Improving Adversarial Transferability via Decision Boundary Adaptation

Jiayu Zhang[1] **Zhiyu Zhu**[2] **Zhibo Jin**[2] **Xinyi Wang**[3] **Huaming Chen**[*4] **Kim-Kwang Raymond Choo**[5]

[1]Suzhou University of Technology
[2]University of Technology Sydney
[3]Universiti Malaya
[4]The University of Sydney
[5]The University of Texas at San Antonio

Abstract

Black-box attacks play a pivotal role in adversarial attacks. However, existing approaches often focus predominantly on attacking from a data-centric perspective, neglecting crucial aspects of the models. To address this issue, we propose a novel approach in this paper, coined Decision Boundary Adaptation (DBA). Our approach innovatively adopts a model-centric viewpoint, leveraging operations on the model to attain properties that enhance transferability. We observe that a flatter curvature of the statistical manifold, influenced by both samples and model parameters, leads to stronger transferability of the adversarial attacks. To leverage this, we introduce the concept of local flatness, providing an evaluation method for local flatness property along with a detailed mathematical proof. Additionally, we demonstrate a consistent relationship between local flatness, the model's decision boundary, and the gradient descent process, showing how flatness can be achieved through gradient descent at the model parameter level. Through extensive evaluation using state-of-the-art adversarial attack techniques, our DBA approach significantly enhances the black-box attack capabilities of all the tested adversarial attack methods. The implementation of our method is available at `https://github.com/LMBTough/DBA`.

1 INTRODUCTION

Artificial intelligence (AI), especially Deep neural networks (DNNs), has showcased impressive success in computer vision tasks, such as image classification [He et al., 2016a, Huang et al., 2017, Sandler et al., 2018, Li, 2022, Gulzar, 2023]. However, the vulnerability of DNNs to adversarial attacks poses critical concerns regarding model safety. Even minor, imperceptible perturbations to input data can lead to erroneous predictions [Goodfellow et al., 2014], significantly undermining the model integrity. To enhance the trustworthiness of DNNs, it is important to proactively identify the potential vulnerabilities in the models. Therefore, developing novel adversarial attack methods against DNNs becomes a critical approach to provide holistic and novel insights of the vulnerabilities in the models.

In particular, the growing prevalence of AI-empowered web applications and online services has introduced both new opportunities and challenges for adversarial attacks. Web services, such as cloud-based machine learning platforms and online APIs, often expose machine learning models to external parties, creating potential attack surfaces. Attackers can exploit the web interfaces by launching black-box or query-based attacks, probing for vulnerabilities without direct access to the model. Moreover, with the widespread image recognition and natural language processing techniques on online platforms, the threat of adversarial attacks on the web has become a major security concern [Apruzzese et al., 2022, Omara and Kantarci, 2024].

Adversarial attacks are broadly categorized into white-box and black-box approaches [Papernot et al., 2017]. Initially, most adversarial attacks are white-box, where attackers have full knowledge of the target model, such as its architecture, parameters, etc [Kurakin et al., 2018, Madry et al., 2017]. However, white-box attacks are often impractical in real-world scenarios. As a result, black-box attacks that do not require any knowledge of the target model have become popular. In black-box attacks, query-based methods simulate the model behaviour by querying the target model for its outputs. While effective, these methods often require frequent and extensive querying to the target model, potentially reducing the stealthiness of the attack. In contrast, transferable attacks do not require any access to the target model, providing better stealth and greater development potential [Dong

[*]Corresponding author: huaming.chen@sydney.edu.au. This work was done while the authors were with The University of Sydney.

et al., 2018, 2019, Lin et al., 2019, Zhang et al., 2022, Long et al., 2022]. Our work focuses on enhancing transferable black-box attacks.

Figure 1 shows a simplified model training process with different emphasis of the attack techniques. Recent methods, such as DI-FGSM [Xie et al., 2019] and MI-FGSM [Dong et al., 2018], enhance the transferability of attacks through data transformations, while others like NAA [Zhang et al., 2022], DANAA [Jin et al., 2023], and MIG [Ma et al., 2023] improve attack transferability by modifying the loss function. SSA optimizes the Backpropagation steps to enhance the attack transferability [Long et al., 2022]. While most approaches optimize based on data or gradients, we note the techniques like LGV [Gubri et al., 2022] and DBA, proposed in this work, enhance attack transferability from the model-centric perspective. More discussion is included in Section. 2.

We observe that, the effectiveness of transferable attacks is closely related to gradient information. Smaller gradient values can enhance the attack transferability [Ge et al., 2023], as they indicate a flatter curvature of the statistical manifold under the combined influence of samples and current parameters, contributing to the generalization [Zhao et al., 2022]. However, relying solely on the magnitude of gradients calculated by L_p norms, within a Euclidean space loss function, may not be optimal. In particular, using such information to assess flatness only considers the first-order property of functions. We further investigate the function curvature with second-order information, such as leveraging the Hessian matrix.

In this paper, we introduce a more general definition of local flatness in loss function curvature from an adversarial attack perspective, providing an approximate calculation method and a thorough proof. We also investigate how the gradient descent process of model parameters helps find a flatter curvature of the loss function during adversarial attack. Our experiments demonstrate a remarkable consistency between local flatness and decision boundaries. With these findings, we discuss the effectiveness of our method from multiple perspectives: local flatness, decision boundaries, and adversarial defense. We summarise the contributions as follows:

- We provide a definition of local flatness during the adversarial attack process, along with a detailed proof, as well as proposing a novel method to achieve flatness.
- We discuss and validate the correlations between local flatness, gradient descent, adversarial defense training, and transferability, and propose DBA method to effectively leverage the gradient descent process to enhance the transferability of adversarial attacks.
- We conduct extensive experiments to compare DBA method against a broad spectrum of transferable attack methods. Significantly, DBA enhances the transferable capability of all methods in the experiments, demonstrating that DBA is a universal and effective approach for improving the transferability of adversarial attacks.

2 RELATED WORK

2.1 WHITE-BOX ATTACKS

White-box attacks are classical methods that grant the adversary access to the target model for the information of structure and parameters. Several methods include Fast Gradient Sign Method (FGSM) [Goodfellow et al., 2014], Basic Iterative Method (BIM) [Kurakin et al., 2018], PGD [Madry et al., 2017], and Carlini&Wagner method (C&W) [Carlini and Wagner, 2017].

FGSM leverages gradient information of the input data to generate adversarial examples by introducing small perturbations in the direction indicated by the sign function. BIM enhances adversarial attacks by iteratively applying subtle perturbations, thereby increasing the attack effectiveness. In parallel, PGD algorithm extended BIM by incorporating a projection step in each iteration to ensure that adversarial examples remain within a predefined perturbation range, thus enhancing the controllability and efficacy of the attack. Conversely, C&W employs a distinct strategy by optimizing a tailored objective function to craft adversarial examples. It focuses on minimizing the size of the perturbations required, aiming to produce high-quality adversarial examples that are imperceptible.

2.2 BLACK-BOX ATTACKS

Black-box attacks aim to address the limited access to the target model's internals. While there are different approaches, they typically share a common goal of improving the effectiveness and transferability of adversarial examples.

Traditional method, such as gradient-based attacks, serves as the cornerstone of attack techniques. Several exemplar methods include DI-FGSM [Xie et al., 2019], MI-FGSM [Dong et al., 2018], TI-FGSM [Dong et al., 2019], and SINI-FGSM [Lin et al., 2019]. DI-FGSM employs random transformations to improving the transferability. MI-FGSM and TI-FGSM add momentum and translation invariance, respectively, to refine attack robustness and success rate. SINI-FGSM further uses Nesterov accelerated gradient with scale-invariance, boosting the robustness and transferability.

In addition to traditional gradient-based methods, Structure Invariant Attack (SIA) [Wang et al., 2023] and Momentum Integrated Gradients (MIG) [Ma et al., 2023] introduce innovative strategies to augment attack transferability. SIA preserves the structural integrity of images through region-specific transformations (i.e., rotation and scaling), while MIG uses integrated gradients and a momentum term to enhance the transferability across models. Other methods

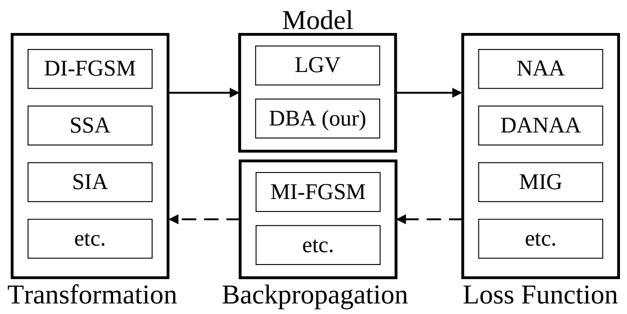


Figure 1: Comparison of adversarial attacks. Solid right-pointing arrows represent forward propagation; Dashed reverse arrows indicate backward propagation

include Penalizing Gradient Norm (PGN) [Ge et al., 2023], Neural Attribution Attack (NAA) [Zhang et al., 2022], and their variants. PGN enhances transferability by exploring the flat regions of a model's loss surface, leveraging flat minima characteristics through repeated gradient ascent on samples, aiming to find a flat direction. For NAA and its advanced form, the Dual Adversarial Neural Attribution Attack [Jin et al., 2023], they target intermediate neurons to create more transferable adversarial examples. These methods assess the significance of neurons, attributing the model's output comprehensively and employing complex formulas to enhance attack efficiency.

Complementary techniques like Gradient Relevance Attack (GRA) [Zhu et al., 2023a] and frequency domain methods such as Spectrum Simulation Attack (SSA) [Long et al., 2022] and Frequency-based Stationary Point Search (FSPS) [Zhu et al., 2023b] represent novel directions. GRA reduces perturbation fluctuations via a gradient relevance framework. SSA manipulates spectral components for better transferability, while FSPS uses frequency information to find effective adversarial directions.

In Figure 1, we outline the model training process where each method targets improving attack transferability, primarily through data or gradient-based strategies. LGV [Gubri et al., 2022] leverages model parameter to simulate multiple models, akin to DI-FGSM's strategy of utilising similar semantic information. However, LGV requires extensive training data, and the data quality could significantly impact the attack effectiveness. Thus, in this work, we focus on model parameters through the lens of local flatness, which results from the interaction between a single sample and the model. This approach is independent of data quality, providing a more robust framework. Unlike LGV, which does not specifically address decision boundaries or flatness properties, our proposed DBA method directly targets these aspects. By optimizing for local flatness, DBA enhances the transferability of attacks across different methods, making it a universal and effective optimization framework.

Furthermore, unlike PGN [Ge et al., 2023], which explores flat regions of the loss surface by manipulating gradients, our method works by adjusting model parameters to identify attack directions that better promote flatness. In the next section, we provide a more detailed discussion of our DBA method and its advantages in enhancing attack transferability.

3 OUR PROPOSED METHOD

In this section, we first provide definitions for adversarial attacks and transferability in the attacks. We then introduce the local flatness and decision boundary adaptation of models and elucidate their impacts on transferability. We finally discuss the consistency between parameter gradient descent and transferability, decision boundary adaptation, and local flatness from multiple perspectives.

3.1 PROBLEM DEFINITION

For adversarial attacks, an adversarial example for a given input sample x and a source model with parameters θ_s is defined as $x_{adv} = x + \delta$, where $x \in \mathbb{R}^m$, $\theta_s \in \mathbb{R}^n$, and δ represents a small perturbation crafted to mislead the model. The objective can be mathematically formulated as:

$$\max_{\delta} L(y^t|x+\delta,\theta_s) \quad \text{s.t.} \quad ||\delta||_p \leq \epsilon \tag{1}$$

where L is the loss function, y is the true label for x, ϵ is the perturbation constraint ensuring the perturbation is small, and $||\cdot||_p$ denotes the L_p norm, typically $p=\infty$ for image data, imposing a maximum change ϵ to any element of x.

We define the transferability T of an adversarial example x_{adv} from a source model with parameters θ_s to a well-trained target model with parameters θ_t as:

$$T(x_{\text{adv}},\theta_s,\theta_t) = \frac{1}{k}\sum_{i=1}^{k} \mathbb{I}\left[f_t(x_i^{\text{adv}}) \neq f_t(x_i)\right] \tag{2}$$

where $f_t(x)$ represents the classification function of the target model, k denotes the number of adversarial samples, and $\mathbb{I}[\cdot]$ is the indicator function, yielding 1 if the condition holds true otherwise 0. High transferability implies that T must be higher value, signifying that a greater number of adversarial examples generated on the source model effectively manipulate the prediction of target model.

3.2 LOCAL FLATNESS

Smaller gradient values of the loss function indicate a flatter loss function landscape at the gradient computation location [Zhao et al., 2022]. The flat local maxima induced by smaller gradients can enhance the transferability of adversarial attacks [Ge et al., 2023]. However, assessing flatness values based solely on gradient magnitudes constitutes an evaluation utilizing only first-order properties in Euclidean space. More generally, we aim to observe the curvature of the statistical manifold, which is influenced by both the current sample and model parameters. This notion, termed **local flatness**, refers to the flatness of the loss function locally around a single sample x and model parameters θ. Evaluating local flatness typically involves the Fisher information matrix [Lehmann and Casella, 2006]. Moreover, starting from the definition of local flatness, we find that transferability arises not only from the gradient of the current sample and its variants but also from the combined effects of the sample and model parameters, an important aspect overlooked by nearly all current transferable adversarial attack methods.

Inspired by the methods of computing Fisher information in [Martin and Elster, 2020], Fisher information inherently serves to assess the amount of information about the sample collected by the model, as discussed in [Ly et al., 2017]. Equation 4 illustrates that as training progresses, the amount of information gathered from the sample decreases, leading to a flatter Fisher information matrix. This flattening effect due to parameter updates (training) is the reason for the increase in flatness. Therefore, we extend this to the sample space to assess its association with local flatness.

Theorem 1 (Directional Finite Difference Estimation)**.** *Given a function f with independent variable x, we can estimate $\Delta x \cdot \frac{\partial f(x)}{\partial x}$ by applying a fixed-direction perturbation Δx to x, as follows:*

$$\begin{aligned}\Delta x \cdot \frac{\partial f(x)}{\partial x} &\approx \frac{f(x+\epsilon \cdot \Delta x) - f(x)}{\epsilon} \\ &\approx \frac{f(x+\epsilon \cdot \Delta x) - f(x-\epsilon \cdot \Delta x)}{2\epsilon}\end{aligned} \tag{3}$$

where ϵ is a scalar. Theorem 1 will be used later to reduce the complexity of computing Fisher information. The *proof* is provided in the appendix.

Theorem 2 (Estimation of Curvature on Statistical Manifold)**.** *Given an adversarial example x and the current model parameters θ, we can employ the Fisher information matrix to estimate the curvature of the statistical manifold.*

$$\begin{aligned}\mathcal{I}_\theta(x) &= E_{y^c \sim P(y|x,\theta)}\left[\frac{\partial \log P_{y^c}(x,\theta)^\top}{\partial x} \cdot \frac{\partial \log P_{y^c}(x,\theta)}{\partial x}\right] \\ &= -E_{y^c \sim P(y|x,\theta)}\left[\mathrm{H}^c\right] \quad s.t. \quad \mathrm{H}^c_{ij} = \frac{\partial^2 \log P(y^c|x,\theta)}{\partial x_i \partial x_j}\end{aligned} \tag{4}$$

where $E_{y^c \sim P(y|x,\theta)}[\cdot]$ denotes the expectation over the conditional distribution of class y^c given input x and parameters θ, $\frac{\partial \log P_{y^c}(x,\theta)}{\partial x}$ represents the gradient of the logarithm of the conditional probability of class y^c with respect to input x, and H^c is the Hessian matrix of the logarithm of the conditional probability with respect to x, with elements $\mathrm{H}^c_{ij} = \frac{\partial^2 \log P(y^c|x,\theta)}{\partial x_i \partial x_j}$.

The Hessian matrix H^c represents the curvature of the statistical manifold and serves as an effective tool for assessing the flatness of the manifold. As illustrated in Theorem 2, if we consider the curvature of all neural network output class functions, the Fisher information matrix can be employed. Furthermore, the Fisher information matrix can also be regarded as a crucial tool for evaluating the changes in the KL divergence of the model's output distribution due to variations in x, as analyzed in the appendix. It is important to note that the magnitude of gradients considers only the output of a single class, $-log\ P_{y^t}(x,\theta)$, where t represents the target label (the computation method of cross-entropy loss function under hard label conditions) [Ge et al., 2023]. However, in reality, for model outputs transformed into a probability distribution, enhancing the output values of other classes can also be an important means to diminish the probability of the current class.

Considering the high computational complexity of $\mathcal{I}_\theta(x) \in \mathbb{R}^{m\times m}$, we can estimate the impact of $\mathcal{I}_\theta(x)$ using the trace of $\mathcal{I}_\theta(x)$, $tr(\mathcal{I}_\theta(x))$. However, using the trace neglects the contribution of many off-diagonal dimensions to curvature. We use the quadratic form of $tr(\mathcal{I}_\theta(x))$ for analysis:

$$tr(\mathcal{I}_\theta(x)) = \sum_{i=1}^{m} e^i \mathcal{I}_\theta(x) e^i \tag{5}$$

where $e^i \in \mathbb{R}^m$ and the i-th dimension of e^i is 1, with the rest being 0. Changes in the dimension that e^i is 0 of x are not considered in the curvature calculation. More generally, we define:

$$flat(\mathcal{I}_\theta(x)) = \Delta x^\top \mathcal{I}_\theta(x) \Delta x \tag{6}$$

as a measure to evaluate the flatness score of x's curvature. With **Theorem 1**, we can approximate and derive **Theorem 3**.

Theorem 3 (Assessment of Local Flatness)**.** *This allows us to assess local flatness using $flat(\mathcal{I}_\theta(x))$ with the following approximation.*

$$\begin{aligned}flat(\mathcal{I}_\theta(x)) \approx \frac{1}{4\varepsilon^2}\Big\langle &\frac{\partial P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}, \\ &\frac{\partial \log P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial \log P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}\Big\rangle \\ \propto \Big\langle &\frac{\partial P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}, \\ &\frac{\partial \log P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial \log P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}\Big\rangle\end{aligned} \tag{7}$$

where $\langle\cdot,\cdot\rangle$ denotes the dot product, we choose Δx as the iterative update vector in adversarial attacks, allowing us to observe the level of flatness during the adversarial attack process. Additionally, we compute the flatness with respect to model parameters θ as follows:

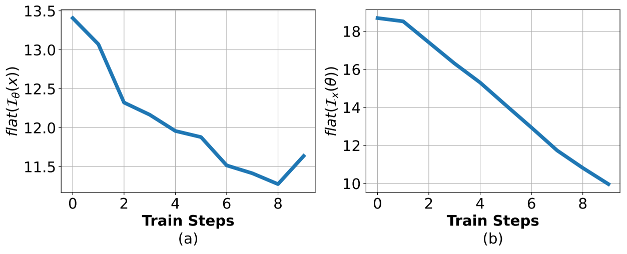


Figure 2: (a) and (b) depict the flatness analysis of $flat(\mathcal{I}_\theta(x))$ and $flat(\mathcal{I}_x(\theta))$ during training. The results are obtained using the Inception-v3 model on the ImageNet dataset.

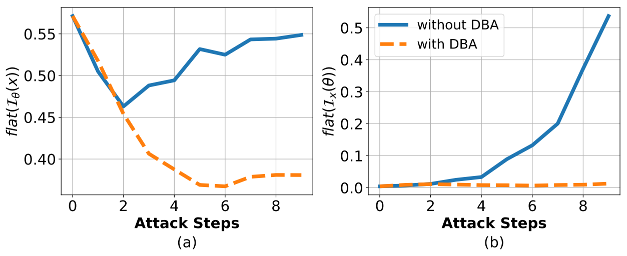


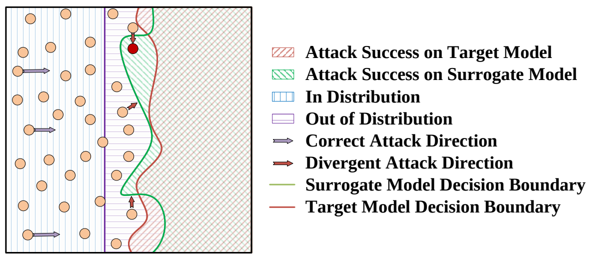
Figure 3: (a) and (b) are the flatness analysis of $flat(\mathcal{I}_\theta(x))$ and $flat(\mathcal{I}_x(\theta))$ during the attack process. The experiments are conducted using Inception-v3 on the ImageNet dataset.

$$flat(\mathcal{I}_x(\theta)) \propto \left\langle \frac{\partial P_y(x, \theta + \varepsilon \cdot \Delta\theta)}{\partial \theta} - \frac{\partial P_y(x, \theta - \varepsilon \cdot \Delta\theta)}{\partial \theta}, \right. \\ \left. \frac{\partial \log P_y(x, \theta + \varepsilon \cdot \Delta\theta)}{\partial \theta} - \frac{\partial \log P_y(x, \theta - \varepsilon \cdot \Delta\theta)}{\partial \theta} \right\rangle \quad (8)$$

The *proof* of **Theorem 3** is provided in the appendix. Fisher information can be employed to assess the compatibility between parameters and samples [Martin and Elster, 2020], where a higher $flat(\mathcal{I}_\theta(x))$ indicates that the sample is extracting more information from the parameters θ (potentially leading to overfitting in attacks). From the perspective of $flat(\mathcal{I}_x(\theta))$, a larger value suggests that the sample is unusual for the parameters θ and is near the decision boundary, characterizing it as an Out-of-Distribution (OOD) sample. Therefore, it is crucial to minimize $flat(\mathcal{I}_\theta(x))$ and $flat(\mathcal{I}_x(\theta))$ during the attack process, for which gradient descent will be employed. Furthermore, we will discuss why samples close to the decision boundary are non-flat and detrimental to the transferability of adversarial attacks, highlighting a consistency between the two.

Assumption 1 (Attack Transferability)**.** *Given an input sample x, and model parameters θ, a smaller $flat(\mathcal{I}_\theta(x))$ leads to stronger attack transferability.*

Assumption 1 constitutes our core assumption, which is an optimized form of the hypothesis regarding the relationship between flatness and transferability proposed in [Ge et al., 2023]. As demonstrated in Figure 2, our DBA method effectively reduces $flat(\mathcal{I}_\theta(x))$ and $flat(\mathcal{I}_x(\theta))$ during the training process.



▨ Attack Success on Target Model
▨ Attack Success on Surrogate Model
▥ In Distribution
▤ Out of Distribution
⇒ Correct Attack Direction
⇒ Divergent Attack Direction
— Surrogate Model Decision Boundary
— Target Model Decision Boundary

Figure 4: Gradient direction information near the decision boundary is divergent and unstable. With DBA, the sample will be converted from Divergent Attack Direction to Correct Attack Direction. (Here we take samples close to the decision boundary as an example, as once a small perturbation is added to these samples, the classification results of the model may change)

In Figure 3, taking BIM as an example, when integrated with our DBA method, it consistently sustains lower $flat(\mathcal{I}_\theta(x))$ and $flat(\mathcal{I}_x(\theta))$ throughout the attack process compared to when used without DBA. Lower values of these metrics indicate a flatter curvature of the statistical manifold.

3.3 DECISION BOUNDARY

Neural networks operate as continuous mappings, where the transition of samples across the decision boundary (resulting in class changes) is a gradual process without abrupt, large-scale shifts in outcomes. This implies that samples near the decision boundary are of lower confidence and more prone to class changes. Training data exhibit high confidence on a well-trained model and are distant from decision boundary. Data close to the training data distribution are considered in-distribution (ID) data, whereas those near the decision boundary are deemed out-of-distribution (OOD) data.

In Figure 4, once an adversarial example approaches the vicinity of the decision boundary, gradient direction information becomes more divergent and unstable. To ensure data remains within the ID range, gradient descent on individual samples can be applied to push them from the decision boundary. The distance from the decision boundary can be inferred from the variation in loss function values during the adversarial attack; the greater the distance, the harder it is to increase the loss function value. Figure 5 illustrates that the difficulty of attacking increases with gradient descent (Samples Distant from the Decision Boundary). It's important to emphasize that the methodologies for calculating the distance to the decision boundary and local flatness are solely dependent on the current model, the sample, and the task itself, and are not influenced by the method used

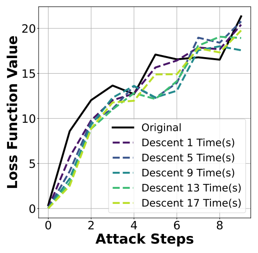


Figure 5: Relationship between the number of attack steps and the loss function value. We define 'Original' as the baseline where no DBA is applied. 'Descent' as the rounds of gradient descent. The smaller the value of the loss function, the more challenging it becomes to execute a successful attack, indicating greater distance from the decision boundary. This experiment uses the Inception-v3 model on the ImageNet dataset.

to compute the gradient of adversarial samples during the attack process. Therefore, the approach remains valid even in scenarios involving complex loss functions.

$$\min_{\theta} E_{(x,y^t)\sim D}[\max_{\delta} L(\theta, x+\delta, y)] \tag{9}$$

(From the perspective of adversarial training) Furthermore, applying gradient descent to adversarial examples can be seen as simulating the process of adversarial defense training, as shown in Equation 9, where adversarial defense training of θ employs gradient descent [Shaham et al., 2018]. Utilizing gradients obtained from adversarially trained parameters and adversarial examples can facilitate attack effectiveness on adversarially trained models. We also experiment the transferability tests on such models.

3.4 DECISION BOUNDARY ADAPTATION (DBA)

We have discussed how gradient descent operations can distance samples from the decision boundary, obtaining more general adversarial gradient information, and effectively increase the flatness of the curvature of the statistical manifold. In this context, we simply introduce gradient descent operations during the process of enhancing attack transferability.

$$\theta^t = \theta^{t-1} - \eta \cdot \frac{\partial L(y^t|x_i^{t-1}, \theta^{t-1})}{\partial \theta^{t-1}} \tag{10}$$

where θ^0 represents the model's initial parameters, and x^{t-1} denotes the adversarial sample iterated $t-1$ times. In each iteration of the adversarial sample, θ^t replaces the original model parameters. Furthermore, as adversarial attack algorithms often iterate over multiple samples simultaneously, the DBA algorithm can perform batch gradient descent operations during parameter updates:

$$\theta^t = \theta^{t-1} - \eta \frac{1}{k} \sum_{i=1}^{k} \frac{\partial L(y^t|x_i^{t-1}, \theta^{t-1})}{\partial \theta^{t-1}} \tag{11}$$

The update to θ affecting x_i^{t-1} is by $\frac{\partial L(y^t|x_i^{t-1},\theta^{t-1})}{\partial \theta^{t-1}}$, which also facilitates distancing from the decision boundary. This approach significantly enhances the efficiency of DBA, serving as an approximation of the single-step DBA, which introduces only one additional forward and backward gradient propagation, and it is negligible compared to the numerous propagations in methods like SSA and NAA. Therefore, there is some increase in computational cost but the added computational cost is manageable and acceptable. Detailed experimental results and pseudocode are in the Appendix G.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Dataset To ensure a rigorous experiment design, we use the same dataset in the NAAZhang et al. [2022] and SSA [Long et al., 2022], thus ensuring consistency in the experimental conditions across all baseline methods, including our algorithm. Accordingly, we conduct the evaluations using a standard image classification dataset that encompasses 1000 randomly chosen images from the ILSVRC 2012 validation set [Russakovsky et al., 2015].

Models We evaluated adversarial robustness across eleven models, including six without defense training: 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), ResNet-101 (Res-101), and ResNet-152 (Res-152) [He et al., 2016b], along with two Vision Transformer models, ViT-B/16 [Dosovitskiy et al., 2020] and MaxViT-T [Tu et al., 2022]. Among them, four models (Inc-v3, Inc-v4, IncRes-v2, Res-152) served as surrogates for attack generation.

We also evaluated three defense-trained models: Inception-v3 Ensemble 3 (Inc-v3-ens3), Inception-v3 Ensemble 4 (Inc-v3-ens4), and Inception-ResNet-v2 Ensemble (IncRes-v2-ens) [Tramèr et al., 2017], which employ ensemble techniques to enhance adversarial robustness. Inc-v3-ens3 and Inc-v3-ens4 aggregate multiple Inception-v3 models, while IncRes-v2-ens extends this approach to Inception-ResNet-v2.

Baseline Methods To evaluate the performance of the models under adversarial attacks, we apply ten baseline adversarial attack methods representing a diverse range of common attacks: BIM [Kurakin et al., 2018], PGD [Madry et al., 2017], DI-FGSM [Xie et al., 2019], TI-FGSM [Dong et al., 2019], MI-FGSM [Dong et al., 2018], SINI-FGSM [Lin et al., 2019], SSA [Long et al., 2022], FSPS [Zhu et al., 2023b], SIA [Wang et al., 2023] and MIG [Ma et al., 2023].

Metrics The main evaluation metric employed to gauge the performance of the models under adversarial attacks is

Table 1: ASR on Defensive Training Models. Each data point represents Baseline/Baseline+DBA(Gap), Green indicating improvement by DBA over the original method, and Red indicates a decrease.

Model	Method	Inc-v3-ens3	Inc-v3-ens4	IncRes-ens	Average
Inc-v3	BIM	12.5/12.6(0.1)	13.1/14.0(0.9)	4.7/6.7(2.0)	10.1/11.1(1.0)
	DI-FGSM	17.3/17.7(0.4)	17.4/19.2(1.8)	9.0/9.1(0.1)	14.57/15.33(0.77)
	MIG	40.3/46.1(5.8)	39.6/43.8(4.2)	21.4/27.4(6.0)	33.77/39.1(5.33)
	MI-FGSM	22.5/21.4(-1.1)	22.5/21.5(-1.0)	10.5/11.0(0.5)	18.5/17.97(-0.53)
	PGD	12.1/12.1(0.0)	12.6/12.9(0.3)	6.5/6.8(0.3)	10.4/10.6(0.2)
	PGN	20.8/23.9(3.1)	20.8/24.9(4.1)	9.7/12.8(3.1)	17.1/20.53(3.43)
	SIA	63.6/63.5(-0.1)	61.9/60.8(-1.1)	36.5/38.6(2.1)	54.0/54.3(0.3)
	SINI-FGSM	39.6/47.3(7.7)	36.7/45.6(8.9)	23.0/26.6(3.6)	33.1/39.83(6.73)
	SSA	74.3/77.2(2.9)	75.0/77.1(2.1)	60.0/64.1(4.1)	69.77/72.8(3.03)
	TI-FGSM	21.8/24.7(2.9)	24.1/27.2(3.1)	12.5/15.9(3.4)	19.47/22.6(3.13)
IncRes-v2	BIM	10.5/12.5(2.0)	11.5/12.6(1.1)	7.1/7.3(0.2)	9.7/10.8(1.1)
	DI-FGSM	16.0/20.7(4.7)	17.3/20.1(2.8)	10.8/13.1(2.3)	14.7/17.97(3.27)
	MIG	61.8/65.2(3.4)	56.4/58.1(1.7)	44.0/47.7(3.7)	54.07/57.0(2.93)
	MI-FGSM	21.2/24.4(3.2)	21.9/22.6(0.7)	12.8/15.0(2.2)	18.63/20.67(2.03)
	PGD	11.9/12.1(0.2)	12.4/12.4(0.0)	6.7/7.8(1.1)	10.33/10.77(0.43)
	PGN	21.9/23.4(1.5)	22.1/24.2(2.1)	14.7/16.6(1.9)	19.57/21.4(1.83)
	SIA	72.4/72.1(-0.3)	63.6/65.2(1.6)	51.7/51.9(0.2)	62.57/63.07(0.5)
	SINI-FGSM	54.7/62.2(7.5)	48.9/53.8(4.9)	38.7/44.0(5.3)	47.43/53.33(5.9)
	SSA	80.3/82.2(1.9)	76.7/77.4(0.7)	76.8/77.0(0.2)	77.93/78.87(0.93)
	TI-FGSM	27.0/34.8(7.8)	28.2/32.9(4.7)	24.8/31.6(6.8)	26.67/33.1(6.43)
Res-152	BIM	11.3/15.6(4.3)	11.7/15.2(3.5)	4.8/7.7(2.9)	9.27/12.83(3.57)
	DI-FGSM	17.1/21.5(4.4)	16.8/22.7(5.9)	9.3/14.7(5.4)	14.4/19.63(5.23)
	MIG	28.0/35.3(7.3)	26.5/32.2(5.7)	15.1/18.8(3.7)	23.2/28.77(5.57)
	MI-FGSM	19.9/23.5(3.6)	19.5/24.0(4.5)	10.3/13.2(2.9)	16.57/20.23(3.67)
	PGD	10.9/13.4(2.5)	12.5/13.5(1.0)	6.1/7.3(1.2)	9.83/11.4(1.57)
	PGN	18.8/21.9(3.1)	21.2/24.5(3.3)	12.3/13.3(1.0)	17.43/19.9(2.47)
	SIA	53.4/54.9(1.5)	49.0/50.5(1.5)	31.9/32.4(0.5)	44.77/45.93(1.17)
	SINI-FGSM	26.3/36.8(10.5)	26.0/32.9(6.9)	13.6/18.2(4.6)	21.97/29.3(7.33)
	SSA	77.9/82.7(4.8)	77.2/80.4(3.2)	68.5/73.4(4.9)	74.53/78.83(4.3)
	TI-FGSM	27.2/35.1(7.9)	30.5/34.8(4.3)	21.3/28.1(6.8)	26.33/32.67(6.33)
ViT-B/16	BIM	21.2/24.1(2.9)	24.4/28.0(3.6)	14.8/16.8(2.0)	20.13/22.97(2.83)
	DI-FGSM	40.7/43.9(3.2)	43.1/45.9(2.8)	33.1/35.6(2.5)	38.97/41.8(2.83)
	MIG	56.7/58.5(1.8)	56.8/58.7(1.9)	50.3/52.2(1.9)	54.6/56.47(1.87)
	MI-FGSM	45.9/46.2(0.3)	46.4/47.5(1.1)	38.5/40.7(2.2)	43.6/44.8(1.2)
	PGD	18.7/20.2(1.5)	20.6/22.0(1.4)	12.8/12.8(0.0)	17.37/18.33(0.97)
	PGN	29.8/35.3(5.5)	32.9/38.0(5.1)	21.7/27.5(5.8)	28.13/33.6(5.47)
	SIA	82.7/85.0(2.3)	83.2/84.2(1.0)	76.8/77.6(0.8)	80.9/82.27(1.37)
	SINI-FGSM	56.6/57.5(0.9)	56.3/58.5(2.2)	51.1/50.2(-0.9)	54.67/55.4(0.73)
	SSA	71.0/71.4(0.4)	71.8/72.7(0.9)	66.3/68.9(2.6)	69.7/71.0(1.3)
	TI-FGSM	30.3/35.2(4.9)	34.6/38.0(3.4)	25.4/29.0(3.6)	30.1/34.07(3.97)

Attack Success Rate (ASR). It quantifies the percentage of adversarial examples that successfully induce misclassifications. A higher attack success rate indicates that the attack method performs better on a specific model. The primary parameter adjusted in our DBA method is the learning rate, with the batch size set to 1 and the training strategy encompassing all steps. Full details on the parameters can be found in the Appendix H and our code repository.

4.2 RESULT

Experimental results are shown in Tables 2 and 1. Each table's results are separated by a forward slash, with the left side representing the original attack success rates of baseline methods on various models, and the right side displaying the success rates after applying our DBA algorithm. The values in parentheses indicate the difference in attack success rates between with and without the use of the DBA method. The last column presents the average attack success rates for both the baseline methods and DBA. For additional experimental data, please refer to the appendix.

In Table 2, the transferability of nearly all attack methods on models without defense training is significantly enhanced under the black-box setting of DBA, with results up to 97%. Only in a minority of cases does DBA's performance slightly decline. Specifically, when using Res-152 as the surrogate model, DBA exhibited higher improvements than the other three models, with an average increase of 8.66%, possibly due to the more general decision boundaries of Res-152. Meanwhile, other three models showed average improvements of 5.41%, 3.45%, and 2.44%, respectively.

For models with defense training, DBA also sustained significant performance enhancements. As depicted in Table 1, DBA continued to be highly effective in these more challenging scenarios, achieving effectiveness in 96% of cases. With defensively trained models, the average Attack Success Rate (ASR) increased by 2.81% after applying DBA.

It is important to note that while the improvement in transferability for certain methods like SIA and SSA might not appear significant, the consistency in the enhancements provided by our DBA method across various attack methods is crucial. This consistency is valuable, as it indicates that our work can be effectively integrated with future algorithms and methods. Such a feature of consistency is commonly appreciated in fields like object detection, evident in the updates within the YOLO series. Our approach lays a foundational strategy that can be universally applied, underscoring its potential for broad applicability and adaptability in advancing adversarial attack methodologies.

4.3 ABLATION EXPERIMENT

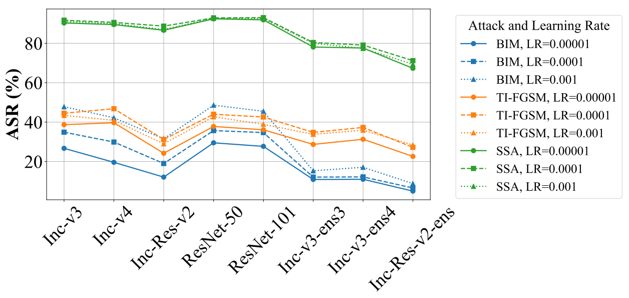


Figure 6: Performance of different attack methods under the influence of various learning rates using the proposed DBA technique on ResNet-152 as the source model.

4.3.1 Impact of Learning Rates on DBA Performance

In this section, ResNet-152 is employed as the source model to investigate the impact of different learning rates on the performance of DBA. The learning rates are set to 0.001, 0.0001, and 0.00001, respectively. As shown in Figure 6, it is observed that for enhancing the BIM attack, the learning rate of 0.001 leads to the best performance, while for the TI-FGSM and SSA methods, a learning rate of 0.0001 yields the optimal results.

Table 2: ASR on models without defense training. Each data group follows the pattern: Baseline/Baseline+DBA (Gap), where Green indicates an improvement by DBA relative to the original method, Red denotes a decrease, and Grey represents white-box attacks, which are not our focus.

Model	Method	Inc-v3	Inc-v4	IncRes-v2	Res-50	Res-101	Res-152	ViT-B/16	MaxViT-T	Average
Inc-v3	BIM	-	27.9/36.8(8.9)	19.7/29.9(10.2)	24.1/30.5(6.4)	21.9/27.7(5.8)	18.7/25.1(6.4)	9.2/11(1.8)	9.8/11.4(1.6)	18.76/24.63(5.87)
	DI-FGSM	-	48.2/52.1(3.9)	40.1/41.4(1.3)	38.4/42.1(3.7)	35.9/39.1(3.2)	32.5/36.3(3.8)	12.6/13.5(0.9)	14.8/16.5(1.7)	31.79/34.43(2.64)
	MIG	-	70.8/81.8(11)	68.5/81.8(13.3)	69.8/79.8(10.0)	64.7/77.1(12.4)	63.9/75.7(11.8)	32.8/37(4.2)	32.4/42.7(10.3)	57.56/67.99(10.43)
	MI-FGSM	-	50.2/56.3(6.1)	46.5/54.3(7.8)	47.0/51.4(4.4)	41.3/46.4(5.1)	41.6/45.9(4.3)	20.8/21.5(0.7)	20.9/23(2.1)	38.33/42.69(4.36)
	PGD	-	24.4/31.5(7.1)	15.9/28(12.1)	22.4/28.1(5.7)	18.8/25.4(6.6)	18.3/21.6(3.3)	9.6/10.8(1.2)	8.3/10.6(2.3)	16.81/22.29(5.47)
	PGN	-	28.9/37.4(8.5)	19.4/29.9(10.5)	27.0/33.2(6.2)	22.7/29.3(6.6)	20.9/27.4(6.5)	13.8/15.8(2)	8.8/12.5(3.7)	20.21/26.5(6.29)
	SIA	-	96.5/96.9(0.4)	95.4/95.6(0.2)	92.3/93.9(1.6)	91.6/92.1(0.5)	90.2/90.9(0.7)	48.2/50.6(2.4)	65/67.3(2.3)	82.74/83.9(1.16)
	SINI-FGSM	-	76.6/87.9(11.3)	75.5/86.6(11.1)	72.9/84.3(11.4)	69.0/81.9(12.9)	66.9/79.1(12.2)	32.1/39.6(7.5)	35/43.7(8.7)	61.14/71.87(10.73)
	SSA	-	88.3/90.9(2.6)	86.4/88.8(2.4)	80.7/84.6(3.9)	80.0/83.7(3.7)	80.7/83.4(2.7)	52.2/56(3.8)	49.1/51.5(2.4)	73.91/76.99(3.07)
	TI-FGSM	-	32.9/40(7.1)	16.5/23.1(6.6)	24.2/27.8(3.6)	20.8/25.5(4.7)	20.7/24.3(3.6)	14/15.2(1.2)	10.6/12.3(1.7)	19.96/24.03(4.07)
IncRes-v2	BIM	33.7/41.4(7.7)	27.7/34(6.3)	-	24.3/29.7(5.4)	20.8/26.2(5.4)	20.4/26(5.6)	9.5/10.2(0.7)	10/12(2)	20.91/25.64(4.73)
	DI-FGSM	55.8/64.2(8.4)	48.8/59.3(10.5)	-	40.0/48.7(8.7)	38.2/47.1(8.9)	37/45.1(8.1)	13.6/14.2(0.6)	16.9/21.2(4.3)	35.76/42.83(7.07)
	MIG	88.2/89.3(1.1)	84.2/85.8(1.6)	-	83.0/83.5(0.5)	80.3/83.5(3.2)	77.5/81.8(4.3)	42.9/43.6(0.7)	43.8/45.8(2)	71.41/73.33(1.91)
	MI-FGSM	60.1/68.2(8.1)	54/61(7)	-	48.4/54.2(5.8)	45.1/52.0(6.9)	43.5/47.8(4.3)	21/21.4(0.4)	20.7/25.2(4.5)	41.83/47.11(5.29)
	PGD	32.1/38.5(6.4)	23.7/29.5(5.8)	-	22.0/26.1(4.1)	19.3/21.0(1.7)	17.5/20.6(3.1)	8.9/10.1(1.2)	8.5/10.2(1.7)	18.86/22.29(3.43)
	PGN	39.7/42.2(2.5)	30.2/33(2.8)	-	30.6/31.4(0.8)	27.0/28.0(1.0)	24.5/26(1.5)	13.3/14.9(1.6)	11.5/12.1(0.6)	25.26/26.8(1.54)
	SIA	96.5/95.7(-0.8)	95.8/95(-0.8)	-	92.7/92.8(0.1)	91.4/91.6(0.2)	91.6/90.9(-0.7)	50.4/51.4(1)	67.2/68.5(1.3)	83.66/83.7(0.04)
	SINI-FGSM	87.7/92.5(4.8)	84.2/89.7(5.5)	-	78.8/86.1(7.3)	78.2/83.1(4.9)	77.6/82.5(4.9)	37.1/40.9(3.8)	41/45.1(4.1)	69.23/74.27(5.04)
	SSA	90.6/89.8(-0.8)	89.4/89.5(0.1)	-	86.2/86.3(0.1)	84.6/84.6(0.0)	85.5/85.3(-0.2)	60/62.4(2.4)	57.3/57.4(0.1)	79.09/79.33(0.24)
	TI-FGSM	42.8/48.6(5.8)	40.6/47.8(7.2)	-	31.5/36.5(5.0)	28.1/35.1(7.0)	29/34.1(5.1)	16.6/20.7(4.1)	12.9/15.2(2.3)	28.79/34.0(5.21)
Res-152	BIM	27.4/48.3(20.9)	21.1/43.2(22.1)	11.8/29.5(17.7)	29.6/50.3(20.7)	26.3/46.8(20.5)	-	9.3/14.2(4.9)	10.8/20.8(10)	19.47/36.16(16.69)
	DI-FGSM	55.1/58.9(3.8)	52.1/55.5(3.4)	41.5/45.3(3.8)	58.8/65.1(6.3)	56.0/63.0(7.0)	-	14.4/19.7(5.3)	24.5/31.6(7.1)	43.2/48.44(5.24)
	MIG	66.9/78(11.1)	60.5/74.7(14.2)	52.5/68.2(15.7)	71.7/81.7(10.0)	68.5/81.1(12.6)	-	22.2/29.9(7.7)	31/36.3(5.3)	53.33/64.27(10.94)
	MI-FGSM	54.6/67.6(13)	48.3/63(14.7)	40/53.3(13.3)	57.2/70.9(13.7)	53.0/68.8(15.8)	-	19.4/24.9(5.5)	24.9/34.9(10)	42.49/54.77(12.29)
	PGD	22.6/42.9(20.3)	18.8/37.4(18.6)	10.2/24.3(14.1)	25.5/44.3(18.8)	21.7/41.7(20.0)	-	9.2/12.8(3.6)	8.4/16.6(8.2)	16.63/31.43(14.8)
	PGN	29.8/34.6(4.8)	26.1/28.6(2.5)	16.7/20.2(3.5)	36.5/39.7(3.2)	32.7/35.3(2.6)	-	16.6/17.6(1)	12.5/14(1.5)	24.41/27.14(2.73)
	SIA	94.7/95.4(0.7)	95.7/96.1(0.4)	91.3/93(1.7)	95.9/96.9(1.0)	95.9/96.3(0.4)	-	46.3/47(0.7)	71.9/75.6(3.7)	84.53/85.76(1.23)
	SINI-FGSM	67.8/81.6(13.8)	62.5/79.9(17.4)	53.8/73.6(19.8)	72.4/86.8(14.4)	68.3/84.2(15.9)	-	22.6/27.9(5.3)	28.3/38.6(10.3)	53.67/67.51(13.84)
	SSA	90.3/92.4(2.1)	89.2/91.5(2.3)	86.4/89(2.6)	92.7/93.4(0.7)	91.7/92.9(1.2)	-	62.9/68.2(5.3)	59.6/66.1(6.5)	81.83/84.79(2.96)
	TI-FGSM	37.3/43(5.7)	38.2/43.9(5.7)	21.7/30(8.3)	38.3/44.4(6.1)	35.4/39.6(4.2)	-	24.1/29.9(5.8)	16.7/21.9(5.2)	30.24/36.1(5.86)
ViT-B/16	BIM	30.4/33.8(3.4)	24.3/26.6(2.3)	17.4/20.5(3.1)	28.4/29.8(1.4)	25.4/29.4(4.0)	23.5/26.5(3)	-	22.8/25(2.2)	24.6/27.37(2.77)
	DI-FGSM	49.9/52.1(2.2)	44.2/47.2(3)	38.3/40.7(2.4)	46.7/49.9(3.2)	43.2/47.9(4.7)	40.6/44.9(4.3)	-	42.5/48(5.5)	43.63/47.24(3.61)
	MIG	67.9/70.5(2.6)	62.1/63.8(1.7)	55.8/57.1(1.3)	66.0/67.5(1.5)	62.4/63.8(1.4)	61.1/61.9(0.8)	-	54.3/56.9(2.6)	61.37/63.07(1.7)
	MI-FGSM	57.7/59(1.3)	53.8/54.1(0.3)	45.9/47.7(1.8)	56.4/56.4(0.0)	54.0/54.1(0.1)	50.7/52.2(1.5)	-	48.6/50.7(2.1)	52.44/53.46(1.01)
	PGD	28/30.9(2.9)	21.2/23.5(2.3)	15.6/18.3(2.7)	27.1/29.7(2.6)	23.4/26.6(3.2)	22.2/24.9(2.7)	-	21.1/24.2(3.1)	22.66/25.44(2.79)
	PGN	37.2/43.9(6.7)	31.8/35.9(4.1)	22.9/29.8(6.9)	34.2/41.4(7.2)	31.2/37.4(6.2)	30.2/36(5.8)	-	28.1/34.6(6.5)	30.8/37.0(6.2)
	SIA	89.9/90.5(0.6)	87.6/87.2(-0.4)	84.2/86(1.8)	88.6/88.5(-0.1)	87.4/87.6(0.2)	85.9/85.8(-0.1)	-	89.8/91.1(1.3)	87.63/88.1(0.47)
	SINI-FGSM	68.3/70.7(2.4)	64.6/66.5(1.9)	59.1/59.7(0.6)	68.0/68.0(0.0)	64.0/64.5(0.5)	60.5/60.4(-0.1)	-	54.8/57.5(2.7)	62.76/63.9(1.14)
	SSA	77.2/77.6(0.4)	72.1/74.4(2.3)	70.3/70.5(0.2)	76.1/75.9(-0.2)	73.3/74.9(1.6)	71.9/74(2.1)	-	62.8/65.2(2.4)	71.96/73.21(1.26)
	TI-FGSM	32.4/35.2(2.8)	29.6/33.2(3.6)	19.3/22.8(3.5)	26.7/30.4(3.7)	26.1/28.9(2.8)	26/29.3(3.3)	-	24.1/28.3(4.2)	26.31/29.73(3.41)

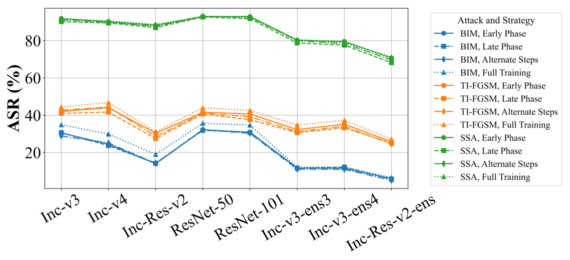


Figure 7: Comparative summary of attack method performances using the ResNet-152 source model under four distinct training strategies with the implementation of the proposed DBA technique. The strategies include Early Phase (First 50%), Late Phase (Last 50%), Alternate Steps (Every 1-step interval), and Full Training (All steps), each demonstrating the effectiveness of the DBA technique in enhancing attack robustness across different phases of training.

4.3.2 Impact of Training Strategies on DBA Performance

In this section, ResNet-152 is employed as the source model to investigate the influence of four distinct training strategies on DBA performance. The strategies consist of training for the first 50% steps, training for the last 50% steps, training at every 1-step interval, and training all steps. The unified learning rate is set to 0.0001. As shown in Figure 7, when enhancing the BIM attack, the strategies of training for the first 50% steps and all steps consistently better. On the other hand, for enhancing the TI-FGSM and SSA methods, the performance of training all steps significantly outperform the other three strategies.

5 CONCLUSION

In this paper, we introduced the Decision Boundary Adaptation (DBA) approach, a simple, efficient, and easily implementable method to enhance adversarial attack transferability across DNN models. DBA exploits the similarity between source and target model decision boundaries, improving adversarial example transferability through boundary manipulation. We evaluated DBA on state-of-the-art adversarial attacks, demonstrating its effectiveness in enhancing black-box attack performance. Our results show that DBA significantly improves adversarial transferability, achieving state-of-the-art performance. However, DBA introduces additional computational overhead ($\sim 5\%$ on average), which, though manageable, may challenge deployment on resource-limited servers. Overall, DBA provides valuable insights for future research on strengthening DNN robustness against black-box adversarial attacks.

References

Giovanni Apruzzese, Mauro Conti, and Ying Yuan. Spacephish: The evasion-space of adversarial attacks against phishing website detectors using machine learning. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 171–185, 2022.

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

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

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Zhijin Ge, Fanhua Shang, Hongying Liu, Yuanyuan Liu, and Xiaosen Wang. Boosting adversarial transferability by achieving flat local maxima. *arXiv preprint arXiv:2306.05225*, 2023.

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

Martin Gubri, Maxime Cordy, Mike Papadakis, Yves Le Traon, and Koushik Sen. Lgv: Boosting adversarial example transferability from large geometric vicinity. In *European Conference on Computer Vision*, pages 603–618. Springer, 2022.

Yonis Gulzar. Fruit image classification model based on mobilenetv2 with deep transfer learning technique. *Sustainability*, 15(3):1906, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016a.

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

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Zhibo Jin, Zhiyu Zhu, Xinyi Wang, Jiayu Zhang, Jun Shen, and Huaming Chen. Danaa: Towards transferable attacks with double adversarial neuron attribution. In *International Conference on Advanced Data Mining and Applications*, pages 456–470. Springer, 2023.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

Erich L Lehmann and George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

Yinglong Li. Research and application of deep learning in image recognition. In *2022 IEEE 2nd International Conference on Power, Electronics and Computer Applications (ICPECA)*, pages 994–999. IEEE, 2022.

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

Yuyang Long, Qilong Zhang, Boheng Zeng, Lianli Gao, Xianglong Liu, Jian Zhang, and Jingkuan Song. Frequency domain model augmentation for adversarial attack. In *European Conference on Computer Vision*, pages 549–566. Springer, 2022.

Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

Wenshuo Ma, Yidong Li, Xiaofeng Jia, and Wei Xu. Transferable adversarial attack for both vision transformers and convolutional networks via momentum integrated gradients. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4630–4639, 2023.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Jörg Martin and Clemens Elster. Inspecting adversarial examples using the fisher information. *Neurocomputing*, 382:80–86, 2020.

Ahmed Omara and Burak Kantarci. An ai-driven solution to prevent adversarial attacks on mobile vehicle-to-microgrid services. *Simulation Modelling Practice and Theory*, page 103016, 2024.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

Zeyu Qin, Yanbo Fan, Yi Liu, Li Shen, Yong Zhang, Jue Wang, and Baoyuan Wu. Boosting the transferability of adversarial attacks with reverse adversarial perturbation. *Advances in neural information processing systems*, 35: 29845–29858, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204, 2018.

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

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

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

Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*, pages 459–479. Springer, 2022.

Xiaosen Wang, Zeliang Zhang, and Jianping Zhang. Structure invariant transformation for better adversarial transferability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4607–4619, 2023.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2730–2739, 2019.

Jianping Zhang, Weibin Wu, Jen-tse Huang, Yizhan Huang, Wenxuan Wang, Yuxin Su, and Michael R Lyu. Improving adversarial transferability via neuron attribution-based attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14993–15002, 2022.

Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *International Conference on Machine Learning*, pages 26982–26992. PMLR, 2022.

Hegui Zhu, Yuchen Ren, Xiaoyan Sui, Lianping Yang, and Wuming Jiang. Boosting adversarial transferability via gradient relevance attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4741–4750, 2023a.

Zhiyu Zhu, Huaming Chen, Jiayu Zhang, Xinyi Wang, Zhibo Jin, Qinghua Lu, Jun Shen, and Kim-Kwang Raymond Choo. Improving adversarial transferability via frequency-based stationary point search. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3626–3635, 2023b.

A *PROOF* OF THEOREM 1

Proof of Theorem 1. Consider an infinitesimal $\mathcal{O}$.

Given $f(x+\epsilon\cdot\Delta x) = f(x)+\epsilon\cdot\Delta x\cdot\frac{\partial f(x)}{\partial x}+\mathcal{O}$, we can express the incremental change in $f(x)$ due to a small perturbation $\epsilon\cdot\Delta x$ as:

$$\frac{f(x+\epsilon\cdot\Delta x)-f(x)}{\epsilon} = \Delta x\cdot\frac{\partial f(x)}{\partial x}+\frac{\mathcal{O}}{\epsilon} \tag{12}$$

This expression implies that the rate of change of $f(x)$ with respect to ϵ approximates the directional derivative along Δx, with $\mathcal{O}/\epsilon$ representing the higher-order terms that become negligible as $\epsilon\to 0$.

To prove $\frac{f(x+\epsilon\cdot\Delta x)-f(x-\epsilon\cdot\Delta x)}{2\epsilon}$ follows similarly, we utilize the symmetric difference quotient, which accounts for the function value at $x+\epsilon\cdot\Delta x$ and $x-\epsilon\cdot\Delta x$, providing a more accurate approximation of the derivative:

$$\frac{f(x+\epsilon\cdot\Delta x)-f(x-\epsilon\cdot\Delta x)}{2\epsilon} = \Delta x\cdot\frac{\partial f(x)}{\partial x}+\frac{\mathcal{O}}{\epsilon} \tag{13}$$

This symmetric formulation further reduces the error in the approximation, leading to a more accurate estimation of the directional derivative in the limit as $\epsilon\to 0$.

□

B *PROOF* OF THEOREM 2:

Proof of Theorem 2. The Fisher information matrix $\mathcal{I}_\theta(x)$ can be expressed as:

$$\mathcal{I}_\theta(x) = E_{y^c\sim P(y|x,\theta)}\left[\frac{\partial\log P_{y^c}(x,\theta)^\top}{\partial x}\cdot\frac{\partial\log P_{y^c}(x,\theta)}{\partial x}\right] \tag{14}$$

This can be expanded as:

$$\mathcal{I}_\theta(x) = \sum_{i=1}^{c}\frac{\partial\log P_{y^c}(x,\theta)^\top}{\partial x}\cdot\frac{\partial\log P_{y^c}(x,\theta)}{\partial x}\cdot P_{y^c}(x,\theta) \tag{15}$$

Considering the expectation of the second derivative of $P_{y^c}(x,\theta)$ with respect to x normalized by $P_{y^c}(x,\theta)$:

$$E_{y^c\sim P(y|x,\theta)}\left[\frac{\partial^2 P_{y^c}(x,\theta)}{\partial x^2}\cdot\frac{1}{P_{y^c}(x,\theta)}\right] = \frac{\partial^2}{\partial x^2}\sum P_{y^c}(x,\theta) = 0 \tag{16}$$

The second derivative of $\log P_{y^c}(x,\theta)$ is given by:

$$\frac{\partial^2}{\partial x^2}\log P_{y^c}(x,\theta) = \frac{\frac{\partial^2 P_{y^c}(x,\theta)}{\partial x^2}}{P_{y^c}(x,\theta)} - \frac{\left(\frac{\partial P_{y^c}(x,\theta)}{\partial x}\right)^\top\cdot\frac{\partial P_{y^c}(x,\theta)}{\partial x}}{P_{y^c}^2(x,\theta)} \tag{17}$$

Therefore, the expectation of the product of the gradients of the log likelihoods is equivalent to the negative expectation of the Hessian matrix of the log likelihood:

$$E_{y^c\sim P(y|x,\theta)}\left[\frac{\partial\log P_{y^c}(x,\theta)^\top}{\partial x}\cdot\frac{\partial\log P_{y^c}(x,\theta)}{\partial x}\right] = -E_{y^c\sim P(y|x,\theta)}\left[\mathrm{H}^c\right] \tag{18}$$

By integrating over the conditional distribution and approximating for all class labels c, we can rewrite the Fisher information matrix as:

$$\mathcal{I}_\theta(x) = \sum_{c=1}^{C}\frac{\partial\log P_{y^c}(x,\theta)}{\partial x}\cdot P_{y^c}(x,\theta)\cdot\left(\frac{\partial\log P_{y^c}(x,\theta)}{\partial x}\right)^\top \tag{19}$$

□

C *PROOF* OF THEOREM 3:

Proof of Theorem 3. The Fisher information matrix $\mathcal{I}_\theta(x)$ is given by:

$$\begin{aligned}\mathcal{I}_\theta(x) &= E_{y^c \sim P(y|x,\theta)}\left[\frac{\partial \log P_{y^c}(x,\theta)^\top}{\partial x} \cdot \frac{\partial \log P_{y^c}(x,\theta)}{\partial x}\right] \\ &= \sum_{c=1}^{C} P_{y^c}(x,\theta) \cdot \frac{\partial \log P_{y^c}(x,\theta)^\top}{\partial x} \cdot \frac{\partial \log P_{y^c}(x,\theta)}{\partial x} \\ &= \sum_{c=1}^{C} \frac{\partial P_{y^c}(x,\theta)^\top}{\partial x} \cdot \frac{\partial \log P_{y^c}(x,\theta)}{\partial x}\end{aligned} \tag{20}$$

The trace of $\mathcal{I}_\theta(x)$ is:

$$\text{tr}(\mathcal{I}_\theta(x)) = \sum_{i=1}^{m} e^i \mathcal{I}_\theta(x) e^i \tag{21}$$

Considering the flatness of $\mathcal{I}_\theta(x)$:

$$\begin{aligned} flat(\mathcal{I}_\theta(x)) &= \Delta x \mathcal{I}_\theta(x) \Delta x^\top \\ &= \sum_{c=1}^{C} \Delta x \cdot \frac{\partial P_{y^c}(x,\theta)^\top}{\partial x} \cdot \Delta x \cdot \frac{\partial \log P_{y^c}(x,\theta)}{\partial x} \\ &= \left\langle \Delta x \cdot \frac{\partial P_y(x,\theta)}{\partial x}, \Delta x \cdot \frac{\partial \log P_y(x,\theta)}{\partial x} \right\rangle \end{aligned} \tag{22}$$

Given:

$$\frac{\Delta x \cdot \frac{\partial P_y(x,\theta)}{\partial x}}{2\epsilon} \approx \frac{1}{2\epsilon}\left(\frac{\partial P_y(x+\epsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial P_y(x-\epsilon\cdot\Delta x,\theta)}{\partial x}\right) \tag{23}$$

$$\frac{\Delta x \cdot \frac{\partial \log P_y(x,\theta)}{\partial x}}{2\epsilon} \approx \frac{1}{2\epsilon}\left(\frac{\partial \log P_y(x+\epsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial \log P_y(x-\epsilon\cdot\Delta x,\theta)}{\partial x}\right) \tag{24}$$

Hence, **Theorem 3** is proven as:

$$\begin{aligned} flat(\mathcal{I}_\theta(x)) &\approx \frac{1}{4\varepsilon^2}\left\langle \frac{\partial P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}, \right. \\ &\qquad \left. \frac{\partial \log P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial \log P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x} \right\rangle \\ &\propto \left\langle \frac{\partial P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x}, \right. \\ &\qquad \left. \frac{\partial \log P_y(x+\varepsilon\cdot\Delta x,\theta)}{\partial x} - \frac{\partial \log P_y(x-\varepsilon\cdot\Delta x,\theta)}{\partial x} \right\rangle \end{aligned} \tag{25}$$

The same can be proven for $flat(\mathcal{I}_x(\theta))$.

□

D *PROOF* OF THE RELATIONSHIP BETWEEN KL DIVERGENCE AND FISHER INFORMATION MATRIX

Proof of the Relationship between KL Divergence and Fisher Information Matrix. The KL divergence between $P_y(x,\theta)$ and $P_y(x+\Delta x,\theta)$ can be approximated as:

$$KL(P_y(x,\theta), P_y(x+\Delta x,\theta)) = \frac{1}{2}\Delta x^\top \cdot \left(\frac{\partial^2 KL(P_y(x,\theta), P_y(x+\Delta x,\theta))}{\partial(x+\Delta x)_i \partial(x+\Delta x)_j}\right)_{\Delta x \to 0} \Delta x + \mathcal{O}$$

The second derivative of the KL divergence at $\Delta x \to 0$ is given by:

$$\left(\frac{\partial^2 KL(P_y(x,\theta), P_y(x+\Delta x,\theta))}{\partial(x+\Delta x)_i \partial(x+\Delta x)_j}\right)_{\Delta x\to 0} = -\sum_{c=1}^{C} P_y(x,\theta)\cdot\left(\frac{\partial^2 \log P_y(x+\Delta x,\theta)}{\partial(x+\Delta x)_i \partial(x+\Delta x)_j}\right)_{\Delta x\to 0} = [\mathcal{I}(\theta)]_{ij}$$

□

This result demonstrates the equivalence between the second derivative of the KL divergence and the Fisher Information Matrix $\mathcal{I}(\theta)$, thereby establishing the relationship between them.

E THE METHOD OF DRAWING VARIATION FIGURE OF $flat(\cdot)$

For the computations depicted in the above figures, we calculated k curves corresponding to the number of samples, normalized all $flat(\cdot)$ values to the 0-1 range using their maximum values, and then plotted the average of these k samples.

F PRACTICAL IMPLEMENTATION STEPS OF DBA

More detailed explanations regarding the practical implementation of DBA. The core implementation of DBA is centered around gradient descent operations to adjust model parameters, optimizing the flatness of decision boundaries. Specifically, the process is as follows:

- **Gradient Information Computation:** Compute gradient information for the sample to assess the properties of the decision boundary in its vicinity.
- **Parameter Update:** Use the computed gradients to update the model parameters via gradient descent, ensuring that samples move away from unstable regions near the decision boundary during the attack process.
- **Iterative Optimization:** Repeat this process iteratively to enhance the stability of attack directions and improve the transferability of adversarial examples.

This method allows DBA to maintain the efficiency of existing adversarial attack algorithms while significantly enhancing cross-model attack capabilities through parameter optimization. We validated this implementation extensively in our experiments, demonstrating that DBA consistently and reliably improves the performance of existing methods.

G DBA IMPLEMENTATION

Algorithm 1 Decision Boundary Adaptation Algorithm

```
Input: parameter of the source model $\theta$, input $x$, target $y$, learning rate of model boundaries $\alpha_1$, learning rate of perturbations $\alpha_2$, warm up step $w$
Output: $x_I$
1: $Init\ x_0 \leftarrow x, \theta_0 \leftarrow \theta$
2: for $i = 1 \cdots I$ do
3:     $\theta^i = \theta^{i-1} + [[i \geq w]] \cdot \alpha_1 \left(\frac{\partial L(y|x_{i-1},\theta^{i-1})}{\partial \theta^{i-1}}\right)$
4:     $\eta_i = \alpha_2 \cdot \text{sign}\left(\frac{\partial L(y|x_{i-1},\theta^{i-1})}{\partial(x_{i-1}+\eta_{i-1})}\right)$
5: end for
6: $x_I = x_{i-1} + \eta_i = x_0 + \sum_{j=1}^{i} \eta_j = x_0 + \eta^i$
```

H PARAMETERS SETTING

All our experiments are conducted on one NVIDIA RTX 6000 Ada graphics card. In our experiments across all models, we set the maximum perturbation of all algorithms to $\frac{16}{255}$, with an attack step size of 10. Notably, we set the batch size to

1; the appendix discusses other batch sizes, highlighting the optimal performance enhancement of DBA at a batch size of 1. Additionally, specific parameters were allocated to each baseline attack method to ensure evaluation consistency and fairness. For the DI-FGSM method, we set the decay parameter to 0, the resize rate to 0.9, and the diversity probability to 0.5. Similarly, for the TI-FGSM method, the decay parameter was set to 0, employing a Gaussian kernel with a length of 15 and a standard deviation of 3. The resize rate and diversity probability were set to 0.9 and 0.5, respectively. For the MI-FGSM method, the decay parameter was set to 1. For the SINI-FGSM method, the decay parameter was set to 1, with parameter m set to 5. For the SSA method, we used a kernel length of 7, standard deviation of 3, momentum of 1, N of 20, σ of 16, and ρ of 0.5.

I RESULT OF DBA WITH EXPENDED ATTACK METHOD

We have supplemented our experiments with the RAP [Qin et al., 2022] method by evaluating its performance both with and without the integration of DBA in Table 3. The results are summarized in the table below. Using Inc-v3 as the source model, it is evident that our DBA method consistently enhances the transferability of RAP. In nearly all black-box models, performance improvements are observed, with an average improvement of 0.66%.

Table 3: Result of RAP with and without DBA

	RAP	RAP with DBA	Gap
Inc-v3	95.20%	95.10%	-0.10%
Inc-v4	46.00%	46.90%	0.90%
IncRes-v2	44.00%	45.10%	1.10%
Res-50	45.70%	45.70%	0.00%
Res-101	41.40%	41.70%	0.30%
Res-152	39.80%	40.70%	0.90%
Inc-v3-ens3	22.30%	23.80%	1.50%
Inc-v3-ens4	22.10%	22.80%	0.70%
IncRes-ens	11.70%	12.80%	1.10%
ViT-B/16	20.60%	20.60%	0.00%
MaxViT-T	21.60%	21.70%	0.10%

J RESULT OF DBA WITH BATCH GRADIENT DESCENT

The data in Table 4 and Table 6 represent the results of all experiments conducted with a batch size of 10. In scenarios where gradient descent is applied to each sample individually, the batch size does not affect the attack outcome, as the computations for each sample are independent. Furthermore, regarding the optimization method mentioned in Eq. 11, we have conducted experiments detailed in this section, which also demonstrate consistent improvement across various settings.

K COMPUTATIONAL EFFICIENCY ANALYSIS OF DBA

Table 5 presents the detailed frame-per-second (FPS) measurements for different adversarial attack methods applied to two target models, with and without incorporating DBA. The results show that DBA introduces a modest computational overhead across all configurations. The maximum overhead observed is below 9%, with an average overhead of approximately 5%. This confirms that DBA is computationally efficient and can be integrated into existing black-box attack pipelines with minimal runtime impact.

Model	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-50	Res-101	Res-152	Average
Inc-v3	BIM	100/98.2(-1.8)	27.5/32.6(5.1)	20.1/24.2(4.1)	24.1/29.4(5.3)	22.2/24.6(2.4)	18/23.5(5.5)	35.32/38.75(3.43)
	PGD	100/100(0)	23.8/27.8(4)	16.6/19.1(2.5)	22.9/23.9(1)	19/20.7(1.7)	18.3/19.6(1.3)	33.43/35.18(1.75)
	DI-FGSM	99.8/99.7(-0.1)	46/50.5(4.5)	38.8/41.3(2.5)	37.8/42.8(5)	34.2/37.8(3.6)	32.7/36(3.3)	48.22/51.35(3.13)
	TI-FGSM	98.6/98.1(-0.5)	32/40.9(8.9)	17.3/24.5(7.2)	24.8/29.8(5)	21.7/25.8(4.1)	20.8/23.9(3.1)	35.87/40.5(4.63)
	MI-FGSM	100/99.8(-0.2)	50.6/64.2(13.6)	45.7/59(13.3)	47.2/58.9(11.7)	42.2/55.2(13)	40.6/54.3(13.7)	54.38/65.23(10.85)
	SINI-FGSM	100/100(0)	77.4/88.1(10.7)	75.2/87.5(12.3)	73.3/84.6(11.3)	69.1/82.4(13.3)	69.4/80.5(11.1)	77.4/87.18(9.78)
	SSA	99.3/99.4(0.1)	87.5/90.5(3)	86.7/88.7(2)	81.9/84.6(2.7)	79.3/83.7(4.4)	80.4/83.3(2.9)	85.85/88.37(2.53)
	NAA	97.4/97.5(0.1)	87.8/87.9(0.1)	85.8/86.4(0.6)	82.8/83.8(1)	82.4/82.8(0.4)	81.4/81.9(0.5)	86.27/86.72(0.45)
Inc-v4	BIM	36.4/41.5(5.1)	99.9/99.6(-0.3)	18.9/26.5(7.6)	23.4/28(4.6)	20.5/25.9(5.4)	19.3/23.7(4.4)	36.4/40.87(4.47)
	PGD	33.4/36.6(3.2)	99.9/99.3(-0.6)	14.3/21.4(7.1)	21.3/25.8(4.5)	17.7/20.2(2.5)	17.5/20(2.5)	34.02/37.22(3.2)
	DI-FGSM	54.9/59.3(4.4)	99.5/98.6(-0.9)	37.4/43.8(6.4)	35.7/43.9(8.2)	32.3/38.4(6.1)	31/39.2(8.2)	48.47/53.87(5.4)
	TI-FGSM	38.7/45.9(7.2)	97.8/97.2(-0.6)	17.6/25.3(7.7)	23.9/28.6(4.7)	20.5/24.9(4.4)	21.2/24.3(3.1)	36.62/41.03(4.41)
	MI-FGSM	60.9/68.1(7.2)	99.8/99.8(0)	45.9/53.6(7.7)	45.6/52.4(6.8)	42.2/47.1(4.9)	41.2/46.9(5.7)	55.93/61.32(5.39)
	SINI-FGSM	86.6/92.6(6)	100/100(0)	78.7/87.2(8.5)	77.6/83.8(6.2)	73.8/82.2(8.4)	72.8/81(8.2)	81.58/87.8(6.22)
	SSA	91.7/91.2(-0.5)	98.9/98.7(-0.2)	86.6/87.9(1.3)	82.8/85.7(2.9)	80.6/83.1(2.5)	82.7/84.7(2)	87.22/88.55(1.33)
	NAA	88/88.2(0.2)	98/98(0)	83.7/83.9(0.2)	82.6/82.2(-0.4)	82.2/82.2(0)	79.9/80(0.1)	85.73/86.9(1.17)
IncRes-v2	BIM	35.4/43.8(8.4)	29.3/33.7(4.4)	99.2/99.4(0.2)	24.5/30.4(5.9)	21.2/26.7(5.5)	21.4/25(3.6)	38.5/43.17(4.67)
	PGD	30.8/37.9(7.1)	23.7/28.4(4.7)	99.4/99.5(0.1)	21.8/24.8(3)	19.7/22.1(2.4)	17.7/19.4(1.7)	35.52/38.68(3.16)
	DI-FGSM	56.7/63.8(7.1)	49.4/59.2(9.8)	98/98.7(0.7)	41.7/48.7(7)	38.4/44.1(5.7)	38.6/45(6.4)	53.8/59.92(6.12)
	TI-FGSM	42.7/51.9(9.2)	41.7/49.2(7.5)	94.7/94.9(0.2)	30.9/37.7(6.8)	29.6/35.4(5.8)	28.7/36.3(7.6)	44.72/50.9(6.18)
	MI-FGSM	61.4/68.9(7.5)	53/60.8(7.8)	99.1/99.5(0.4)	48.2/55.3(7.1)	46.6/51.5(4.9)	44.7/50.1(5.4)	58.83/64.35(5.52)
	SINI-FGSM	87.4/92.2(4.8)	83.6/88(4.4)	99.9/99.9(0)	79.2/84.2(5)	77.3/84.9(7.6)	75/81.9(6.9)	83.73/88.52(4.79)
	SSA	89.4/90.5(1.1)	89.4/89.5(0.1)	97.7/97.8(0.1)	85.9/87.3(1.4)	83.9/85.6(1.7)	85.7/86.3(0.6)	88.67/89.5(0.83)
	NAA	83.9/84.3(0.4)	80.5/81.2(0.7)	92.2/92.5(0.3)	79.5/79.8(0.3)	78.1/78.6(0.5)	77.4/77.1(-0.3)	81.93/82.25(0.32)
Res-152	BIM	25.6/47.8(22.2)	21.6/42.2(20.6)	11.7/31.5(19.8)	28.6/48.6(20)	26.1/45.5(19.4)	24/44.2(20.2)	22.93/43.3(20.37)
	PGD	22.9/39.4(16.5)	18.4/32(13.6)	10.5/23.1(12.6)	24.5/43.1(18.6)	21.7/36.8(15.1)	20.4/36.1(15.7)	19.73/35.08(15.35)
	DI-FGSM	53.8/59.6(5.8)	51.5/55.2(3.7)	40.1/45.7(5.6)	60.7/63.6(2.9)	57.6/62.1(4.5)	54.9/60.8(5.9)	53.1/57.83(4.73)
	TI-FGSM	37.7/45.4(7.7)	37.3/44.2(6.9)	23.3/31.4(8.1)	38.2/44.1(5.9)	34.6/42.5(7.9)	34.7/43.1(8.4)	34.3/41.78(7.48)
	MI-FGSM	55/72.7(17.7)	48.8/69.5(20.7)	39.7/60(20.3)	58.3/74.1(15.8)	52.7/72.5(19.8)	52/72(20)	51.08/70.13(19.05)
	SINI-FGSM	68.1/86.5(18.4)	63.2/83.8(20.6)	53.6/79.1(25.5)	72.6/90.4(17.8)	68.4/88.8(20.4)	68.3/87(18.7)	65.7/85.93(20.23)
	SSA	90.4/91.7(1.3)	89/90.6(1.6)	86.6/88.7(2.1)	92.4/92.9(0.5)	91.9/93(1.1)	92.8/93.4(0.6)	90.52/91.72(1.2)
	NAA	88.1/89(0.9)	86.9/87.5(0.6)	81.6/83.1(1.5)	89.3/91.1(1.8)	90.2/90.9(0.7)	88.8/89.8(1)	87.48/88.57(1.09)

Table 4: Attack Success Rates on Non-defensive Training Models. Each data group follows this pattern: Baseline/Baseline+DBA(Gap), with green indicating enhancement by DBA over the original method, while red indicates a decrease. Importantly, it should be noted that, compared to white-box attacks, more attention should be focused on black-box attacks.

Table 5: Comparison of attack efficiency in frames per second (FPS) with and without DBA. Although DBA introduces a slight computational overhead, the decrease in FPS remains marginal (mostly within 5% on average), indicating that DBA maintains high efficiency across various models and attack methods.

Attack	Model	FPS (No DBA)	FPS (With DBA)	Overhead
MIG	Inception-v3	0.4524	0.4355	3.88%
	MaxViT-T	0.1786	0.1731	3.20%
PGN	Inception-v3	0.5919	0.5490	7.83%
	MaxViT-T	0.2307	0.2121	8.73%
SIA	Inception-v3	0.1659	0.1602	3.62%
	MaxViT-T	0.1603	0.1576	1.74%
SSA	Inception-v3	0.6360	0.5891	7.97%
	MaxViT-T	0.3074	0.2974	3.34%

Model	Attack	Inc-v3-ens3	Inc-v3-ens4	IncRes-v2-ens	Average
Inc-v3	BIM	12.4/13.4(1)	13.5/13.3(-0.2)	4.9/6.2(1.3)	10.27/10.97(0.7)
	PGD	12.6/12.3(-0.3)	12.4/13.4(1)	6.5/6.2(-0.3)	10.5/10.63(0.13)
	DI-FGSM	17/19.2(2.2)	17.5/18.9(1.4)	8/9.3(1.3)	14.17/15.8(1.63)
	TI-FGSM	22/27(5)	24.4/28.9(4.5)	12.9/15.5(2.6)	19.77/23.8(4.03)
	MI-FGSM	21.8/25.3(3.5)	22.6/26.3(3.7)	10.1/12.9(2.8)	18.17/21.5(3.33)
	SINI-FGSM	39.5/47.5(8)	38.1/45.6(7.5)	22.7/27.7(5)	33.43/40.27(6.84)
	SSA	74.5/79.6(5.1)	73.4/77.3(3.9)	59.2/64.8(5.6)	69.03/73.9(4.87)
	NAA	57.8/57.1(-0.7)	55.2/56.3(1.1)	34.7/34.2(-0.5)	49.23/49.2(-0.03)
Inc-v4	BIM	10.2/11.7(1.5)	12.4/11.9(-0.5)	5.5/6(0.5)	9.37/9.87(0.5)
	PGD	11.7/12.1(0.4)	11.5/11.7(0.2)	6/5.5(-0.5)	9.73/9.77(0.04)
	DI-FGSM	14/16.7(2.7)	15.4/16.2(0.8)	7.5/9.3(1.8)	12.3/14.07(1.77)
	TI-FGSM	19.3/23(3.7)	20.3/26.1(5.8)	13.2/15.8(2.6)	17.6/21.63(4.03)
	MI-FGSM	20.6/22.4(1.8)	18.3/19.9(1.6)	10.8/12.1(1.3)	16.57/18.13(1.56)
	SINI-FGSM	46.9/54.2(7.3)	43.4/50.8(7.4)	29.4/33.8(4.4)	39.9/46.27(6.37)
	SSA	74.9/77.2(2.3)	72.1/77.4(5.3)	63.8/67(3.2)	70.27/73.87(3.6)
	NAA	59.7/59.9(0.2)	56.7/57.3(0.6)	39.8/39.5(-0.3)	52.07/52.23(0.16)
IncRes-v2	BIM	9.4/11.7(2.3)	9.9/12.4(2.5)	6.5/7.1(0.6)	8.6/10.4(1.8)
	PGD	11.8/11.6(-0.2)	10.8/12(1.2)	7/7.1(0.1)	9.87/10.23(0.36)
	DI-FGSM	16.3/20.4(4.1)	16.8/18.6(1.8)	10/12.6(2.6)	14.37/17.2(2.83)
	TI-FGSM	28.2/35.3(7.1)	28.2/34.8(6.6)	25.3/33.7(8.4)	27.23/34.6(7.37)
	MI-FGSM	21.9/25.5(3.6)	22.5/24(1.5)	13.4/16.3(2.9)	19.27/21.93(2.66)
	SINI-FGSM	55.8/62.5(6.7)	49/55.1(6.1)	40.9/45.6(4.7)	48.57/54.4(5.83)
	SSA	80.7/81.7(1)	77.3/79.4(2.1)	76.9/77.4(0.5)	78.3/79.5(1.2)
	NAA	63.8/63.7(-0.1)	56.7/57.2(0.5)	50.5/51.2(0.7)	57/57.37(0.37)
Res-152	BIM	11.8/15.3(3.5)	11.7/17.1(5.4)	5/8.8(3.8)	9.5/13.73(4.23)
	PGD	11.9/12.3(0.4)	11.4/12.8(1.4)	5.5/7.2(1.7)	9.6/10.77(1.17)
	DI-FGSM	15.9/23(7.1)	16.6/24.5(7.9)	8.6/13.6(5)	13.7/20.37(6.67)
	TI-FGSM	28.3/36.8(8.5)	31.1/38.5(7.4)	21.7/29.5(7.8)	27.03/34.93(7.9)
	MI-FGSM	19/28.5(9.5)	19.9/28(8.1)	10/15.5(5.5)	16.3/24(7.7)
	SINI-FGSM	25.4/41.3(15.9)	26.5/38.7(12.2)	13.3/21(7.7)	21.73/33.67(11.94)
	SSA	78.1/80.4(2.3)	77.3/79.1(1.8)	67.3/71.2(3.9)	74.23/76.9(2.67)
	NAA	41.1/43.5(2.4)	40/41.2(1.2)	23.2/23.9(0.7)	34.77/36.2(1.43)

Table 6: Attack Success Rates on Defensive Training Models. Each data group follows this pattern: Baseline/Baseline+DBA(Gap), with green indicating enhancement by DBA over the original method, while red indicates a decrease.