Pixel2Feature Attack (P2FA): Rethinking the Perturbed Space to Enhance Adversarial Transferability

Renpu Liu [*1] **Hao Wu** [*2] **Jiawei Zhang** [3] **Xin Cheng** [1] **Xiangyang Luo** [2] **Bin Ma** [4] **Jinwei Wang** [5]

Abstract

Adversarial examples have been shown to deceive Deep Neural Networks (DNNs), raising widespread concerns about this security threat. More seriously, as different DNN models share critical features, feature-level attacks can generate transferable adversarial examples, thereby deceiving black-box models in real-world scenarios. Nevertheless, we have theoretically discovered the principle behind the limited transferability of existing feature-level attacks: Their attack effectiveness is essentially equivalent to perturbing features in one step along the direction of feature importance in the feature space, despite performing multiple perturbations in the pixel space. This finding indicates that existing feature-level attacks are inefficient in disrupting features through multiple pixel-space perturbations. To address this problem, we propose a P2FA that efficiently perturbs features multiple times. Specifically, we directly shift the perturbed space from pixel to feature space. Then, we perturb the features multiple times rather than just once in the feature space with the guidance of feature importance to enhance the efficiency of disrupting critical shared features. Finally, we invert the perturbed features to the pixels to generate more transferable adversarial examples. Numerous experimental results strongly demonstrate the superior transferability of P2FA over State-Of-The-Art (SOTA) attacks.

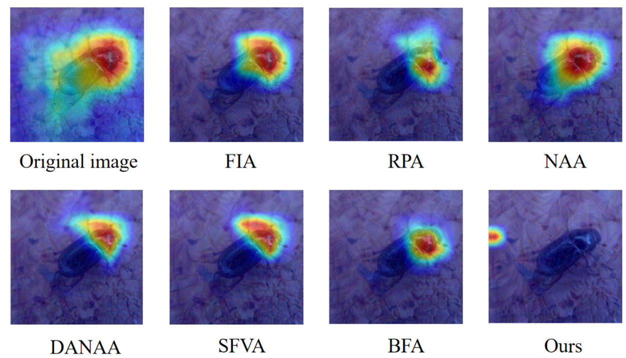

Figure 1. Comparison of existing feature-level attacks (FIA (Wang et al., 2021), RPA (Zhang et al., 2022b), NAA (Zhang et al., 2022a), DANAA (Jin et al., 2023), SFVA (Ren et al., 2023), and BFA (Wang et al., 2024b)) and our attack. Adversarial examples are generated on the source model (ResNet-152) (He et al., 2016a) to attack the target model (ResNet-50) (He et al., 2016a). Our attack keeps models from capturing important features of the object and focusing on entirely irrelevant regions instead, while existing feature-level attacks make models still focus on object-related areas.

1. Introduction

DNNs have achieved significant success in various machine learning tasks (Girshick, 2015; He et al., 2016b; Krizhevsky et al., 2012). However, DNNs are vulnerable to adversarial examples (Szegedy, 2013), which add elaborate and imperceptible perturbations to original images to mislead DNNs. The existence of adversarial examples has raised concerns about the security of sensitive applications, such as autonomous driving and face recognition. Research on adversarial examples not only enhances the understanding of their underlying principles and the drawbacks of DNNs, but also contributes to improving the adversarial robustness of DNNs (Goodfellow et al., 2014), ensuring their stability and accuracy in diverse application scenarios.

Numerous adversarial attacks (Carlini & Wagner, 2017; Xiao et al., 2018; Xie et al., 2019) have been proposed to generate adversarial examples, which are typically categorized into two types based on the attacker's knowledge of the target model: white-box attacks and black-box attacks. White-box attacks imply that the attacker has full access

[*]Equal contribution [1]School of Computer Science, Nanjing University of Information Science and Technology, Nanjing, China [2]State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China [3]School of Cyber Security and Information Law, Chongqing University of Posts and Telecommunications, Chongqing, China [4]School of Cyber Security, Qilu University of Technology (Shandong Academy of Sciences), Shandong, China [5]College of Cryptology and Cyber Science, Nankai University, Tianjin, China. Correspondence to: Jinwei Wang <wjwei_2004@163.com>, Xiangyang Luo <luoxy_ieu@sina.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

to the knowledge of the target model (e.g., structure and parameters) to generate adversarial examples via gradient information. In contrast, in black-box attacks, the attacker attacks only through the inputs and outputs of the target model without detailed information about the internal structure, parameters, training data, and so on, which makes black-box attacks more challenging and realistic. Therefore, we focus on black-box attacks in this work.

Based on attack strategies, black-box attacks can be further categorized as query-based attacks and transfer-based attacks. Query-based attacks (Brendel et al., 2017; Ilyas et al., 2018) approximate the gradient information through queries or employ intelligent search algorithms to explore the input space, thereby generating adversarial examples. However, query-based attacks are impractical in many real-world scenarios, such as face recognition and autonomous driving, because a large number of queries are not allowed in these scenarios. In contrast, transfer-based attacks (Lin et al., 2019; Dong et al., 2019; Wang & He, 2021) are more realistic and flexible, as they do not require any knowledge of the target model. They first attack the local white-box surrogate model and then transfer the obtained adversarial examples directly to the unknown target model. The ability of adversarial examples to cross models and retain effectiveness is known as transferability.

Recently, feature-level attacks (Zhou et al., 2018; Lu et al., 2020) have explored methods to improve the transferability of adversarial examples by targeting intermediate layers rather than directly perturbing the output layer of the surrogate model. Since key features are shared between different DNN models (Ganeshan et al., 2019; Naseer et al., 2018), these feature-level attacks enhance transferability by maximizing internal feature distortion to generate transferable adversarial examples.

In feature-level attacks, the evaluation of feature importance significantly affects the effectiveness of the attacks. FIA (Wang et al., 2021) uses random pixel dropping transformation and aggregates the transformed gradient to assess feature importance, suppressing model-specific features and highlighting object-related features. RPA (Zhang et al., 2022b) builds on FIA by recognizing the correlation between neighboring elements in a natural image. It then applies a random patch transformation to alter model-specific features, which improves the identification of key object features. In addition, NAA (Zhang et al., 2022a) attributes the model's output to each intermediate layer neuron and employs an approximation scheme for neuron attribution, allowing for an estimate of feature importance with reduced computational overhead. Based on this approach, DANAA (Jin et al., 2023) utilizes adversarial nonlinear path selection to expand the attack points, yielding more accurate feature importance. SFVA (Ren et al., 2023) estimates feature importance using Combined Feature Enhancement Transformation (CFET). BFA (Wang et al., 2024b) categorizes image features into white-box and black-box features, and assesses feature importance by disrupting the white-box features.

However, existing feature-level attacks still suffer from inefficiency. Specifically, by analyzing the loss functions of the above feature-level attacks, we mathematically demonstrate that they actually aim to perturb features along the direction of feature importance in the feature space. We experimentally validate this conclusion. Our results shows that the effect of existing feature-level attacks, which perturb **multiple times** in **pixel space**, is no different from perturbing features in **one step** along the direction of feature importance in **feature space**. This inefficient perturbation in pixel space limits the attacks' ability to enhance the transferability.

To address the inefficiency of existing feature-level attacks, this paper proposes P2FA. Specifically, we shift the perturbation from pixel space to feature space and perturb important features multiple times along the direction of feature importance within the feature space. The perturbed features are then used to generate corresponding adversarial examples via the feature inversion method. As illustrated in Fig. 1, compared to existing feature-level attacks, the adversarial example generated by the proposed P2FA keeps models from capturing important features of the object and focusing on entirely irrelevant regions instead. Extensive experiments on the ImageNet-NIPS dataset (Kurakin et al., 2018) demonstrate that P2FA significantly improve adversarial transferability.

The main contributions of this paper are summarized below:

- We have theoretically identified the principle underlying feature-level attacks: These feature-level attacks aim to perturb feature in one step in feature space, along the direction of feature importance. We verified the conclusion experimentally. Our findings reveal the inefficiency of existing feature-level attacks in disrupting important features, which limits their transferability.
- We propose P2FA, a new attack paradigm, to address the inefficiency issue and enhance adversarial transferability. P2FA generates transferable adversarial examples by perturbing important features multiple times in the feature space and then inverting the perturbed features back onto the image.
- Extensive experiments have demonstrated that the proposed P2FA outperforms SOTA attacks in terms of transferability.

2. Related Work

Since Szegedy et al. (2013) discovered the adversarial example, numerous adversarial attack algorithms have been proposed to demonstrate the vulnerability of neural networks. In this paper, we focus on transfer-based attacks, which initially target a local white-box surrogate model and then apply the resulting adversarial examples to attack an unknown target model. The paper further examines two primary categories of transfer-based attacks: feature-level attacks and input transformations.

2.1. Feature-level Attack

Feature-level attacks primarily target the internal feature maps of the model. A common approach in feature-level attacks is to expand the gap in these feature maps during the generation of adversarial examples.

Feature Importance-aware Attack (FIA) (Wang et al., 2021) . FIA obtains feature importance by introducing an aggregated gradient, which is computed by applying random pixel dropping to the original image. The feature importance directs the adversarial examples to disrupt the key features, thereby enhancing their transferability. We denote the feature map of the k-th layer of surrogate model f as $f_k(\cdot)$, and the feature importance as Δ_k. The objective function of FIA is presented in Eq. (1):

$$\arg\min_{x^{adv}} \sum (\Delta_k \odot f_k(x^{adv})), \quad (1)$$

where $\odot$ represents the Hadamard product operation and x^{adv} denotes the generated adversarial example.

Random Patch Attack (RPA) (Zhang et al., 2022b). RPA improves upon FIA by introducing the patch-wise random transformation to the original image, altering model-specific features to obtain a more accurate measure of feature importance.The objective function of RPA is presented in Eq. (1). The difference from FIA's objective function lies in their distinct Δ_k.

Neuron Attribution-based Attack (NAA) (Zhang et al., 2022a). NAA proposes a neuron attribution-based attack that attributes model's output exclusively to each neuron in the intermediate layer. The objective functions of NAA are presented in Eqs. (2) and (3):

$$A_k = (f_k(x^{adv}) - f_k(x')) \odot IA, \quad (2)$$

$$\arg\min_{x^{adv}} \sum_{A_k^{(j)} \geq 0} h_p(A_k^{(j)}) - \gamma \sum_{A_k^{(j)} < 0} h_n(-A_k^{(j)}), \quad (3)$$

where x' represents the baseline image, $\odot$ represents the Hadamard product operation, IA represents the the integration of the gradient along a straight line from the features of the baseline image to the features of the input, A_k represents the attribution of all neurons in the k-th layer of the model, and the parameter γ is used to balance the positive and negative attributions, $A_k^{(j)}$ denotes the j-th dimension of the attribution A_k, while $h_p(\cdot)$ and $h_n(\cdot)$ are the transformation functions applied to the positive and negative attributions, respectively. In practical, γ is set to 1, and both $h_p(\cdot)$ and $h_n(\cdot)$ are set to linear functions.

Double Adversarial Neuron Attribution Attack (DANAA) (Jin et al., 2023). DANAA improves upon NAA by attributing the model's output to an intermediate layer using adversarial nonlinear paths, thereby providing a more accurate measure of the weights of individual neurons. The objective function of DANAA is presented in Eq. (4):

$$\sum (f_k(x^{adv}) - f_k(x')) \odot \gamma_k, \quad (4)$$

where x^{adv} denotes the generated adversarial example, x' represents the baseline image, γ_k represents the the integration of the gradient along a non-linear path from the features of the baseline image to the features of the input.

Salient Feature Variance Attack (SFVA) (Ren et al., 2023). SFVA applies CFET (patch level mask, random noise, scale transformation) to a clean copy of the image to estimate the feature weight of the k-th layer. This is used to compute the positive and negative salient variances, PV_k and NV_k. The objective function of SFVA is presented in Eq. (5):

$$\arg\max_{x^{adv}} \lambda_1 \cdot \tau_P(PV_k) + \tau_N(NV_k), \quad (5)$$

where $\tau_P(\cdot)$ and $\tau_N(\cdot)$ are transformation functions, and λ_1 is the variance factor. In practical, λ_1 is set to 1, and $\tau_P(\cdot)$ and $\tau_N(\cdot)$ are set to identity functions.

Black-box Feature-driven Attack (BFA) (Wang et al., 2024b). BFA divides image features into white-box features and black-box features, obtains the fitted image by disrupting the white-box features, and computes the fitted gradient for the images with different fitting degrees. Finally, the objective function is constructed based on the obtained fitted gradient and feature map. The objective function of BFA is presented in Eqs. (6) and (7):

$$F = f_k(x^{adv}) \odot I + I^2, \quad (6)$$

$$\arg\max_{x^{adv}} \sum_{F \geq 0} F - \omega \cdot \sum_{F < 0} -F, \quad (7)$$

where x^{adv} denotes the generated adversarial example, $\odot$ represents the Hadamard product operation, I represents the fitted gradient and ω is the weight assigned to the control positive and negative factors.

Table 1. The attack success rates of the adversarial examples generated by FIA and its corresponding feature inversion attack (i.e.,FIA*) against the other models, using Inception-v3 as a surrogate model.

	Inception-v3	Inception-v4	Inception-ResNet-V2	ResNet-50	ResNet-152	Vgg-16	Vgg-19
FIA	98.8	86.2	78.7	82.0	75.4	85.1	83.7
FIA*	99.3	87.2	79.4	82.9	75.8	85.8	84.5

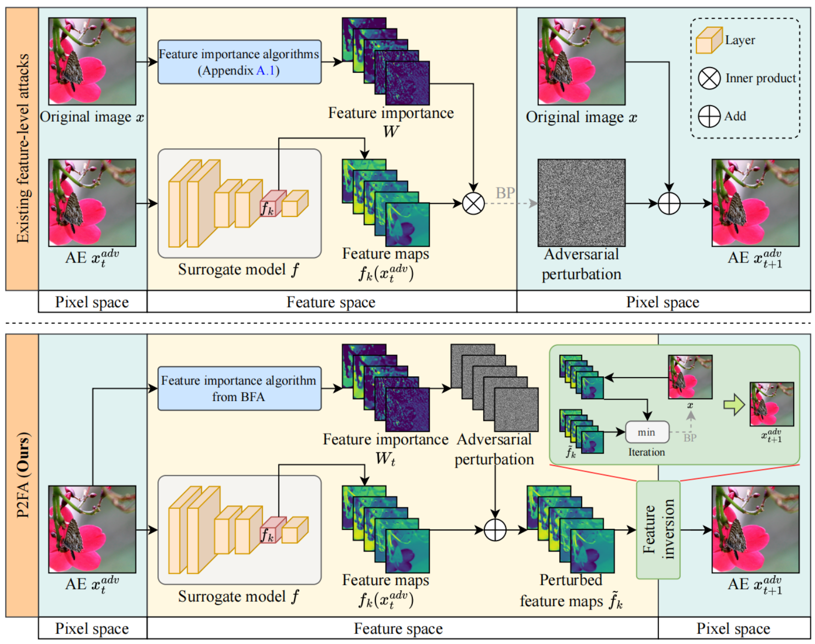

Figure 2. Schematic representation of the perturbations in pixel space and feature space for the existing feature-level attacks (top) and the proposed P2FA (bottom). The objective function of existing feature-level attacks is to maximize the inner product of feature importance W and feature maps $f_k(x_t^{adv})$, then generate adversarial examples by adding adversarial perturbation in the pixel space through backpropagation. P2FA directly adds adversarial perturbation to the feature maps in the feature space and generates adversarial examples through feature inversion.

2.2. Input Transformation

Input transformation improves the transferability of adversarial examples by data augmentation. Specifically,

Diverse Input Method (DIM) (Xie et al., 2019). DIM applies image transformations (random resizing and padding) in each iteration to improve the transferability of the adversarial examples.

Translation-Invariant Method (TIM) (Dong et al., 2019). TIM improves the transferability of the adversarial examples by optimizing the perturbation across a set of translated images. This approach reduces the sensitivity of the generated adversarial examples to the discriminative regions of the attacked white-box model.

Patch-wise Iterative Method (PIM) (Gao et al., 2020). PIM introduces an amplification factor and a projection kernel into the step size during each iteration. This modification enables the generated adversarial noise to exhibit aggregation characteristics and cover diverse discriminative regions across different DNNs, thereby enhancing the transferability of adversarial examples.

3. Methodology

3.1. Threat Model

We assume that the surrogate model is represented as $f(\cdot) : x \mapsto y$, where x denotes the original image and y denotes the corresponding ground-truth label. Our objective is to generate an adversarial example $x^{adv} = x + \epsilon$ by adding a perturbation ϵ to a original image, such that the target model $f^t(\cdot)$ is misled, resulting in $f^t(x^{adv}) \neq y$. In feature-level attacks, the generation of adversarial examples can be formulated as the following optimization problem (Detailed proof can be found in A.1 of the Appendix):

$$\arg\max_{x^{adv}} \langle W, f_k(x^{adv}) \rangle, s.t. ||x^{adv} - x||_p \leq \epsilon, \quad (8)$$

where W denotes the feature importance, $f_k(\cdot)$ denotes the k-th layer feature map of the surrogate model f, and $\langle \cdot, \cdot \rangle$ denotes the inner product. ℓ_p-norm is adopted to measure the distance between x and x^{adv}, and $p = \infty$ in this work.

3.2. Motivation

Building on the established threat model, we identify and analyze the issue of attack inefficiency in existing feature-level attacks. To provide a clear explanation, we need to introduce the following lemma.

Lemma 3.1. ***Cauchy–Schwarz inequality***. *Let $u, v \in \mathbb{R}^n$ be arbitrary vectors. Then*

$$\langle u, v \rangle \leq \|u\|_2 \|v\|_2$$

with equality holding if and only if $u = s \cdot v$ and $s \geq 0$.

Let us begin with the proof. First, we introduce $f_k(x)$, which is independent of the optimization parameter, and rewrite Eq. (8) as follows

$$\begin{aligned} &\arg\max_{x^{adv}} \langle W, f_k(x^{adv}) - f_k(x) \rangle, \\ &\quad s.t. \ \|x^{adv} - x\|_p \leq \epsilon \end{aligned} \quad (9)$$

Then, according to the Cauchy–Schwarz inequality, it follows that for any $x^{adv} \in \{x^{adv} | \|x^{adv} - x\|_p \leq \epsilon\}$, the following inequality still holds:

$$\begin{aligned} &\left\langle W, f_k(x^{adv}) - f_k(x) \right\rangle \\ \leq &\|W\|_2 \|f_k(x^{adv}) - f_k(x)\|_2 \end{aligned} \quad (10)$$

with equality holding if and only if $f_k(x^{adv}) - f_k(x) = s \cdot W$ $(s \geq 0)$. In other words, when

$$f_k(x^{adv}) = f_k(x) + s \cdot W, \ \ s.t. \ \|x^{adv} - x\|_p \leq \epsilon, \quad (11)$$

Eq. (9) achieves its maximum value $s \cdot \|W\|_2^2$, and its optimal solution is also the optimal solution of Eq. (8).

In summary, we only need to perturb the feature $f_k(x)$ in one step along the direction of W to obtain the perturbed feature $f_k(x) + s \cdot W$, and obtain adversarial example through feature inversion that satisfies $\|x^{adv} - x\|_p \leq \epsilon$. The obtained adversarial example is the optimal solution of Eq. (8). In practice, we additionally incorporate a clip function to ensure that the crafted adversarial examples remain within the ϵ-ball of x.

Consequently, we conclude that the feature-level attacks perturb the intermediate layer features of the surrogate model toward $f_k(x) + s \cdot W$. This suggests that **the effect of the existing feature-level attacks, which involve multiple perturbations in the pixel space, is effectively equivalent to perturbing the features in one step along the direction of feature importance in the feature space**.

Algorithm 1 Pixel2Feature Attack

Input: an original image x and ground-truth label y, surrogate model f, intermediate layer k, the step size λ, and the number of perturbations T
Output: The adversarial image x^{adv}
Initialize $x_0^{adv} = x, g_0 = 0, \mu = 1.$
for $t = 0$ **to** $T - 1$ **do**
  Calculate the feature importance W_t of x_t^{adv}
  $g_{t+1} = \mu \cdot g_t + W_t$
  $\tilde{f}_k = f_k(x_t^{adv}) + \lambda \cdot \frac{g_{t+1}}{||g_{t+1}||_2}$
  $x_{t+1}^{adv} = FeatureInversion(\tilde{f}_k)$
end for
return x_T^{adv}

To validate the above conclusion, we conducted the following experiments:

We employ existing feature-level attacks as baselines and derive corresponding feature inversion attacks based on the aforementioned conclusions. Specifically, we first perturb the features along the direction of feature importance as defined by the baseline in the feature space, and then generate corresponding adversarial examples through feature inversion. The aforementioned conclusions are validated if the attack success rates and perturbation distribution of adversarial examples generated by the baselines and the corresponding feature inversion attacks are similar. Due to page limitations, Table 1 presents the experimental results under the setting where the baseline attack is FIA and the surrogate model is Inception-v3 (Szegedy et al., 2016). The complete experimental results are provided in Table 5 in A.2 of the Appendix.

We employ the Structural Similarity Index Measure (SSIM) (Wang et al., 2004) to compute the distribution of the adversarial example perturbations generated by baselines and their corresponding feature inversion attacks. The computed SSIM value greater than 0.99 indicates that the perturbations are almost identical (The complete experimental results are provided in Table 5 in A.2 of the Appendix).The experimental results indicate that the attack success rates of the adversarial examples generated by baselines and their corresponding feature inversion attacks are comparable, and their perturbation distributions are almost identical. This validates the aforementioned conclusion.

Therefore, the effect of the existing feature-level attacks, which involve multiple perturbations in the pixel space to destroy important feature, is effectively equivalent to perturbing the features in one step along the direction of feature importance in the feature space. This inefficiency limits the transferability of existing feature-level attacks.

3.3. Pixel2Feature Attack

Inspired by the conclusion in Sec. 3.2, we propose P2FA to address the inefficiency of existing feature-level attacks in disrupting important features. It perturbs important features multiple times in the feature space along the direction of dynamic feature importance, thereby enhancing the transferability of adversarial examples. Specifically, as illustrated in Fig. 2, unlike existing feature-level attacks, P2FA transforms the perturbation space from pixel space to feature space. As described in Eq. (12), the optimization parameters are no longer adversarial examples in pixel space, but feature maps in feature space. We solve the optimization problem through multi-step iteration to achieve multiple perturbations of features in the feature space:

$$\arg\max_{f_k} J(f_k, y), \tag{12}$$

where f_k denotes the feature map of the k-th layer of the surrogate model f, and J denotes the cross-entropy loss. In gradient-based attacks, cross-entropy $J(x, y) = -\mathbb{1}_y \cdot \log \mathrm{softmax}(f(x))$, where $\mathbb{1}_y$ is the one-hot vector encoding class y, is used to update the input image x. We shift it to $J(f_k, y) = -\mathbb{1}_y \cdot \log \mathrm{softmax}(f_k^{post}(f_k))$ for feature updates, where f_k^{post} is the post-k-th-layer model part.

To ensure that the perturbation effectively disrupts important features, we adopt the feature importance to guide the perturbation direction. Unlike existing feature-level attacks that maintain fixed feature importance, we dynamically update the feature importance in each iteration to capture more refined perturbation directions. In addition, to further improve the adversarial transferability, we introduce momentum to stabilize the update direction within the feature space and prevent the perturbed features from converging to a local optimum. The detailed update process for solving Eq. (12) is as follows:

$$g_{t+1} = \mu \cdot g_t + W_t, \tag{13}$$

$$\tilde{f}_k = f_k + \lambda \cdot \frac{g_{t+1}}{||g_{t+1}||_2}, \tag{14}$$

where μ denotes the decay factor in the momentum, λ denotes the step size, and the feature importance W_t is derived from the state-of-the-art BFA (Wang et al., 2024b) and applied iteratively as follows:

$$W_t = \sum_{n=1}^{N} \nabla_{f_k(x_n^{IF})} J(f_k(x_n^{IF}), y), \tag{15}$$

where $x_n^{IF} = x_t^{adv} + \gamma \cdot \frac{\nabla_{x_{n-1}^{IF}} J(f_k(x_{n-1}^{IF}), y)}{\|\nabla_{x_{n-1}^{IF}} J(f_k(x_{n-1}^{IF}), y)\|_2}$, x_n^{IF} denotes the fitted image, x_t^{adv} denotes adversarial examples generated by white-box attacks, γ denotes the perturbation size, $f_k(x_n^{IF})$ denotes the black-box features of the fitted image. Finally, we employ the classical feature inversion algorithm (Du et al., 2018) to map the optimal perturbed features f_k^* back to the pixel space, generating adversarial examples. The detailed workflow of our P2FA is outlined in Algorithm 1. Code is available at: `https://github.com/WH-Lrp/P2FA`.

4. Experiments

4.1. Setup

Dataset. For a fair comparison, we adhere to previous work by utilizing the ImageNet-NIPS dataset (Kurakin et al., 2018), which comprises 1000 images for the NIPS 2017 adversarial competition.

Target Models. We employ seventeen classification models as target models to evaluate the performance of different attacks. Among these, seven are normally trained models: 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), ResNet-50 (Res-50) (He et al., 2016a), ResNet-152 (Res-152) (He et al., 2016a), VGG16 (Vgg-16) (Simonyan, 2014), and VGG19 (Vgg-19) (Simonyan, 2014). In addition, we select six advanced defense methods, including Adv-Inc-v3 (Kurakin et al., 2016), Adv-Ens-IncRes-v2 (Tramèr et al., 2017), random resizing and padding (R&P) (Xie et al., 2017), feature distillation (FD) (Liu et al., 2019), JPEG compression (JPEG) (Guo et al., 2017) and bit-depth compression (Bit-Red) (Liu et al., 2017). Finally, we select four vision transformers, including PiT-S (Heo et al., 2021), CaiT-S (Touvron et al., 2021b), DeiT-B (Touvron et al., 2021a), Swin-B (Liu et al., 2021).

Baseline Methods. We select six advanced feature-level attack methods as our baselines: FIA (Wang et al., 2021), RPA (Zhang et al., 2022b), NAA (Zhang et al., 2022a), DANAA (Jin et al., 2023), SFVA (Ren et al., 2023), and BFA (Wang et al., 2024b). In addition, we integrated all the methods with two classic input transformation methods: PIM (Gao et al., 2020) and DIM (Xie et al., 2019), to compare the compatibility of the different attacks. When PIM and DIM are combined with the baselines and P2FA, we denote them as PIDI-FIA, PIDI-RPA, PIDI-NAA, PIDI-DANAA, PIDI-SFVA, PIDI-BFA, PIDI-P2FA. We also integrated all the methods with two up-to-date input transformation methods: BSR (Wang et al., 2024a) and SIA (Wang et al., 2023)(The experimental results are provided in Table 6 and Table 7 in A.3 of the Appendix.)

Implementation Details. For a fair comparison, we adhere

Table 2. Success rate of different attacks against normally trained models. The first column shows surrogate models, the first row lists target models and the last column represents the average attack success rate. The best results are highlighted in **bold**.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-50	Res-152	Vgg-16	Vgg-19	Average
Inc-v3	FIA	98.8	86.2	78.7	82.0	75.4	85.1	83.7	84.3
	RPA	98.7	86.6	81.8	85.4	79.5	86.4	85.0	86.2
	NAA	97.7	81.0	73.9	78.4	72.1	81.9	79.2	80.6
	DANAA	98.1	82.3	76.7	81.3	73.6	84.8	81.8	82.7
	SFVA	98.8	87.0	83.0	86.0	80.9	87.7	85.2	86.9
	BFA	**100.0**	95.1	92.0	91.9	90.5	93.3	92.4	93.6
	P2FA(**Ours**)	**100.0**	**96.5**	**94.9**	**95.2**	**92.5**	**94.8**	**96.0**	**95.7**
Inc-v4	FIA	83.9	96.7	77.2	79.6	72.6	84.8	83.2	82.6
	RPA	87.5	98.1	80.3	82.8	78.6	88.1	87.6	86.1
	NAA	81.4	97.8	73.7	77.3	69.9	84.8	82.5	81.1
	DANAA	82.1	98.0	77.5	81.1	74.7	85.8	84.9	83.4
	SFVA	86.7	98.2	79.0	82.6	78.2	88.6	87.1	85.8
	BFA	94.8	99.5	90.6	91.4	88.9	93.6	92.6	93.1
	P2FA(**Ours**)	**96.3**	**100.0**	**92.2**	**93.6**	**91.1**	**95.8**	**95.0**	**94.9**
IncRes-v2	FIA	75.7	74.5	92.9	69.8	62.5	77.0	73.7	75.2
	RPA	78.8	78.9	91.9	76.9	68.0	78.6	76.9	78.6
	NAA	70.4	70.0	91.9	68.4	59.0	73.9	72.1	72.2
	DANAA	76.9	76.1	94.2	73.5	66.7	78.4	77.3	77.6
	SFVA	81.0	78.7	93.7	77.7	71.3	81.1	79.8	80.5
	BFA	92.3	91.6	99.3	87.1	83.2	87.0	86.3	89.5
	P2FA(**Ours**)	**93.7**	**93.5**	**100.0**	**90.8**	**86.5**	**91.1**	**91.5**	**92.4**
Res-152	FIA	76.2	78.4	62.5	98.0	**100.0**	91.9	88.7	85.1
	RPA	87.4	86.3	74.2	99.0	**100.0**	95.1	94.4	90.9
	NAA	81.7	79.5	71.8	97.8	99.9	92.5	91.2	87.8
	DANAA	84.8	83.2	74.7	98.4	**100.0**	93.1	92.7	89.6
	SFVA	86.8	85.6	78.2	99.4	**100.0**	94.9	94.2	91.3
	BFA	92.7	92.3	85.4	99.6	**100.0**	97.0	96.9	94.8
	P2FA(**Ours**)	**94.9**	**94.4**	**88.7**	**99.9**	**100.0**	**98.0**	**98.4**	**96.3**

to the parameter settings of FIA (Wang et al., 2021). Specifically, we set the maximum perturbation to $\epsilon = 16$ and the number of integrations steps for the aggregated gradient to $N = 30$. In addition, we set the decay factor to $\mu = 1.0$ for all the baselines, as they are optimized using the momentum method. For the input transformation methods, we set the transformation probability of the DIM to 0.7, the amplification factor of the PIM to 2.5 and the kernel size to 3. We select consistent intermediate layers for the feature-level attacks: Mixed_5b for the Inc-v3 model, feature.6 for the Inc-v4 model, Conv2d_4a for the IncRes-v2 model and the last layer of block2 for the Res-152 model. For the proposed P2FA, We set the step size to 10^5, the number of perturbations to 3, and utilize the feature importance derived from BFA.

4.2. Comparison of Transferability

In this section, we compare the transferability between the proposed P2FA and the baselines. We select Inc-v3, Inc-v4, IncRes-v2, and Res-152 as surrogate model to generate adversarial examples, respectively, and then attack other normally trained models, defense models and vision transformers.

Attacking Normally Trained Models. Table 2 report the attack success rates of the adversarial examples generated by the proposed P2FA and baselines on normally trained models. The results demonstrate that P2FA exhibits a significant advantage over baseline attacks, with an average improvement of 2.1% compared to the best results of baselines. In addition, our method can be integrated with the classic input transformation methods, PIM and DIM, to further improve the transferability. As shown in Table 3, the attack success rate of PIDI-P2FA significantly improves to 97.4% after combining PIM and DIM. In contrast, PIDI-BFA, the top-performing baseline, achieves an average attack success rate of 93.3%. Furthermore, Table 6 also demonstrates the superiority of P2FA when combined with BSR or SIA in attacking normally trained models. All these experiments validate the effectiveness of P2FA.

Attacking Defense Models. Next, we evaluate the effectiveness of our proposed P2FA and other baseline attacks on the defense models. To enhance transferability, we integrate all attack methods with the classic input transformation methods PIM and DIM. The results in Table 4 demonstrate that

Table 3. Success rate of different attacks integrated with PIM and DIM against normally trained models. The first column shows source models, the first row lists target models and the last column represents the average attack success rate. The best results are highlighted in **bold**.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-50	Res-152	Vgg-16	Vgg-19	Average
Inc-v3	PIDI-FIA	99.1	89.3	84.7	86.9	79.0	87.2	86.8	87.6
	PIDI-RPA	99.7	90.4	86.9	89.7	85.2	89.9	89.1	90.1
	PIDI-NAA	98.0	83.0	78.6	83.3	75.9	85.3	83.0	83.9
	PIDI-DANAA	98.9	86.1	81.9	85.7	81.2	88.9	85.8	86.9
	PIDI-SFVA	99.1	88.8	85.4	88.3	83.7	89.6	88.4	89.0
	PIDI-BFA	**100.0**	95.7	93.0	93.2	91.2	93.8	92.7	94.2
	PIDI-P2FA(**Ours**)	**100.0**	**98.4**	**97.3**	**97.3**	**96.7**	**97.0**	**97.4**	**97.7**
Inc-v4	PIDI-FIA	87.6	97.9	83.0	84.6	79.4	88.1	87.5	86.9
	PIDI-RPA	91.4	99.1	86.1	87.1	84.3	90.0	89.6	89.7
	PIDI-NAA	84.5	98.1	79.1	83.4	75.7	88.4	85.2	84.9
	PIDI-DANAA	85.7	97.5	80.8	84.1	78.6	88.0	87.2	86.0
	PIDI-SFVA	88.0	98.4	81.1	86.0	81.6	90.4	87.9	87.6
	PIDI-BFA	95.3	99.6	91.8	92.9	89.4	93.2	93.1	93.6
	PIDI-P2FA(**Ours**)	**98.9**	**100.0**	**96.9**	**97.2**	**95.8**	**97.9**	**97.9**	**97.8**
IncRes-v2	PIDI-FIA	80.3	77.2	94.5	75.8	69.2	78.3	78.0	79.0
	PIDI-RPA	83.3	81.2	94.1	80.0	73.3	82.0	78.9	81.8
	PIDI-NAA	72.4	69.5	92.8	69.0	60.4	73.4	71.5	72.7
	PIDI-DANAA	80.2	78.8	93.0	77.9	68.8	80.8	77.3	79.5
	PIDI-SFVA	81.8	79.1	94.9	78.3	72.7	79.9	78.2	80.7
	PIDI-BFA	92.6	90.8	99.1	87.2	83.1	88.1	86.3	89.6
	PIDI-P2FA(**Ours**)	**96.4**	**96.2**	**100.0**	**94.3**	**91.8**	**93.7**	**93.4**	**95.1**
Res-152	PIDI-FIA	88.6	87.6	79.6	98.8	**100.0**	95.3	94.2	92.0
	PIDI-RPA	91.5	90.0	83.2	99.6	**100.0**	96.3	96.2	93.8
	PIDI-NAA	85.4	84.6	79.9	98.2	99.9	94.3	92.3	90.7
	PIDI-DANAA	89.0	86.9	83.3	98.8	**100.0**	95.4	94.3	92.5
	PIDI-SFVA	90.7	89.3	83.3	99.6	**100.0**	96.2	96.6	93.7
	PIDI-BFA	93.9	93.1	89.2	99.7	**100.0**	97.4	97.4	95.8
	PIDI-P2FA(**Ours**)	**99.1**	**98.4**	**97.7**	**99.9**	**100.0**	**99.4**	**99.2**	**99.1**

our proposed P2FA achieves an average attack success rate of 62.6%, surpassing all baselines by over 7.0%. Furthermore, Table 7 also demonstrates the superiority of P2FA when combined with BSR or SIA in attacking defense models. This indicates that our attack method is highly effective in targeting defense models.

Attacking Vision Transformers. Finally, we evaluate the effectiveness of our proposed P2FA and other baseline attacks on the vision transformers. We integrate all attack methods with the classic input transformation methods PIM and DIM. The results in Table 4 demonstrate that our proposed P2FA achieves an average attack success rate of 74.9%, surpassing all baselines by over 15%. Furthermore, Table 7 also demonstrates the superiority of P2FA when combined with BSR or SIA in attacking vision transformers. This indicates that our attack method poses a significant threat to advanced vision transformers.

4.3. Ablation Study

In this section, we conduct an ablation study to analyze the three key factors that affect our proposed P2FA. The first factor is the step size λ, which determines the magnitude of perturbations along the direction of feature importance in the feature space. The second factor is the number of perturbations T, representing how many times features are perturbed along the direction of feature importance. The final factor is the effect of the choice of feature importance on the performance of our algorithm. Due to page limitations, the figures for the ablation study are shown in A.4 of the Appendix.

Step Size. We analyze the effect of using different step sizes on P2FA. Using Inc-v3 as the surrogate model, we evaluate step sizes of 1, 10, 10^2, 10^3, 10^4, 10^5, 10^6 to analyze the transferability of the adversarial examples generated with different step sizes. The results are shown in Fig. 3, indicate that the attack success rate stabilizes when the step size exceeds 10^4, reaching its peak at 10^5. Consequently, we select 10^5 as the optimal step size.

Number of Perturbations. We analyze the effect of the number of perturbations along the direction of feature importance in the feature space on P2FA. Using Inc-v3 as a surrogate model, we vary the number of perturbations from

Table 4. Success rate of different attacks integrated with PIM and DIM against defense models and vision transformers. The first column shows source models, the first row lists target models and the last column represents the average attack success rate. The best results are highlighted in **bold**.

Model	Attack	Adv-Inc-v3	Adv-Ens-IncRes-v2	R&P	FD	JPEG	Bit-Red	PiT-S	CaiT-S	DeiT-B	Swin-B	Average
	PIDI-FIA	65.0	34.4	37.3	53.9	46.7	34.7	54.6	39.7	43.4	27.4	43.7
	PIDI-RPA	66.7	41.2	45.7	52.8	50.8	41.9	62.2	47.2	49.0	34.1	49.2
	PIDI-NAA	55.4	34.8	39.3	41.6	45.3	34.4	56.4	40.4	44.9	32.8	42.5
Inc-v3	PIDI-DANAA	59.7	39.2	48.0	47.4	50.8	42.3	63.5	48.5	51.0	38.2	48.9
	PIDI-SFVA	61.3	40.5	46.1	48.8	49.2	40.3	62.2	46.0	51.5	35.9	48.2
	PIDI-BFA	72.4	45.9	50.5	57.8	57.1	46.3	73.4	59.3	59.6	46.2	56.9
	PIDI-P2FA(**Ours**)	**76.7**	**51.8**	**58.0**	**63.3**	**61.3**	**52.8**	**85.2**	**73.0**	**75.9**	**60.3**	**65.8**
	PIDI-FIA	60.7	36.6	40.7	52.2	43.7	37.3	54.7	40.1	40.3	32.5	43.9
	PIDI-RPA	62.2	42.0	44.8	52.3	49.0	42.1	62.4	45.8	47.6	39.3	48.8
	PIDI-NAA	51.5	35.5	38.9	41.9	44.5	35.8	58.2	41.9	45.1	37.4	43.1
Inc-v4	PIDI-DANAA	55.2	40.7	46.4	45.7	44.2	44.3	61.5	49.1	50.5	43.5	48.1
	PIDI-SFVA	58.0	44.2	47.9	47.7	47.4	44.5	63.7	50.4	50.7	43.6	49.8
	PIDI-BFA	66.5	46.6	51.4	53.5	51.4	46.1	72.7	56.4	58.7	49.6	55.3
	PIDI-P2FA(**Ours**)	**71.8**	**52.5**	**57.9**	**59.6**	**55.0**	**52.1**	**84.9**	**73.2**	**74.6**	**65.7**	**64.7**
	PIDI-FIA	64.2	40.9	42.5	54.6	44.7	41.2	43.2	31.7	32.0	21.8	41.7
	PIDI-RPA	66.6	52.1	52.7	58.1	52.6	52.1	53.1	43.1	43.3	30.4	50.4
	PIDI-NAA	51.8	37.8	38.6	46.6	42.3	37.9	42.1	30.9	33.5	24.5	38.6
IncRes-v2	PIDI-DANAA	58.6	47.8	52.5	51.8	47.8	50.7	53.4	41.5	42.4	34.6	48.1
	PIDI-SFVA	58.4	46.9	47.7	52.1	50.1	46.9	51.4	43.6	45.1	33.4	47.6
	PIDI-BFA	72.6	61.2	61.4	60.8	56.8	60.8	64.6	52.9	53.1	40.8	58.5
	PIDI-P2FA(**Ours**)	**79.9**	**65.8**	**68.8**	**67.7**	**62.9**	**68.0**	**78.5**	**68.7**	**67.4**	**54.7**	**68.2**
	PIDI-FIA	57.3	33.5	38.9	54.1	45.5	34.5	52.9	36.7	41.8	30.2	42.5
	PIDI-RPA	59.8	40.4	46.6	53.7	51.1	39.8	65.1	47.3	53.5	42.2	50.0
	PIDI-NAA	52.8	38.4	44.7	47.1	48.4	38.3	68.1	50.9	56.5	47.6	49.3
Res-152	PIDI-DANAA	55.0	41.2	49.0	50.6	50.6	44.1	72.5	56.0	60.5	54.1	53.4
	PIDI-SFVA	57.6	42.5	50.2	52.8	51.9	42.9	71.4	56.3	61.2	52.6	53.9
	PIDI-BFA	61.6	45.0	50.6	56.4	55.8	45.3	78.2	59.7	66.6	55.5	57.5
	PIDI-P2FA(**Ours**)	**71.6**	**55.2**	**62.6**	**66.8**	**63.6**	**55.8**	**92.4**	**82.2**	**84.9**	**76.4**	**71.2**

1 to 10 to analyze their effect on transferability. The results, as illustrated in Fig. 4, demonstrate that the attack success rate peaks when the number of perturbations is 3. Consequently, we select 3 as the optimal number of perturbations for our study.

Feature Importance. We analyze the impact of employing different feature importance assessment methods on P2FA. Using Inc-v3 as a surrogate model, we incorporate the feature importance metrics from FIA, RPA, NAA, DANAA, SFVA, and BFA into our algorithm. The results are shown in Fig. 5, indicating that the highest performance is achieved when utilizing the feature importance derived from BFA. Therefore, we select the feature importance of BFA and dynamically compute it in each iteration.

5. Conclusion

In this paper, we propose P2FA to generate highly transferable adversarial examples. The proposed P2FA shifts the space of perturbation from pixel space to feature space directly and perturbs the features multiple times along the direction of feature importance in the feature space to more efficiently destroy important features. We conducted extensive experiments to demonstrate the superior performance of P2FA as compared to those state-of-the-art methods, and our method can serve as a benchmark for evaluating the robustness of various models.

Acknowledgment

This work is supported by National Natural Science Foundation of China (Grants No. 62472229, U24B20179, U23A20305, U23B2022, 62371145, 62272255, 62302248), the National Key R and D Program of China (Grants No. 2021QY0700), Zhongyuan Science and Technology Innovation Leading Talent Project of China (Grants No. 214200510019) and the Open Project the State Key Laboratory of Tibetan Intelligent Information Processing and Applica tion (Key Laboratory of Tibetan information processing, Ministry of Education) (Grants No. 2024-Z-003).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.

Dong, Y., Pang, T., Su, H., and 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.

Du, M., Liu, N., Song, Q., and Hu, X. Towards explanation of dnn-based prediction with guided feature inversion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1358–1367, 2018.

Ganeshan, A., BS, V., and Babu, R. V. Fda: Feature disruptive attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8069–8079, 2019.

Gao, L., Zhang, Q., Song, J., Liu, X., and Shen, H. T. Patch-wise attack for fooling deep neural network. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pp. 307–322. Springer, 2020.

Girshick, R. Fast r-cnn. *arXiv preprint arXiv:1504.08083*, 2015.

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

Guo, C., Rana, M., Cisse, M., and Van Der Maaten, L. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

He, K., Zhang, X., Ren, S., and 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, 2016b.

Heo, B., Yun, S., Han, D., Chun, S., Choe, J., and Oh, S. J. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11936–11945, 2021.

Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box adversarial attacks with limited queries and information. In *International conference on machine learning*, pp. 2137–2146. PMLR, 2018.

Jin, Z., Zhu, Z., Wang, X., Zhang, J., Shen, J., and Chen, H. Danaa: Towards transferable attacks with double adversarial neuron attribution. In *International Conference on Advanced Data Mining and Applications*, pp. 456–470. Springer, 2023.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Kurakin, A., Goodfellow, I., Bengio, S., Dong, Y., Liao, F., Liang, M., Pang, T., Zhu, J., Hu, X., Xie, C., et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*, pp. 195–231. Springer, 2018.

Lin, J., Song, C., He, K., Wang, L., and Hopcroft, J. E. Nesterov accelerated gradient and scale invariance for adversarial attacks. *arXiv preprint arXiv:1908.06281*, 2019.

Liu, J., Sun, W., and Liu, Y. Bit-depth enhancement via convolutional neural network. In *International Forum on Digital TV and Wireless Multimedia Communications*, pp. 255–264. Springer, 2017.

Liu, Z., Liu, Q., Liu, T., Xu, N., Lin, X., Wang, Y., and Wen, W. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 860–868. IEEE, 2019.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Lu, Y., Jia, Y., Wang, J., Li, B., Chai, W., Carin, L., and Velipasalar, S. Enhancing cross-task black-box transferability of adversarial examples with dispersion reduction. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 940–949, 2020.

Naseer, M., Khan, S. H., Rahman, S., and Porikli, F. Task-generalizable adversarial attack based on perceptual metric. *arXiv preprint arXiv:1811.09020*, 2018.

Ren, Y., Zhu, H., Sui, X., and Liu, C. Crafting transferable adversarial examples via contaminating the salient feature variance. *Information Sciences*, 644:119273, 2023.

Simonyan, K. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Szegedy, C. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and 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.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021a.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42, 2021b.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Wang, K., He, X., Wang, W., and Wang, X. Boosting adversarial transferability by block shuffle and rotation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24336–24346, 2024a.

Wang, M., Wang, J., Ma, B., and Luo, X. Improving the transferability of adversarial examples through black-box feature attacks. *Neurocomputing*, 595:127863, 2024b.

Wang, X. and He, K. Enhancing the transferability of adversarial attacks through variance tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1924–1933, 2021.

Wang, X., Zhang, Z., and Zhang, J. Structure invariant transformation for better adversarial transferability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4607–4619, 2023.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Wang, Z., Guo, H., Zhang, Z., Liu, W., Qin, Z., and Ren, K. Feature importance-aware transferable adversarial attacks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7639–7648, 2021.

Xiao, C., Li, B., Zhu, J.-Y., He, W., Liu, M., and Song, D. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., and 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.

Zhang, J., Wu, W., Huang, J.-t., Huang, Y., Wang, W., Su, Y., and 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, 2022a.

Zhang, Y., Tan, Y.-a., Chen, T., Liu, X., Zhang, Q., and Li, Y. Enhancing the transferability of adversarial examples with random patch. In *IJCAI*, pp. 1672–1678, 2022b.

Zhou, W., Hou, X., Chen, Y., Tang, M., Huang, X., Gan, X., and Yang, Y. Transferable adversarial perturbations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 452–467, 2018.

A. Appendix.

A.1. Proof

FIA: The optimization problem of FIA is

$$\arg\min_{x^{adv}} \sum (\bar{\Delta}_k^x \odot f_k(x^{adv})) = \left\langle \bar{\Delta}_k^x, f_k(x^{adv}) \right\rangle, \tag{16}$$

where $\bar{\Delta}_k^x$ denotes the aggregate gradient,

$$\bar{\Delta}_k^x = \frac{\sum_{n=1}^{N} \Delta_k^{\mathcal{T}_n(x)}}{\| \sum_{n=1}^{N} \Delta_k^{\mathcal{T}_n(x)} \|_2}, \;\; \mathcal{T}_n(x) = x \odot M_{p_d}^n, \;\; M_{p_d}^n \sim Bernoulli(1 - p_d), \tag{17}$$

where $\Delta_k^x = \frac{\partial l(x,y)}{\partial f_k(x)}$, and $l(\cdot, y)$ denotes the logits output with respect to the true label y. Let $W = -\bar{\Delta}_k^x$, The optimization problem of FIA can ultimately be rewritten as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle. \tag{18}$$

RPA: The only difference between RPA and FIA is the different transformation $\mathcal{T}$ in Eq. (17), which will result in different feature importance W, but does not affect the structure of the optimization problem. Therefore, the optimization problem of RPA can also be rewritten as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle. \tag{19}$$

NAA: The optimization problem of NAA is

$$\arg\min_{x^{adv}} \sum_{A_k^{(j)} \geq 0} [h_p(A_k^{(j)})] - \gamma \sum_{A_k^{(j)} < 0} [h_n(-A_k^{(j)})], \tag{20}$$

where

$$\begin{aligned} A_k &= (f_k(x^{adv}) - f_k(x')) \odot IA \\ &= (f_k(x^{adv}) - f_k(x')) \odot \frac{1}{n} \sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)}, \end{aligned} \tag{21}$$

where $F(\cdot, y)$ denotes the softmax output of the true label y and $x_m = (1 - \frac{m}{n})x' + \frac{m}{n}x$. In practical, $\gamma = 1$ and $f_p(\cdot), f_n(\cdot)$ are linear functions mapping to themselves, i.e.$h_p(x) = x, h_n(x) = x$. Eq. (20) can be rewritten as

$$\arg\min_{x^{adv}} \sum_j A_k^{(j)} = \left\langle f_k(x^{adv}) - f_k(x'), \frac{1}{n} \sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)} \right\rangle, \tag{22}$$

which is equivalent to

$$\arg\min_{x^{adv}} \left\langle f_k(x^{adv}), \frac{\sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)}}{\left\| \sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)} \right\|_2} \right\rangle. \tag{23}$$

Let $W = -\frac{\sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)}}{\left\| \sum_{m=1}^{n} \frac{\partial F(x_m, y)}{\partial f_k(x_m)} \right\|_2}$, The optimization problem of NAA can eventually be rewritten as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle. \tag{24}$$

DANAA: DANAA differs from NAA only in that the paths used to compute x_m in W in Eq. (23) are different, which will result in a different W, but does not affect the structure of the optimization problem. Therefore, the optimization problem of DANAA can also be written as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle . \tag{25}$$

SFVA: The optimization problem of SFVA is

$$\arg\max_{x^{adv}} \lambda \cdot \tau_P(PV_k) + \tau_N(NV_k). \tag{26}$$

where the positive salient variance PV_k of the k-th layer feature map is denoted as

$$PV_k = \sum [M^p(x) - M^p(x^{adv})], \tag{27}$$

the positive salient maps $M^p(x)$ is denoted as

$$M_i^p(x) = \begin{cases} M_i(x) & \text{if } M_i(x) \geq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{28}$$

the negative salient variance NV_k of the k-th layer feature map is denoted as

$$NV_k = \sum [M^n(x) - M^n(x^{adv})], \tag{29}$$

the negative salient maps $M^n(x)$ is denoted as

$$M_i^n(x) = \begin{cases} M_i(x) & \text{if } M_i(x) \leq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{30}$$

the salient feature maps $M(x) = \hat{W}^* \odot f_k(x)$, where$\hat{W}^*$ denotes the optimal feature weights. In practical, $\lambda = 1$ and $\tau_P(\cdot), \tau_N(\cdot)$ are identity functions, i.e. $\tau_P(x) = x, \tau_N(x) = x$. Eq. (26) can be rewritten as

$$\begin{aligned} \arg\max_{x^{adv}} PV_k + NV_k &= \sum [M^p(x) - M^p(x^{adv})] + \sum [M^n(x) - M^n(x^{adv})] \\ &= \left[\sum M^p(x) + \sum M^n(x)\right] - \left[\sum M^p(x^{adv}) + \sum M^n(x^{adv})\right] \\ &= \sum_i M_i(x) - \sum_i M_i(x^{adv}) \\ &= \sum_i \left[\hat{W}^* \odot f_k(x) - \hat{W}^* \odot f_k(x^{adv})\right] \\ &= \left\langle \hat{W}^*, f_k(x) - f_k(x^{adv}) \right\rangle . \end{aligned} \tag{31}$$

Let $W = -\hat{W}^*$ and removing the terms that are not related to the optimization parameters, the optimization problem of SFVA can finally be rewritten as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle . \tag{32}$$

BFA: The optimization problem of BFA is

$$\arg\max_{x^{adv}} \sum_{F \geq 0} F - \omega \cdot \sum_{F<0} -F. \tag{33}$$

where $F = f_k(x^{adv}) \odot I + I^2$ and I denotes the fitted gradient. In the official code, ω is set to 1.0 which degrades the optimization problem to

$$\arg\max_{x^{adv}} \sum F = \sum f_k(x^{adv}) \odot I + I^2. \tag{34}$$

Since I^2 is a constant term independent of the optimization parameters, Eq. (34) can be rewritten as

$$\arg\max_{x^{adv}} \sum f_k(x^{adv}) \odot I = \left\langle I, f_k(x^{adv}) \right\rangle . \tag{35}$$

Let $W = I$, the optimization problem of BFA can be finally rewritten as

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle . \tag{36}$$

In summary, the above optimization problems of feature-level attacks are all equivalent to

$$\arg\max_{x^{adv}} \left\langle W, f_k(x^{adv}) \right\rangle . \tag{37}$$

where W denotes the feature importance.

A.2. The Attack Success Rates of the Adversarial Examples Generated by Feature-level Attacks and Their Corresponding Feature Inversion Attacks (Table 5)

A.3. Success Rate of Different Attacks Integrated with BSR or SIA Against Normally Trained Models, Defense Models and Vision Transformers (Table 6, Table 7)

A.4. Ablation Study Figures(Fig. 3, Fig. 4, Fig. 5)

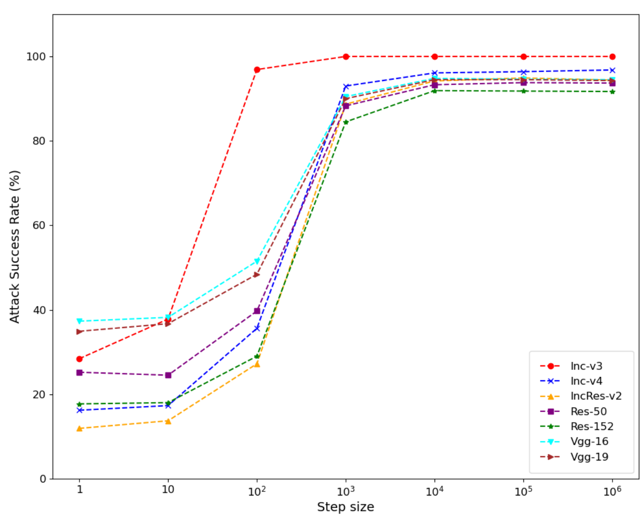

Figure 3. The effect of the choice of step size on attack success rate. Using Inc-v3 as a surrogate model, different step sizes are chosen to generate adversarial examples and their success rates are reported against different target models.

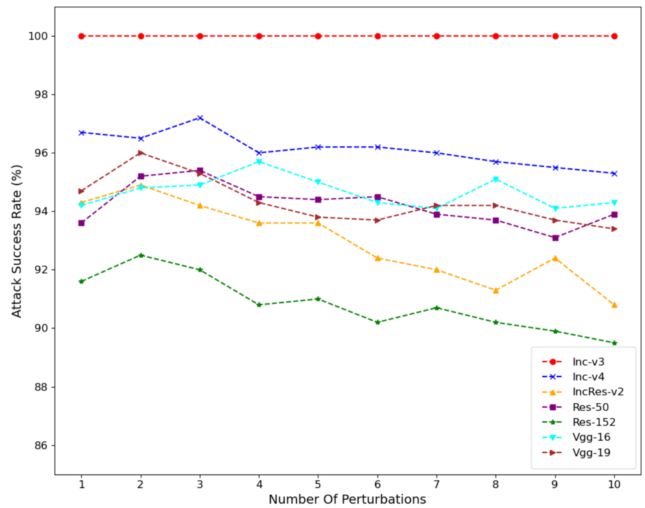

Figure 4. The effect of the choice of the number of perturbations on attack success rate. Using Inc-v3 as a surrogate model, different numbers of perturbations are chosen to generate adversarial examples and their success rates are reported against different target models.

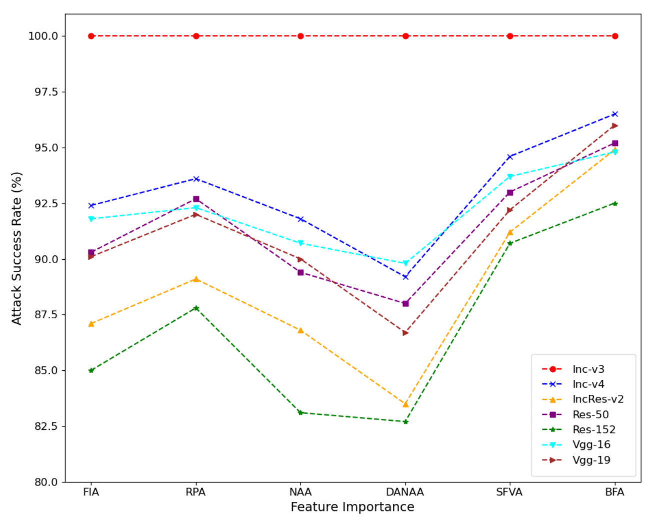

Figure 5. The effect of feature importance selection on attack success rate. Using Inc-v3 as a surrogate model, the proposed feature importance assessment methods in different feature-level attacks are selected to generate adversarial examples and their success rates are reported against different target models.

Table 5. The attack success rates and the value of SSIM of the adversarial examples generated by feature-level attacks and their corresponding Feature Inversion Attacks. The first column shows surrogate models, the first row lists target models and the last column represents the value of SSIM. Average represent the average attack success rates. * denotes Feature Inversion Attack.

Model	Attack	Inception-v3	Inception-v4	Inception-ResNet-V2	ResNet-50	ResNet-152	Vgg-16	Vgg-19	Average	SSIM
Inc-v3	FIA	98.8	86.2	78.7	82.0	75.4	85.1	83.7	84.3	0.9978
	FIA*	99.3	87.2	79.4	82.9	75.8	85.8	84.5	85.0	
	RPA	98.7	86.6	81.8	85.4	79.5	86.4	85.0	86.2	0.9966
	RPA*	99.0	87.1	82.4	86.0	80.4	86.6	85.9	86.8	
	NAA	97.7	81.0	73.9	78.4	72.1	81.9	79.2	80.6	0.9972
	NAA*	98.6	81.8	74.7	79.2	73.0	82.3	80.1	81.4	
	DANAA	98.1	82.3	76.7	81.3	73.6	84.8	81.8	82.7	0.9945
	DANAA*	98.8	83.3	77.5	82.1	74.5	85.3	82.4	83.4	
	SFVA	98.8	87.0	83.0	86.0	80.9	87.7	85.2	86.9	0.9937
	SFVA*	99.3	87.7	84.1	86.2	81.2	88.0	86.1	87.5	
	BFA	100.0	95.1	92.0	91.9	90.5	93.3	92.4	93.6	0.9945
	BFA*	100.0	96.1	92.3	92.6	91.6	94.2	92.7	94.2	
Inc-v4	FIA	83.9	96.7	77.2	79.6	72.6	84.8	83.2	82.6	0.9967
	FIA*	84.4	97.2	78.1	80.3	72.7	85.2	84.1	83.1	
	RPA	87.5	98.1	80.3	82.8	78.6	88.1	87.6	86.1	0.9974
	RPA*	88.1	98.7	81.1	83.3	79.4	89.0	88.2	86.8	
	NAA	81.4	97.8	73.7	77.3	69.9	84.8	82.5	81.1	0.9956
	NAA*	82.2	98.4	74.7	78.1	70.1	85.5	83.2	81.7	
	DANAA	82.1	98.0	77.5	81.1	74.7	85.8	84.9	83.4	0.9956
	DANAA*	82.5	98.6	78.7	82.2	75.2	86.1	85.3	84.1	
	SFVA	86.7	98.2	79.0	82.6	78.2	88.6	87.1	85.8	0.9984
	SFVA*	87.2	98.6	79.9	83.3	79.2	88.9	87.8	86.4	
	BFA	94.8	99.5	90.6	91.4	88.9	93.6	92.6	93.1	0.9947
	BFA*	95.3	99.6	91.2	91.9	89.6	94.2	93.4	93.6	
IncRes-v2	FIA	75.7	74.5	92.9	69.8	62.5	77.0	73.7	75.2	0.9936
	FIA*	76.6	75.1	93.0	70.8	63.2	77.7	74.2	75.8	
	RPA	78.8	78.9	91.9	76.9	68.0	78.6	76.9	78.6	0.9951
	RPA*	79.2	79.6	92.5	77.8	68.9	79.7	77.4	79.3	
	NAA	70.4	70.0	91.9	68.4	59.0	73.9	72.1	72.2	0.9977
	NAA*	70.9	70.7	92.7	69.2	59.9	74.5	73.1	73.0	
	DANAA	76.9	76.1	94.2	73.5	66.7	78.4	77.3	77.6	0.9959
	DANAA*	77.4	76.7	94.9	74.4	67.2	79.3	78.3	78.3	
	SFVA	81.0	78.7	93.7	77.7	71.3	81.1	79.8	80.5	0.9964
	SFVA*	81.8	79.4	94.3	78.7	71.8	81.9	80.3	81.2	
	BFA	92.3	91.6	99.3	87.1	83.2	87.0	86.3	89.5	0.9951
	BFA*	93.2	92.6	99.5	88.0	84.1	87.8	87.1	90.3	
Res-152	FIA	76.2	78.4	62.5	98.0	100.0	91.9	88.7	85.1	0.9973
	FIA*	76.9	79.2	63.2	98.5	100.0	92.5	89.5	85.7	
	RPA	87.4	86.3	74.2	99.0	100.0	95.1	94.4	90.9	0.9975
	RPA*	87.9	87.2	74.8	99.3	100.0	95.9	95.2	91.5	
	NAA	81.7	79.5	71.8	97.8	99.9	92.5	91.2	87.8	0.9963
	NAA*	82.3	80.4	72.2	98.5	99.9	93.4	92.0	88.4	
	DANAA	84.8	83.2	74.7	98.4	100.0	93.1	92.7	89.6	0.9966
	DANAA*	85.6	84.1	75.1	98.8	100.0	93.7	93.1	90.1	
	SFVA	86.8	85.6	78.2	99.4	100.0	94.9	94.2	91.3	0.9961
	SFVA*	87.2	86.4	78.9	99.6	100.0	95.5	95.2	91.8	
	BFA	92.7	92.3	85.4	99.6	100.0	97.0	96.9	94.8	0.9976
	BFA*	93.3	92.8	86.2	99.7	100.0	97.8	97.8	95.4	

Table 6. Success rate of different attacks integrated with BSR or SIA against normally trained models. The first column shows source models, the first row lists target models and the last column represents the average attack success rate. The best results are highlighted in **bold**.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-50	Res-152	Vgg-16	Vgg-19	Average
Inc-v3	BSR-FIA	99.7	86.3	78.9	84.9	75.4	88.1	85.1	85.5
	BSR-RPA	99.8	89.4	84.3	87.4	79.9	89.6	88.1	88.4
	BSR-NAA	99.9	85.1	78.7	83.0	76.6	86.9	84.1	84.9
	BSR-DANAA	99.2	85.8	79.0	82.5	76.8	87.0	85.6	85.1
	BSR-SFVA	99.9	87.8	83.4	85.8	78.8	88.1	85.8	87.1
	BSR-BFA	**100.0**	91.2	86.7	88.5	84.5	90.6	89.6	90.2
	BSR-P2FA(**Ours**)	**100.0**	**98.7**	**96.2**	**97.3**	**96.2**	**97.3**	**97.7**	**97.6**
	SIA-FIA	**100.0**	94.0	89.1	92.0	86.8	93.5	93.2	92.7
	SIA-RPA	99.9	94.8	91.6	93.6	91.0	94.3	94.9	94.3
	SIA-NAA	99.9	93.9	90.2	92.4	88.9	92.8	92.2	92.9
	SIA-DANAA	99.9	95.3	91.8	94.0	90.6	94.5	93.1	94.2
	SIA-SFVA	**100.0**	95.1	91.3	94.8	90.2	94.2	93.5	94.2
	SIA-BFA	**100.0**	97.0	94.7	96.4	93.9	96.3	96.1	96.3
	SIA-P2FA(**Ours**)	**100.0**	**99.5**	**98.9**	**99.4**	**98.6**	**99.3**	**99.5**	**99.3**
Inc-v4	BSR-FIA	88.1	99.5	80.1	86.2	79.3	92.6	89.9	88.0
	BSR-RPA	91.4	**100.0**	82.8	87.8	81.5	92.8	92.4	89.8
	BSR-NAA	87.8	99.9	79.2	83.4	75.9	89.8	88.4	86.3
	BSR-DANAA	87.9	98.6	80.9	86.0	79.1	90.9	89.6	87.6
	BSR-SFVA	90.3	99.6	80.5	86.0	79.4	91.0	89.9	88.1
	BSR-BFA	92.6	99.4	86.4	89.6	84.0	93.4	93.0	91.2
	BSR-P2FA(**Ours**)	**98.8**	**100.0**	**97.3**	**98.1**	**97.1**	**98.1**	**98.2**	**98.2**
	SIA-FIA	94.0	**100.0**	86.9	92.8	87.8	95.1	94.6	93.0
	SIA-RPA	96.1	**100.0**	90.7	93.4	90.3	96.5	96.2	94.7
	SIA-NAA	94.5	99.9	89.7	92.9	88.8	95.0	94.0	93.5
	SIA-DANAA	94.6	99.8	92.0	93.6	89.6	95.3	94.8	94.2
	SIA-SFVA	95.2	**100.0**	90.4	93.6	90.6	95.8	95.2	94.4
	SIA-BFA	96.6	99.7	93.5	95.2	92.6	96.1	96.2	95.7
	SIA-P2FA(**Ours**)	**99.6**	**100.0**	**98.7**	**99.6**	**98.9**	**99.6**	**99.6**	**99.4**
IncRes-v2	BSR-FIA	81.8	77.5	98.7	75.5	66.3	81.9	79.3	80.1
	BSR-RPA	85.0	82.9	98.7	80.4	72.6	85.2	82.2	83.9
	BSR-NAA	77.5	75.6	97.0	73.2	66.7	78.9	75.5	77.8
	BSR-DANAA	80.5	78.3	95.7	77.2	67.3	81.8	79.3	80.0
	BSR-SFVA	83.6	79.6	96.9	78.2	71.2	82.2	80.5	81.7
	BSR-BFA	88.2	87.1	99.0	83.2	75.4	86.8	84.2	86.3
	BSR-P2FA(**Ours**)	**95.7**	**95.8**	**100.0**	**93.5**	**90.1**	**93.6**	**92.8**	**94.5**
	SIA-FIA	91.3	89.8	**99.9**	85.5	80.0	88.1	87.8	88.9
	SIA-RPA	94.1	92.6	**99.9**	89.7	85.5	91.3	91.5	92.1
	SIA-NAA	91.8	90.0	99.7	88.6	83.6	90.0	87.9	90.2
	SIA-DANAA	93.7	91.1	99.2	89.4	85.4	90.9	91.6	91.6
	SIA-SFVA	94.3	91.6	99.4	90.7	87.2	90.6	90.5	92.0
	SIA-BFA	95.4	94.2	99.5	92.5	90.1	93.7	92.6	94.0
	SIA-P2FA(**Ours**)	**99.5**	**98.4**	**99.9**	**98.1**	**97.4**	**98.0**	**98.3**	**98.5**
Res-152	BSR-FIA	79.0	77.9	64.3	98.8	**100.0**	94.3	93.3	86.8
	BSR-RPA	86.5	83.1	70.6	99.5	**100.0**	96.3	95.7	90.2
	BSR-NAA	84.0	82.9	73.7	98.6	**100.0**	95.1	93.4	89.7
	BSR-DANAA	88.0	86.1	76.9	99.1	99.9	95.4	94.4	91.4
	BSR-SFVA	87.3	84.9	77.1	99.5	**100.0**	95.9	94.7	91.3
	BSR-BFA	90.6	88.8	80.2	99.6	**100.0**	97.3	96.7	93.3
	BSR-P2FA(**Ours**)	**98.8**	**98.3**	**96.4**	**99.9**	**100.0**	**99.8**	**99.7**	**99.0**
	SIA-FIA	86.0	85.7	73.3	99.6	**100.0**	96.5	96.2	91.0
	SIA-RPA	92.2	89.5	80.8	99.9	**100.0**	98.1	97.6	94.0
	SIA-NAA	92.2	90.7	84.0	99.8	**100.0**	97.7	97.9	94.6
	SIA-DANAA	94.2	92.7	87.5	99.8	**100.0**	98.5	98.1	95.8
	SIA-SFVA	93.1	91.9	84.6	99.9	**100.0**	98.1	97.9	95.1
	SIA-BFA	95.3	94.3	87.7	99.8	**100.0**	98.3	98.6	96.3
	SIA-P2FA(**Ours**)	**99.2**	**98.8**	**96.8**	**100.0**	**100.0**	**99.9**	**99.9**	**99.2**

Table 7. Success rate of different attacks integrated with BSR or SIA against defense models and vision transformers. The first column shows source models, the first row lists target models and the last column represents the average attack success rate. The best results are highlighted in **bold**.

Model	Attack	Adv-Inc-v3	Adv-Ens-IncRes-v2	R&P	FD	JPEG	Bit-Red	PiT-S	CaiT-S	DeiT-B	Swin-B	Average
Inc-v3	BSR-FIA	52.6	24.3	28.1	38.1	40.0	23.8	40.0	24.4	29.9	18.9	32.0
	BSR-RPA	56.1	29.4	34.2	40.1	43.6	29.6	49.6	31.8	37.7	22.7	37.5
	BSR-NAA	53.8	33.4	37.8	40.6	44.0	33.5	54.8	38.0	42.7	28.2	40.7
	BSR-DANAA	56.0	34.0	38.8	41.2	44.9	33.8	55.5	38.3	44.9	31.0	41.8
	BSR-SFVA	55.2	32.9	38.3	41.6	45.7	33.2	54.3	36.9	43.0	28.2	40.9
	BSR-BFA	57.7	31.6	37.3	42.2	46.4	31.8	59.3	37.5	44.5	29.3	41.8
	BSR-P2FA(**Ours**)	**77.8**	**50.2**	**56.3**	**61.5**	**60.6**	**50.4**	**82.8**	**69.3**	**74.4**	**55.7**	**63.9**
	SIA-FIA	62.4	33.7	37.8	48.7	47.8	33.8	56.3	38.5	43.1	28.2	43.0
	SIA-RPA	67.8	40.9	43.6	50.1	53.9	40.6	66.5	46.0	50.2	36.8	49.6
	SIA-NAA	65.1	45.4	49.5	49.4	55.7	45.0	71.6	54.0	57.4	44.3	53.7
	SIA-DANAA	69.1	48.8	52.9	53.0	58.2	48.6	74.5	57.8	59.4	45.7	56.8
	SIA-SFVA	66.4	45.3	49.4	52.2	56.9	45.1	70.6	53.6	55.1	42.6	53.7
	SIA-BFA	72.6	46.2	49.4	54.2	59.2	46.1	76.9	59.0	62.7	46.2	57.3
	SIA-P2FA(**Ours**)	**83.2**	**61.9**	**63.9**	**72.1**	**70.9**	**61.7**	**91.0**	**81.4**	**81.9**	**69.5**	**73.8**
Inc-v4	BSR-FIA	48.4	26.2	31.0	38.4	39.6	26.3	45.4	26.4	31.9	21.1	33.5
	BSR-RPA	50.8	29.3	34.6	39.5	39.4	29.3	48.5	30.3	34.7	25.4	36.2
	BSR-NAA	51.0	34.1	38.4	38.8	42.9	33.8	55.8	38.6	41.1	30.8	40.5
	BSR-DANAA	54.9	36.3	39.9	40.4	41.7	36.3	55.6	41.3	44.4	32.1	42.3
	BSR-SFVA	51.5	34.4	40.5	39.7	43.7	34.6	55.8	39.7	42.3	32.2	41.4
	BSR-BFA	56.0	32.5	37.3	41.0	41.2	31.7	57.4	38.5	42.6	31.6	41.0
	BSR-P2FA(**Ours**)	**74.5**	**54.6**	**60.1**	**59.1**	**58.9**	**54.5**	**86.5**	**70.9**	**75.5**	**62.6**	**65.7**
	SIA-FIA	57.9	34.1	36.9	46.1	44.8	33.9	54.2	34.8	40.1	29.6	41.2
	SIA-RPA	60.5	39.3	43.3	48.6	47.9	39.7	62.5	42.3	45.8	35.8	46.6
	SIA-NAA	62.2	47.6	50.1	48.5	52.4	47.1	68.7	52.1	53.9	46.4	52.9
	SIA-DANAA	64.9	50.2	52.5	51.7	54.5	50.3	72.6	55.3	57.1	46.4	55.6
	SIA-SFVA	60.8	48.0	51.2	49.7	55.2	47.4	72.7	55.7	55.9	48.8	54.5
	SIA-BFA	65.7	46.5	50.2	53.2	55.0	46.5	73.9	55.8	58.7	46.7	55.2
	SIA-P2FA(**Ours**)	**79.7**	**63.4**	**65.8**	**69.6**	**67.7**	**63.8**	**91.7**	**82.8**	**83.1**	**75.4**	**74.3**
IncRes-v2	BSR-FIA	57.1	32.6	33.1	43.3	41.2	32.7	33.8	21.3	25.6	12.7	33.3
	BSR-RPA	63.5	42.7	43.4	48.5	47.2	42.8	46.3	30.9	35.1	20.4	42.1
	BSR-NAA	56.4	41.2	42.6	43.5	46.1	41.0	45.0	33.5	34.3	22.8	40.6
	BSR-DANAA	59.0	44.3	44.4	46.0	46.1	44.2	49.5	36.5	37.6	23.9	43.2
	BSR-SFVA	58.5	42.5	44.7	47.5	48.5	42.4	48.9	36.2	38.8	24.2	43.2
	BSR-BFA	65.4	45.7	47.1	49.2	47.3	45.7	52.9	37.6	39.4	25.0	45.5
	BSR-P2FA(**Ours**)	**79.7**	**64.7**	**67.6**	**66.1**	**62.5**	**65.3**	**76.0**	**63.9**	**66.9**	**48.9**	**66.2**
	SIA-FIA	68.1	45.8	44.9	56.8	52.9	46.0	48.5	34.0	38.7	23.3	45.9
	SIA-RPA	74.3	58.1	56.6	60.9	60.3	57.8	62.4	46.7	49.4	33.3	56.0
	SIA-NAA	70.8	61.0	60.7	58.5	60.5	61.2	62.9	53.0	50.7	40.2	58.0
	SIA-DANAA	73.4	65.3	64.7	60.3	64.6	64.8	67.9	56.3	54.9	42.2	61.4
	SIA-SFVA	70.6	61.9	61.6	61.9	64.3	62.4	68.5	56.7	55.8	42.4	60.6
	SIA-BFA	78.1	67.6	66.2	63.9	66.3	67.3	72.1	60.3	59.8	44.8	64.6
	SIA-P2FA(**Ours**)	**87.8**	**76.7**	**75.7**	**77.2**	**75.1**	**76.3**	**87.8**	**80.9**	**81.9**	**67.8**	**78.7**
Res-152	BSR-FIA	41.4	20.4	24.6	36.2	33.4	20.4	35.8	20.1	26.8	17.1	27.6
	BSR-RPA	43.4	24.9	31.1	38.4	40.7	24.9	48.7	26.2	34.3	23.5	33.6
	BSR-NAA	47.0	34.1	40.1	41.2	45.2	34.4	63.8	43.8	50.5	38.2	43.8
	BSR-DANAA	47.7	33.6	39.6	42.2	45.4	32.9	63.8	42.1	50.2	38.3	43.6
	BSR-SFVA	48.6	33.4	39.2	43.2	45.7	32.7	62.5	42.4	49.5	35.9	43.3
	BSR-BFA	48.4	30.1	37.6	42.3	46.5	30.2	61.4	38.3	49.6	32.9	41.7
	BSR-P2FA(**Ours**)	**70.5**	**53.4**	**61.5**	**62.9**	**61.5**	**52.7**	**90.3**	**77.3**	**83.2**	**72.9**	**68.6**
	SIA-FIA	44.9	25.6	29.9	40.7	38.8	25.7	46.2	26.0	33.7	24.8	33.6
	SIA-RPA	48.4	30.4	37.3	43.9	44.4	30.8	56.4	33.2	43.0	33.7	40.2
	SIA-NAA	54.7	41.2	47.4	48.8	51.2	40.5	74.0	53.3	60.9	49.5	52.2
	SIA-DANAA	55.5	41.4	47.6	50.2	52.3	41.1	72.6	51.5	59.9	47.6	52.0
	SIA-SFVA	54.5	40.4	46.1	48.7	52.1	39.9	71.9	51.1	57.2	45.4	50.7
	SIA-BFA	54.8	37.4	43.6	50.2	54.1	37.8	73.6	49.1	59.6	45.9	50.6
	SIA-P2FA(**Ours**)	**70.2**	**55.0**	**59.4**	**68.4**	**66.8**	**54.8**	**90.4**	**80.6**	**83.1**	**73.9**	**70.3**