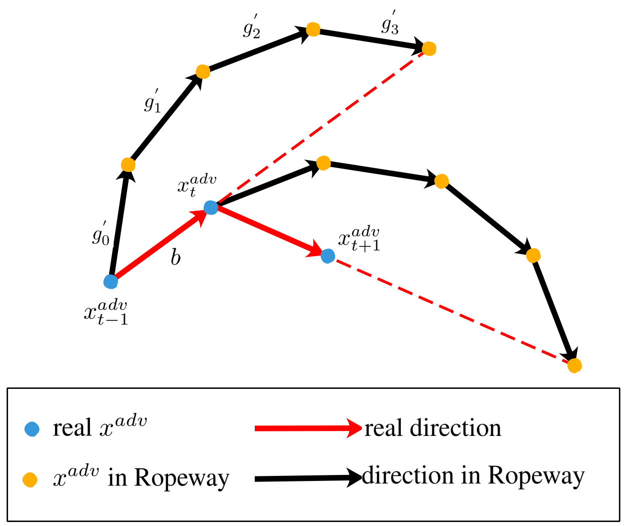


Figure 1: Illustration of the update direction of the adversarial example in Ropeway.

It has to be said that, although we simplified FGSM (i.e., Eq. (1)) in the derivation of the gradient shift hypothesis, this hypothesis is essentially a study of the properties of DNNs rather than a study of specific attacks. So we can directly apply this to existing attacks to boost the transferability of adversarial examples. Taking MI-FGSM as an example, we only need to add b to get the new update formula of x^{adv}:

$$g_t = \mu \cdot g_{t-1} + \frac{b}{\|b\|_1}, \tag{6}$$

$$x_{t+1}^{adv} = x_t^{adv} + \alpha \cdot sign(g_t), \tag{7}$$

where α is the step length and the calculation formula of $x_0^{'}$ in Eq. (4) should be modified as $x_0^{'} = x_t^{adv}$. In order to reduce the number of hyperparameters, β in Eq. (4) is set equal to α in Eq. (7). From the perspective of process of adversarial examples generation, our proposed method adjust the current gradient with the gradient of next n steps to obtain a better gradient direction (see Fig. 1), just like the point go forward along the "Ropeway" between the current position and the end position. That's why we name this method Ropeway.

4.2 Algorithm

In fact, Ropeway shows a stronger performance advantage when combined with both input transform and MI-FGSM. We illustrate it in Algorithm 1 how Ropeway is combined with DIM and MI-FGSM. In the Ropeway of each iteration (i.e., lines 5-10), we do not clip the perturbation. On the one hand this is due to the consideration of reducing the amount of computation; on the other hand, empirically, no clip yields better performance.

5 Experimental Results

5.1 Experimental Setup

Dataset: We evaluate on 1,000 images from the ILSVRC 2012 validation set provided by Lin et al. [6].

Baselines: We adopt four input transformations as our baselines, i.e., DIM [10], TIM [8], SIM [6] and DI-Admix which combined with Admix[11] and DIM. All the input transformations are integrated into MI-FGSM [5].

Algorithm 1 Ropeway combined with DIM and MI-FGSM

Input: Classifier f, loss function J, benign example x, ground-truth label y, max pertubation ϵ, number of iterations T, transformation $T(\cdot)$ and it's probability p, number of steps in Ropeway N
Output: An adversarial example x^{adv}
$\alpha = \epsilon/T; g_0 = 0; x_0^{adv} = x; n = 0$
for $t = 0 \to T - 1$ **do**
  while $n \leq N$ **do**
    $x_n^{'} = x_t^{adv}$
    Transform $x_n^{'}$ and calculate the gradient: $g_n^{'} = \nabla_{x_n^{'}} J(T(x_n^{'}, p), y)$
    Save $g_n^{'}$
    Calculate the $x_{n+1}^{'}$: $x_{n+1}^{'} = x_n^{'} + \alpha \cdot g_n^{'}$
    $n = n + 1$
  end while
  Calculate the bias b by Eq. (5)
  Update momentum by Eq. (6) with b
  Update x_{t+1}^{adv} by Eq. (7)
  Clip x_{t+1}^{adv} with x and ϵ
end for
return x_T^{adv}

Models: We evaluate adversarial examples on six popular normally trained models and five adversarially trained models. Normally trained models are Inception-V3 (Inc-v3) [22], Inception-V4 (Inc-v4) [23], VGG19 (Vgg-19) [24], ResNet-101 (Res-101) [25], ResNet-152 (Res-152) [25] and Iception-ResNet-V2 (IncRes-v2) [23]. Adversarially trained models [26, 27] are Adv-Inc-v3, Ens3-Inc-v3, Ens4-Inc-v3, Adv-IncRes-v2 and Ens-adv-IncRes-v2.

Attack settings: The maximum perturbation $\epsilon = 16$, number of iteration $T = 10$, step size $\alpha = \epsilon/10$ and the momentum decay factor $\mu = 1.0$. We adopt the Gaussian kernel with size 7×7 for TIM, the transformation probability $p = 0.5$ for DIM, and the number of copies $m = 5$ for SIM. We set $m_1 = 5, \gamma_i = 1/2^i, m_2 = 3, \eta = 0.2$ for Admix and $n = 10$, $\beta = \alpha$ for Ropeway.

Other details: DIM, TIM, SIM, and Admix are all implemented with different versions of Python and TensorFlow, but Ropeway is implemented with PyTorch. For a fair comparison, we do not use the code provided by authors of each model to directly produce adversarial examples. Insteadly, we reference their source code and rewritten all baseline attacks with PyTorch. For the 5 adversarially training models, we are in TensorFlow environment to evaluate the generated adversarial examples. We publish the information of version and library on our codes. All resources are publicly available.

5.2 Evaluation on normally trained models

Firstly, we evaluate the adversarial examples on 6 normally trained models, namely Inc-v3, Inc-v4, Vgg-19, Res-101, Res-152 and IncRes-v2, with DIM, TIM, SIM ,DI-Admix attacks and their Ropeway version. We craft adversaries on 3 normally trained networks, i.e., Inc-v3, Vgg-19 and Res-152, respectively. The attack success rates, namely the misclassification rates on target models, are shown in Table 1. The models attacken are on rows and the models tested are on columns.

It can be seen that whichever the attack is, Ropeway outperforms on all tests. Under the white-box attack setting, Ropeway maintains a high attack success rate. Under the black-box attack setting, when the original model is Inc-v3 and the target model is IncRes-v2, the success rate of DIM is 62.5%, while that of R-DIM is as high as 81.9%, and that is a tremendous improvement. These experimental data show that Ropeway is very effective in boosting the transferability of adversarial examples, and proves the gradient shift hypothesis to some extent as well.

5.3 Evaluation on adversarially trained models

In general, adversarially trained models having strong robustness, so we evaluate the performance on adversarially trained models (i.e., Adv-Inc-v3, Ens3-Inc-v3, Ens4-Inc-v3, Adv-IncRes-v2 and Ens-

Table 1: Success rate on normally trained models. The attacks start with "R-" denotes combining with Ropeway. "*" indicates white-box attacks.

model	attack	Inc-v3	Inc-v4	Vgg-19	Res-101	Res-152	IncRes-v2
Inc-v3	DIM	99.6%*	65.8%	70.2%	56.2%	55.6%	62.5%
	R-DIM	99.9%*	84.6%	82.3%	69.5%	70.5%	81.9%
	SIM	99.9%*	68.2%	76.5%	67.3%	68.8%	66.3%
	R-SIM	100%*	74.5%	82.5%	69.5%	73.6%	72.5%
	TIM	99.8%*	49.5%	62.5%	46.7%	48.1%	45.7%
	R-TIM	100%*	63.1%	71.9%	55.9%	57.5%	60.4%
	DI-Admix	100%*	92.9%	93.6%	90.7%	93.1%	92.1%
	R-DI-Admix	100%*	96%	95.7%	92.6%	94.6%	95.6%
Vgg-19	DIM	56.8%	59.5%	100%*	59.1%	57.9%	47.9%
	R-DIM	68.1%	75.6%	100%*	70.3%	69%	60.9%
	SIM	59.8%	58.2%	100%*	68.2%	64.3%	45.4%
	R-SIM	66.1%	68.3%	100%*	71.4%	70.5%	54.1%
	TIM	59.8%	57.9%	100%*	62.9%	61.1%	49.2%
	R-TIM	69.2%	72.2%	100%*	70.5%	66.9%	58.5%
	DI-Admix	79.8%	80.1%	100%*	82.5%	82.8%	68.2%
	R-DI-Admix	84.3%	88.9%	100%*	87.0%	85.3%	77.3%
Res-152	DIM	79.9%	76.9%	91.8%	93.7%	100%*	73.5%
	R-DIM	87.5%	85.9%	96.8%	98.8%	100%*	84.7%
	SIM	65.9%	53.4%	80.9%	94.2%	100%*	53.3%
	R-SIM	68.9%	63.7%	87.5%	98.2%	100%*	60.5%
	TIM	67.8%	60.8%	77.6%	92.7%	100%*	58.9%
	R-TIM	75.8%	69.1%	83.6%	96.2%	100%*	68.5%
	DI-Admix	92.4%	89.9%	97.1%	99.1%	100%*	86.4%
	R-DI-Admix	95.7%	95.4%	99.0%	99.8%	100%*	94.4%

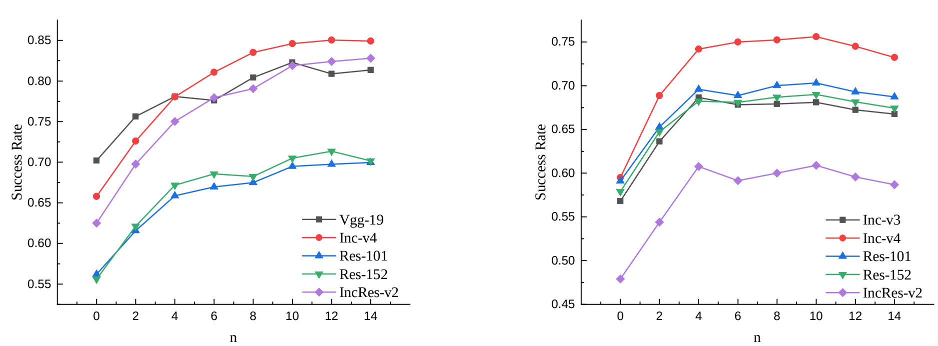
adv-IncRes-v2). Same with Section 5.2, we use 3 original models (i.e., Inc-v3, Vgg-19 and Res-152) to generate adversarial examples and the results are shown in Table 2 (The models attacken are on rows and the models tested are on columns). It can be seen that Ropeway still shows strong performance even in the face of adversarially training models. Among all input transformations attacks, Ropeway has the most significant improvement on DIM, with an average of 11.81 percentage points in 15 experiments. R-DI-Admix has the highest attack success rate, with an average attack success rate as high as 75.89% in 15 experiments.

5.4 Ablation Studies

Since we set $\beta = \alpha$, there is only one hyperparameter in Ropeway, i.e., n. The key of Ropeway is n which will directly impact the transferability of adversarial examples. To highlight n, we set different value (i.e., $n = 0, 2, 4, 6, 8, 10, 12, 14$) in the R-DIM attack for a comprehensive comparison. We create adversarial examples on Inc-v3 and Vgg-19 (Fig. 2(a) and Fig. 2(b), respectively), and then evaluate them on other normally trained models (Section 5.1). Because the attack success rate of the white-box attack is almost always 100%, we omit the white-box attack test, and the experimental results are shown in Fig. 2. In the figures, the horizontal axis represents the size of n, and the vertical axis represents the attack success rate. R-DIM degenerates into DIM when $n = 0$ (see Algorithm 1) and the attack success rate has been significantly improved when $n = 2$. The attack success rate increases as n increases where adversarial examples are crafted on Inc-v3 (see Fig. 2(a)) and the inflection point of attack success rate occurs when $n = 10$ where adversarial examples are crafted on Vgg-19 (see Fig. 2(b)).This is why we choose $n = 10$ in this paper.

Table 2: Success rate on adversarially trained models. The attacks start with "R-" denotes combining with Ropeway.

Model	Attack	Adv-Inc-v3	Ens3-Inc-v3	Ens4-Inc-v3	Adv-IncRes-v2	Ens-adv-IncRes-v2
inc-v3	DIM	53.5%	48.1%	44.9%	43.5%	34.6%
	R-DIM	73.8%	65.4%	60.4%	62.4%	48.7%
	SIM	61.9%	56.1%	54.4%	48.6%	38.5%
	R-SIM	66.3%	59.2%	56.6%	55.3%	42.3%
	TIM	42.9%	40.0%	37.7%	36.0%	29.2%
	R-TIM	54.8%	50.6%	47.8%	46.4%	36.8%
	DI-Admix	89.1%	85.8%	83.8%	81.3%	72.0%
	R-DI-Admix	92.3%	90.2%	87.3%	86.6%	75.9%
vgg-19	DIM	38.3%	34.5%	32.3%	28.0%	23.7%
	R-DIM	46.6%	46.2%	37.4%	38.4%	30.7%
	SIM	42.0%	39.9%	36.7%	31.7%	26.7%
	R-SIM	47.0%	45.7%	39.0%	36.8%	31.9%
	TIM	45.9%	43.6%	42.2%	35.9%	32.6%
	R-TIM	52.8%	52.2%	47.9%	43.2%	39.1%
	DI-Admix	60.1%	56.9%	51.4%	49.7%	41.1%
	R-DI-Admix	63.0%	62.1%	53.7%	56.0%	47.0%
res-152	DIM	65.2%	64.6%	59.5%	60.8%	50.9%
	R-DIM	74.8%	75.2%	67.7%	71.0%	60.8%
	SIM	50.6%	48.9%	45.4%	43.5%	37.7%
	R-SIM	57.7%	53.8%	51.0%	49.9%	41.4%
	TIM	57.0%	54.7%	53.0%	50.5%	47.0%
	R-TIM	65.8%	62.7%	60.1%	58.4%	53.7%
	DI-Admix	84.9%	81.5%	77.3%	78.5%	72.9%
	R-DI-Admix	89.1%	88.1%	83.8%	86.2%	77.0%



(a) Inc-v3 (b) Vgg-19

Figure 2: The success rate changes as the changes of n in Ropeway. The adversarial examples are crafted on (a) Inc-v3 and (b) Vgg-19 respectively.

6 Conclusion

In this work, We revisit and explain the nature of common input transformations attack, such as DIM, SIM, TIM and Admix. Inspired by these methods, a hypothesis that the gradient of different models trained on same datasets are related is proposed. Based on this, we proposed a new method called Ropeway which can boost the transferability of adversarial examples without harm the success rate under white-box attacks. Specifically, for each iteration of generating examples, the gradients of next n steps are used to adjust the gradient of current step in Ropeway. Extensive experiments are conducted to demonstrate the superior performance of Ropeway. we hope the gradient shift hypothesis and our attack will provide a new perspective to understand DNNs and the adversarial examples.

References

[1] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[2] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[3] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

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

[6] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. *arXiv preprint arXiv:1908.06281*, 2019.

[7] Junhua Zou, Zhisong Pan, Junyang Qiu, Xin Liu, Ting Rui, and Wei Li. Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting. In *European Conference on Computer Vision*, pages 563–579. Springer, 2020.

[8] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019.

[9] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv preprint arXiv:2002.05990*, 2020.

[10] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019.

[11] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He. Admix: Enhancing the transferability of adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16158–16167, 2021.

[12] Zhibo Wang, Hengchang Guo, Zhifei Zhang, Wenxin Liu, Zhan Qin, and Kui Ren. Feature importance-aware transferable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7639–7648, 2021.

[13] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 452–467, 2018.

[14] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4733–4742, 2019.

[15] Muzammal Naseer, Salman H Khan, Shafin Rahman, and Fatih Porikli. Task-generalizable adversarial attack based on perceptual metric. *arXiv preprint arXiv:1811.09020*, 2018.

[16] Aditya Ganeshan, Vivek BS, and R Venkatesh Babu. Fda: Feature disruptive attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8069–8079, 2019.

[17] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[18] Junhua Zou, Zhisong Pan, Junyang Qiu, Yexin Duan, Xin Liu, and Yu Pan. Making adversarial examples more transferable and indistinguishable. *arXiv preprint arXiv:2007.03838*, 2020.

[19] Xiaosen Wang, Jiadong Lin, Han Hu, Jingdong Wang, and Kun He. Boosting adversarial transferability through enhanced momentum. *arXiv preprint arXiv:2103.10609*, 2021.

[20] Donggon Jang, Sanghyeok Son, and Dae-Shik Kim. Strengthening the transferability of adversarial examples using advanced looking ahead and self-cutmix.

[21] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1924–1933, 2021.

[22] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[23] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[26] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4.1. The calculation of bias b is derived from experimental experience rather than theoretical derivation.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.2.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Section 3.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See README.md file in the code provided.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5.1.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Although there is randomness in the experiment, after many experiments we have found that the effect of randomness on the experimental results is only in a very small range and is not a decisive factor.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 5.1 and README.md file in the code provided.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We show the source of the existing assets very clearly.
   (b) Did you mention the license of the assets? [Yes] The all of assets we used can be used to do this research.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] We do not provide supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Section 5.1. The datasets and models we use are publicly available and can be used to do the work of this paper.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] The data we used does not contain such information and content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]