PID-Based Approach to Adversarial Attacks

Chen Wan[1, 2], **Biaohua Ye**[1, 2], **Fangjun Huang**[1, 2*]

[1] School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 510006, China
[2] Guangdong Provincial Key Laboratory of Information Security Technology, Guangzhou 510006, China
wanchen18@outlook.com, yebh3@mail2.sysu.edu.cn, huangfj@mail.sysu.edu.cn

Abstract

Adversarial attack can misguide the deep neural networks (DNNs) with adding small-magnitude perturbations to normal examples, which is mainly determined by the gradient of the loss function with respect to inputs. Previously, various strategies have been proposed to enhance the performance of adversarial attacks. However, all these methods only utilize the gradients in the present and past to generate adversarial examples. Until now, the trend of gradient change in the future (*i.e.*, the derivative of gradient) has not been considered yet. Inspired by the classic proportional-integral-derivative (PID) controller in the field of automatic control, we propose a new PID-based approach for generating adversarial examples. The gradients in the present and past, and the derivative of gradient are considered in our method, which correspond to the components of P, I and D in the PID controller, respectively. Extensive experiments consistently demonstrate that our method can achieve higher attack success rates and exhibit better transferability compared with the state-of-the-art gradient-based adversarial attacks. Furthermore, our method possesses good extensibility and can be applied to almost all available gradient-based adversarial attacks.

Introduction

Deep Neural Networks (DNNs) have been demonstrated to be highly vulnerable to adversarial attacks, which are designed carefully by adding small-magnitude perturbations to original inputs (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2014). Adversarial examples can be hardly distinguished from normal examples by human observers but can misguide the DNNs to produce incorrect outputs. Thus, adversarial examples may result in security risks of misidentification and misdirection in deep learning applications such as autonomous vehicles and language translation systems (Athalye et al. 2018; Hein and Andriushchenko 2017). Previously, a variety of algorithms have been proposed to generate adversarial examples such as fast gradient sign method (FGSM) (Goodfellow, Shlens, and Szegedy 2014), Carlini & Wagner's method (C&W) (Carlini and Wagner 2017) and generative adversarial networks method (GANs) (Xiao et al. 2018). Among these methods, FGSM has lower cost and better performance in general, and it has received more and more attention recently (Liu et al. 2017; Madry et al. 2018; Cheng et al. 2019; Zheng, Chen, and Ren 2019).

The improved methods of FGSM include making full use of the gradient (*i.e.*, better optimization algorithm) and data augmentation. Generally, better optimization algorithms such as Momentum Iterative (MI-FGSM) (Dong et al. 2018) and Nesterov Iterative (NI-FGSM) (Lin et al. 2020) try to prevent the generated adversarial examples from falling into poor local maxima, and data augmentation strategies, *e.g.*, diverse inputs (DI) (Xie et al. 2019b), translation-invariant (TI) (Dong et al. 2019) and scale-invariant (SI) (Lin et al. 2020), strive to prevent the generated adversarial examples from overfitting the model. Nevertheless, the abovementioned adversarial attacks only consider the gradients in the present and past to update the added adversarial perturbations. To the best of our knowledge, the trend of gradient change in the future (*i.e.*, the derivative of gradient) has not been taken into consideration.

In this study, we propose a new PID-based approach to generate the adversarial examples, which takes the present gradient, past gradient, and the derivative of gradient of the loss function into account. Note that PID controller is a classic controller in the field of automatic control (Ang, Chong, and Li 2005). The basic philosophy behind it is to exploit the present, past and derivative of the prediction error to control a feedback system. An et al. (2018) firstly pointed out that the backpropagation in neural networks employed in machine learning shares some similarity to the feedback in PID control, and the idea of PID control can be utilized to optimize the parameters of DNNs. In our opinion, the PID control also has a close relationship with the process of generating the adversarial examples. For example, if we view the gradient of loss function in DNNs as prediction error in PID controller, FGSM can be considered as a P controller since it uses the present gradient to generate the added adversarial perturbations, and MI-FGSM and NI-FGSM can be considered as two PI controllers since they both use the gradients in the present and past to generate the added adversarial perturbations. In this paper, we integrate the idea of PID control into adversarial attacks to form a series of new PID-based adversarial attacks. Our contributions are listed as follows.

- We first revealed the relationship between the optimiza-

[*]Corresponding author.

tion process of adversarial attack and PID control, and established a close connection between the field of classical control and the adversarial attack, which opens a new door for solving the optimization problem of adversarial attack.

- A series of new PID-based adversarial attacks are proposed in this paper, which can not only achieve higher success rates, but also possess better transferability. Moreover, compared with those P and PI-based adversarial attacks, our new PID-based methods may have high executive efficiency, *i.e.*, with consuming the same time, the performance obtained by our algorithm is better in general.
- Our new strategy possesses good extensibility and can be easily applied to almost all available gradient-based adversarial attacks.

Background

In this section, we give a brief overview of the gradient-based adversarial attack methods and PID controller.

Gradient-based Adversarial Attack Methods

Let x denote a original image and y denote the corresponding ground-truth label. A given classifier $f(x)$ outputs a label $\hat{y}$ as the prediction for an input image x. An adversarial example x^{adv} is generated by adding small perturbations to x to mislead the classifier, *i.e.*, $f\left(x^{adv}\right) \neq y$. We use $J(x, y)$ to denote the cross-entropy loss function of the classifier. To generate the adversarial example, the goal is to maximize the loss function $J\left(x^{adv}, y\right)$. In this work, we utilize the L_∞ norm to measure the perceptibility of adversarial perturbations, *i.e.*, $\left\|x^{adv} - x\right\|_\infty \leq \epsilon$, where ϵ describes the magnitude of perturbations.

Fast gradient sign method (FGSM) (Goodfellow, Shlens, and Szegedy 2014) generates an adversarial example by calculating the gradient of the loss function, and performs the one-step update as

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

where $\nabla_x J$ is the gradient of the loss function with respect to x, and $sign(\cdot)$ is the sign function to restrict the perturbation in the L_∞ norm bound.

Momentum Iterative Fast Gradient Sign Method (MI-FGSM) (Dong et al. 2018) adopts momentum term to stabilize the update directions for perturbations. Note that different from the original FGSM, MI-FGSM applies gradient updates multiple times with a small step size (*i.e.*, iterative gradient-based attack), which firstly proposed by Kurakin, Goodfellow, and Bengio (2016). The update procedure of MI-FGSM is described as follows.

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} \tag{2}$$

$$x_{t+1}^{adv} = Clip_x^\epsilon\left\{x_t^{adv} + \alpha \cdot sign\left(g_{t+1}\right)\right\} \tag{3}$$

where $x_0^{adv} = x$, $g_0 = 0$, g_t is the accumulated gradient at iteration t, μ is the decay factor of the momentum term, α is the step size, and the $Clip_x^\epsilon\{\cdot\}$ function indicates that the generated adversarial examples are clipped within the ϵ-ball of the original image.

Nesterov Iterative Fast Gradient Sign Method (NI-FGSM) (Lin et al. 2020) proposes to improve the transferability of adversarial examples by integrating Nesterov accelerated gradient proposed by Nesterov (1983) into the iterative gradient-based attack, and the update procedure is formalized as follows.

$$x_t^{nes} = x_t^{adv} + \alpha\mu \cdot g_t \tag{4}$$

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x J\left(x_t^{nes}, y\right)}{\left\|\nabla_x J\left(x_t^{nes}, y\right)\right\|_1} \tag{5}$$

$$x_{t+1}^{adv} = Clip_x^\epsilon\left\{x_t^{adv} + \alpha \cdot sign\left(g_{t+1}\right)\right\} \tag{6}$$

where $x_0^{adv} = x$, $g_0 = 0$, and x_t^{nes} means that x_t^{adv} makes a jump in the direction of the previously accumulated gradient.

PID Controller

The PID controller originated in the 19th century and has been the most popular controller in industry due to its simplicity and interpretability (Astrom and Hagglund 2001). Specifically, the PID controller continuously calculates the prediction error $e(t)$, *i.e.*, the difference between the expected optimal output and the measured system output (Ang, Chong, and Li 2005). According to the proportional (P), integral (I), and derivative (D) terms of $e(t)$, the system control signal $u(t)$ is obtained, which can be described as

$$u(t) = k_p e(t) + k_i \int_0^t e(t) + k_d \frac{d}{dt} e(t) \tag{7}$$

where k_p, k_i, and k_d are the gain coefficients on the P, I and D components, and $e(t)$, $\int_0^t e(t)$ and $\frac{d}{dt}e(t)$ represent prediction errors in the present and past, and the change trend of prediction error in the future, respectively. Note that in the field of classic control, it seems very simple to introduce D controller into PI controller to form the PID controller. However, compared with the PI controller, the PID controller has made a revolutionary improvement. The fundamental reason is that D controller can predict the changing trend of the error $e(t)$, and thus the introduction of the D controller can reduce the overshoot, enhance system stability and result in fast response.

Method

In this section, we reveal the relationship between the adversarial attack and PID control firstly. Then, how to apply the idea of PID control to optimize two state-of-the-art adversarial attacks, *i.e.*, MI-FGSM and NI-FGSM, will be exemplified. Note that our method is a general method and can be applied to almost all gradient-based attacks to improve the performance of adversarial examples.

Relationship Between FGSM Series and PID Control

In our opinion, adversarial attack can be abstracted as such a controller, whose input is the gradient of loss function and the output is the added perturbations. The added perturbations are mainly determined by the gradient of the loss function, which is equivalent to the error single $e(t)$ in PID controller. The goal of the adversarial attack is to maximize the loss function. Nowadays, many existing methods have been proposed to maximize the loss $J(x_t^{adv}, y)$ by updating x_t^{adv} using the gradients $\nabla_x J\left(x_t^{adv}, y\right)$. Next, we will give some theoretical analysis to demonstrate that FGSM and its improved versions all can be regarded as P or PI controllers.

FGSM – P controller. According to Eq. (1), the added perturbations of the adversarial examples generated by FGSM can be described as follows.

$$x^{adv} - x = \epsilon \cdot sign\left(\nabla_x J\left(x, y\right)\right) \tag{8}$$

By viewing the gradient $\nabla_x J\left(x, y\right)$ in Eq. (8) as the error $e(t)$ in Eq. (7), we can found that FGSM only uses the present gradient to obtain the adversarial examples. Thus, FGSM can be regarded as a P controller.

MI-FGSM – PI controller. According to Eq. (2), MI-FGSM gathers the gradient information up to the t-th iteration for generating adversarial examples. Thus, Eq. (2) can be rewritten as

$$\begin{aligned} g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} \\ &= \mu^2 \cdot g_{t-1} + \mu \cdot \frac{\nabla_x J\left(x_{t-1}^{adv}, y\right)}{\left\|\nabla_x J\left(x_{t-1}^{adv}, y\right)\right\|_1} + \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} \\ &= \cdots \\ &= \sum_{i=0}^{t-1} \left(\mu^{t-i} \cdot \frac{\nabla_x J\left(x_i^{adv}, y\right)}{\left\|\nabla_x J\left(x_i^{adv}, y\right)\right\|_1}\right) + \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} \end{aligned} \tag{9}$$

As seen in Eq. (9), the first and second items correspond to the normalized gradients in the past and present, respectively. For simplicity, we represent these two items as

$$P_t = \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1}, I_t = \sum_{i=0}^{t-1} \left(\mu^{t-i} \cdot \frac{\nabla_x J\left(x_i^{adv}, y\right)}{\left\|\nabla_x J\left(x_i^{adv}, y\right)\right\|_1}\right)$$

Then, according to Eq. (3), the adversarial examples generated by MI-FGSM at each iteration can be rewritten as

$$x_{t+1}^{adv} = Clip_x^{\epsilon}\left\{x_t^{adv} + \alpha \cdot sign\left(I_t + P_t\right)\right\} \tag{10}$$

Neglect the $Clip_x^{\epsilon}\left\{\cdot\right\}$ function, and we can get

$$x_{t+1}^{adv} - x_t^{adv} = \alpha \cdot sign\left(I_t + P_t\right) \tag{11}$$

In Eq. (11), the left side represents the added perturbations at each iteration. As seen, MI-FGSM can be regarded as a PI controller. Analogously, NI-FGSM can also be regarded as a PI controller.

PID-based Adversarial Attack

As exemplified above, MI-FGSM and NI-FGSM both try to maximize the loss function $J(x_t^{adv}, y)$ via updating x_t^{adv}. However, in their optimization process, they only take into account the gradients $\nabla_x J\left(x_t^{adv}, y\right)$ in the past and present. According to the characteristics of D controller in the field of classical control (*i.e.*, reducing the overshoot, enhancing system stability and fast response, *etc.*), if we can introduce D controller take into account the changing trend of gradient in the future, the optimal adversarial examples may be easier to be found. Thus in the next, we will integrate the D term into the existing gradient-based adversarial attacks to form a series of new PID-based attacks, named MID-FGSM and NID-FGSM, in which the item of D represents the consideration of gradient changing trend in the future. The obtained MID-FGSM is summarized in Algorithm 1, and the core procedure of it can be formalized as follows.

Algorithm 1 MID-FGSM

Input: A clean example x with ground-truth label y; a classifier f with loss function J.
Parameter: The size of perturbation ϵ; iterations T, decay factor μ, and control parameter k_d.
Output: Adversarial example x^{adv}.

```
1:  Let $\alpha = \epsilon/T$, $x_0^{adv} = x$, $g_0 = 0$, $D_0 = 0$;
2:  for $t = 0$ to $T - 1$ do
3:      Get $x_t^b$ by $x_t^b = x_t^{adv} - \alpha \cdot D_t$;
4:      Input $x_t^b$ to $f$ and obtain gradient $\nabla_{x^b} J(x_t^b, y)$;
5:      Input $x_t^{adv}$ to $f$ and obtain gradient $\nabla_x J(x_t^{adv}, y)$;
6:      Update $D_{t+1}$ by Eq. (14);
7:      Update $g_{t+1}$ by Eq. (12);
8:      Update $x_{t+1}^{adv}$ by Eq. (13);
9:  end for
10: return $x^{adv} = x_T^{adv}$.
```

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} - D_{t+1} \tag{12}$$

$$x_{t+1}^{adv} = Clip_x^{\epsilon}\left\{x_t^{adv} + \alpha \cdot sign\left(g_{t+1}\right)\right\} \tag{13}$$

where D_{t+1} represents the accumulation of the derivative of gradients.

In our algorithm, D_0 is equal to 0, and D_{t+1} is updated by accumulating the derivative of gradient, which refers to the difference between the gradients of $J(x_t^{adv}, y)$ and $J(x_t^b, y)$

$$D_{t+1} = D_t + k_d \cdot B_t \tag{14}$$

where $B_t = \frac{\nabla_x J\left(x_t^{adv}, y\right)}{\left\|\nabla_x J\left(x_t^{adv}, y\right)\right\|_1} - \frac{\nabla_{x^b} J\left(x_t^b, y\right)}{\left\|\nabla_{x^b} J\left(x_t^b, y\right)\right\|_1}$, k_d is a control parameter, x_t^b is an intermediate state between x_{t-1}^{adv} and x_t^{adv}.

The reason why we replace x_{t-1}^{adv} with x_t^b is that the discrepancy between the gradients of $J(x_{t-1}^{adv}, y)$ and $J(x_t^{adv}, y)$ is generally too large, which may cause the adversarial examples to deviate from the global optimum. x_t^b is obtained by tuning the x_t^{adv} backward a little bit and can be described as follows.

$$x_t^b = x_t^{adv} - \alpha \cdot D_t \tag{15}$$

where α is the step size of each iteration, and D_t represents the accumulation of the derivative of gradients at iteration t.

Subsequently, we rewrite Eq. (12) to the format of PID controller for better understanding of the proposed algorithm. Via the same derivation process as that in Eq. (9), we can obtain Eq. (16) according to Eq. (14).

$$D_{t+1} = k_d \cdot \sum_{i=0}^{t} B_i \tag{16}$$

where $B_i = \frac{\nabla_x J(x_i^{adv}, y)}{\left\|\nabla_x J(x_i^{adv}, y)\right\|_1} - \frac{\nabla_{x^b} J(x_i^b, y)}{\left\|\nabla_{x^b} J(x_i^b, y)\right\|_1}$.

Substituting Eq. (16) into Eq. (12), we have

$$g_{t+1} = \begin{cases} I_t + P_t - \frac{k_d}{(1-\mu)} \sum_{i=0}^{t} \left(1 - \mu^{t-i}\right) B_i & \mu \neq 1 \\ I_t + P_t - k_d \sum_{i=0}^{t} (t - i + 1) B_i & \mu = 1 \end{cases} \tag{17}$$

where

$P_t = \frac{\nabla_x J(x_t^{adv}, y)}{\left\|\nabla_x J(x_t^{adv}, y)\right\|_1}$, $I_t = \sum_{i=0}^{t-1} \left(\mu^{t-i} \cdot \frac{\nabla_x J(x_i^{adv}, y)}{\left\|\nabla_x J(x_i^{adv}, y)\right\|_1} \right)$.

The items of P_t, I_t and B_i represent gradients in the present and past, and the derivative of gradient, respectively. Note that if $k_d = 0$, MID-FGSM degenerates to MI-FGSM.

Similarly, we can integrate the idea of PID control into NI-FGSM to obtain NID-FGSM. Starting with $x_0^{adv} = x$, $g_0 = 0$ and $D_0 = 0$, the update procedure of NID-FGSM can be formalized as follows.

$$x_t^{nes} = x_t^{adv} + \alpha\mu \cdot g_t \tag{18}$$

$$x_t^b = x_t^{adv} - \alpha \cdot D_t \tag{19}$$

$$B_t^* = \left(\frac{\nabla_x J\left(x_t^{nes}, y\right)}{\left\|\nabla_x J\left(x_t^{nes}, y\right)\right\|_1} - \frac{\nabla_{x^b} J\left(x_t^b, y\right)}{\left\|\nabla_{x^b} J\left(x_t^b, y\right)\right\|_1} \right) \tag{20}$$

$$D_{t+1} = D_t + k_d \cdot B_t^* \tag{21}$$

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x J\left(x_t^{nes}, y\right)}{\left\|\nabla_x J\left(x_t^{nes}, y\right)\right\|_1} - D_{t+1} \tag{22}$$

$$x_{t+1}^{adv} = Clip_x^{\epsilon} \left\{ x_t^{adv} + \alpha \cdot sign\left(g_{t+1}\right) \right\} \tag{23}$$

where D_t represents the accumulation of the derivative of gradients at iteration t, g_t denotes the update directions of perturbations at iteration t, and μ denotes the decay factor of g_t. Note that if $k_d = 0$, NID-FGSM degenerates to NI-FGSM.

The Extensibility of PID-based Adversarial Attack

Our PID-based approach possesses good extensibility and can be combined with the data augmentation strategies (*e.g.*, diverse inputs (DI) (Xie et al. 2019b), translation-invariant (TI) (Dong et al. 2019) and scale-invariant (SI) (Lin et al. 2020)) to further improve the attack performance. Note that in our algorithm, in each iteration, there are two states, *i.e.*, x_t^b and x_t^{adv}. For each state, the corresponding loss function can be obtained and the gradients are computed according to the obtained loss function. The abovementioned three data augmentation methods all can be integrated into these two states. We apply DI, TI and SI into our NID-FGSM and a series of new powerful adversarial attacks named DI-NID-FGSM, TI-DI-NID-FGSM and SI-TI-DI-NID-FGSM are proposed in this paper.

Experiments

In this section, we conduct extensive experiments on the ImageNet validation set (Deng et al. 2009) to evaluate the effectiveness of the proposed methods. All of our experiments are conducted on the Tensorflow DNN computing framework (Abadi et al. 2016) and run with four parallel NVIDIA GeForce GTX 1080Ti GPUs.

Experimental Settings

The test dataset consists of $10,000$ images (resized to $299 \times 299 \times 3$) chosen randomly from the ImageNet validation set, which are almost correctly classified by all the testing models described below. Four normally trained models and ten defense models are selected in our test.

- Four normally trained models, *i.e.*, Inception-v3 (Inc-v3) (Szegedy et al. 2016), Inception-v4 (Inc-v4) (Szegedy et al. 2017), Inception-Resnet-v2 (IncRes-v2) (Szegedy et al. 2017) and Resnet-v2-152 (Res-152) (He et al. 2016).
- Ten defense models,
  - three adversarially trained models, *i.e.*, Inc-v3$_{\text{ens3}}$, Inc-v3$_{\text{ens4}}$ and IncRes-v2$_{\text{ens}}$ (Tramèr et al. 2018);
  - the top-3 defense strategies in the NIPS 2017 competition, *i.e.*, high-level representation guided denoiser (HGD rank-1) (Liao et al. 2018), random resizing and padding (R&P rank-2) (Xie et al. 2018) and the rank-3 submission (NIPS-r3[1]);
  - four recently proposed defense methods, *i.e.*, purifying perturbations via image compression model (Comdefend) (Jia et al. 2019), randomized smoothing (RS) (Cohen, Rosenfeld, and Kolter 2019), feature distillation (FD$_1$) (Liu et al. 2019) and feature denoising (FD$_2$) (Xie et al. 2019a).

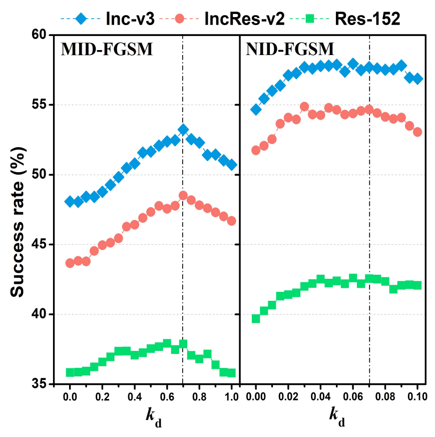
To evaluate the effectiveness of the proposed PID-based approach, two state-of-the-art algorithms MI-FGSM (Dong et al. 2018) and NI-FGSM (Lin et al. 2020) are selected for comparison. Note that our new PID-based approaches are named MID-FGSM and NID-FGSM. In order to demonstrate the extensibility of our proposed PID-based approach, the abovementioned three data augmentation strategies (*i.e.*, diverse inputs (DI) (Xie et al. 2019b), translation-invariant (TI) (Dong et al. 2019) and scale-invariant (SI) (Lin et al. 2020)) are integrated into the original NI-FGSM and our proposed NID-FGSM for comparison. NI-FGSM integrated with the three augmentation strategies are named DI-NI-FGSM, TI-DI-NI-FGSM and SI-TI-DI-NI-FGSM, and NID-FGSM integrated with the three data augmentation strategies are termed DI-NID-FGSM, TI-DI-NID-FGSM and SI-TI-DI-NID-FGSM, respectively.

In our experiments, the hyper-parameters, *i.e.*, the maximum perturbations of each pixel, the number of iterations, the step size, and the default decay factor are set as $\epsilon = 16$, $T = 10$, $\alpha = \epsilon/T = 1.6$, and $\mu = 1.0$, respectively. The transformation probabilities of DI augmentation strategy are set as 0.5, the size of the Gaussian kernels in TI augmentation strategy are set as 15×15, and the numbers of scale copies in SI augmentation strategy are set as 5.

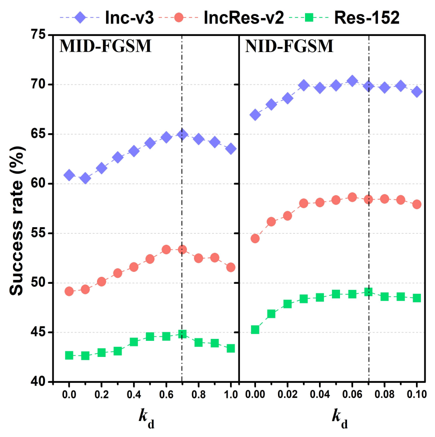
The Control Parameter k_d

In our algorithm, the parameter k_d is an important factor that should be optimized. In order to find the optimal value of k_d, two normally trained models (*i.e.*, Inc-v3 and Inc-v4) are considered as the target models in our test. For MID-FGSM attack, the parameter k_d is selected in the range from 0 to 1,

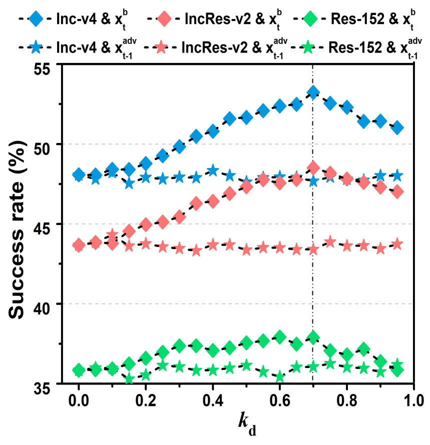
[1]https://github.com/anlthms/nips-2017/tree/master/mmd



(a)



(b)



(c)

Figure 1: (a) The success rates (%) of black-box attacks (the adversarial examples are obtained by attacking Inc-v3). (b) The success rates (%) of black-box attacks (the adversarial examples are obtained by attacking Inc-v4). (c) The success rates (%) of black-box attacks (the adversarial examples are obtained by attacking Inc-v3).

and for NID-FGSM attack, k_d is selected in the range from 0 to 0.1. Firstly, we attack Inc-v3 using MID-FGSM and NID-FGSM separately to obtain adversarial examples. The success rates of the obtained adversarial examples against Inc-v4, IncRes-v2 and Res-152 are shown in Figure 1(a). Then, we attack Inc-v4 using MID-FGSM and NID-FGSM separately to obtain adversarial examples. The success rates of the obtained adversarial examples against Inc-v3, IncRes-v2 and Res-152 are shown in Figure 1(b). As seen, the optimal values of k_d for MID-FGSM and NID-FGSM are approximately 0.7 and 0.07 respectively, which demonstrates that the optimal values of k_d generally keep stable for different target models. Therefore, we set the value of k_d as 0.7 for MID-FGSM and 0.07 for NID-FGSM in the following experiments.

The Effectiveness of x_t^b

According to our above analysis, the intermediate state x_t^b is introduced to prevent the obtained adversarial examples from falling into local optimal state. To explain the importance of x_t^b, we replace x_t^b in Eq. (14) with x_{t-1}^{adv} directly, and a new method named MID-FGSM' can be obtained. We use MID-FGSM and MID-FGSM' to attack Inc-v3 separately, and two groups of adversarial examples can be obtained, where the parameter k_d is selected in the range from 0 to 0.9. The success rates of the obtained adversarial examples against the Inc-v4, IncRes-v2 and Res-152 are shown in Figure 1c, respectively. Note that in Figure 1(c), the label "Inc-v4 & x_t^b" represents the success rates of the obtained adversarial examples (generated by MID-FGSM) against Inc-v4. The label "Inc-v4 & x_{t-1}^{adv}" represents the success rates of the obtained adversarial examples (generated by MID-FGSM') against Inc-v4. The other labels in Figure 1c have similar meanings. It can be observed that with selecting different values of k_d, the success rates obtained by MID-FGSM is higher than MID-FGSM', which demonstrates the effectiveness of x_t^b.

Attacking the Single Model

In this subsection, we firstly utilize MI-FGSM, MID-FGSM, NI-FGSM and NID-FGSM to attack the four normally trained models separately, and then the adversarial examples can be obtained. The success rates of the obtained adversarial examples against the seven models (*i.e.*, Inc-v3, Inc-v4, IncRes-v2, Res-152, Inc-v3$_{\text{ens3}}$, Inc-v3$_{\text{ens4}}$ and IncRes-v2$_{\text{ens}}$) are listed in Table 1. As seen, in contrast to MI-FGSM and NI-FGSM, our method can achieve higher success rates. It is also observed that NID-FGSM outperforms all other attacks. The success rates of NID-FGSM achieves approximately 100% under the white-box setting and consistently outperforms NI-FGSM by $1\% \sim 6\%$ under the black-box setting. The results validate the effectiveness of our algorithm and indicate that our proposed method can serve as a powerful strategy to improve the transferability of adversarial examples.

Next, we evaluate the effectiveness of our approach combined with the abovementioned three data augmentation strategies (*i.e.*, DI (Xie et al. 2019b), TI (Dong et al. 2019) and SI (Lin et al. 2020)). As we mentioned above, the baseline attack NI-FGSM integrated with the three augmentation strategies are named DI-NI-FGSM, TI-DI-NI-FGSM and SI-TI-DI-NI-FGSM, and our proposed NID-FGSM integrated with the three data augmentation strategies are termed DI-NID-FGSM, TI-DI-NID-FGSM and SI-TI-DI-NID-FGSM, respectively. All these methods are utilized to attack the normally trained models separately, and then the adversarial examples can be obtained. We test the efficiency of the obtained adversarial examples on seven modes (*i.e.*, Inc-v3, Inc-v4, IncRes-v2, Res-152, Inc-v3$_{\text{ens3}}$, Inc-v3$_{\text{ens4}}$ and IncRes-v2$_{\text{ens}}$). The success rates of DI-NI-FGSM and DI-NID-FGSM are reported in Table 2(a), TI-DI-NI-FGSM and TI-DI-NID-FGSM are reported in Table 2(b), and SI-TI-DI-NI-FGSM and SI-TI-DI-NID-FGSM are reported in Table 2(c). As seen, the success rates of our methods can achieve nearly 100% under the white-box setting, and outperform

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3ens3	Inc-v3ens4	IncRes-v2ens
Inc-v3	MI-FGSM	99.91*	48.08	43.66	35.84	13.31	12.08	6.28
	MID-FGSM	99.85*	52.40	48.18	37.96	13.02	11.73	6.47
	NI-FGSM	99.90*	54.67	51.75	39.69	12.89	11.68	5.78
	NID-FGSM	**99.92***	**57.71**	**54.67**	**42.56**	**14.37**	**13.58**	**6.86**
Inc-v4	MI-FGSM	60.97	99.81*	49.24	42.56	16.49	14.30	7.65
	MID-FGSM	64.85	99.24*	53.03	44.47	16.42	14.41	8.16
	NI-FGSM	66.68	99.91*	54.83	45.57	15.27	13.26	6.87
	NID-FGSM	**69.80**	**99.94***	**58.27**	**49.00**	**17.92**	**15.55**	**8.42**
IncRes-v2	MI-FGSM	61.39	54.07	97.17*	45.28	20.80	17.05	12.21
	MID-FGSM	61.91	55.27	95.34*	44.79	20.24	15.96	12.22
	NI-FGSM	64.81	56.11	98.77*	44.90	18.19	14.62	10.53
	NID-FGSM	**69.44**	**61.70**	**98.88***	**50.02**	**22.11**	**17.99**	**12.88**
Res-152	MI-FGSM	58.44	53.88	50.87	98.55*	24.05	20.30	13.13
	MID-FGSM	62.83	58.47	54.81	98.49*	24.66	21.15	13.85
	NI-FGSM	65.39	60.72	58.09	98.83*	23.21	19.46	12.36
	NID-FGSM	**68.90**	**64.86**	**61.56**	**98.90***	**26.86**	**22.82**	**14.88**

Table 1: The success rates (%) of adversarial attacks against the seven models. * indicates the white-box attacks.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3ens3	Inc-v3ens4	IncRes-v2ens
Inc-v3	DI-NI-FGSM	99.89*	64.78	60.10	46.88	13.91	12.91	6.28
	DI-NID-FGSM	**99.94***	**72.86**	**68.01**	**54.60**	**18.13**	**17.18**	**8.48**
Inc-v4	DI-NI-FGSM	73.74	99.70*	61.67	51.61	16.30	14.19	7.72
	DI-NID-FGSM	**79.84**	**99.78***	**69.46**	**58.99**	**20.94**	**18.96**	**10.75**
IncRes-v2	DI-NI-FGSM	71.41	65.38	97.53*	51.83	20.46	16.66	11.67
	DI-NID-FGSM	**77.16**	**72.79**	**97.55***	**59.95**	**27.55**	**23.24**	**17.28**
Res-152	DI-NI-FGSM	78.07	75.37	72.46	99.00*	28.63	23.84	15.23
	DI-NID-FGSM	**83.79**	**81.16**	**79.47**	**99.03***	**38.04**	**32.62**	**22.15**

(a) Comparison of DI-NI-FGSM and DI-NID-FGSM.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3ens3	Inc-v3ens4	IncRes-v2ens
Inc-v3	TI-DI-NI-FGSM	99.57*	55.99	46.53	41.49	37.80	35.67	26.95
	TI-DI-NID-FGSM	**99.78***	**61.44**	**51.61**	**45.42**	**43.11**	**40.78**	**31.87**
Inc-v4	TI-DI-NI-FGSM	61.19	98.86*	47.48	42.53	37.51	35.80	29.39
	TI-DI-NID-FGSM	**66.00**	**98.95***	**52.74**	**46.08**	**42.34**	**40.37**	**33.41**
IncRes-v2	TI-DI-NI-FGSM	61.34	58.18	94.04*	48.13	44.79	41.50	42.11
	TI-DI-NID-FGSM	**65.97**	**63.35**	**94.69***	**52.48**	**50.80**	**47.82**	**48.91**
Res-152	TI-DI-NI-FGSM	64.69	60.47	56.89	**98.75***	53.76	51.12	45.74
	TI-DI-NID-FGSM	**68.49**	**65.00**	**61.49**	98.68*	**59.09**	**57.05**	**51.95**

(b) Comparison of TI-DI-NI-FGSM and TI-DI-NID-FGSM.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3ens3	Inc-v3ens4	IncRes-v2ens
Inc-v3	SI-TI-DI-NI-FGSM	**99.93***	73.98	64.42	58.61	59.91	59.65	46.78
	SI-TI-DI-NID-FGSM	99.90*	**77.13**	**67.79**	**62.82**	**65.45**	**64.96**	**52.63**
Inc-v4	SI-TI-DI-NI-FGSM	76.16	99.75*	65.76	59.29	60.20	59.39	51.36
	SI-TI-DI-NID-FGSM	**79.54**	**99.76***	**70.05**	**63.71**	**65.89**	**64.82**	**56.92**
IncRes-v2	SI-TI-DI-NI-FGSM	78.17	76.78	**97.91***	67.82	71.27	68.77	68.51
	SI-TI-DI-NID-FGSM	**79.92**	**78.49**	97.86*	**70.44**	**74.97**	**73.00**	**72.59**
Res-152	SI-TI-DI-NI-FGSM	75.31	72.70	69.62	**99.58***	71.61	70.05	63.78
	SI-TI-DI-NID-FGSM	**77.13**	**74.61**	**72.21**	99.57*	**75.40**	**74.13**	**67.96**

(c) Comparison of SI-TI-DI-NI-FGSM and SI-TI-DI-NID-FGSM.

Table 2: The success rates (%) of adversarial attacks against the seven models. * indicates the white-box attacks.

the baseline attacks (*i.e.*, DI-NI-FGSM, TI-DI-NI-FGSM and SI-TI-DI-NI-FGSM) by $1\% \sim 10\%$ under the black-box setting and by $2\% \sim 10\%$ when attacking the adversarially trained models.

Attacking the Ensemble of Models

In this subsection, we further validate the effectiveness of the proposed method with attacking the ensemble of models. Following the ensemble method described in Dong et al. (2018), we use NI-FGSM, NID-FGSM, DI-NI-FGSM, DI-

Attack	Inc-v3ens3	Inc-v3ens4	IncRes-v2ens	HGD	R&P	NIPS-r3	ComDefend	RS	FD_1	FD_2
NI-FGSM	40.22	38.84	22.28	26.57	22.81	32.29	43.12	23.44	43.37	15.77
NID-FGSM	**47.75**	**41.72**	**27.57**	**35.55**	**28.32**	**38.66**	**48.69**	**24.85**	**49.58**	**15.97**
DI-NI-FGSM	43.70	38.20	24.20	30.63	26.27	37.16	46.36	24.95	47.10	15.98
DI-NID-FGSM	**56.50**	**50.50**	**35.10**	**49.83**	**38.53**	**49.51**	**55.51**	**27.52**	**56.03**	**16.34**
TI-DI-NI-FGSM	75.30	72.80	69.00	75.19	68.34	70.40	69.87	55.66	73.03	18.48
TI-DI-NID-FGSM	**79.90**	**78.00**	**74.40**	**80.00**	**74.39**	**75.52**	**74.12**	**58.93**	**77.28**	**18.91**
SI-TI-DI-NI-FGSM	91.50	90.60	87.30	90.50	86.60	88.53	88.18	75.21	90.62	22.42
SI-TI-DI-NID-FGSM	**92.70**	**92.00**	**88.80**	**91.37**	**88.39**	**89.82**	**89.90**	**78.00**	**92.07**	**23.28**

Table 3: Under the black-box setting, the success rates (%) of adversarial attacks against the advanced defense methods.

NID-FGSM, TI-DI-NI-FGSM, TI-DI-NID-FGSM, SI-TI-DI-NI-FGSM and SI-TI-DI-NID-FGSM to attack the ensemble of models which are composed of Inc-v3, Inc-v4, IncRes-v2 and Res-152 with the same ensemble weight and then the adversarial examples can be obtained. The success rates of the obtained adversarial examples against ten defense models are shown in Table 3. Our approaches consistently outperform the baseline attacks (*i.e.*, NI-FGSM, DI-NI-FGSM, TI-DI-NI-FGSM and SI-TI-DI-NI-FGSM) by $0.2\% \sim 19.2\%$, which demonstrates the effectiveness of our proposed PID-based approach.

Discussion of the Executive Efficiency

In this subsection, we firstly discuss the executive efficiency of the approach according to the number of iterations and success rate. The number of iterations is selected in the range from 1 to 10 and the maximum perturbations added on each pixel $\epsilon = 16$. We attack Inc-v3 using MI-FGSM, MID-FGSM, NI-FGSM and NID-FGSM separately and then four groups of adversarial examples can be obtained. The success rates of the obtained adversarial examples against Inc-v4, IncRes-v2 and Res-152 are shown in Figure 2, respectively. Note that in Figure 2, the label "Inc-v4 vs. MI-FGSM" represents the success rates of the obtained adversarial examples (generated by MI-FGSM) against Inc-v4. The other labels in Figure 2 have similar meanings. It can be observed that with the same number of iterations, MID-FGSM and NID-FGSM yield higher attack success rates than MI-FGSM and NI-FGSM under the black-box setting.

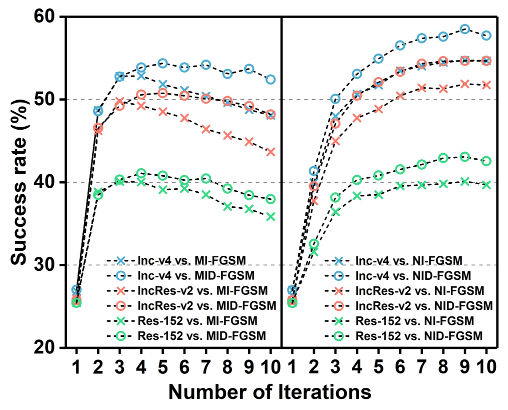


Figure 2: The success rates (%) of black-box attacks.

Then, we further discuss the executive efficiency of the algorithm from the run time and success rate. We use MI-FGSM, MID-FGSM, NI-FGSM and NID-FGSM to attack Inc-v3 separately, and the number of iterations are set as 5 and 10. The run time for generating the 10, 000 adversarial examples and the success rates of the obtained adversarial examples against Inc-v4, IncRes-v2 and Res-152 are shown in Table 4. It can be observed that under the same number of iterations, the run time of PID-based approach is higher than baseline attacks (*i.e.*, MI-FGSM and NI-FGSM). That is because our method needs to calculate the gradients two times in each iteration. However, we would like to emphasize that with consuming the same time, our method can get higher success rates. As seen in Table 4, the success rates of MID-FGSM/NID-FGSM with 5 iterations are higher than those of MI-FGSM/NI-FGSM with 10 iterations. This indicates that the introduction of D term may result in fast response.

Attack	Iteration	Run time	Inc-v4	IncRes-v2	Res-152
MI-FGSM	5	761.68	51.84	48.52	39.09
	10	1284.82	48.08	43.66	35.84
MID-FGSM	5	1273.58	54.37	50.76	40.79
	10	2292.23	52.40	48.18	37.96
NI-FGSM	5	763.99	51.73	48.85	38.51
	10	1287.14	54.67	51.75	39.69
NID-FGSM	5	1278.16	54.92	52.08	40.81
	10	2300.61	57.71	54.67	42.56

Table 4: The run time (s) for generating the adversarial examples and the success rates (%) of black-box attacks.

Conclusion

In this paper, we first uncover the close relationship between adversarial attacks and PID control, and a series of new PID-based approaches to generate adversarial examples are proposed. The main advantages of our method are as follows. 1) The adversarial examples generated by our algorithm can attack the white-box models more efficiently, meanwhile exhibit higher transferability against the black-box models and defense models. 2) Our method possesses good extensibility and can be applied to almost all available gradient-based adversarial attacks. 3) Our new PID-based approaches have high executive efficiency, *i.e.*, with consuming the same time, the performance of our method is better than those of P and PI-based approaches.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62072481, 61772572), and the NSFC-NRF Scientific Cooperation Program (61811540409).

References

Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, 265–283.

An, W.; Wang, H.; Sun, Q.; Xu, J.; Dai, Q.; and Zhang, L. 2018. A pid controller approach for stochastic optimization of deep networks. In *CVPR*, 8522–8531.

Ang, K. H.; Chong, G.; and Li, Y. 2005. PID control system analysis, design, and technology. *IEEE transactions on control systems technology* 13(4): 559–576.

Astrom, K. J.; and Hagglund, T. 2001. The future of PID control. *Control Engineering Practice* 9(11): 1163–1175.

Athalye, A.; Engstrom, L.; Ilyas, A.; and Kwok, K. 2018. Synthesizing robust adversarial examples. In *ICML*, 284–293.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, 39–57.

Cheng, S.; Dong, Y.; Pang, T.; Su, H.; and Zhu, J. 2019. Improving black-box adversarial attacks with a transfer-based prior. In *NeurIPS*, 10934–10944.

Cohen, J. M.; Rosenfeld, E.; and Kolter, J. Z. 2019. Certified adversarial robustness via randomized smoothing. In *ICML*, 1310–1320.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*, 248–255.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *CVPR*, 9185–9193.

Dong, Y.; Pang, T.; Su, H.; and Zhu, J. 2019. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *CVPR*, 4312–4321.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* .

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *ECCV*, 630–645.

Hein, M.; and Andriushchenko, M. 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2266–2276.

Jia, X.; Wei, X.; Cao, X.; and Foroosh, H. 2019. Comdefend: An efficient image compression model to defend adversarial examples. In *CVPR*, 6084–6092.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* .

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

Lin, J.; Song, C.; He, K.; Wang, L.; and Hopcroft, J. E. 2020. Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks. In *ICLR*.

Liu, Y.; Chen, X.; Liu, C.; and Song, D. 2017. Delving into transferable adversarial examples and black-box attacks. In *ICLR*.

Liu, Z.; Liu, Q.; Liu, T.; Xu, N.; Lin, X.; Wang, Y.; and Wen, W. 2019. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *CVPR*, 860–868.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *ICLR*.

Nesterov, Y. 1983. A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2). In *Doklady an ussr*, volume 269, 543–547.

Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 4278–4284.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *CVPR*, 2818–2826.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2018. Ensemble adversarial training: Attacks and defenses. In *ICLR*.

Xiao, C.; Li, B.; Zhu, J.-Y.; He, W.; Liu, M.; and Song, D. 2018. Generating adversarial examples with adversarial networks. In *IJCAI*, 3905–3911.

Xie, C.; Wang, J.; Zhang, Z.; Ren, Z.; and Yuille, A. 2018. Mitigating adversarial effects through randomization. In *ICLR*.

Xie, C.; Wu, Y.; Maaten, L. v. d.; Yuille, A. L.; and He, K. 2019a. Feature denoising for improving adversarial robustness. In *CVPR*, 501–509.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019b. Improving transferability of adversarial examples with input diversity. In *CVPR*, 2730–2739.

Zheng, T.; Chen, C.; and Ren, K. 2019. Distributionally adversarial attack. In *AAAI*, 2253–2260.