BounceAttack: A Query-Efficient Decision-based Adversarial Attack by Bouncing into the Wild

Jie Wan[1†], Jianhao Fu[1†], Lijin Wang[1], Ziqi Yang[1,2*]
[1] *Zhejiang University*
[2] *ZJU-Hangzhou Global Scientific and Technological Innovation Center*
{*wanjie, 3180102970, wanglijin, yangziqi*}*@zju.edu.cn*

2024 IEEE Symposium on Security and Privacy (SP) | 979-8-3503-3130-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/SP54263.2024.00068

***Abstract*—Deep neural networks are vulnerable to adversarial attacks. We study such threats in the decision-based black-box setting where the adversary could obtain only the predicted labels of the victim classifier within limited queries and aims at performing targeted and untargeted adversarial attacks under different perturbation constraints. In this paper, we propose BounceAttack as a query-efficient attack method. We propose the bounce vector which encourages the iterations to maximally explore the adversarial space towards the optimal adversarial example within limited queries to improve the query efficiency. We perform extensive experiments on various benchmark datasets and models. Experimental results show that BounceAttack achieves both high query efficiency and small perturbation size. BounceAttack outperforms existing attack methods. For example, BounceAttack achieves 48.1% smaller perturbation compared to the state-of-the-art attack methods on average using the same number of model queries.**

1. Introduction

Deep neural networks (DNNs) have made state-of-the-art progress in many fields, including automatic driving, image classification, and face recognition. However, DNNs are vulnerable to adversarial attacks [1], [2], where imperceptible perturbations are added to benign inputs and thus lead to false classification results. Existing adversarial attacks generally fall into two categories based on the adversary's capabilities. In the first category, the adversary is assumed to have access to the details of the victim model (e.g., model structure, parameters, and the training data), such that the adversary could find the optimal adversarial perturbation using gradient information in a white-box manner [2]–[6]. In the second category, the adversary has limited information about the victim model and usually treats it as a black box where the adversary could obtain only the output information such as the predicted label or additionally the confidence vector [7]–[11]. In this work, we focus on the black-box scenario, and more specifically, only the predicted label is available to the adversary.

Depending on the knowledge that the adversary could obtain, existing black-box adversarial attacks can be largely classified into transfer-based attacks, score-based attacks, and decision-based attacks. In the transfer-based attack [12]–[15], the adversary has access to an auxiliary dataset and trains a substitute model for the victim model. Hence, the adversary could craft the adversarial example against the substitute model in a white-box manner and expects this adversarial example to transfer to the victim model. In the score-based attack [7], [16], the adversary can obtain the confidence vectors of the victim model and thus performs the attack by observing the output changes on different perturbations. In this work, we consider the decision-based attack [17]–[24], which is the most practical and challenging case because the adversary obtains only the predicted label of the victim model on a query sample. In the real-world scenario, the number of model queries performed by the attacker is usually limited either for cost reasons or for evading detection. For example, a commercial machine learning model is usually hosted as a service for a charge, or protection mechanisms are deployed to detect malicious queries. Query efficiency is an important metric for evaluating a decision-based adversarial attack.

Existing studies on decision-based attacks generally follow three different strategies: gradient estimation, direction optimization, and geometry acceleration. Gradient estimation-based methods start with an adversarial candidate sample and try to estimate the gradient of this sample as the search direction of a new candidate in each iteration until convergence or failure. For example, HSJA [18], AHA [19], and RamBoAttack [25] are proposed to estimate the gradient. Direction optimization-based methods, on the other hand, start with the benign sample and optimize the direction of searching the adversarial example (e.g., Opt [20] and Sign-Opt [21] attacks). Lastly, some studies utilize geometrical properties of decision boundaries to accelerate the searching process of the adversarial example, such as RayS [22], SurFree [23], Tangent [26] and Triangle [24] attacks.

Unfortunately, existing attacks are limited in either query efficiency or attack success rate. In particular, current gradient estimation-based methods do not efficiently estimate the iteration direction such that they require a variety of iterations to generate a satisfying adversarial example, which wastes a mass of queries. The direction optimization-based methods are shown to incur significant adversarial perturbations within the same number of queries compared with

† *Equal contribution*
* *Corresponding author*

DOI 10.1109/SP54263.2024.00068

gradient estimation-based methods, especially under the L_∞ constraint [18], [22]. The geometry acceleration-based methods can reduce the size of adversarial perturbation quickly but are only suitable for certain attack scenarios.

In this paper, we propose *BounceAttack*[1] as a query-efficient adversarial attack based on HSJA [18], which follows the gradient-estimation strategy. The major change of BounceAttack is that we use the orthogonal component of the estimated gradient of HSJA as the direction, to guide the iterations to explore the adversarial space the most within the same step size. As a result, BounceAttack can significantly reduce the number of queries. Specifically, the goal of each iteration in the attack process is to produce a new adversarial candidate sample with smaller perturbations than the old adversarial candidate sample until convergence. The search direction of the new sample is a key factor that affects the query efficiency. HSJA directly uses the estimated gradient as such search direction in each iteration. However, we find that when the estimated gradient is decomposed into an orthogonal component and a parallel component in terms of the hyper-line connecting the benign sample and the adversarial candidate sample, only the orthogonal component makes contributions to reducing the size of perturbations (as proven in Section 4.2). Therefore, BounceAttack uses such an orthogonal component as the search direction instead of the original estimated gradient, which results in a more efficient search than HSJA. Such an orthogonal component is referred to as *bounce direction* in this paper. Intuitively, the bounce direction leads an iteration to generate a new adversarial candidate sample with a large reduction of the perturbation size, as if it was "bounced" off the aforementioned hyper-line to explore the adversarial space with high query efficiency.

In addition to the bounce direction, we design momentum and smooth search to boost the performance of BounceAttack. Specifically, the momentum is designed to regularize the bounce direction with historical bounce directions in the past iterations in order to stabilize the *bounce search*. When the adversarial candidate sample is close to the optimal solution with the increase of the number of queries, BounceAttack stops the bounce search and switches to HSJA-style search (referred to as *smooth search* in this paper) by using the original estimated gradient. This is because when the search is about to converge, the bounce direction might result in strong updates of adversarial candidates which causes fluctuation in the search process. BounceAttack switches to smooth search for gentle updates until a satisfying adversarial sample is found.

We perform extensive experiments to evaluate BounceAttack on various datasets and models, and compare BounceAttack with existing attack methods [18], [19], [21]–[25]. Experimental results show that BounceAttack can achieve high query efficiency and attack success rate for both targeted and untargeted attack settings under both L_2 and L_∞ constraints. For example, BounceAttack consumes 37% to 76% fewer model queries to generate successful adversarial examples than existing methods, and given the same query limit, BounceAttack achieves 1-7 times higher attack success rate.

1. Code available at https://github.com/RaymondDawn/BounceAttack

Contributions. We summarize our contributions in the following.

- We propose BounceAttack as a query-efficient decision-based adversarial attack. We design novel bounce direction to improve the query efficiency while maintaining a high attack success rate.
- We design BounceAttack as a widely applicable attack in the sense that BounceAttack supports both targeted and untargeted adversarial attacks under either L_2 or L_∞ constraints.
- We perform extensive experiments on various benchmark datasets and models to evaluate BounceAttack and compare it with existing attack methods. BounceAttack is shown to outperform existing attack methods in both query efficiency and attack success rate.

Roadmap. In Section 2, we introduce the related work and the conception of adversarial attacks under different constraints and in different settings. In Section 3, we state the problem in our paper. In Section 4, we introduce BounceAttack. In Section 5, we describe our experimental results with evaluation metrics. In Section 6, we conclude our work.

2. Background

2.1. Adversarial Attack

In this paper, we focus on the supervised classification tasks, in which Deep Neural Networks are used to identify which sample space a certain image belongs to. The existing study [3] has shown that there is a hyper-plane called decision boundary that separates the sample spaces of a model. For any hyper-plane, the gradient direction is perpendicular to every vector inside the space. To perform an adversarial attack, an adversary seeks to add a minimum perturbation onto the benign sample to make it cross the decision boundary so as to be predicted as another class.

Generally, according to the goal of adversaries, adversarial attacks can be divided into targeted and untargeted ones. The sample space that the adversary tends to move the benign sample to is called the *adversarial space*. To measure the distance of the movement, researchers use the concepts of L_2 and L_∞ constraints.

2.2. White-Box Attack

The adversarial attack is first studied in the white-box setting by Szegedy et al. [1], where the adversary has full information about the training data, model structure, and model weights. Goodfellow et al. design FGSM [2] and I-FGSM [5] to generate adversarial samples by maximizing

the classification loss. To measure the robustness of classification models, PGD [27] is proposed as the "first-order" adversarial attack to iteratively search the gradient directions. To break the distillation defense [28], C&W [4] is raised to use a differentiable function as the restriction of optimization framework and achieves a high attack success rate. Besides L_2 and L_∞ constraints, L_0 constraint is considered in JSMA [29], which uses the L_0 norm as a mathematical representation of the perceptual distance and constructs a saliency map by using the forward derivative output from the last layer of the deep neural network, reducing the workload of adversarial sample attacks. Gradient-descent evasion attack [6] is inspired by Golland's discriminating direction technique and can evade both linear and non-linear classifiers. DeepFool [3] can iteratively search for the minimum perturbation along the normal vector of the decision boundary to produce an adversarial sample, which can be applied in both linear and non-linear classifiers.

2.3. Black-Box Attack

Different from the white-box attack, the black-box attack is more practical for real-world applications. In the black-box setting, the adversary is not aware of the structures of the models, the datasets used in training or validation, or whether defense mechanisms are deployed or not. The adversary is assumed to know the model input and the corresponding model output, such as the confidence vectors or only the predicted labels.

2.3.1. Transfer-Based Attack. The transfer-based attack is proposed by Papernot et al. [30], which transforms the black-box problem into the white-box problem. To conduct black-box attacks on the victim models, it generates adversarial samples on the substitute models trained on the same or similar datasets. The fundamental idea of these attack methods is to improve the transferability of the adversarial samples generated on the substitute models even with different model structures [31]. The transferability is largely related to the similarity of subspaces between the victim model and the substitute model [32]. Adversarial samples generated on the ensemble models can obtain higher transferability even in the targeted attack setting [33]. Some components like momentum [12] are introduced under the framework of FGSM to enhance the transferability of substitute models. Processing the input with the resize transformation method [15] can also improve the attack transferability.

2.3.2. Score-Based Attack. Score-based attacks assume that the model returns the full or partial confidence vectors (e.g., top-1 or top-5) to users. The adversary modifies the input sample according to the change of the confidence vector from the last iteration. ZOO [7] uses zero-order optimization to estimate the gradient, changing the black-box attack to the white-box attack and greatly improving the query efficiency. Ilyas et al. [10] introduce gradient priors into the black-box attack problem and give an optimization-based algorithm that can integrate any prior information. Ilyas et al. [9] use NES to maximize the expected value of the loss function along the search distribution. GenAttack [16] is a gradient-free optimization attack method using genetic algorithms to craft adversarial samples.

2.3.3. Decision-Based Attack. The decision-based attack is also called the hard label-only attack because the adversary cannot acquire any information except the predicted label of a certain input. It is first studied by Brendel et al. [17], who propose the Boundary Attack to walk along the decision boundary without a convergence guarantee. We classify the existing decision-based attacks into three kinds: gradient estimation attacks, direction optimization attacks, and geometry acceleration attacks.

The gradient estimation attacks find the gradient at the decision boundary and use it to guide the next search direction. HSJA [18] is proposed to find the adversarial sample with the minimum distance to the benign sample where the adversary moves the candidate sample along the gradient direction and projects it back to the decision boundary in each iteration. AHA [19] gathers historical information from previous queries to refine the mean of the random distribution and the step coefficient. After that, AHA makes the next estimation based on them with the converging direction and the random unit. RamBoAttack [25] proves that the final distance between the generated adversarial sample and the benign sample has a significant connection with the chosen initial sample and the target label. RamBoAttack proposes Block Descent in the framework of Rambo to avoid sticking to the local minimum but sacrifices the query efficiency. The gradients generated by these methods do not guide the optimal direction given the same step size in one iteration. BounceAttack, on the contrary, results in a maximized distance reduction as proved in Section 4.2.

The direction optimization-based attacks first adjust the search direction from the benign sample along a sampled unit vector with the step size which is calculated by the change of perturbation size from the last iteration and then normalize the search direction. Opt [20] is proposed to evaluate the search direction with a single unit. Sign-Opt [21] improves Opt [20] by the $\mathbf{sign}(\cdot)$ operation to identify the gradient direction estimated by a set of unit vectors more accurately. They are difficult to acquire a small-size perturbation compared with other attack methods in [26], [24], and [22]. Therefore, BounceAttack does not adopt this attack strategy in our paper.

The geometry acceleration attacks are proposed for faster-converging speed with small adversarial perturbation size. RayS [22] divides the benign image into small batches and greedily judges the flip of each batch which could craft an adversarial sample in each iteration. SurFree [23] pays attention to the geometrical properties of the decision boundary and iteratively draws circles to walk along different directions. Tangent [26] regards the gradient space of a certain boundary point as a d-dimensional spherical space and draws a tangent line to maximize the step size in the search direction. Triangle [24] updates the next sample in the attack process based on the properties of the triangle in each

TABLE 1. NOTATIONS AND EXPLANATION

Notation	Explanation
$F(\cdot)$	victim model
$\mathbf{X}^*$	benign sample
$\mathbf{X}_a$	adversarial sample
$\mathbf{X}_t$	the adversarial candidate sample at the decision boundary
$\hat{\mathbf{X}}_t$	the intermediate sample in the adversarial space
$\mathbf{sign}(\cdot)$	the function obtaining the sign of the input
$\mathbf{X}^*\mathbf{X_t}$	the hyper-line connecting $\mathbf{X}^*$ and $\mathbf{X}_t$
L_p	norm constraint where $p = 2$ or $p = \infty$
ϵ	perturbation threshold that determines the attack success
t	the number of queries
$\Phi_{\mathbf{X}^*}(\mathbf{X})$	the indication function of whether $\mathbf{X}$ is adversarial
$\mathbf{D}(\mathbf{X}_1, \mathbf{X}_2)$	the distance between $\mathbf{X}_1$ and $\mathbf{X}_2$
λ_t	the step size in the t-th iteration
$\mathbf{clip}(\cdot)$	the function clipping image pixel values into valid range

iteration. These methods can effectively search for the adversarial sample in the beginning stage but are only suitable for a certain attack scenario (i.e., only in a special attack setting or under a special attack constraint). BounceAttack, on the contrary, supports both targeted and untargeted attacks and both L_2 and L_∞ perturbation constraints.

3. Problem Statement

In this paper, we consider the black-box scenario where the adversary has access to only the predicted label of the victim model on a benign sample and aims at crafting an adversarial example for it. Specifically, given a victim model F and a benign sample $\mathbf{X}^*$ which is correctly classified by F as the label $y^* = F(\mathbf{X}^*)$, the adversary seeks an adversarial sample $\mathbf{X}_a$ with negligible perturbation such that $F(\mathbf{X}_a)$ produces the label as what the adversary expects. Formally, $\mathbf{X}_a$ satisfies the following target.

$$\min \mathbf{D}(\mathbf{X}^*, \mathbf{X}_a), \mathbf{s.t.} \Phi_{\mathbf{X}^*}(\mathbf{X}_a) = 1 \tag{1}$$

where $\mathbf{D}(\mathbf{X}^*, \mathbf{X}_a)$ measures the size of adversarial perturbation added to $\mathbf{X}^*$. We consider L_2 and L_∞ constraints in this paper. $\Phi_{\mathbf{X}^*}(\mathbf{X}_a)$ is a function indicating whether $\mathbf{X}_a$ is an adversarial sample in terms of $\mathbf{X}^*$. Suppose O_A denotes the adversarial space and O_B denotes the benign space of $\mathbf{X}^*$. In the targeted attack, O_A represents the space of an adversary-chosen class, while in the untargeted attack, O_A represents the space of any class other than the class of O_B. We formally define function Φ in the following.

$$\Phi_{\mathbf{X}^*}(\mathbf{X}) = \begin{cases} 1, & \mathbf{X} \in O_A \\ -1, & \mathbf{X} \in O_B \end{cases} \tag{2}$$

Generally speaking, the adversarial sample $\mathbf{X}_a$ achieves the minimum perturbation usually when $\mathbf{X}_a$ is located in the vicinity of the decision boundary Π. We list the notations used in this paper in Table 1.

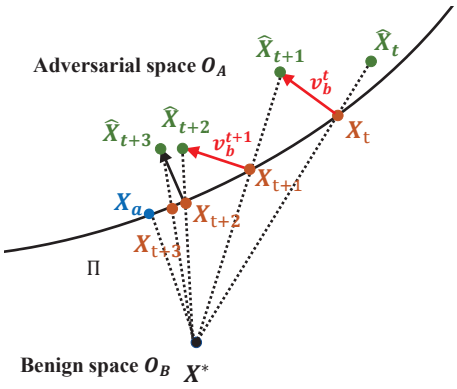


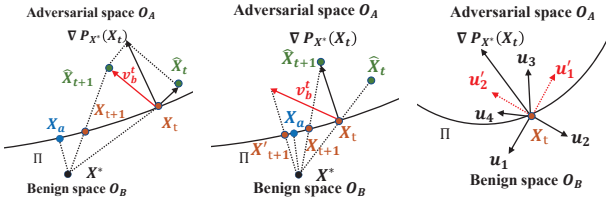
Figure 1. Overview of BounceAttack. Starting with the intermediate sample $\hat{\mathbf{X}}_t$, BounceAttack first finds the adversarial candidate $\mathbf{X}_t$ by binary search and then generates the next intermediate sample $\hat{\mathbf{X}}_{t+1}$ along the bounce direction $\mathbf{v}_b^t$. BounceAttack keeps the iterative bounce search process until the change of distance is small (i.e., $\mathbf{X}_{t+2}$), where BounceAttack switches to smooth search to get close to $\mathbf{X}_a$.

4. Approach

4.1. BounceAttack Overview

Following the same gradient-estimation strategy as adopted in previous methods [18], BounceAttack starts with an adversary-chosen sample of a random class in the untargeted setting and a certain class in the targeted setting and optimizes the perturbation size in an iterative manner. In particular, an iteration $t + 1$ produces an adversarial candidate sample $\mathbf{X}_{t+1}$ which is supposed to have a smaller perturbation size than $\mathbf{X}_t$ generated in the last iteration t. With the number of iterations increasing, the perturbation size converges to a small constant value and the attack ends.

We present the overview of BounceAttack in Fig. 1. Given a benign sample $\mathbf{X}^*$, BounceAttack, compared to HSJA, improves the query efficiency by maximizing the distance between $\mathbf{X}_{t+1}$ and $\mathbf{X}_t$ in each iteration such that $\mathbf{X}_{t+1}$ can quickly converge to a satisfying adversarial sample. To this end, we propose *bounce direction* as the replacement of the original estimated gradient in HSJA to guide the search of the new $\mathbf{X}_{t+1}$ in one iteration. The search process is referred to as *bounce search* in this paper. Specifically, an iteration t of BounceAttack starts with an intermediate sample $\hat{\mathbf{X}}_t$ in the adversarial space which is generated in the last iteration. The adversarial candidate sample $\mathbf{X}_t$ is positioned as the intersection point of the hyper-line $\mathbf{X}^*\hat{\mathbf{X}}_t$ and the decision boundary Π by binary search because it has the least size of perturbation while being adversarial among all the data points on the hyper-line $\mathbf{X}^*\hat{\mathbf{X}}_t$. The candidate sample $\mathbf{X}_{t+1}$ is generated in the next iteration $t+1$ from a new intermediate sample $\hat{\mathbf{X}}_{t+1}$. In HSJA, $\hat{\mathbf{X}}_{t+1}$ is generated by moving along the estimated gradient of $\mathbf{X}_t$. BounceAttack, on the contrary, uses the orthogonal component (i.e., bounce direction denoted as $\mathbf{v}_b^t$) by projecting such gradient onto the hyper-line orthogonal to $\mathbf{X}^*\hat{\mathbf{X}}_t$ to generate $\hat{\mathbf{X}}_{t+1}$, because it can drive $\mathbf{X}_{t+1}$ farthest away from $\mathbf{X}_t$



(a) Bounce Search (b) Smooth Search (c) Gradient Estimation

Figure 2. (a) illustrates the bounce search process. BounceAttack decomposes the gradient and uses the bounce direction $\mathbf{v}_b^t$ to produce the next intermediate sample $\hat{\mathbf{X}}_{t+1}$. (b) illustrates the smooth search process. BounceAttack uses the estimated gradient instead of the bounce direction to generate the next intermediate sample $\hat{\mathbf{X}}_{t+1}$. (c) illustrates the gradient estimation process. The gradient is estimated by the average of several sampled unit vectors $\mathbf{u}_i$ weighted by the output of the indication function Φ.

as proven in Section 4.2, which can significantly speed up the convergence of the adversarial candidate to a satisfying adversarial sample.

In general, BounceAttack consists of three steps to generate an adversarial sample: attack initialization, bounce search, and smooth search. In the attack initialization, we randomly sample a data point $\hat{\mathbf{X}}_1$ in the adversarial space O_A as the initial intermediate sample in the 1st iteration. In the targeted attack, $\hat{\mathbf{X}}_1$ is randomly drawn from the data distribution of the target class, while in the untargeted attack, $\hat{\mathbf{X}}_1$ is randomly drawn from the data distribution of a randomly-chosen class. BounceAttack first launches the above-mentioned bounce search to quickly reduce the perturbation size with the least number of iterations. When the bounce search is about to converge (i.e., the perturbation size of $\mathbf{X}_{t+1}$ is not significantly reduced in terms of $\mathbf{X}_t$), BounceAttack switches to the *smooth search* which uses the original estimated gradient as HSJA does instead of the bounce direction. This is because the bounce direction, compared to the original gradient, might lead to a large distance between $\mathbf{X}_{t+1}$ and $\mathbf{X}_t$ which causes $\mathbf{X}_{t+1}$ to fluctuate around the optima. When the perturbation size of the generated candidate sample is smaller than the threshold ϵ or the number of model queries exceeds the limits, BounceAttack stops.

4.2. Bounce Search

The bounce search uses the bounce direction in each iteration to generate a new adversarial candidate sample $\mathbf{X}_{t+1}$ farthest away from $\mathbf{X}_t$ along the decision boundary. We illustrate one iteration of bounce search in Fig. 2a.

The iteration t starts with an intermediate sample $\hat{\mathbf{X}}_t$ ($\hat{\mathbf{X}}_1$ is initialized by the adversary). The candidate sample $\mathbf{X}_t$ is selected as the intersection point of the hyper-line $\mathbf{X}^*\hat{\mathbf{X}}_t$ and the decision boundary Π because it has the least size of perturbation among points on $\mathbf{X}^*\hat{\mathbf{X}}_t$. To this end, we use binary search to find $\mathbf{X}_t$. The stop condition of the binary search is when the distance between the low point and the high point is no greater than a threshold $\theta = \frac{\gamma}{d}$. We set d

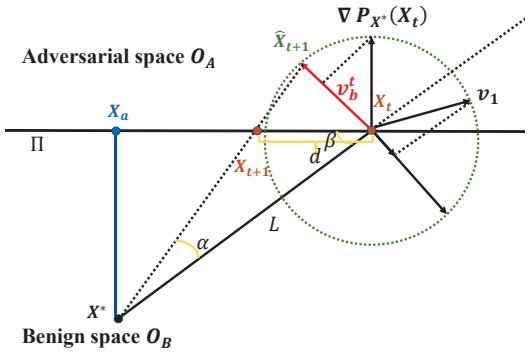


Figure 3. Proof of bounce search. Given an adversarial candidate sample $\mathbf{X}_t$, BounceAttack decomposes the estimated gradient of $\mathbf{X}_t$ and normalizes it. The distance of movement d calculated as $\mathbf{D}(\mathbf{X}_t, \mathbf{X}_{t+1})$ can be maximized when $\mathbf{X}^*\hat{\mathbf{X}}_{t+1}$ is the tangent line of the circle with radius λ_t centred at $\mathbf{X}_t$, where $\hat{\mathbf{X}}_{t+1}$ can be approximated by following the bounce vector $\mathbf{v}_b^t$.

as the product of the shape dimensions of the benign sample (i.e., channel×width×height), and γ is set to avoid sticking into the dead loop due to floating point precision when BounceAttack conducts the binary search. Let $P_{\mathbf{X}^*}(\mathbf{X}_t)$ denote the objective function measuring how adversarial $\mathbf{X}_t$ is in terms of $\mathbf{X}^*$. We formally define $P_{\mathbf{X}^*}(\mathbf{X})$ in the following.

$$P_{\mathbf{X}^*}(\mathbf{X}) = \begin{cases} \max\limits_{i \neq y^*} F_i(\mathbf{X}) - F_{y^*}(\mathbf{X}), & \text{untargeted} \\ F_{y^+}(\mathbf{X}) - \max\limits_{i \neq y^+} F_i(\mathbf{X}), & \text{targeted} \end{cases} \tag{3}$$

where y^* is the label of the benign sample $\mathbf{X}^*$ and y^+ is the adversary-chosen label in the targeted attack. The indication function $\Phi_{\mathbf{X}^*}(\mathbf{X})$ in Eq. (2) is then updated as:

$$\Phi_{\mathbf{X}^*}(\mathbf{X}) = \mathbf{sign}(P_{\mathbf{X}^*}(\mathbf{X})) \tag{4}$$

The bounce direction is the orthogonal component (i.e., bounce vector $\mathbf{v}_b^t$) when the gradient $\nabla P_{\mathbf{X}^*}(\mathbf{X}_t)$ of $P_{\mathbf{X}^*}(\mathbf{X}_t)$ with respect to $\mathbf{X}_t$ is decomposed by being projected onto the hyper-line $\mathbf{X}^*\mathbf{X}_t$.

$$\mathbf{v}_b^t = \nabla P_{\mathbf{X}^*}(\mathbf{X}_t) - \frac{\nabla P_{\mathbf{X}^*}(\mathbf{X}_t)(\mathbf{X}_t - \mathbf{X}^*)}{||\mathbf{X}_t - \mathbf{X}^*||_p} \tag{5}$$

where $p = 2$ or ∞ in this paper.

The next intermediate sample $\hat{\mathbf{X}}_{t+1}$ is generated by moving from $\mathbf{X}_t$ along the direction $\mathbf{v}_b^t$ with a step size λ_t as described in the following.

$$\hat{\mathbf{X}}_{t+1} = \mathbf{clip}\left(\mathbf{X}_t + \lambda_t \mathbf{v}_b^t\right) \tag{6}$$

where λ_t is gradually decreased as the iteration t grows. We set λ_t in the following.

$$\lambda_t = \frac{\mathbf{D}\left(\mathbf{X}_t, \mathbf{X}^*\right)}{\sqrt{t}} \tag{7}$$

Note that if $\Phi_{\mathbf{X}^*}(\hat{\mathbf{X}}_{t+1}) \neq 1$, we further recursively decrease λ_t by half $\lambda_t = \lambda_t/2$ until $\Phi_{\mathbf{X}^*}(\hat{\mathbf{X}}_{t+1}) = 1$.

Algorithm 1 BounceAttack Process

Input: F: target classifier; $\mathbf{X}^*$: benign sample; B_0: initial number of sampled unit vectors; T: number of iterations; μ: decay factor in the momentum
Output: potential adversarial sample

Initialization: randomly sample $\hat{\mathbf{X}}_1$, $\mathbf{v}_b^0 = \mathbf{0}$, counter $c = 0$, $\mu = 0.5$
for $t = 1; t \leq T; t++$ **do**
  $\mathbf{X}_t = \text{binarySearch}(\hat{\mathbf{X}}_t, \mathbf{X}^*)$
  if $\frac{\mathbf{D}(\mathbf{X}^*, \mathbf{X}_t)}{\mathbf{D}(\mathbf{X}^*, \mathbf{X}_{t-1})} > 0.99$ and $t > 1$ **then**
    c++
  else
    $c = 0$
  end if
  $B_t = B_0 * \sqrt{t}$
  $\nabla P_{\mathbf{X}^*}(\mathbf{X}_t) = \text{gradientEstimation}\,(\mathbf{X}_t, \Phi_{\mathbf{X}^*}(\mathbf{X}_t), B_t)$
  if $c < 3$ **then**
    $\mathbf{v}_b^t = \text{bounceDirection}\,(\nabla P_{\mathbf{X}^*}(\mathbf{X}_t), \mathbf{X}^*\mathbf{X_t})$
  else
    $\mathbf{v}_b^t = \nabla P_{\mathbf{X}^*}(\mathbf{X}_t)$
  end if
  Update $\mathbf{v}_b^t$ using Eq. (13)
  $\lambda_t = \frac{\mathbf{D}(\mathbf{X}_t, \mathbf{X}^*)}{\sqrt{t}}$
  while $\Phi_{\mathbf{X}^*}(\mathbf{X}_t + \lambda_t \mathbf{v}_b^t) \neq 1$ **do**
    $\lambda_t = \lambda_t / 2$
  end while
  $\hat{\mathbf{X}}_{t+1} = \mathbf{X}_t + \lambda_t \mathbf{v}_b^t$
end for
return $\text{binarySearch}(\hat{\mathbf{X}}_{t+1}, \mathbf{X}^*)$

Theorem 1. *Given a fixed step size λ_t, the bounce direction $\mathbf{v}_b^t$ can result in an intermediate sample $\hat{\mathbf{X}}_{t+1}$ that produces an adversarial candidate sample $\mathbf{X}_{t+1}$ with the maximal distance $\mathbf{D}(\mathbf{X}_t, \mathbf{X}_{t+1})$.*

Proof: The decision boundary Π is a continuously connected hyper-space [34] and we can approximate it as a linear hyper-plane in the vicinity of $\mathbf{X}_t$ as shown in Fig. 3. Suppose d denotes the distance $\mathbf{D}(\mathbf{X}_t, \mathbf{X}_{t+1})$ and L denotes $\mathbf{D}(\mathbf{X}^*, \mathbf{X}_t)$. It is clear that $\hat{\mathbf{X}}_{t+1}$ locates on the circle centered at $\mathbf{X}_t$ with radius λ_t. Let α represent the angle between $\mathbf{X}^*\mathbf{X}_{t+1}$ and $\mathbf{X}^*\mathbf{X_t}$ and β denote the angle between $\mathbf{X}^*\mathbf{X}_t$ and the decision boundary Π. We can obtain d based on the property of triangles in the following.

$$\begin{aligned} \frac{d}{\sin\alpha} &= \frac{L}{\sin(\pi - (\alpha + \beta))} \\ d &= \frac{L}{\cos\beta + \frac{\sin\beta}{\tan\alpha}} \end{aligned} \tag{8}$$

where β is a constant, so the distance d is proportional to the angle α. Obviously, α is maximum when $\mathbf{X}^*\hat{\mathbf{X}}_{t+1}$ is tangential to the circle centred at $\mathbf{X}_t$ with radius λ_t. In this case, we can calculate α in the following.

$$\alpha = \arctan(\frac{\lambda_t}{\sqrt{L^2 - \lambda_t^2}}) = \arctan(\frac{1}{\sqrt{\frac{L^2}{\lambda_t^2} - 1}}) \tag{9}$$

Assuming $\frac{L}{\lambda_t}$ is sufficiently large, we have $\sqrt{\frac{L^2}{\lambda_t^2} - 1} \approx \sqrt{\frac{L^2}{\lambda_t^2}} = \frac{L}{\lambda_t}$, and α can thus be approximated as follows.

$$\alpha \approx \arctan(\frac{\lambda_t}{L}) \tag{10}$$

This indicates that d is approximately maximal when $\mathbf{X}_t\hat{\mathbf{X}}_{t+1}$ is orthogonal to $\mathbf{X}^*\mathbf{X}_t$, which is exactly the direction of the bounce vector $\mathbf{v}_b^t$. □

We estimate the gradient $\nabla P_{\mathbf{X}^*}(\mathbf{X}_t)$ by following the same strategy of HSJA [18]. Specifically, we use the Monte Carlo method to sample several unit vectors from the Gaussian distribution $U = \{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_{B_t}\}$ as shown in Fig. 2c where each $\mathbf{u}_i$ has the same dimension as $\mathbf{X}^*$ and B_t denotes the number of sampled unit vectors $\mathbf{u}_i$. In order to determine if $\mathbf{u}_i$ points to the adversarial space O_A, We generate a series of new samples $\mathbf{X}_t + \delta\mathbf{u}_i$ by adding perturbation of size δ to $\mathbf{X}_t$ along the direction of each $\mathbf{u}_i$. Hence, we can use the indication function $\Phi_{\mathbf{X}^*}(\cdot)$ to test if $\mathbf{X}_t + \delta\mathbf{u}_i$ is in O_A or not. As a result, the gradient can be estimated by the mean of $\mathbf{u}_i$ weighted by its sign. Formally, the gradient $\nabla P_{\mathbf{X}^*}(\mathbf{X})$ is estimated in the following where $\overline{\Phi}_{\mathbf{X}^*}$ is the mean of $\Phi_{\mathbf{X}^*}(\cdot)$ on the B_t samples.

$$\overline{\Phi}_{\mathbf{X}^*} = \frac{1}{B_t}\sum_{i=1}^{B_t} \Phi_{\mathbf{X}^*}\left(\mathbf{X}_t + \delta\mathbf{u}_i\right) \tag{11}$$

$$\nabla P_{\mathbf{X}^*}(\mathbf{X}_t) = \frac{1}{B_t - 1}\sum_{i=1}^{B_t} \left(\Phi_{\mathbf{X}^*}\left(\mathbf{X}_t + \delta\mathbf{u}_i\right) - \overline{\Phi}_{\mathbf{X}^*}\right)\mathbf{u}_i \tag{12}$$

where $B_t = B_0\sqrt{t}$ and the initial B_0 is set to 100 in our paper.

Such estimation of the gradient assures that the gradient always points to the adversarial space O_A and is orthogonal to the decision boundary. As a result, the bounce vector which is the orthogonal component of the gradient also points to the adversarial space O_A. This explains why we use the orthogonal component of the gradient rather than that of the randomly sampled vectors as the bounce vector. For example, in Fig. 3, even if $\mathbf{v}_1$ points to O_A, its orthogonal component points to the benign space O_B which cannot be used as a bounce direction.

4.3. Momentum

In order to stabilize the bounce direction and avoid local optima, we introduce the momentum in BounceAttack by regularizing the bounce direction $\mathbf{v}_b^t$ with historical bounce directions in the past iterations. The μ is essential for the storage of past gradients, and a proper μ can make the attack process more efficient. The bounce vectors produced in each sample point might be different due to the randomness of the

unit vectors. The momentum can smooth the search process. When μ for momentum is small, the search direction will be greatly affected by the newly generated bounce direction. When μ for momentum is large, the search direction will be greatly driven by the historical information of past iterations. Specifically, the bounce direction is updated and then normalized in the following.

$$\mathbf{v}_b^t = \mathbf{v}_b^t + \mu \mathbf{v}_b^{t-1}$$
$$\mathbf{v}_b^t = \begin{cases} \frac{\mathbf{v}_b^t}{||\mathbf{v}_b^t||_2} & , p = 2 \\ \mathbf{sign}\left(\mathbf{v}_b^t\right) & , p = \infty \end{cases} \tag{13}$$

where the decay factor μ is a hyper-parameter which we set to 0.5 in our paper. The initial $\mathbf{v}_b^0$ is set to $\mathbf{0}$.

4.4. Smooth Search

Although the bounce direction can lead to a fast convergence of the adversarial candidate sample, it might cause the adversarial candidate sample to fluctuate back and forth across the optimal adversarial sample $\mathbf{X}_a$ in the end as shown in Fig. 2b. This is because when $\mathbf{X}_t$ is near the $\mathbf{X}_a$, the angle between $\mathbf{X}^*\mathbf{X_t}$ and the gradient $\nabla P_{\mathbf{X}^*}(\mathbf{X}_t)$ becomes small. As a result, the bounce direction which is orthogonal to $\mathbf{X}^*\mathbf{X_t}$ might lead even a small movement to cause a large change of $\mathbf{X}_{t+1}$ to cross the optima $\mathbf{X}_a$ (e.g., $\mathbf{X}_{t+1}^{'}$ in Fig. 2b). Therefore, BounceAttack switches to smooth search to mitigate such a situation. Specifically, it uses the estimated gradient $\nabla P_{\mathbf{X}^*}(\mathbf{X}_t)$ instead of the bounce vector $\mathbf{v}_b^t$ in the subsequent iterations. The condition of such a switch is when the perturbation reduction becomes insignificant (i.e., $\frac{\mathbf{D}(\mathbf{X}^*,\mathbf{X}_t)}{\mathbf{D}(\mathbf{X}^*,\mathbf{X}_{t-1})} > 0.99$, where $t > 1$) continuously for three iterations.

4.5. BounceAttack Process

The attack process of BounceAttack is detailed in Algorithm 1. We first initialize the attack (Line 1). In each iteration, BounceAttack finds the adversarial candidate $\mathbf{X}_t$ on the decision boundary using binary search (Line 3) and then estimates the gradient at $\mathbf{X}_t$ (Line 10). The bounce direction is calculated from the gradient (Line 12) and updated using the momentum mechanism (Line 16). The bounce step size is dynamically adjusted according to the iteration number t (Line 17-20). A new intermediate sample $\hat{\mathbf{X}}_{t+1}$ is generated (Line 21). When the iteration is about to converge (Line 4), BounceAttack switches to the smooth search and uses the gradient instead of the bounce direction to generate the new intermediate sample (Line 14). At last, the adversarial sample can be generated (Line 23).

5. Experiments

In this section, we conduct extensive experimental comparisons of BounceAttack with existing attacks. To fully understand how BounceAttack works, we change μ to see how the hyper-parameter affects BounceAttack and conduct ablation studies on each component of BounceAttack. We measure the performance of BounceAttack against three defense mechanisms.

5.1. Experimental Setup

5.1.1. Datasets. To evaluate BounceAttack more comprehensively, various standard datasets are adopted in the experiments, including CIFAR10 [35], CIFAR100 [35] and ImageNet2012 [36]. These datasets are commonly used to evaluate the classification accuracy and robustness of models [18], [19], [23], [24]. CIFAR10 and CIFAR100 are composed of tiny images with the size $3\times32\times32$. CIFAR10 has 10 classes with 6,000 images per class. In each class, 5,000 images are used for training and 1,000 images for testing. CIFAR100 has 100 classes with 600 images per class. In each class, 500 images are used for training and 100 images for testing. ImageNet is widely used in computer vision tasks [37]–[39] and has millions of images. We use ImageNet in our experiments to be consistent with previous work and process each image to $3\times224\times224$. ImageNet has 1,000 classes. In each class, around 1,300 images are used for training and 50 images for testing. To avoid sticking into the dead loop, we set the $\gamma = 1$ in CIFAR10 and CIFAR100 and $\gamma = 10$ in ImageNet to control the binary search threshold.

To ensure the fairness of the experiments, we select 200 correctly classified image-label pairs from the test dataset to evaluate the performance of all the attack methods. The first 100 pairs are used as the benign samples and the other 100 pairs are used as the initial target adversarial sample for the targeted attack.

5.1.2. Victim Models. We consider the Convolutional Neural Network (CNN) based models as our victim models, including ResNet50 [40] and DenseNet121 [41] implemented by mmlab [42], which first introduces deep learning into computer vision area and publishes dozens of papers each year in the top conferences. For ResNet50 on the three datasets, we apply the pretrained weights by mmlab. For DenseNet121 on CIFAR10 and CIFAR100, we train the model with the default configuration file by mmlab [42].

5.1.3. Metrics. We consider two metrics to evaluate the performance of BounceAttack.

We use **Median L Norm (MLN)** to measure the size of adversarial perturbation. We consider L_2 and L_∞ as the adversarial perturbation constraints. We refer to **MLN** as *distance* in the remaining part of our paper.

$$\mathbf{MLN} = \text{median}(\{\|\mathbf{X}^* - \mathbf{X}_i\|_p \,|0 \le i < N\}) \tag{14}$$

where N is set as 100 which means the 100 correctly classified image-label pairs, $\mathbf{X}_i$ is the i-th generated adversarial sample, and p is the constraint type (i.e., $p = 2$ or ∞).

The **Attack Success Rate (ASR)** is also a key factor in measuring the quality of adversarial attack methods. It measures the rate of the number of samples for which an

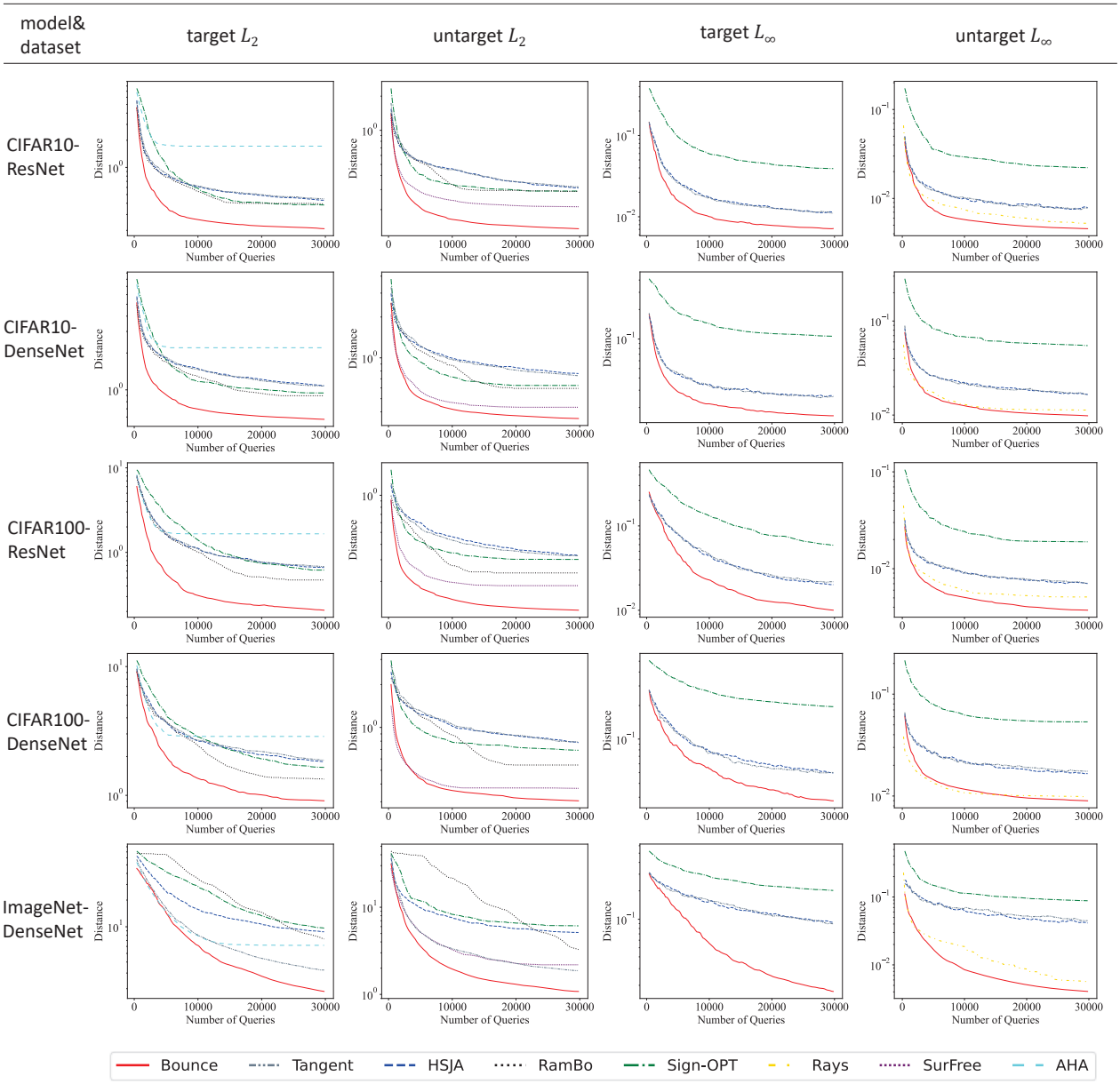


Figure 4. Median distance with log scale v.s. the number of model queries on CIFAR10 (with ResNet and DenseNet), CIFAR100 (with ResNet and DenseNet), and ImageNet (with DenseNet) from top to bottom rows. 2nd column: targeted L_2. 3rd column: untargeted L_2. 4th column: targeted L_∞. 5th column: untargeted L_∞.

attack method can successfully generate adversarial samples within a perturbation size ϵ over the N testing samples.

$$\textbf{ASR} = \frac{1}{N}\sum_{i=0}^{N-1} \Psi_{\mathbf{X}^*}(\mathbf{X}_i) \tag{15}$$

Where $\Psi_{\mathbf{X}^*}(\mathbf{X}_i) = 1$ if $\|\mathbf{X}^* - \mathbf{X}_i\|_p \leq \epsilon$, otherwise $\Psi_{\mathbf{X}^*}(\mathbf{X}_i) = 0$.

5.1.4. Baseline Attack Methods. We select HSJA [18], Sign-OPT [21], RamBo [25], Rays [22], Tangent [26], SurFree [23] and AHA [19] as the baseline attack methods. We use HSJA as the backbone of RamBo as their paper does. We use the same hyper-parameters as those in their original papers. We make comparisons with the baseline attack methods according to their supported attack settings as shown in Table 6.

5.1.5. Defense mechanisms. To evaluate the performance of BounceAttack against the defense mechanisms, we select three defense methods: Defensive Distillation [28], Adversarial Training [27] and TRADES [43] to protect the victim

TABLE 2. MEDIAN L_2 DISTANCE OF THE GENERATED ADVERSARIAL SAMPLES BY DIFFERENT ATTACKS USING DIFFERENT NUMBERS OF QUERIES AT 1,000, 5,000, AND 30,000 ON CIFAR10 (WITH RESNET AND DENSENET), CIFAR100 (WITH RESNET AND DENSENET) AND IMAGENET (WITH DENSENET). N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Data	Model	Attack	1K							5K							30K						
			Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent
CIFAR10	ResNet	untargeted	**0.610**	0.983	1.278	0.895	0.619	N.A.	1.001	**0.208**	0.537	0.407	0.520	0.286	N.A.	0.543	**0.136**	0.310	0.291	0.289	0.212	N.A.	0.318
		targeted	**1.670**	2.477	5.710	2.009	N.A.	4.989	2.523	**0.366**	0.830	1.058	0.782	N.A.	2.804	0.860	**0.209**	0.432	0.383	0.395	N.A.	2.779	0.449
	Densenet	untargeted	1.303	2.065	2.215	1.854	**1.202**	N.A.	2.275	**0.504**	1.204	1.128	0.932	1.127	N.A.	1.231	**0.357**	0.766	0.626	0.595	0.432	N.A.	0.740
		targeted	**2.663**	3.504	5.731	3.059	N.A.	4.908	3.661	**0.913**	1.804	1.748	1.694	N.A.	2.985	1.848	**0.575**	1.080	0.943	0.895	N.A.	2.981	1.076
CIFAR100	ResNet	untargeted	**0.456**	0.926	0.969	0.785	0.558	N.A.	1.041	**0.177**	0.599	0.418	0.450	0.243	N.A.	0.542	**0.117**	0.325	0.302	0.235	0.185	N.A.	0.323
		targeted	**3.495**	5.459	7.812	5.163	N.A.	5.729	5.344	**0.558**	1.632	2.799	1.631	N.A.	3.397	1.704	**0.207**	0.662	0.617	0.472	N.A.	3.239	0.680
	Densenet	untargeted	1.011	1.932	1.983	1.931	**0.866**	N.A.	2.259	**0.435**	1.232	0.961	1.199	0.452	N.A.	1.311	**0.305**	0.782	0.691	0.544	0.373	N.A.	0.791
		targeted	**6.050**	7.286	9.390	6.847	N.A.	7.140	7.160	**2.009**	3.699	4.226	3.743	N.A.	4.310	3.721	**0.908**	1.823	1.650	1.337	N.A.	4.018	1.879
ImageNet	Densenet	untargeted	17.797	20.296	34.128	42.097	**16.997**	N.A.	20.289	**3.579**	9.837	11.828	39.195	5.189	N.A.	5.168	**1.081**	5.125	6.114	3.273	2.186	N.A.	1.874
		targeted	**39.301**	55.079	65.605	67.885	N.A.	51.465	45.701	**14.398**	25.211	41.467	66.543	N.A.	17.577	16.309	**1.863**	8.780	9.632	7.281	N.A.	7.407	3.249

TABLE 3. MEDIAN L_∞ DISTANCE OF THE GENERATED ADVERSARIAL SAMPLES BY DIFFERENT ATTACKS USING DIFFERENT NUMBERS OF QUERIES AT 1,000 5,000, AND 30,000 ON CIFAR10 (WITH RESNET AND DENSENET), CIFAR100 (WITH RESNET AND DENSENET) AND IMAGENET (WITH DENSENET). N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Data	Model	Attack	1K					5K					30K				
			Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent
CIFAR10	ResNet	untargeted	0.024	0.027	0.116	**0.021**	0.027	**0.008**	0.013	0.036	0.010	0.012	**0.005**	0.008	0.022	0.005	0.008
		targeted	**0.083**	0.100	0.299	N.A.	0.090	**0.016**	0.027	0.096	N.A.	0.026	**0.007**	0.011	0.039	N.A.	0.011
	Densenet	untargeted	0.043	0.053	0.187	**0.030**	0.052	**0.016**	0.026	0.083	0.017	0.026	**0.010**	0.017	0.055	0.011	0.017
		targeted	**0.100**	0.115	0.371	N.A.	0.117	**0.029**	0.044	0.186	N.A.	0.045	**0.016**	0.026	0.107	N.A.	0.026
CIFAR100	ResNet	untargeted	**0.015**	0.019	0.078	0.015	0.020	**0.006**	0.011	0.033	0.008	0.011	**0.004**	0.007	0.019	0.005	0.007
		targeted	**0.177**	0.189	0.396	N.A.	0.180	**0.046**	0.078	0.209	N.A.	0.082	**0.010**	0.020	0.059	N.A.	0.022
	Densenet	untargeted	0.034	0.047	0.159	**0.023**	0.045	**0.014**	0.027	0.080	0.014	0.027	**0.009**	0.017	0.053	0.010	0.018
		targeted	**0.198**	0.221	0.472	N.A.	0.217	**0.084**	0.114	0.337	N.A.	0.112	**0.028**	0.049	0.196	N.A.	0.050
ImageNet	Densenet	untargeted	**0.059**	0.136	0.325	0.060	0.140	**0.017**	0.079	0.148	0.025	0.077	**0.004**	0.041	0.088	0.006	0.045
		targeted	**0.256**	0.277	0.484	N.A.	0.279	**0.124**	0.200	0.350	N.A.	0.195	**0.017**	0.092	0.202	N.A.	0.090

TABLE 4. ASR OF DIFFERENT ATTACKS CONSTRAINED BY DIFFERENT ϵ UNDER L_2 NORM USING DIFFERENT NUMBERS OF QUERIES AT 1,000 5,000, AND 30,000 ON CIFAR10 (WITH RESNET AND DENSENET), CIFAR100 (WITH RESNET AND DENSENET) AND IMAGENET (WITH DENSENET). N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Data	Model	Attack	ϵ	1K							5K							30K						
				Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent
CIFAR10	ResNet	untargeted	0.5	**0.40**	0.16	0.12	0.17	0.36	N.A.	0.11	**0.92**	0.45	0.60	0.46	0.80	N.A.	0.46	**0.99**	0.83	0.91	0.91	0.96	N.A.	0.86
		targeted	1.0	**0.30**	0.16	0.04	0.24	N.A.	0.01	0.16	**0.97**	0.74	0.48	0.74	N.A.	0.06	0.70	**1.00**	0.99	0.99	1.00	N.A.	0.06	1.00
	Densenet	untargeted	0.5	0.16	0.05	0.04	0.07	**0.17**	N.A.	0.03	**0.51**	0.07	0.17	0.16	0.46	N.A.	0.09	**0.71**	0.20	0.28	0.42	0.56	N.A.	0.22
		targeted	1.0	**0.16**	0.04	0.03	0.04	N.A.	0.03	0.03	**0.60**	0.13	0.17	0.17	N.A.	0.07	0.14	**0.88**	0.42	0.54	0.57	N.A.	0.07	0.42
CIFAR100	ResNet	untargeted	0.5	**0.56**	0.25	0.26	0.26	0.50	N.A.	0.19	**0.95**	0.43	0.65	0.54	0.86	N.A.	0.48	**0.99**	0.77	0.84	0.94	0.97	N.A.	0.78
		targeted	1.0	**0.09**	0.01	0.00	0.02	N.A.	0.00	0.00	**0.75**	0.21	0.12	0.22	N.A.	0.02	0.21	**0.98**	0.84	0.80	0.92	N.A.	0.02	0.84
	Densenet	untargeted	0.5	0.24	0.07	0.07	0.08	**0.28**	N.A.	0.08	**0.59**	0.10	0.16	0.17	0.55	N.A.	0.12	**0.74**	0.21	0.28	0.46	0.62	N.A.	0.23
		targeted	1.0	**0.03**	0.01	0.02	0.02	N.A.	0.01	0.02	**0.19**	0.04	0.04	0.03	N.A.	0.02	0.03	**0.56**	0.11	0.14	0.27	N.A.	0.02	0.08
ImageNet	Densenet	untargeted	5.0	0.13	0.10	0.05	0.03	**0.17**	N.A.	0.14	**0.60**	0.20	0.18	0.04	0.47	N.A.	0.48	**1.00**	0.50	0.35	0.68	0.92	N.A.	0.98
		targeted	15	**0.02**	0.00	0.01	0.00	N.A.	0.01	0.02	**0.53**	0.13	0.05	0.00	N.A.	0.31	0.43	**0.99**	0.91	0.79	0.94	N.A.	0.96	0.97

model.

Adversarial training is regarded as one of the most effective defense mechanisms against adversarial attacks [2], [27] and is widely used to evaluate the attacks (e.g., AHA, RamBo, RayS, and Tangent). Defensive distillation is conducted in HSJA so we add it to our experiments, where we set the distillation temperature $T = 2$. TRADES is proposed to balance the robustness and the classification accuracy in adversarial training. In our experiment, we set the regulation parameter $\beta = 6.0$ to produce a more robust model to see the performance of each attack method. We set the number of model queries up to 30,000 in the defense experiments and record the **ASR** under different ϵ values.

5.2. Effectiveness of BounceAttack

We examine the effectiveness of BounceAttack and other baseline attack methods under different constraints (i.e., L_2 and L_∞) and attack settings (i.e., targeted and untargeted). In each experiment, we restrict the maximum number of model queries to 30,000. To have a more intuitive presentation of the experimental results, we plot several figures to show how the median distance and the **ASR** change with the number of model queries and organize the experimental data into tables when the number of model queries K satisfies K=1,000, 5,000 and 30,000.

TABLE 5. ASR OF DIFFERENT ATTACKS CONSTRAINED BY DIFFERENT ϵ UNDER L_∞ NORM USING DIFFERENT NUMBERS OF QUERIES AT 1,000 5,000, AND 30,000 ON CIFAR10 (WITH RESNET AND DENSENET), CIFAR100 (WITH RESNET AND DENSENET) AND IMAGENET (WITH DENSENET). N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Data	Model	Attack	ϵ	1K					5K					30K				
				Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent
CIFAR10	ResNet	untargeted	0.01	0.19	0.13	0.01	**0.21**	0.11	**0.64**	0.38	0.03	0.54	0.36	**0.91**	0.66	0.05	0.89	0.66
		targeted	0.03	**0.12**	0.11	0.00	N.A.	0.12	**0.81**	0.59	0.10	N.A.	0.57	**1.00**	0.99	0.31	N.A.	0.97
	Densenet	untargeted	0.01	0.11	0.05	0.00	**0.13**	0.05	**0.27**	0.06	0.00	0.27	0.06	**0.53**	0.19	0.00	0.45	0.20
		targeted	0.03	**0.13**	0.05	0.02	N.A.	0.06	**0.52**	0.24	0.03	N.A.	0.27	**0.85**	0.59	0.04	N.A.	0.63
CIFAR100	ResNet	untargeted	0.01	0.32	0.21	0.07	**0.34**	0.22	**0.77**	0.45	0.09	0.66	0.45	**0.92**	0.73	0.13	0.89	0.70
		targeted	0.03	**0.02**	0.00	0.00	N.A.	0.01	**0.28**	0.12	0.00	N.A.	0.12	**0.88**	0.68	0.10	N.A.	0.67
	Densenet	untargeted	0.01	0.11	0.04	0.04	**0.21**	0.04	**0.36**	0.10	0.04	0.32	0.11	**0.57**	0.18	0.04	0.53	0.22
		targeted	0.03	**0.03**	0.03	0.01	N.A.	0.03	**0.14**	0.04	0.01	N.A.	0.04	**0.55**	0.14	0.02	N.A.	0.19
ImageNet	Densenet	untargeted	0.03	0.25	0.11	0.04	**0.30**	0.07	**0.71**	0.12	0.07	0.59	0.14	**1.00**	0.28	0.11	1.00	0.32
		targeted	0.20	**0.19**	0.16	0.01	N.A.	0.14	**0.80**	0.51	0.11	N.A.	0.56	**1.00**	0.99	0.49	N.A.	0.98

TABLE 6. SUPPORTED ATTACK SETTINGS OF DIFFERENT METHODS.

	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	RayS
L_2-tar	✓	✓	✓	✓		✓	✓	
L_2-untar	✓	✓	✓	✓	✓		✓	
L_∞-tar	✓	✓	✓				✓	
L_∞-untar	✓	✓	✓				✓	✓

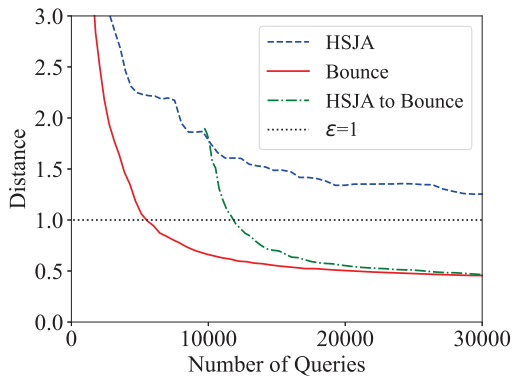


Figure 5. Targeted-L_2 distance v.s. the number of model queries on CIFAR10 (with ResNet) using BounceAttack and HSJA attacks. Although HSJA fails on this sample, the change to use BounceAttack can reduce the distance and generate a successful adversarial sample under $\epsilon = 1$.

5.2.1. Distance Against Queries. We plot the curves of median distance with log scale against the number of model queries in Fig. 4. The result of the median distance when the number of model queries equals 1,000, 5,000, and 30,000 is organized in Table 2 and Table 3.

BounceAttack is able to craft an adversarial sample with a limited number of model queries to obtain a small distance in the attack process as shown in Table 2 and Table 3. BounceAttack can reduce the distance quickly within 1,000 model queries at the beginning of the attack. Within 5,000 model queries, BounceAttack acquires the smallest distance, which reduces the distance by over 40% compared with other attack methods across all the experiments. Within 30,000 model queries, BounceAttack outperforms other methods with a distance reduction of 9%-92% across all experiments.

Notably, although some methods like SurFree in the untargeted-L_2 scenario and RayS in the untargeted-L_∞ scenario reduce the distance more quickly than BounceAttack in the early stage of the attack (i.e., within 1,000 model queries), BounceAttack outperforms them with around 2,000 model queries. RayS is a greedy attack method that searches quickly in the early stage of the attack but will acquire a local minimum value. SurFree searches for the next direction based on the circle it draws from the last iteration which gives no guarantee to point to the direction with the largest reduction of distance.

A more intuitive result is presented in Fig. 4. BounceAttack can keep reducing the distance when other attack methods converge to a large distance. For example, in the untargeted-L_2 scenario, all attack methods except SurFree and BounceAttack cannot reduce the distance anymore even if the distance is large, and this is more obvious in the small-size datasets like CIFAR10 and CIFAR100.

We present the visualized trajectories of BounceAttack in Fig. 10 in Appendix A. In the untargeted setting, BounceAttack can hide the adversarial perturbations into the benign image within 1,000 model queries in the visual perspective. In the targeted setting, BounceAttack needs around 5,000 model queries to hide the target image into the benign image. With the attack going on, the generated samples look more and more similar to the benign sample. After 1,000 and 5,000 model queries, the generated image is visually indistinguishable compared with the benign sample in the untargeted and targeted settings respectively.

5.2.2. Attack Success Rate. For attacks on different datasets, in different attack settings, and under different attack constraints, we assign a different value to ϵ to measure the success. In CIFAR10 and CIFAR100, we assign the same ϵ in each attack scenario. In ImageNet, we increase the ϵ due to the complexity of the dataset. The result of the **ASR** at 1,000, 5,000, and 30,000 model queries under different ϵ is organized in Table 4 and Table 5. In Fig. 6, we plot the curve of **ASR** against the number of model queries with a fixed ϵ.

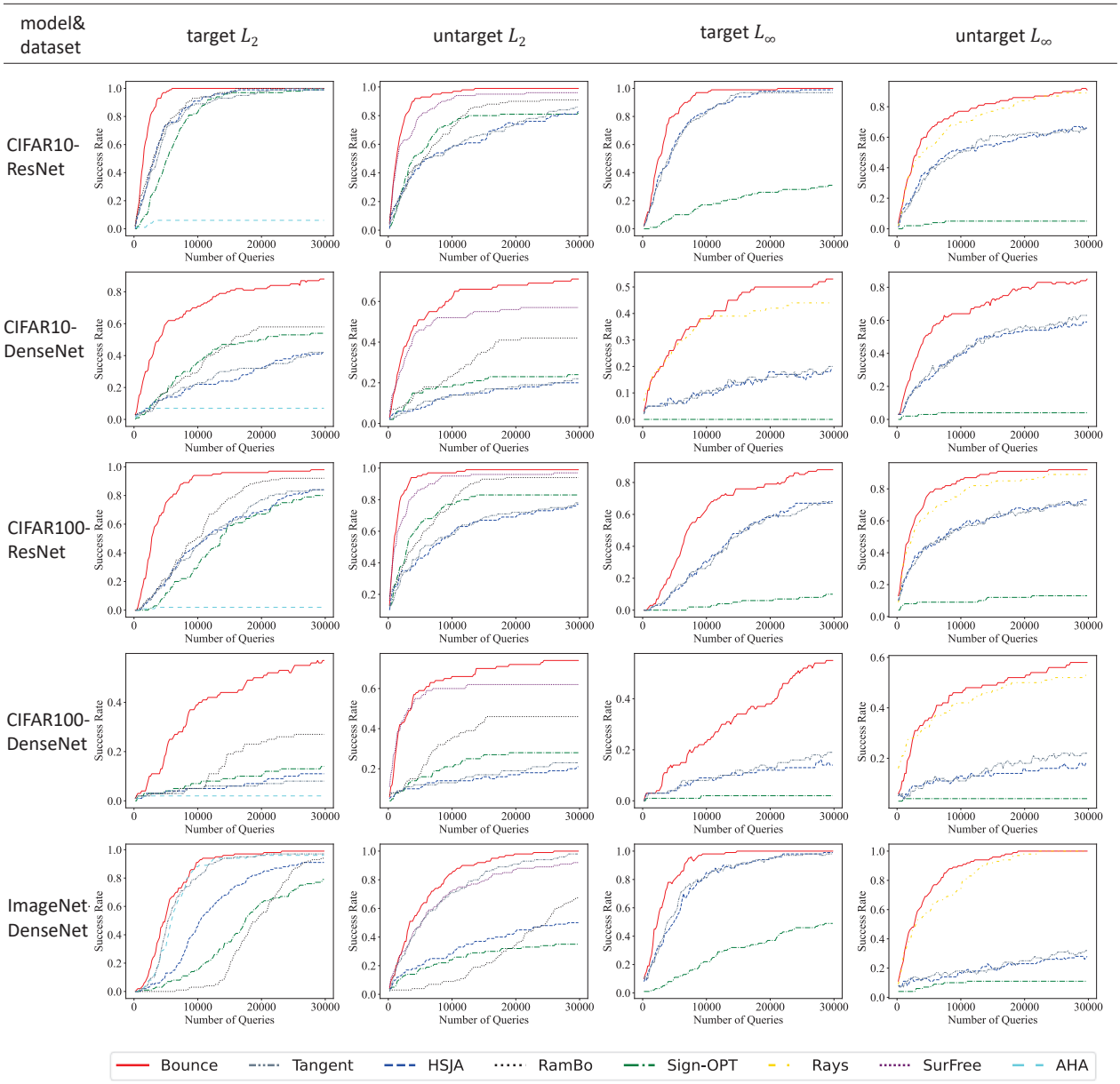


Figure 6. ASR against the number of model queries on datasets of CIFAR10, CIFAR100, and ImageNet with models of ResNet and DenseNet from top to bottom. Columns are targeted l_2, untargeted l_2, targeted l_∞, untargeted l_∞ from left to right.

BounceAttack can achieve a relatively high **ASR** in the attack process. Within 10,000 model queries, the curve of BounceAttack rises fast which means that the distance of the samples is reduced quickly in the beginning stage of the attack. As the number of model queries increases, the **ASR** of BounceAttack is constantly higher than other attacks (e.g., when the numbers of queries are 5,000 and 30,000 in Table 4 and Table 5). At 30,000 model queries, BounceAttack achieves the highest **ASR** among all the attacks. This shows the excellent ability of BounceAttack to produce adversarial samples even if other attack methods fail.

In our experiments, some attack methods show unsatisfactory performance, such as AHA and Sign-OPT. Sign-OPT ends with a large distance, especially under L_∞ constraint, where its ASR is no more than 50%. Sign-OPT searches the smallest perturbation by optimizing the perturbation direction and perturbation size. In each iteration, it generates a new direction with a small movement from the previous direction, which means that the search direction changes slowly and might fall into the local minimum. AHA cannot support datasets with small-size images well (i.e., CIFAR10

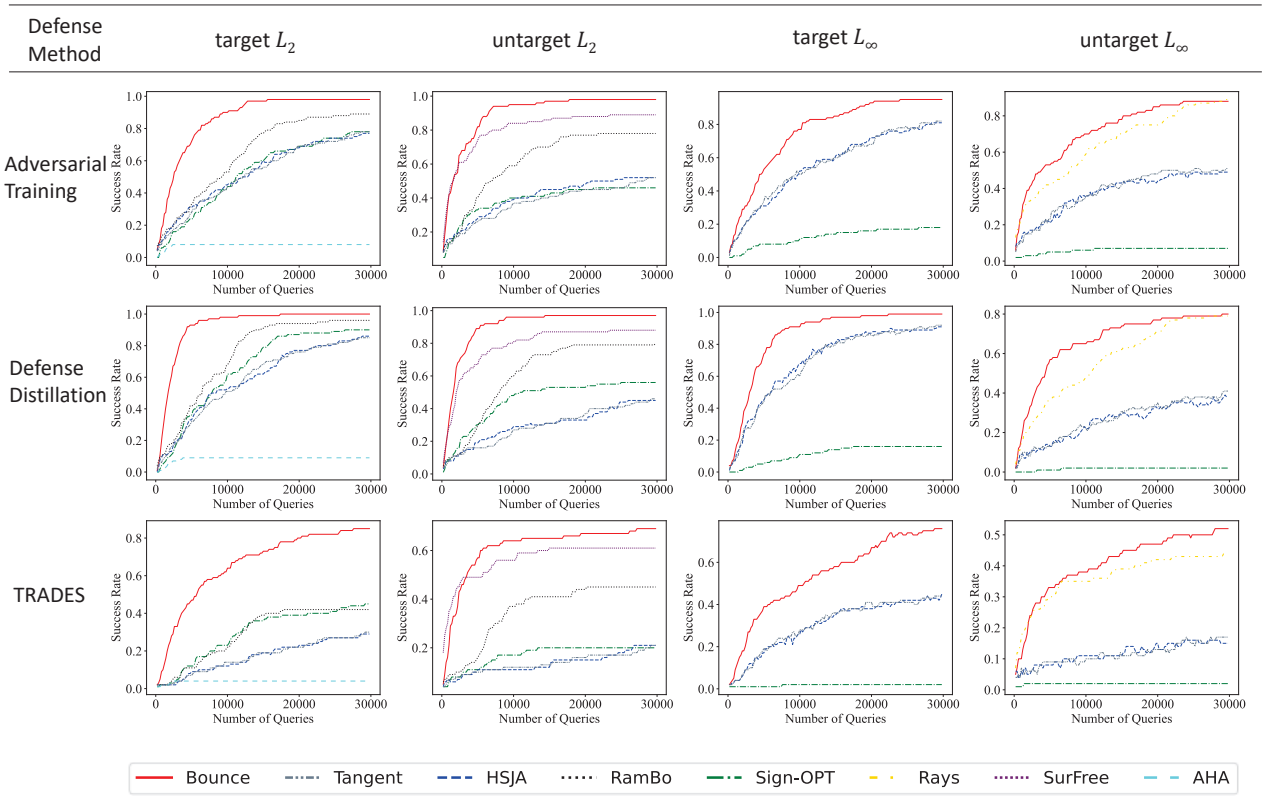


Figure 7. Attack performance against defense mechanisms. The figure shows ASR versus the number of model queries on CIFAR10 with ResNet against adversarial training, defensive distillation, and TRADES from top to bottom rows. Columns are targeted l_2, untargeted l_2, targeted l_∞, untargeted l_∞ from left to right.

and CIFAR100), which is also verified in our experiments (i.e., the target-L_2 column in Fig. 6).

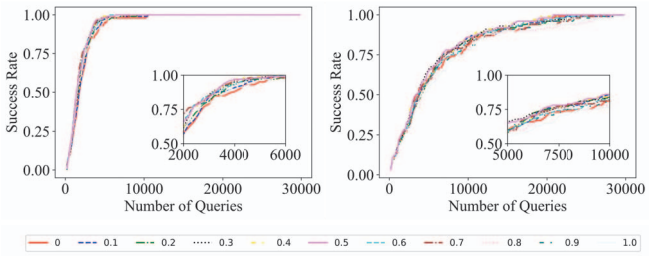
To gain insights on how BounceAttack succeeds in scenarios where existing attacks fail, we compare the attack processes of HSJA and BounceAttack on the same randomly chosen sample in Fig. 5. Given the success threshold $\epsilon = 1$, HSJA fails to generate an adversarial sample within 30,000 queries, but BounceAttack succeeds. However, if we switch the attack method to BounceAttack at the 25th iteration of HSJA, the attack quickly converges to a distance smaller than ϵ, resulting in a successful adversarial sample. This demonstrates that, compared to the estimated gradient used in HSJA, the bounce direction used in BounceAttack indeed improves the query efficiency as well as the attack success rate.

5.3. Performance against Defense Mechanisms

We evaluate the performance of BounceAttack against the defense mechanisms on ResNet50 trained on CIFAR10. In Fig. 7, we present the curve of the **ASR** against the number of model queries. We set the attack success threshold $\epsilon = 1, 2, 4$ under the L_2 constraint and $\epsilon = 0.05, 0.1, 0.2$ under L_∞ constraint, and present the **ASR** at 5,000 model queries in Table 8 and Table 9 in Appendix A.

BounceAttack shows outstanding attack performance. It can still acquire a fast-rising speed in the **ASR** and maintain a high **ASR** in the final stage in each sub-figure in Fig. 7. The performance of BounceAttack in the experiment against TRADES indicates the superiority of BounceAttack facing strong defense mechanisms.

BounceAttack can achieve a high **ASR** when the ϵ value is small (i.e., $\epsilon = 1$ under L_2 constraint and $\epsilon = 0.05$ under L_∞ constraint), where a large rate of generated samples by other attack methods fail to satisfy the requirements, as shown in Table 8 and Table 9. SurFree and RayS can also achieve a high **ASR** due to the use of geometric information but are still outperformed by BounceAttack. We calculate the rate of the average **ASR** of other attack methods to that of BounceAttack and find that other attack methods can only reach 18%-87% **ASR** compared with BounceAttack across all the experiments when we measure the success with the smallest ϵ value in L_2 and L_∞ respectively. In Table 8, when we fix $\epsilon = 1, 2, 4$ respectively, BounceAttack can acquire a high attack success rate when the $\epsilon = 1$ and can reach 100% in 50% scenarios when $\epsilon = 2$. This means that BounceAttack can generate a small-size perturbation and meet the attack goal against the defense mechanisms even if ϵ is strict. A similar result under L_∞ constraint can be observed in Table 9.



(a) CIFAR10 targeted l_2 (b) ImageNet untargeted l_2

Figure 8. The effect of the momentum decay factor μ on the attack performance of BounceAttack from 0.0 to 1.0 with an interval of 0.1. (a) ASR versus the number of queries with $\epsilon = 1.0$ in the CIFAR10-ResNet-targeted-L_2 setting. (b) ASR versus the number of queries with $\epsilon = 5.0$ in the ImageNet-DenseNet-untargeted-L_2 setting.

TABLE 7. ABLATION STUDY RESULTS. "B, M, S, BOUNCE" STANDS FOR THE BOUNCE SEARCH, MOMENTUM, SMOOTH SEARCH, AND THE COMPLETE BOUNCE ATTACK RESPECTIVELY.

Dataset	constraints	HSJA	HSJA+B	HSJA+M	HSJA+BS	HSJA+BM	Bounce
CIFAR10	L_2-tar	1.130	0.622	0.628	0.616	0.614	**0.611**
	L_2-untar	0.816	0.388	0.390	0.387	0.386	**0.384**
	L_∞-tar	0.028	0.022	0.024	0.020	0.021	**0.019**
	L_∞-untar	0.018	0.013	0.015	0.012	0.012	**0.011**
CIFAR100	L_2-tar	2.501	1.411	1.453	1.328	1.222	**1.220**
	L_2-untar	0.888	0.371	0.375	0.367	0.370	**0.366**
	L_∞-tar	0.070	0.054	0.056	0.051	0.049	**0.042**
	L_∞-untar	0.020	0.013	0.013	0.011	0.012	**0.011**

5.4. Effect of Hyper-Parameter

To show how the hyper-parameter μ for momentum affects the performance of BounceAttack, we set the value of μ from 0.0 to 1.0 with an interval of 0.1 and depict the relations of the **ASR** against the number of queries in the CIFAR10-ResNet-targeted-L_2 setting in Fig. 8 (a) and ImageNet-DenseNet-untargeted-L_2 setting in Fig. 8 (b).

The curves with $\mu = 0.5$ are effective in the attack process in the sense that they rise relatively quickly and eventually reach 100% **ASR** in both Fig. 8 (a) and Fig. 8 (b). We also enlarge the area before BounceAttack reaches 100% **ASR** in Fig. 8, and find that μ values that are either too large (e.g., $\mu = 1.0$) or too small (e.g., $\mu = 0.0$) will produce sub-optimal results in the **ASR** curves. An improper μ value will cause either a slower converging speed in the beginning stage of the attack or a lower **ASR** at the end of the attack. To balance the efficiency and the final **ASR**, we choose $\mu = 0.5$ in our experiments.

5.5. Ablation study

BounceAttack has three additional components compared with HSJA: bounce search, momentum, and smooth search. To show the significance of each component, we add each component to HSJA to attack the DenseNet121 models trained on CIFAR10 and CIFAR100. We present the performance measured by median distance in Table 7. Notably, the smooth search works only together with the bounce search because it is used to change the bounce direction to the estimated gradient directly.

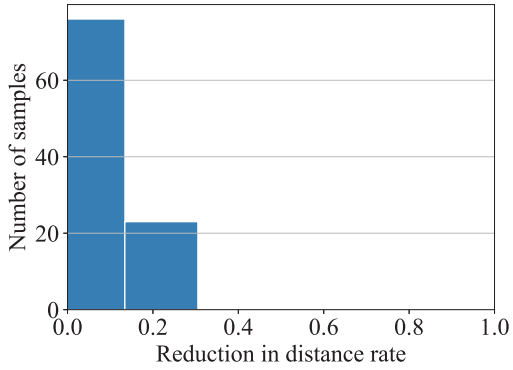


Figure 9. The frequency distribution histogram on CIFAR100-DenseNet-targeted-L_2 to illustrate the reduction in the ratio of the final distance to the switch-time distance in BounceAttack. The x-axis is the portion rate and the y-axis is the frequency. It illustrates how much distance can be reduced when switching to the smooth search.

5.5.1. Effect of Bounce Search. To study the effects of bounce search, we add the bounce search on "HSJA" and on "HSJA+M" respectively to compare their performance with the previous ones. In Table 7, comparing the results in the columns of "HSJA" and "HSJA+B", we can find that if we only use the bounce search to decompose the estimated gradients, the reduction of the distance can maximally reach 45%. Comparing the columns of "HSJA+M" and "HSJA+BM", bounce search can keep reducing 1%-20% distance even if the distance is already small.

5.5.2. Effect of Momentum. To study the effects of momentum, we add momentum on "HSJA" and "HSJA+BS" respectively to compare the improvements. From the columns of "HSJA" and "HSJA+M" in Table 7, the reduction of distance is 37% on average. When we combine the momentum with the bounce search (i.e., the columns of "HSJA+B" and "HSJA+BM"), it can be observed that the distance is reduced from columns "HSJA+B" and "HSJA+BM". A similar result can be reached in the columns "HSJA+BS" and "Bounce". These findings support that momentum can boost the bounce search.

5.5.3. Effect of Smooth Search. To study the effects of the smooth search, we first observe the rate of samples that conduct the smooth search in the attack process. We conduct experiments on "HSJA+B" and "HSJA+BS" to measure the performance of the smooth search, and the result is that the involvement of smooth search can reduce the distance by 1%-10%.

We present a frequency distribution histogram of the experimental result on the CIFAR10-DenseNet-targeted-L_2 scenario in Fig. 9. It illustrates that the distance of a large number of samples (i.e., more than 70% samples) can be

reduced by around 10% due to the smooth search. This means that the smooth search plays a role in fine-tuning the adversarial sample with a smaller distance in BounceAttack.

6. Conclusion

In this paper, we propose BounceAttack as an extension of HSJA. It decomposes the estimated gradient for the decision-based black-box attack. BounceAttack shows good performance under both L_2 and L_∞ constraints in both targeted and untargeted settings. We provide an analysis of why the bounce vector can help to obtain a small perturbation distance. Extensive experiments are conducted to show that BounceAttack is outstanding in both query efficiency and attack success rate. Ablation studies are conducted to figure out how each component of BounceAttack affects its performance. We hope this work can be a new baseline attack method in the decision-based black-box attack.

Acknowledgment

We would like to express our sincere gratitude to the anonymous reviewers and Shepherd for their invaluable feedback and constructive comments, which greatly contributed to the enhancement of this paper. This work was supported in part by National Natural Science Foundation of China (62102353), and by National Key R&D Program of China (2020AAA0107700).

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. 1, 2

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014. 1, 2, 9

[3] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582. 1, 2, 3

[4] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. IEEE, 2017, pp. 39–57. 1, 3

[5] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*. Chapman and Hall/CRC, 2018, pp. 99–112. 1, 2

[6] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*. Springer, 2013, pp. 387–402. 1, 3

[7] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 15–26. 1, 3

[8] J. Li, R. Ji, H. Liu, J. Liu, B. Zhong, C. Deng, and Q. Tian, "Projection & probability-driven black-box attack," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 362–371. 1

[9] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2137–2146. 1, 3

[10] A. Ilyas, L. Engstrom, and A. Madry, "Prior convictions: Black-box adversarial attacks with bandits and priors," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=BkMiWhR5K7 1, 3

[11] C. Guo, J. Gardner, Y. You, A. G. Wilson, and K. Weinberger, "Simple black-box adversarial attacks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2484–2493. 1

[12] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193. 1, 3

[13] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," *arXiv preprint arXiv:1908.06281*, 2019. 1

[14] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2730–2739. 1

[15] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4312–4321. 1, 3

[16] M. Alzantot, Y. Sharma, S. Chakraborty, H. Zhang, C.-J. Hsieh, and M. B. Srivastava, "Genattack: Practical black-box attacks with gradient-free optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019, pp. 1111–1119. 1, 3

[17] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," in *International Conference on Learning Representations*, 2018. 1, 3

[18] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*. IEEE, 2020, pp. 1277–1294. 1, 2, 3, 4, 6, 7, 8

[19] J. Li, R. Ji, P. Chen, B. Zhang, X. Hong, R. Zhang, S. Li, J. Li, F. Huang, and Y. Wu, "Aha! adaptive history-driven attack for decision-based black-box models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 168–16 177. 1, 2, 3, 7, 8

[20] M. Cheng, T. Le, P.-Y. Chen, H. Zhang, J. Yi, and C.-J. Hsieh, "Query-efficient hard-label black-box attack: An optimization-based approach," in *International Conference on Learning Representations*, 2018. 1, 3

[21] M. Cheng, S. Singh, P. Chen, P.-Y. Chen, S. Liu, and C.-J. Hsieh, "Sign-opt: A query-efficient hard-label adversarial attack," *arXiv preprint arXiv:1909.10773*, 2019. 1, 2, 3, 8

[22] J. Chen and Q. Gu, "Rays: A ray searching method for hard-label adversarial attack," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1739–1747. 1, 2, 3, 8

[23] T. Maho, T. Furon, and E. Le Merrer, "Surfree: a fast surrogate-free black-box attack," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 430–10 439. 1, 2, 3, 7, 8

[24] X. Wang, Z. Zhang, K. Tong, D. Gong, K. He, Z. Li, and W. Liu, "Triangle attack: A query-efficient decision-based adversarial attack," *arXiv preprint arXiv:2112.06569*, 2021. 1, 2, 3, 7

[25] V. Q. Vo, E. Abbasnejad, and D. C. Ranasinghe, "Ramboattack: A robust query efficient deep neural network decision exploit," *arXiv preprint arXiv:2112.05282*, 2021. 1, 2, 3, 8

[26] C. Ma, X. Guo, L. Chen, J.-H. Yong, and Y. Wang, "Finding optimal tangent points for reducing distortions of hard-label attacks," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 19 288–19 300. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/a113c1ecd3cace2237256f4c712f61b5-Paper.pdf 1, 3, 8

[27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017. 3, 8, 9

[28] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 582–597. 3, 8

[29] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2016, pp. 372–387. 3

[30] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017, pp. 506–519. 3

[31] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016. 3

[32] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv:1704.03453*, 2017. 3

[33] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv preprint arXiv:1611.02770*, 2016. 3

[34] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, "Empirical study of the topology and geometry of deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3762–3770. 6

[35] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Technical report*, 2009. 7

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255. 7

[37] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009. 7

[38] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788. 7

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 7

[40] ——, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645. 7

[41] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708. 7

[42] M. Contributors, "Openmmlab's image classification toolbox and benchmark," https://github.com/open-mmlab/mmclassification, 2020. 7

[43] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International conference on machine learning*. PMLR, 2019, pp. 7472–7482. 8

Appendix A.

TABLE 8. ASR AGAINST DIFFERENT DEFENSE MECHANISMS OF DIFFERENT ATTACKS MEASURED BY DIFFERENT ϵ UNDER L_2 CONSTRAINT ON CIFAR10 WITH RESNET. N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Defense	Attack	$\epsilon = 1$							$\epsilon = 2$							$\epsilon = 4$						
		Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent	Bounce	HSJA	Sopt	RamBo	SurFree	AHA	Tangent
Adversarial Training	untargeted	**0.99**	0.67	0.73	0.70	0.94	N.A.	0.60	**1.00**	0.95	0.97	0.96	1.00	N.A.	0.94	**1.00**	1.00	1.00	0.97	1.00	N.A.	1.00
	targeted	**0.72**	0.31	0.24	0.33	N.A.	0.08	0.28	**0.92**	0.70	0.46	0.68	N.A	0.22	0.68	**1.00**	0.93	0.82	0.95	N.A	0.63	0.94
TRADES	untargeted	**0.87**	0.34	0.49	0.44	0.75	N.A.	0.30	**0.99**	0.81	0.93	0.84	0.95	N.A.	0.83	**1.00**	1.00	1.00	1.00	1.00	N.A.	1.00
	targeted	**0.47**	0.09	0.12	0.11	N.A.	0.04	0.07	**0.92**	0.45	0.55	0.49	N.A	0.15	0.44	**1.00**	0.96	0.97	0.96	N.A	0.53	0.97
Defensive Distillation	untargeted	**0.99**	0.63	0.76	0.73	0.95	N.A.	0.56	**1.00**	0.98	0.99	0.98	1.00	N.A.	0.96	**1.00**	1.00	1.00	1.00	1.00	N.A.	1.00
	targeted	**0.93**	0.33	0.36	0.42	N.A.	0.09	0.32	**1.00**	0.86	0.71	0.89	N.A	0.36	0.89	**1.00**	1.00	0.94	0.99	N.A	0.80	1.00

TABLE 9. ASR AGAINST DIFFERENT DEFENSE MECHANISMS OF DIFFERENT ATTACKS MEASURED BY DIFFERENT ϵ UNDER L_∞ CONSTRAINT ON CIFAR10 WITH RESNET. N.A. MEANS THAT THE CORRESPONDING ATTACK IS NOT APPLICABLE TO THE ATTACK SETTING.

Defense	Attack	$\epsilon = 0.05$					$\epsilon = 0.1$					$\epsilon = 0.2$				
		Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent	Bounce	HSJA	Sopt	RayS	Tangent
Adversarial Training	untargeted	**0.99**	0.95	0.54	0.99	0.95	**1.00**	1.00	0.87	1.00	1.00	**1.00**	1.00	0.98	1.00	1.00
	targeted	**0.69**	0.53	0.15	N.A.	0.51	**0.89**	0.76	0.31	N.A.	0.74	**1.00**	0.98	0.53	N.A.	0.98
TRADES	untargeted	**0.94**	0.88	0.25	0.94	0.87	**1.00**	1.00	0.54	1.00	1.00	**1.00**	1.00	0.92	1.00	1.00
	targeted	**0.70**	0.46	0.04	N.A.	0.48	**0.94**	0.91	0.12	N.A.	0.90	**1.00**	1.00	0.44	N.A.	1.00
Defensive Distillation	untargeted	**1.00**	0.98	0.42	1.00	0.98	**1.00**	1.00	0.89	1.00	1.00	**1.00**	1.00	1.00	1.00	1.00
	targeted	**0.91**	0.76	0.14	N.A.	0.74	**0.98**	0.93	0.41	N.A.	0.92	**1.00**	0.99	0.76	N.A.	1.00

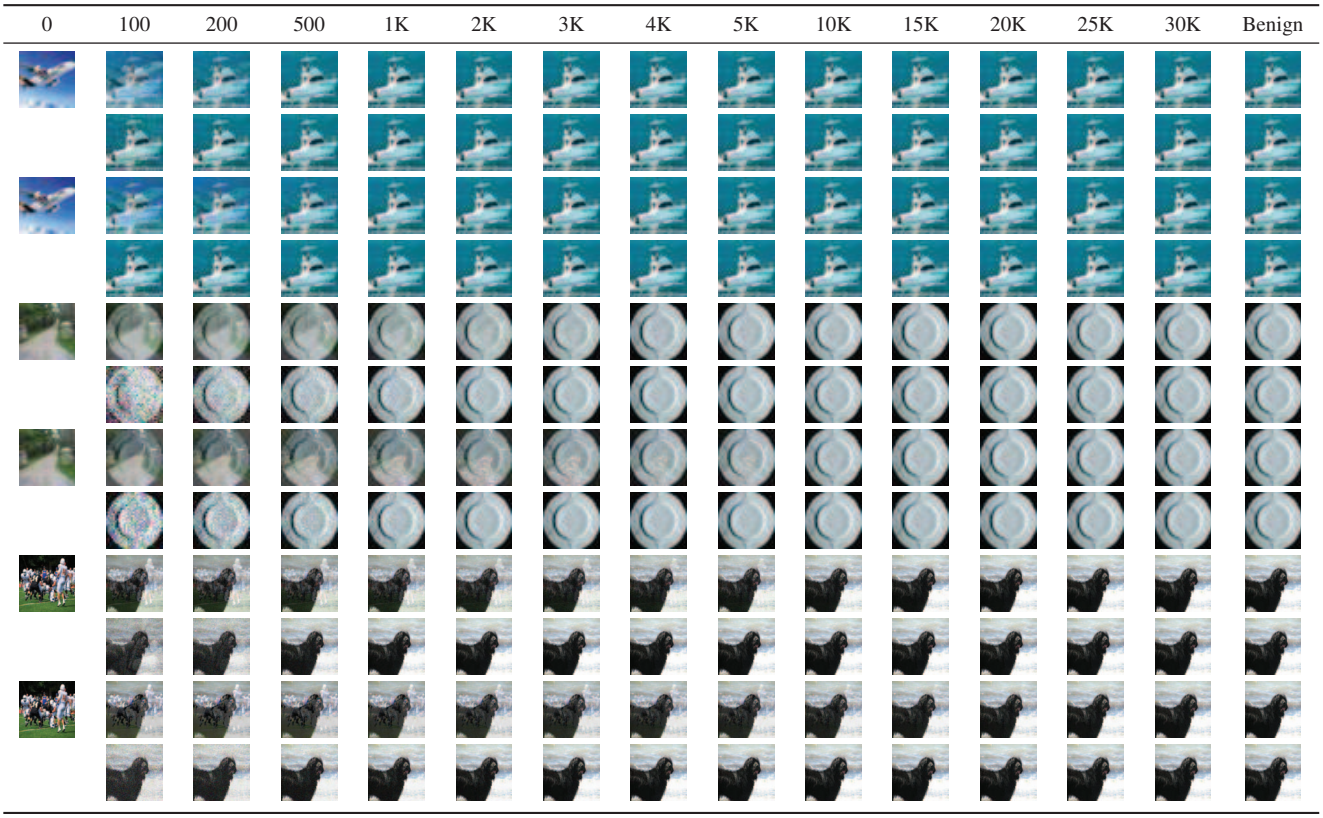


Figure 10. Visualized trajectories of BounceAttack for generating targeted and untargeted adversarial samples under L_2 and L_∞ constraints on randomly selected images in CIFAR10, CIFAR100 and ImageNet. 1st column: initialized intermediate sample $\hat{\mathbf{X}}_1$. 2nd-14th columns: generated adversarial candidates $\mathbf{X}_t$ when t =100, 200, 500, 1K, 2K, 3K, 4K, 5K, 10K, 15K, 20K, 25K and 30K. 15th column: The benign image $\mathbf{X}^*$. 1st row: targeted L_2 on CIFAR10. 2nd row: untargeted L_2 on CIFAR10. 3rd row: targeted L_∞ on CIFAR10. 4th row: untargeted L_∞ on CIFAR10. 5th row: targeted L_2 on CIFAR100. 6th row: untargeted L_2 on CIFAR100. 7th row: targeted L_∞ on CIFAR100. 8th row: untargeted L_∞ on CIFAR100. 9th row: targeted L_2 on ImageNet. 10th row: untargeted L_2 on ImageNet. 11th row: targeted L_∞ on ImageNet. 12th row: untargeted L_∞ on ImageNet.

Appendix B.
Meta-Review

B.1. Summary

This paper proposes a novel decision-based black-box attack that uses an orthogonalized ”bouncing” strategy to reduce the number of queries to find adversarial examples. The paper describes the attack and its intuition, and performs a thorough comparison with prior attacks against standard models on CIFAR-10, CIFAR-100 and ImageNet.

B.2. Scientific Contributions

- Provides a Valuable Step Forward in an Established Field.
- Creates a New Tool to Enable Future Science.

B.3. Reasons for Acceptance

1) The proposed attack is simple, intuitive, and effective, and an improvement over the state of the art.
2) The authors extensively evaluate various black-box attacks and investigate the influence of adversarial training and model distillation on the attack's performance.
3) The theoretical discussions are helpful to explain why the attack is effective and clarify the differences against HJSA.