Dashed Line Defense: Plug-And-Play Defense Against Adaptive Score-Based Query Attacks

Yanzhang Fu* research@fyzdalao.xyz
Jizhou Luo*† luojizhou@hit.edu.cn
Zizheng Guo* zguo.research@gmail.com
* Harbin Institute of Technology † *Corresponding Author*

Abstract

Score-based query attacks pose a serious threat to deep learning models by crafting adversarial examples (AEs) using only black-box access to model output scores, iteratively optimizing inputs based on observed loss values. While recent runtime defenses attempt to disrupt this process via output perturbation, most either require access to model parameters or fail when attackers adapt their tactics. In this paper, we first reveal that even the state-of-the-art plug-and-play defense can be bypassed by adaptive attacks, exposing a critical limitation of existing runtime defenses. We then propose Dashed Line Defense (DLD), a plug-and-play post-processing method specifically designed to withstand adaptive query strategies. By introducing ambiguity in how the observed loss reflects the true adversarial strength of candidate examples, DLD prevents attackers from reliably analyzing and adapting their queries, effectively disrupting the AE generation process. We provide theoretical guarantees of DLD's defense capability and validate its effectiveness through experiments on ImageNet, demonstrating that DLD consistently outperforms prior defenses—even under worst-case adaptive attacks—while preserving the model's predicted labels.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved remarkable success across various domains, yet they remain vulnerable to adversarial examples (AEs) — inputs that differ only slightly from legitimate samples but cause the model to make incorrect predictions (Goodfellow et al., 2015). This vulnerability raises serious security concerns, especially given the rapid advancement of black-box score-based query attacks (SQAs). These attacks leverage zeroth-order optimization (ZOO) to craft AEs by querying only the model's output scores, without requiring access to internal parameters, making them highly practical and hard to defend against.

To mitigate these threats, researchers have explored two main categories of defenses against SQAs: training-time and runtime defenses. Training-time defenses, known as adversarial training (AT) (Madry et al., 2018), enhance robustness by incorporating AEs during model training but are costly and often impractical when data or model access is limited. Runtime defenses instead attempt to disrupt the attacker's optimization by perturbing outputs during prediction (Qin et al., 2021a,b), avoiding retraining but often degrading accuracy or requiring model modifications. Among the recent runtime defenses, the state-of-the-art Adversarial Attack on Attackers (AAA) (Chen et al., 2022) stands out for its plug-and-play design, offering strong protection claims without altering model predictions or requiring access to internal parameters.

However, as we reveal in Section 4.1, AAA remains vulnerable to adaptive attacks, exposing a fundamental limitation of current runtime defenses against black-box adversaries: they often underestimate attacker adaptivity by assuming static or uninformed adversaries, while the true effectiveness of any defense depends critically on its ability to withstand adaptive adversaries who design attack strategies to target the defense itself. This observation echoes findings from the white-box setting (Tramer et al., 2020), where defenses that ignore attacker adaptivity were shown to provide only illusory robustness.

Building on the observed weakness of the state-of-the-

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

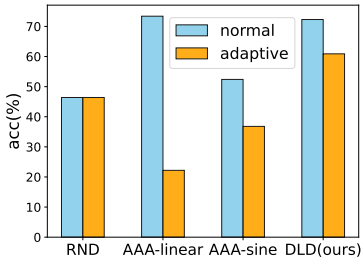


Figure 1: Under-attack accuracy of four defenses, with blue and orange bars indicating normal and adaptive attacks, respectively. Note that RND reduces accuracy on non-adversarial samples.

art defense, we propose Dashed Line Defense (DLD), a plug-and-play post-processing method that preserves predicted labels and remains effective against adaptive attacks. DLD employs a non-continuous mapping between the attacker-observed loss and the true loss (illustrated later in Figure 3), introducing ambiguity in how the observed loss reflects the adversarial effectiveness of AE candidates. This ambiguity disrupts smooth optimization paths and greatly increases difficulty for attackers to analyze and bypass the defense.

We provide a theoretical analysis characterizing DLD's defense strength against both standard and adaptive attacks. Experiments on the ImageNet dataset show that AAA, despite the state-of-the-art runtime defense, suffers severe degradation under adaptive attacks (see Figure 1 and Section 5.2), whereas DLD maintains high accuracy even under worst-case adaptive scenarios. Furthermore, DLD consistently outperforms the prior plug-and-play approach RND (Qin et al., 2021b), achieving higher accuracy while preserving model predictions. These results highlight the necessity of designing defenses that account for attacker adaptivity, which is DLD's central motivation.

Our main contributions are threefold:

- We reveal a fundamental limitation of the state-of-the-art runtime defense (AAA) against adaptive SQA tactics, exposing a concrete gap between claimed and actual protection under adaptivity.
- We propose Dashed Line Defense (DLD), a novel plug-and-play post-processing method that confuses the attack optimization process and is difficult to bypass.
- We provide a formal analysis that characterizes DLD's defense strength, and through experiments on ImageNet, show that DLD outperforms prior plug-and-play defenses (including AAA and RND) even under worst-case adaptive attacks.

2 RELATED WORK

Generating Adversarial Examples began with pioneering works (Szegedy et al., 2014; Goodfellow et al., 2015), which showed DNNs are vulnerable to small perturbations. In white-box settings, where model gradients are accessible, attacks like C&W (Carlini and Wagner, 2017) generate AEs using gradient-based optimization. In black-box settings, early methods relied on substitute models to approximate gradients (Papernot et al., 2017; Li et al., 2020; Xie et al., 2019a).

In 2017, Chen et al. (2017) proposed a ZOO-based attack that outperformed substitute model-based methods in black-box settings. This breakthrough sparked extensive subsequent research employing either gradient estimation-based ZOO (Ilyas et al., 2018; Bhagoji et al., 2018; Ilyas et al., 2019; Al-Dujaili and O'Reilly, 2020) or random search-based ZOO (Guo et al., 2019; Alzantot et al., 2019; Andriushchenko et al., 2020). However, these methods mainly focus on efficiently generating the next candidate during optimization, while overlooking challenging scenarios like non-convex landscapes or stochastic noise, leading to suboptimal performance against defenses.

Defenses to Adversarial Examples can be categorized based on how they interact with the model: (1) modifying the training process, (2) altering the model's internal structure, or (3) adding an external plug-and-play module.

The first approach, AT (Madry et al., 2018; Tramer et al., 2018), augments the training data with AEs during training. Although effective, it reduces clean accuracy, incurs high computational cost (Shafahi et al., 2019), and often generalizes poorly to unseen attacks (Tramer et al., 2020). Its effectiveness remains largely empirical (Schmidt et al., 2018), and recent theoretical work on data debugging (Guo et al., 2025) suggests that certifying the effects of training-time data manipulations like AT may be fundamentally difficult.

The second approach enhances robustness by modifying the model's architecture or computation. Examples include weight noise injection (Qin et al., 2021a), feature denoising (Xie et al., 2019b), and randomized smoothing (Cohen et al., 2019). These methods can improve robustness against standard attacks but often rely on gradient obfuscation and fail under adaptive attacks (Athalye et al., 2018).

The third approach adds plug-and-play modules without modifying the model. Pre-processing defenses perturb the input before prediction (Byun et al., 2022; Qin et al., 2021b), but the resulting change to output

scores is indirect and unpredictable, potentially altering the model's prediction. Post-processing defenses, such as AAA (Chen et al., 2022), modify the output scores after prediction, ensuring the predicted label remains unchanged. However, as shown in this paper, even such state-of-the-art post-processing defenses can be bypassed by adaptive attacks.

While there also exist methods that detect attacks via historical queries (Chen et al., 2020a; Li et al., 2025; Park et al., 2025), their focus is somewhat orthogonal to this work. Such detection inherently limits the number of queries an attacker can make, forcing attackers to prioritize query efficiency over time efficiency.

Adaptive Attacks on Defenses have evolved in response to defense proposals. Many defenses relying on obfuscated gradients were soon broken by adaptive attacks (Athalye et al., 2018). Tramer et al. (2020) outlined general principles for designing adaptive attacks, and Yao et al. (2021) developed a tool to automate the process of crafting such attacks. These successful adaptive attacks primarily target white-box settings with full model access. While black-box attacks can be incorporated into adaptive strategies, query efficiency is often neglected. To our knowledge, adaptive attacks in black-box settings remain largely underexplored.

3 PRELIMINARIES

3.1 Score-Based Black-Box Attack

We consider a black-box classifier $f : \mathcal{X} \to \mathcal{Y}$, where $f_y(\mathbf{x})$ denotes the output score for class y given input $\mathbf{x}$, and the predicted label is $f(\mathbf{x}) = \arg\max_y f_y(\mathbf{x})$. Only the output scores $f_y(\mathbf{x})$ for each class y can be accessed through queries, while the model's internal parameters and gradients are inaccessible. Let $\mathcal{N}(\mathbf{x}_0, \epsilon) = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}_0\| \le \epsilon\}$ denote the neighborhood around a clean input $\mathbf{x}_0$. An adversarial example is formally defined as follows:

Definition 3.1. Let $f : \mathcal{X} \to \mathcal{Y}$ be a classifier, $\mathbf{x}_0 \in \mathcal{X}$ a clean input, and ϵ_{n} a perturbation budget. An input $\mathbf{x} \in \mathcal{X}$ is a $(f, \mathbf{x}_0, \epsilon_{\mathrm{n}})$-adversarial example (AE) if

$$\mathbf{x} \in \mathcal{N}(\mathbf{x}_0, \epsilon_{\mathrm{n}}) \quad \text{and} \quad f(\mathbf{x}) \neq f(\mathbf{x}_0).$$

Given a correctly classified input $\mathbf{x}_0 \in \mathcal{X}$ with ground-truth label $y \in \mathcal{Y}$, a query budget n, and a perturbation budget ϵ_{n}, the goal of a black-box score-based query attack (SQA) is to craft a $(f, \mathbf{x}_0, \epsilon_{\mathrm{n}})$-AE using no more than n queries to f.

Since the model is black-box and only the output scores are accessible, the attack is typically modeled as a ZOO process that minimizes the margin loss:

$$\mathcal{L}(\mathbf{x}, y) = f_y(\mathbf{x}) - \max_{t \neq y} f_t(\mathbf{x}), \tag{1}$$

Algorithm 1: standard-SQA($\mathbf{x}_0, G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}$)

Input: clean sample $\mathbf{x}_0$, example generator $G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}$
Output: AE candidate $\mathbf{x}$
$y_0 \leftarrow f(\mathbf{x}_0)$, $\mathbf{x} \leftarrow \mathbf{x}_0$;
while $\mathcal{L}(\mathbf{x}, y_0) > 0$ ***and*** *not out of query budget* **do**
  $\mathbf{x}_{\mathrm{new}} \leftarrow G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}(\mathbf{x})$;
  if $\mathcal{L}(\mathbf{x}_{new}, y_0) < \mathcal{L}(\mathbf{x}, y_0)$ **then**
    $\mathbf{x} \leftarrow \mathbf{x}_{\mathrm{new}}$;
return x

where $y \in \mathcal{Y}$ is a reference label that defines the margin. It is typically chosen as the predicted label of the initial input $\mathbf{x}_0$. A negative margin indicates that the model no longer favors label y, and thus serves as a criterion for attack success [1]. If y is unspecified, we use the shorthand notation:

$$\mathcal{L}(\mathbf{x}) = f_{f(\mathbf{x})}(\mathbf{x}) - \max_{t \neq f(\mathbf{x})} f_t(\mathbf{x}). \tag{2}$$

Each iteration, the attacker uses a generator $G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}(\cdot) : \mathcal{X} \to \mathcal{N}(\mathbf{x}_0, \epsilon_{\mathrm{n}})$ to produce a new candidate $\mathbf{x}^{\mathrm{new}}$. It may implement heuristic strategies to explore the neighborhood of $\mathbf{x}_0$ within the allowed perturbation budget. Then, the attacker compares $\mathbf{x}^{\mathrm{new}}$ with current best sample $\mathbf{x}^k$ and performs the update:

$$\mathbf{x}^{k+1} = \begin{cases} \mathbf{x}^{\mathrm{new}}, & \mathcal{L}(\mathbf{x}^{\mathrm{new}}, y) < \mathcal{L}(\mathbf{x}^k, y) \\ \mathbf{x}^k, & \text{otherwise} \end{cases} \tag{3}$$

This iterative procedure is summarized in Algorithm 1. The attack is deemed successful if it finds some $\mathbf{x}$ such that $\mathcal{L}(\mathbf{x}, y) < 0$; otherwise, it fails when the query budget n is exhausted without success.

Note that generating $\mathbf{x}^{\mathrm{new}}$ using $G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}$ may involve multiple queries to the model, so the number of iterations can be fewer than the query budget n. We write $G = G_{(\mathbf{x}_0,\epsilon_{\mathrm{n}})}$ for brevity whenever the context permits.

3.2 Plug-and-Play Defense

Under black-box constraints, defenders can adopt plug-and-play defenses that perturb model outputs to mislead attackers, causing observed loss values to differ from the true ones. These defenses fall into two categories: preprocessing and postprocessing.

Given input $\mathbf{x}$, let $\mathcal{L}_{\mathrm{real}}(\mathbf{x})$ be the true margin loss and $\mathcal{L}_{\mathrm{observe}}(\mathbf{x})$ the attacker-observed loss. Preprocessing

[1] This loss is designed for untargeted attacks. In targeted attacks, the label t is a fixed target class chosen by the attacker rather than being arbitrary. This paper focuses on untargeted settings.

applies a (typically randomized) transformation $\mathcal{D}_{\text{pre}}$ before inference:

$$\mathcal{L}_{\text{observe}}(\mathbf{x}) = \mathcal{L}_{\text{real}}(\mathcal{D}_{\text{pre}}(\mathbf{x})).$$

Postprocessing perturbs the loss directly after prediction:

$$\mathcal{L}_{\text{observe}}(\mathbf{x}) = \mathcal{D}_{\text{post}}(\mathcal{L}_{\text{real}}(\mathbf{x})).$$

Preprocessing may harm prediction accuracy on benign inputs, since $f(\mathcal{D}_{\text{pre}}(\mathbf{x}))$ may differ from $f(\mathbf{x})$. In contrast, postprocessing preserves the original prediction while modifying only the loss.

3.3 Robustness Measure

We adopt a formal robustness notion proposed by Katz et al. (2017a) to characterize the stability of a classifier's output scores under small input perturbations.

Definition 3.2 (Global Robustness)**.** A classifier $f : \mathcal{X} \to \mathcal{Y}$ is said to be (δ, ϵ)-globally-robust in an input region D **iff**

$$\forall \mathbf{x}_1, \mathbf{x}_2 \in D,\ \|\mathbf{x}_1 - \mathbf{x}_2\| \leq \delta \\ \Rightarrow\ \forall y \in \mathcal{Y},\ |f_y(\mathbf{x}_1) - f_y(\mathbf{x}_2)| < \epsilon.$$

This definition requires that for any two inputs $\mathbf{x}_1, \mathbf{x}_2 \in D \subseteq \mathcal{X}$ that are close to each other, the model produces similar confidence scores for all classes. We will later use this notion to formalize assumptions about model behavior and to quantify the cost and guarantees of the proposed defense.

4 DEFENSE AGAINST ADAPTIVE ATTACK

4.1 Adaptive Attack on Existing Work

We begin by examining why and how the state-of-the-art defense method AAA (Chen et al., 2022) is vulnerable to adaptive attacks.

AAA is the first plug-and-play post-processing defense, whose loss mapping function $\mathcal{D}_{\text{post}}$ is defined as

$$\mathcal{D}_{\text{post}}(\mathcal{L}) = 2\lfloor \mathcal{L}/\tau \rfloor \tau + \tau - \mathcal{L},$$

where τ is a constant determining the length of each monotonic interval. The blue curve in Figure 2 shows the image of this $\mathcal{D}_{\text{post}}$.

This version, known as AAA-linear, was later found to be vulnerable after reviewer feedback: by reversing the descent direction within each interval, it enables attackers to adapt by switching to maximization. Specifically, Eq. 3 can be replaced with:

$$\mathbf{x}^{k+1} = \begin{cases} \mathbf{x}^k, & \mathcal{L}(\mathbf{x}^{\text{new}}, y) < \mathcal{L}(\mathbf{x}^k, y) \\ \mathbf{x}^{\text{new}}, & \mathcal{L}(\mathbf{x}^{\text{new}}, y) \geq \mathcal{L}(\mathbf{x}^k, y) \end{cases} \tag{4}$$

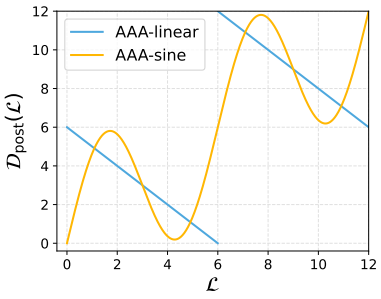


Figure 2: $\mathcal{D}_{\text{post}}$ of AAA

Algorithm 2: reverse-tactic-SQA($\mathbf{x}_0, G, t$)

Input: clean sample $\mathbf{x}_0$, sample generator G, threshold t
Output: AE candidate $\mathbf{x}$
Initialize $reverse \leftarrow$ `False`, $cnt \leftarrow 0$;
$y_0 \leftarrow f(\mathbf{x}_0)$, $\mathbf{x} \leftarrow \mathbf{x}_0$, $loss \leftarrow \mathcal{L}(\mathbf{x}, y_0)$;
while $loss > 0$ ***and*** *not out of query budget* **do**
  $\mathbf{x}_{\text{new}} \leftarrow G(\mathbf{x})$;
  $loss_{\text{new}} \leftarrow \mathcal{L}(\mathbf{x}_{\text{new}}, y_0)$;
  if $\neg((loss_{new} < loss) \oplus reverse)$ **then**
    $cnt \leftarrow cnt + 1$;
  if $cnt > t$ **then**
    $cnt \leftarrow 0$, $reverse \leftarrow \neg reverse$;
  if $(loss_{new} < loss) \oplus reverse$ **then**
    $cnt \leftarrow 0$, $\mathbf{x} \leftarrow \mathbf{x}_{\text{new}}$, $loss \leftarrow loss_{\text{new}}$;
return $\mathbf{x}$

To address this vulnerability, AAA-sine was proposed. With a constant α (0.7 by default), $\mathcal{D}_{\text{post}}$ is defined as

$$\mathcal{D}_{\text{post}}(\mathcal{L}) = \mathcal{L} - \alpha\tau \sin(\pi(1 - (2\mathcal{L} - (2\lfloor \mathcal{L}/\tau \rfloor + 1)\tau)/\tau)).$$

By combining increasing and decreasing intervals, this $\mathcal{D}_{\text{post}}$ aims to defend against both minimization and maximization steps, as shown in Figure 2. AAA-sine is therefore claimed to be robust to adaptive attacks. However, this design still has fundamental flaws. Each monotonic interval in $\mathcal{D}_{\text{post}}$ is only capable of misleading either minimization or maximization steps, but not both simultaneously. To exploit this weakness, we design a simple adaptive tactic (Algorithm 2) that alternates between minimization and maximization. The attack begins with standard minimization until no improvement is observed for t consecutive iterations. It then switches to maximization using Eq. 4 until the next stagnation, after which the direction-reversing cycle repeats. Despite its simplicity, this direction reversing tactic significantly degrades AAA's effectiveness. For instance, the under-attack accuracy of AAA-sine drops from 61.7% to 41.5% (see Section 5.2).

4.2 Dashed Line Defense

To address the weaknesses of post-processing defenses discussed above, we propose a novel method called Dashed Line Defense (DLD), where $\mathcal{D}_{\text{post}}$ is specifically designed to interfere with both minimization and maximization steps. A key insight behind DLD is that $\mathcal{D}_{\text{post}}$ must be non-smooth; otherwise, any monotonic interval of a continuous and smooth $\mathcal{D}_{\text{post}}$ would allow attackers to infer a reliable gradient direction, rendering the direction-reversal tactic (Algorithm 2) effective. The DLD mechanism is formally defined below.

Definition 4.1. Given $\tau > 0$, $h \in [0, 1]$, and $S \subseteq [0, 1)$, a (τ, h, S)-DLD is a post-processing module whose loss mapping $\mathcal{D}_{\text{post}}$ is defined as follows:

$$\mathcal{L}_{\text{bias}}(\mathcal{L}; \tau) = \lfloor \mathcal{L}/\tau \rfloor \cdot \tau \tag{5}$$

$$\begin{aligned}\mathcal{L}_{\text{high}}(\mathcal{L}; \tau, h) = \mathcal{L}_{\text{bias}}(\mathcal{L}; \tau) + h\tau \\ + (1-h)(\mathcal{L} - \mathcal{L}_{\text{bias}}(\mathcal{L}; \tau))\end{aligned} \tag{6}$$

$$\mathcal{L}_{\text{low}}(\mathcal{L}; \tau, h) = \mathcal{L}_{\text{bias}}(\mathcal{L}; \tau) + h\tau - h\left(\mathcal{L} - \mathcal{L}_{\text{bias}}(\mathcal{L}; \tau)\right) \tag{7}$$

$$\mathcal{D}_{\text{post}}(\mathcal{L}; \tau, h, S) = \begin{cases} \mathcal{L}_{\text{high}}(\mathcal{L}; \tau, h) & \text{if } \frac{\mathcal{L} - \mathcal{L}_{\text{bias}}(\mathcal{L}; \tau)}{\tau} \in S \\ \mathcal{L}_{\text{low}}(\mathcal{L}; \tau, h) & \text{otherwise} \end{cases} \tag{8}$$

For readability, fixed parameters such as τ, h, and S are omitted from notations like $\mathcal{D}_{\text{post}}(\mathcal{L}; \tau, h, S)$ when clear from context. The loss mapping function $\mathcal{D}_{\text{post}}$ in DLD is defined over periodic intervals of length τ, with each interval starting at a point given by Eq. (5). In practice, S is typically chosen as a union of narrow subintervals, e.g., $(\epsilon, 2\epsilon) \cup (3\epsilon, 4\epsilon) \cup \cdots \cup (1-\epsilon, 1)$, where ϵ is a small positive constant. As illustrated in Fig. 3, the image of $\mathcal{D}_{\text{post}}$ resembles a staircase-like pattern formed by dashed "less-than" symbols. The upper half of each symbol corresponds to $\mathcal{L}_{\text{high}}$, and the lower half corresponds to $\mathcal{L}_{\text{low}}$.

The implementation of $\mathcal{D}_{\text{post}}$ is straightforward. After the model produces the output scores, a lightweight post-processing module modifies them in a label-preserving manner: it adjusts the difference between the top two scores—that is, the margin loss—to match the output of $\mathcal{D}_{\text{post}}$. This incurs negligible computational overhead. Since the predicted label is not changed, DLD does not modify the decision boundary and therefore lies outside the scope of decision-based (label-only) attacks (e.g., Cheng et al. (2019); Chen et al. (2020b)); in such settings, DLD neither helps nor harms defense.

Note that $|\mathcal{L} - \mathcal{D}_{\text{post}}(\mathcal{L})| \leq \tau$, which bounds the impact on the model's prediction confidence. In our implementation, we modify only the score of the highest-scoring class while keeping all other scores unchanged,

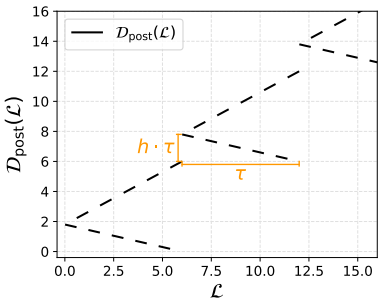


Figure 3: $\mathcal{D}_{\text{post}}$ of (τ, h, S)-DLD with $\tau = 8$, $h = 0.3$, and $S = (0.1, 0.2) \cup (0.3, 0.4) \cup \cdots \cup (0.9, 1)$. Although the image may appear as dashed lines, it is actually composed of multiple solid lines.

which suffices to realize the desired margin transformation. As a result, under this implementation, if a model f is (δ, ϵ)-globally-robust, then its DLD-defended version remains $(\delta, \epsilon + 2\tau)$-globally-robust. Increasing τ may enhance defense but could also introduce greater distortion to the model's outputs.

Given τ, h, and S, the function $\mathcal{D}_{\text{post}}$ defined by DLD introduces ambiguity in the relationship between observed loss and the quality of AE candidates. As illustrated in Figure 3, each "<" interval consists of two branches: if a sample's true loss maps to the upper branch ($\mathcal{L}_{\text{high}}$), a better sample (smaller true loss) yields a smaller observed loss; but if it maps to the lower branch ($\mathcal{L}_{\text{low}}$), a better sample produces a larger observed loss. Since the attacker cannot determine whether a sample corresponds to $\mathcal{L}_{\text{high}}$ or $\mathcal{L}_{\text{low}}$, deciding whether to accept a higher or lower loss becomes unreliable. Formally, $\mathcal{D}_{\text{post}}$ satisfies:

- For any $\mathcal{L}_1, \mathcal{L}_2$ with $\mathcal{L}_{\text{bias}}(\mathcal{L}_1) = \mathcal{L}_{\text{bias}}(\mathcal{L}_2)$, we have $\mathcal{L}_{\text{low}}(\mathcal{L}_1) \leq \mathcal{L}_{\text{high}}(\mathcal{L}_2)$.
- If $\mathcal{L}_{\text{bias}}(\mathcal{L}_1) = \mathcal{L}_{\text{bias}}(\mathcal{L}_2)$, $\mathcal{L}_1 < \mathcal{L}_2$, and $\mathcal{D}_{\text{post}}(\mathcal{L}_2) = \mathcal{L}_{\text{low}}(\mathcal{L}_2)$, then $\mathcal{D}_{\text{post}}(\mathcal{L}_1) > \mathcal{D}_{\text{post}}(\mathcal{L}_2)$.
- If $\mathcal{L}_{\text{bias}}(\mathcal{L}_1) = \mathcal{L}_{\text{bias}}(\mathcal{L}_2)$, $\mathcal{L}_1 < \mathcal{L}_2$, and $\mathcal{D}_{\text{post}}(\mathcal{L}_2) = \mathcal{L}_{\text{high}}(\mathcal{L}_2)$, then $\mathcal{D}_{\text{post}}(\mathcal{L}_1) < \mathcal{D}_{\text{post}}(\mathcal{L}_2)$.

These properties ensure discontinuities that simultaneously mislead both minimization and maximization.

During minimization, suppose the attacker holds a current best sample $\mathbf{x}^k$ with true loss $\mathcal{L}^k$, where $\mathcal{D}_{\text{post}}(\mathcal{L}^k) = \mathcal{L}_{\text{high}}(\mathcal{L}^k)$. If a new candidate $\mathbf{x}^{\text{new}}$ has a smaller true loss $\mathcal{L}^{\text{new}} < \mathcal{L}^k$ but maps to $\mathcal{L}_{\text{low}}(\mathcal{L}^{\text{new}})$, it will be accepted, yet the second property ensures that any smaller true loss in the same "<" interval will be assigned a higher observed loss. This creates a local "trap" that misleads further minimization.

The attacker can avoid the trap only in two ways: (1) generating a much better $\mathbf{x}^{\text{new}}$ with significantly lower loss to jump into the next interval, or (2) landing in regions where $\mathcal{D}_{\text{post}}(\mathcal{L}^{\text{new}}) = \mathcal{L}_{\text{high}}(\mathcal{L}^{\text{new}})$. The first is rare, while the second is also difficult, with only about a 1/2 chance per trial for typical S, and an overall probability that becomes very small over successive iterations. Eventually, with high probability, the observed loss stabilizes near $\mathcal{L}_{\text{bias}}(\mathcal{L}^k)$. Switching to maximization merely flips the roles of $\mathcal{L}_{\text{high}}$ and $\mathcal{L}_{\text{low}}$, highlighting DLD's symmetric disruption of both optimization directions.

4.3 Theoretical Analysis of DLD

To formally understand the defensive behavior of DLD, we consider the following three assumptions to characterize the properties of the model, the clean input data, and the attacker's optimizer, respectively.

Assumption 4.1. The victim model f is $(r, \epsilon/2)$-globally-robust within the region $\mathcal{N}(\mathbf{x}_0, \epsilon_{\text{n}})$.

Assumption 4.2. $\mathcal{L}_{\text{real}}(\mathbf{x}_0) \geq \tau$.

Assumption 4.3. For all $\mathbf{x}$, $\|G_{(\mathbf{x}_0,\epsilon_{\text{n}})}(\mathbf{x}) - \mathbf{x}\| \leq r$.

For theoretical tractability, we introduce a randomized version of DLD. This variant approximates the deterministic (τ, h, S)-DLD while allowing us to model $\mathcal{D}_{\text{post}}$ outputs as i.i.d. Bernoulli trials.

Definition 4.2. A (τ, h, p)-random-DLD is a post-processing defense module where $\mathcal{D}_{\text{post}}(\mathcal{L})$ is randomly chosen as follows:

$$\mathrm{P}\big(\mathcal{D}_{\text{post}}(\mathcal{L}) = \mathcal{L}_{\text{high}}(\mathcal{L})\big) = p,$$
$$\mathrm{P}\big(\mathcal{D}_{\text{post}}(\mathcal{L}) = \mathcal{L}_{\text{low}}(\mathcal{L})\big) = 1 - p,$$

where $\mathcal{L}_{\text{high}}$ and $\mathcal{L}_{\text{low}}$ are defined in Eqs. (6) and (7), respectively.

Note that (τ, h, p)-random-DLD and (τ, h, S)-DLD share the same good properties and are very similar in practice; only the random version is easier to bypass, and is therefore introduced solely for theoretical analysis (see Appendix A for details).

We now evaluate DLD's robustness against standard score-based black-box attacks. The following theorem considers a strong adversary with an *infinite query budget* using the standard-SQA procedure under the margin loss defined in Eq. (1), and shows that the success probability remains exponentially small. The proof is in Appendix B.

Theorem 4.1. Under assumptions 4.1, 4.2, and 4.3, the success probability of standard-SQA$(\mathbf{x}_0, G_{(\mathbf{x}_0,\epsilon_{\text{n}})})$ against a (τ, h, p)-random-DLD defended f with infinite queries is at most

$$p^{\left\lfloor \frac{\tau-\epsilon}{\epsilon} \right\rfloor}.$$

This confirms the earlier intuition: even with unlimited queries, breaking DLD via standard-SQA is exponentially unlikely. A larger τ further increases the attack difficulty.

Remark 4.1. Assumption 4.1 is reasonable, as prior work (Katz et al., 2017a,b) provided a method to verify it. By contrast, the assumptions used in RND (Qin et al., 2021b)—requiring the model to be Lipschitz continuous and its derivative to also be Lipschitz continuous—are overly restrictive in practice. Even then, their analysis yields little insight: lower bounds were needed, but only upper bounds were obtained.

The next theorem quantifies how DLD disrupts reverse-tactic-SQA under the same margin loss defined in Eq. (1) by forcing frequent direction switches. The proof is in Appendix B.

Theorem 4.2. Under assumptions 4.1, 4.2, and 4.3, the expected number of reversals (i.e., the binary flag *reverse* flips from `True` to `False`, or vice versa) in reverse-tactic-SQA$(\mathbf{x}_0, G_{(\mathbf{x}_0,\epsilon_{\text{n}})}, t)$ against (τ, h, p)-random-DLD defended model f is at least

$$2\left\lfloor \frac{\tau - 2\epsilon}{\epsilon} \right\rfloor \cdot p(1-p).$$

This result highlights the inherent limitation of Algorithm 2 under DLD. Empirically (see Section 5.2), the attack's effectiveness is even lower than this bound, since the proof assumes an optimally capable attacker.

4.4 DLD under Adaptive Attack

In the presence of defensive mechanisms like DLD, attackers should take into account the potential influence of the defense when designing their strategy. A practical heuristic approach is to occasionally accept worse candidates in order to escape local traps during the optimization process, rather than strictly following the descent direction. This idea is illustrated in Algorithm 3, where the function *Prob* determines the acceptance probability of a non-improving candidate in each iteration. This reflects an adaptive mindset that accounts for defensive interference.

In this paper, we, as defenders, also adopt an adaptive mindset, assuming that attackers may employ heuristic tactics. Specifically, we adopt a simulated annealing–style acceptance strategy as a representative tactic. This method makes minimal changes to a standard SQA while adding the flexibility needed to escape local traps. As shown in Section 5.2, even under this adaptive tactic—where DLD exhibits its worst-case performance—DLD remains as effective as the state-of-the-art AAA.

Algorithm 3: randomized-tactic-SQA($\mathbf{x}_0, G, Prob$)

Input: clean sample $\mathbf{x}_0$, sample generator G, probability generator $Prob$
Output: AE candidate $\mathbf{x}$
$y_0 \leftarrow f(\mathbf{x}_0)$, $\mathbf{x} \leftarrow \mathbf{x}_0$;
while $\mathcal{L}(\mathbf{x}, y_0) > 0$ ***and*** *not out of query budget* **do**
  $\mathbf{x}_{\text{new}} \leftarrow G(\mathbf{x})$;
  if $\mathcal{L}(\mathbf{x}_{new}, y_0) < \mathcal{L}(\mathbf{x}, y_0)$ **then**
    $\mathbf{x} \leftarrow \mathbf{x}_{\text{new}}$;
  else
    with probability $Prob$, $\mathbf{x} \leftarrow \mathbf{x}_{\text{new}}$;
return x

5 EXPERIMENTS

We assess the effectiveness of DLD on ImageNet against adaptive attacks and compare it with existing plug-and-play defenses. Our experiments demonstrate DLD's superior robustness and analyzing the impact of its key parameters. Code is available at https://github.com/fyzdalao/Dashed-Line-Defense.

5.1 Setup

We compare DLD against two plug-and-play baselines: RND, a state-of-the-art pre-processing defense that injects noise into the model input, and AAA, a post-processing defense previously described. For RND, we adopt two noise levels, $\nu = 0.01$ and $\nu = 0.02$, following its original settings. For AAA, we implement both its linear and sinusoidal variants with original default parameters ($\tau = 6$, $\alpha = 0.7$). For DLD, we evaluate two variants: the deterministic (τ, h, S)-DLD, where $S = (0, 0.02] \cup (0.04, 0.06] \cup \cdots \cup (0.96, 0.98]$, and the randomized (τ, h, p)-random-DLD with $p = |S| = 0.5$. Both use $\tau = 6$ and $h = 0.3$.

In line with prior work, the evaluation uses 1,000 correctly classified ImageNet samples, one randomly selected from each class. We use PyTorch-pretrained WideResNet-50, RegNet-Y 1.6GF, and MaxViT-T, which we observed to exhibit increasing robustness in this order. AEs are generated by Square Attack (Andriushchenko et al., 2020) with an ℓ_∞ noise budget $\epsilon_{\text{n}} = 0.05$ and query budget $n = 2500$.

In this work, we exclusively use Square Attack for generating AEs, as the recent benchmark BlackboxBench (Zheng et al., 2025) identifies it as the strongest black-box SQA available, significantly outperforming other methods. Echoing this, we observed in initial tests that commonly used alternatives such as Bandits Attack (Ilyas et al., 2019) and Sign Hunter (Al-Dujaili and O'Reilly, 2020) were not strong enough to yield meaningful experimental results in our setting.

To evaluate robustness against adaptive tactics, we modify Square Attack in four ways. "Standard" uses the original update rule (Algorithm 1); "reverse" flips the update direction (Algorithm 2); "explore" introduces fixed-probability stochastic exploration (Algorithm 3); and "SA" applies simulated annealing with a time-decaying acceptance probability (Algorithm 3); see Appendix C for settings.

5.2 Numerical Results

Table 1 presents the test accuracy of various defense methods under different attack tactics. Each row corresponds to an attack tactic, and each column represents a defense method applied to a given model. For each defense, the lowest accuracy across all attack tactics is underlined, reflecting the worst-case performance under adaptive attacks.

We first compare the deterministic $(6, 0.3, S)$-DLD with the randomized $(6, 0.3, 0.5)$-random-DLD, shown in the two rightmost columns of Table 1. Across all attack tactics, the two variants achieve nearly identical performance, suggesting similar underlying mechanisms: in the randomized version, the probability that $\mathcal{D}_{\text{post}} = \mathcal{L}_{\text{high}}$ is explicitly set to p, whereas in the deterministic version this probability is not strictly p but is approximately so in practice. Therefore, theoretical conclusions derived for the randomized version also apply to its deterministic counterpart.

Then, from a game-theoretic perspective, the interaction between attack tactics and defense methods can be viewed as a two-player game, where the underlined accuracy in the DLD column corresponds to the global Nash equilibrium—i.e., the result when both attacker and defender adopt optimal strategies. The fact that this equilibrium lies with DLD, with its underlined value substantially exceeding those of the baselines, indicates that DLD provides stronger defense against adaptive attacks.

Appendix D presents the attack success rate (ASR) trends during attack iterations, and Appendix E provides results with ℓ_2-bounded noise budgets.

5.3 Hyper-Parameters of DLD

The choice of S controls whether $\mathcal{D}_{\text{post}}$ returns $\mathcal{L}_{\text{high}}$ or $\mathcal{L}_{\text{low}}$. We define S as a union of small intervals within $(0, 1)$: $S = \bigcup_k ((k - ratio) \cdot step,\ k \cdot step) \cap (0, 1)$, where $step \in (0, 1]$ and $ratio \in [0, 1]$. When $1/step$ is an integer, this simplifies to $S = \bigcup_k ((k - ratio) \cdot step,\ k \cdot step)$ with $|S| = ratio$.

Table 1: The test accuracy under attacks and defenses (%)

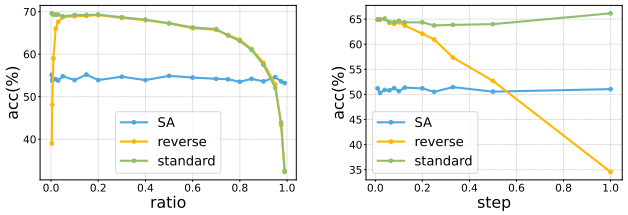
Model	Attack		Defense						
	generator G	tactic	None	RND		AAA		DLD	
				0.01	0.02	sine	linear	rand	determine
WideResNet	None		100	97.4	95.8	100	100	100	100
	Square Attack	standard	1.6	46.4	53.3	52.4	73.4	70.2	72.3
		explore	36.4	52.8	53.1	55.8	70.8	64.8	64.8
		SA	58.1	56.7	56.2	55.9	61.9	61.3	60.9
		reverse	5.9	55.5	55.7	36.8	22.2	71.9	71.9
RegNet-Y	None		100	97.7	95.7	100	100	100	100
	Square Attack	standard	3.1	49.8	53.8	61.7	79.9	76.9	76.4
		explore	32.4	56.7	56.3	62.1	76.7	69.2	69.4
		SA	61.5	62.5	58.4	62.6	68.5	66.6	66.3
		reverse	6.9	57.6	58.2	41.5	17.5	78.6	77.7
MaxViT-T	None		100	98.6	97.5	100	100	100	100
	Square Attack	standard	15.1	71.3	73.0	70.5	91.9	90.3	91.0
		explore	56.7	76.1	73.6	78.7	89.6	84.6	84.2
		SA	78.0	78.4	76.1	77.3	82.2	82.2	81.4
		reverse	39.1	77.0	76.3	56.3	40.6	90.8	90.9

A smaller $|S|$ improves defense by increasing loss ambiguity (Theorem 4.1) but risks making $\mathcal{D}_{\text{post}}$ too continuous, aiding Algorithm 2. Conversely, larger $|S|$ better preserves the model output but weakens defense due to reduced distortion. We evaluate this trade-off by running Square Attack on $(6, 0.3, S)$-DLD with *step* $= 0.04$ and *ratio* from 0.005 to 0.99, using a larger noise budget $\epsilon_{\text{n}} = 0.07$ to amplify effects. Figure 4a shows that defense performance remains strong at large *ratio*, degrading only when *ratio* > 0.9. At the other end, the reverse tactic is effective only when *ratio* < 0.015. These results suggest that while *ratio* $= 0.5$ is a safe default, DLD remains effective across a broad *ratio* range, enabling flexible trade-offs between robustness and output fidelity.

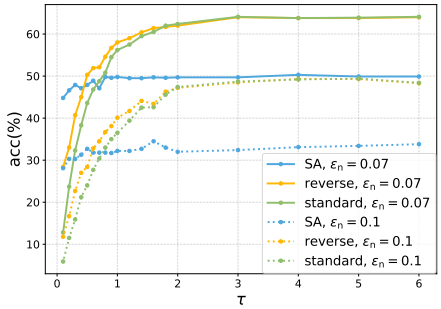
We also assess the effect of *step* by fixing *ratio* $= 0.5$ and varying *step* from 0.01 to 1 (Figure 4b). As *step* increases, DLD increasingly resembles AAA, resulting in a minor performance gain against the standard attack but a significant degradation under the reverse tactic. Nonetheless, with moderate *step*, performance under reverse tactic remains above that of the SA tactic, leaving the attack-defense Nash equilibrium unchanged.

Figure 4 shows that variations in *ratio* and *step* have little impact on performance under SA tactic. Note that both lower *ratio* and larger *step* make DLD more like AAA-linear. Thus, DLD and AAA-linear should exhibit similar performance under SA tactic, which is indeed consistent with Section 5.2.

The choice of τ, as discussed in Section 4.2, determines the extent to which DLD alters the output scores. We vary τ while fixing *step* $= 0.04$ and *ratio* $= 0.5$ under three Square Attack variants (Figure 4c), using $n = 2500$ and $\epsilon_{\text{n}} \in \{0.07, 0.1\}$. Smaller τ introduces less output distortion but weakens defense, especially when the noise budget ϵ_{n} is large. In contrast, larger τ consistently strengthens defense without abrupt drops, making τ easy to tune: when exact prediction scores are less critical, a larger τ can be safely chosen to enhance protection.



(a) Accuracy vs. ratio (b) Accuracy vs. step



(c) Accuracy vs. τ

Figure 4: Accuracy under Square Attack with different DLD parameters.

6 CONCLUSION

In this work, we revealed that the current state-of-the-art runtime defense against SQA remains vulnerable to adaptive tactics. To address this gap, we introduced Dashed Line Defense (DLD), a plug-and-play post-processing method with a non-continuous loss mapping to confuse attack optimization. Our theoretical analysis and ImageNet experiments demonstrate that DLD provides strong defense and outperforms prior approaches even under worst-case adaptive scenarios. We believe this work highlights the need for defenses that explicitly account for attacker adaptivity, while also offering insights for developing more adaptive and diverse attack tactics, inspiring future research on both sides of the attack–defense landscape.

References

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015. URL `https://arxiv.org/abs/1412.6572`.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Chengyu Qin, Xiaoyu Wang, Xinyang Zhang, Huan Yu, and Pin-Yu Chen. Random weight noise for improved robustness: A case study in visual classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 6414–6424, 2021a.

Zeyu Qin, Yanbo Fan, Hongyuan Zha, and Baoyuan Wu. Random noise defense against query-based black-box attacks. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:7650–7663, 2021b.

Sizhe Chen, Zhehao Huang, Qinghua Tao, Yingwen Wu, Cihang Xie, and Xiaolin Huang. Adversarial attack on attackers: Post-process to mitigate black-box score-based query attacks. *Advances in Neural Information Processing Systems (NeurIPS)*, 35: 14929–14943, 2022.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1633–1645, 2020.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, pages 39–57. Ieee, 2017.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM Asia Conference on Computer and Communications Security*, pages 506–519, 2017.

Maosen Li, Cheng Deng, Tengjiao Li, Junchi Yan, Xinbo Gao, and Heng Huang. Towards transferable targeted attack. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 641–649, 2020.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2730–2739, 2019a.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *ACM Workshop on Artificial Intelligence and Security*, pages 15–26, 2017.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning (ICML)*, pages 2137–2146. PMLR, 2018.

Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *European Conference on Computer Vision (ECCV)*, pages 154–169, 2018.

Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In *International Conference on Learning Representations (ICLR)*, 2019.

Abdullah Al-Dujaili and Una-May O'Reilly. Sign bits are all you need for black-box attacks. In *International Conference on Learning Representations (ICLR)*, 2020.

Chuan Guo, Jacob Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Weinberger. Simple black-box adversarial attacks. In *International Conference on Machine Learning (ICML)*, pages 2484–2493. PMLR, 2019.

Moustafa Alzantot, Yash Sharma, Supriyo Chakraborty, Huan Zhang, Cho-Jui Hsieh, and Mani B Srivastava. Genattack: Practical black-box

attacks with gradient-free optimization. In *Genetic and Evolutionary Computation Conference*, pages 1111–1119, 2019.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: A query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision (ECCV)*, pages 484–501. Springer, 2020.

Florian Tramer, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Zizheng Guo, Jun Wu, Pengyu Chen, Yanzhang Fu, and Dongjing Miao. Data debugging is NP-hard for classifiers trained with SGD. In *International Computing and Combinatorics Conference (COCOON)*, 2025.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 501–509, 2019b.

Jeremy M Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, pages 1310–1320, 2019.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, pages 274–283, 2018.

Junyoung Byun, Hyojun Go, and Changick Kim. On the effectiveness of small input noise for defending against query-based black-box attacks. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3051–3060, 2022.

Steven Chen, Nicholas Carlini, and David Wagner. Stateful detection of black-box adversarial attacks. In *ACM Workshop on Security and Privacy on Artificial Intelligence*, pages 30–39, 2020a.

Shaofei Li, Ziqi Zhang, Haomin Jia, Yao Guo, Xiangqun Chen, and Ding Li. Query provenance analysis: Efficient and robust defense against query-based black-box attacks. In *IEEE Symposium on Security and Privacy (SP)*, pages 1641–1656. IEEE, 2025.

Jeonghwan Park, Niall McLaughlin, and Ihsen Alouani. Mind the gap: Detecting black-box adversarial attacks in the making through query update analysis, 2025. URL `https://arxiv.org/abs/2503.02986`.

Chengyuan Yao, Pavol Bielik, Petar Tsankov, and Martin Vechev. Automated discovery of adaptive attacks on adversarial defenses. *Advances in Neural Information Processing Systems (NeurIPS)*, 34: 26858–26870, 2021.

Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Towards proving the adversarial robustness of deep neural networks. *Electronic Proceedings in Theoretical Computer Science*, 257:19–26, September 2017a. ISSN 2075-2180. doi: 10.4204/eptcs.257.3. URL `http://dx.doi.org/10.4204/EPTCS.257.3`.

Minhao Cheng, Huan Zhang, Cho Jui Hsieh, Thong Le, Pin Yu Chen, and Jinfeng Yi. Query-efficient hard-label black-box attack: An optimization-based approach. In *International Conference on Learning Representations (ICLR)*, 2019.

Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *IEEE Symposium on Security and Privacy (SP)*, pages 1277–1294. IEEE, 2020b.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017b.

Meixi Zheng, Xuanchen Yan, Zihao Zhu, Hongrui Chen, and Baoyuan Wu. Blackboxbench: A comprehensive benchmark of black-box adversarial attacks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [No]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [No]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix

A Motivation for Randomized DLD

To facilitate the theoretical analysis of DLD, we rely on a key property: for any input, the output of $\mathcal{D}_{\text{post}}$ equals $\mathcal{L}_{\text{high}}$ with probability p and $\mathcal{L}_{\text{low}}$ with probability $1-p$. This approximately holds for deterministic DLD if we set $|S| = p$ (see Table 2). It does not hold exactly, unless we can guarantee or reasonably assume that the margin loss (input to $\mathcal{D}_{\text{post}}$) is uniformly distributed. Unfortunately, this assumption is not realistic. Even with an idealized model input distribution, the margin loss distribution is shaped by complex model behavior, which remains poorly understood due to limited progress in deep model interpretability.

Table 2: Proportion of $\mathcal{D}_{\text{post}} = \mathcal{L}_{\text{high}}$ after applying $(6, 0.3, S)$-DLD to MaxViT-T predictions on the ImageNet validation set. S is a union of small, evenly spaced intervals in $(0, 1)$.

$\lvert S\rvert$	Proportion of $\mathcal{D}_{\text{post}} = \mathcal{L}_{\text{high}}$ (%)
0.1	9.906
0.2	19.988
0.3	29.740
0.4	39.864
0.5	49.930
0.6	59.798
0.7	69.908
0.8	80.036
0.9	90.086

To formalize this probabilistic behavior, we introduce randomized version of DLD (Definition 4.2), which explicitly enforces the desired probabilities p and $1-p$. Since both versions behave similarly in practice, we argue that theoretical results for randomized DLD can meaningfully approximate those for the deterministic version.

One might ask: why not adopt randomized DLD directly? The answer lies in a critical vulnerability—randomized DLD is susceptible to adaptive attacks. If the attacker knows the defense, they can repeatedly query the same input until two distinct outputs appear. The larger loss must correspond to $\mathcal{L}_{\text{high}}$, which is strictly increasing, and can then be minimized to generate an adversarial example.

The cost of this tactic is not prohibitive. Let X be the number of queries needed to observe two different outputs. This is a variant of the non-uniform coupon collector problem, and we can show $\mathbb{E}[X] = 1 + 1/(p(1-p))$. When $p = 0.5$, the expected number of queries is only 5. That is, the attacker needs only about $5\times$ more queries to reliably succeed, avoiding the traps introduced by DLD. Using more extreme values of p can increase this cost, but introduces new trade-offs (see Section 5.3) and still doesn't eliminate the attack vector.

Although one might mitigate this vulnerability by tricks like varying p across regions, allowing inconsistent outputs for the same input is generally undesirable. Therefore, we adopt the deterministic version of DLD as the default method in this paper.

B Proofs

B.1 Proof for Theorem 4.1

Proof. For convenience, we assume that the generator G is sufficiently strong such that for any input $\mathbf{x}$, we have $\mathcal{L}_{\text{real}}(G(\mathbf{x})) \leq \mathcal{L}_{\text{real}}(\mathbf{x})$. If this assumption does not hold, the attack would be even less effective, making the

theorem stronger.

In ecah iteration, the standard-SQA$(\mathbf{x}_0, G)$ accepts or rejects the generated example. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ denote the sequence of inputs accepted during the attack (i.e., those that lead to observed loss values no greater than their predecessors).

By Assumptions 4.1 and 4.3, we have, for all $\mathbf{x}$ and any class y,

$$|f_y(\mathbf{x}) - f_y(G(\mathbf{x}))| < \frac{\epsilon}{2}.$$

This implies that the difference in real loss between consecutive accepted examples is bounded:

$$|\mathcal{L}_{\text{real}}(\mathbf{x}) - \mathcal{L}_{\text{real}}(G(\mathbf{x}))| < \epsilon.$$

Therefore, for all $j > 0$, we have

$$|\mathcal{L}_{\text{real}}(\mathbf{x}_j) - \mathcal{L}_{\text{real}}(\mathbf{x}_{j-1})| < \epsilon.$$

If the standard-SQA ever accepts an $\mathbf{x}_j$ that $\mathcal{L}_{\text{real}}(\mathbf{x}_j) \geq \epsilon$ and $\mathcal{L}_{\text{observed}}(\mathbf{x}_j) = \mathcal{D}_{\text{post}}(\mathcal{L}_{\text{real}}(\mathbf{x}_j)) = \mathcal{L}_{\text{low}}(\mathcal{L}_{\text{real}}(\mathbf{x}_j))$, then because of the property of $\mathcal{D}_{\text{post}}$, any $G(\mathbf{x}_j)$ will meet $\mathcal{L}_{\text{observed}}(G(\mathbf{x}_j)) > \mathcal{L}_{\text{observed}}(\mathbf{x}_j)$. Thus, future iterations of the standard-SQA will not accept any examples, making $m = j$. The standard-SQA fails.

The standard-SQA will succeed if an example $\mathbf{x}_i$ is accepted such that $\mathcal{L}_{\text{real}}(\mathbf{x}_i) < \epsilon$ because the query budget is infinite. The standard-SQA will succeed only if an example $\mathbf{x}_i$ is accepted such that $\mathcal{L}_{\text{real}}(\mathbf{x}_i) < \epsilon$ because $\forall j > 0$, $|\mathcal{L}_{\text{real}}(\mathbf{x}_j) - \mathcal{L}_{\text{real}}(\mathbf{x}_{j-1})| < \epsilon$. If such an $\mathbf{x}_i$ is accepted, then for all $j \in \{1, 2, ..., i-1\}$, $\mathcal{L}_{\text{observed}}(\mathbf{x}_j) = \mathcal{L}_{\text{high}}(\mathcal{L}_{\text{real}}(\mathbf{x}_j))$.

Note $\forall j > 0$, $|\mathcal{L}_{\text{real}}(\mathbf{x}_j) - \mathcal{L}_{\text{real}}(\mathbf{x}_{j-1})| < \epsilon$, due to assumption 4.2, it will take at least $\lfloor \frac{\tau - \epsilon}{\epsilon} \rfloor + 1$ iterations for standard-SQA to obtain $\mathbf{x}_i$. Thus, $i - 1 \geq \lfloor \frac{\tau - \epsilon}{\epsilon} \rfloor$. The probability of success

$$\begin{aligned}
&\mathrm{P}\left(\bigcap_{j \in \{1,2,\ldots,i-1\}} \left(\mathcal{L}_{\text{observed}}\left(\mathbf{x}_j\right) = \mathcal{L}_{\text{high}}\left(\mathcal{L}_{\text{real}}\left(\mathbf{x}_j\right)\right)\right)\right) \\
&= \prod_{j=1}^{i-1} \mathrm{P}\left(\mathcal{L}_{\text{observed}}(\mathbf{x}_j) = \mathcal{L}_{\text{high}}(\mathcal{L}_{\text{real}}(\mathbf{x}_j))\right) \\
&\leq \prod_{j=1}^{\lfloor \frac{\tau - \epsilon}{\epsilon} \rfloor} \mathrm{P}(\mathcal{L}_{\text{observed}}(\mathbf{x}_j) = \mathcal{L}_{\text{high}}(\mathcal{L}_{\text{real}}(\mathbf{x}_j))) \\
&= \prod_{j=1}^{\lfloor \frac{\tau - \epsilon}{\epsilon} \rfloor} p \\
&= p^{\lfloor \frac{\tau - \epsilon}{\epsilon} \rfloor}.
\end{aligned}$$

□

B.2 Proof for Theorem 4.2

Proof. For convenience, we assume that the generator G is sufficiently strong such that for any input $\mathbf{x}$, we have $\mathcal{L}_{\text{real}}(G(\mathbf{x})) \leq \mathcal{L}_{\text{real}}(\mathbf{x})$. If this assumption does not hold, the attack would be even less effective, making the theorem stronger. Without loss of generality, assume that in every iteration the algorithm always accepts the newly generated example. Arrange the examples accepted by the algorithm in order of occurrence, represented as sequence $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m$.

Denote $\mathcal{L}_{\text{observed}}(\mathbf{x}_i) = \mathcal{L}_{\text{high}}(\mathcal{L}_{\text{real}}(\mathbf{x}_i))$ as event U_i. According to Definition 4.2, for $i \geq 1$, $\mathrm{P}(U_i) = p$, and $\mathrm{P}\left(\overline{U_i}\right) = 1 - p$. For $i \geq 2$, the optimization direction must be reversed in the i-th iteration when: 1) $U_i \wedge \overline{U_{i-1}}$ or 2) $\overline{U_i} \wedge U_{i-1}$.

The times of conducting direction reverse can be calculated using an indicator

$$X_i = \begin{cases} 1, & \left(U_i \wedge \overline{U_{i-1}}\right) \vee \left(\overline{U_i} \wedge U_{i-1}\right) \\ 0, & \textit{otherwise} \end{cases}, i \geq 2$$

We have

$$\begin{aligned}\mathrm{E}(X_i) &= \mathrm{E}(X_i|U_i)\mathrm{P}(U_i) + \mathrm{E}\left(X_i|\overline{U_i}\right)\mathrm{P}\left(\overline{U_i}\right)\\ &= \mathrm{P}(X_i = 1|U_i)\mathrm{P}(U_i) + \mathrm{P}\left(X_i = 1|\overline{U_i}\right)\mathrm{P}\left(\overline{U_i}\right)\\ &= \mathrm{P}\left(U_i \wedge \overline{U_{i-1}}\right) + \mathrm{P}\left(\overline{U_i} \wedge U_{i-1}\right)\\ &= 2p(1-p)\end{aligned}$$

Let $N = \lfloor \frac{\tau-\epsilon}{\epsilon} \rfloor$. According to the proof of Theorem 4.1, it will take at least $N+1$ iterations for the algorithm to succeed. The expectation of total times of conducting direction reverse in N iterations

$$\mathrm{E}\left(\sum_{i=1}^{N} X_i\right) \geq \mathrm{E}\left(\sum_{i=2}^{N} X_i\right) = \sum_{i=2}^{N} \mathrm{E}(X_i) = 2(N-1)p(1-p) = 2\lfloor\frac{\tau-2\epsilon}{\epsilon}\rfloor p(1-p).$$

□

C Parameter Settings for Adaptive Tactics

To examine the potential vulnerabilities of runtime defenses and stress-test our proposed DLD method, we construct several adaptive variants based on Square Attack.

The **standard** variant follows the original update rule used by Square Attack and similar methods, where each iteration accepts the candidate with a lower loss value. This corresponds to Algorithm 1.

The **reverse** variant, corresponding to Algorithm 2, reverses the update direction after the optimization stagnates. In our experiments, the stagnation threshold t is set to 23. This value was empirically chosen to give the attacker the greatest advantage and the defender the greatest challenge.

The **explore** variant corresponds to Algorithm 3, in which each iteration accepts a worse candidate with fixed probability. We set $Prob = 0.5$, empirically selected to be the most favorable to the attacker and the most difficult for the defense.

The **SA (simulated annealing)** variant also follows Algorithm 3, but uses a dynamic acceptance probability based on a simulated annealing schedule. The initial temperature is set to 25 and decays by a factor of 0.997 after each iteration. If no new candidate is accepted for 20 consecutive steps, the temperature is reset to 25. The acceptance probability is given by:

$$Prob = \exp\left(-\frac{\mathcal{L}_{\text{new}} - \mathcal{L}}{\texttt{tmp}}\right)$$

where $\mathcal{L}_{\text{new}}$ is the loss value of the worse candidate, $\mathcal{L}$ is the loss of the current best candidate, and `tmp` is the current temperature. These settings were also empirically chosen to favor the attacker and stress the defense.

All experiments were run on NVIDIA RTX 3090 GPUs, and other CUDA devices are expected to work as well. For methods involving randomness, we use the default seed 19260817 (a prime number) and run each experiment once.

D ASR Trend during Attacks

As a supplement to Table 1, Figure 5 shows how the attack success rate (ASR)—that is, the proportion of initially clean samples that have been successfully turned into AEs—evolves over attack iterations for different defenses. For the WideResNet-50 model, we plot the worst-case scenario for each defense: reverse tactic for AAA, standard tactic for RND ($\nu = 0.01$) and for the undefended model, and SA tactic for DLD. Note that the vertical axis represents the attacker's success rate, where lower values indicate stronger defense performance.

E Numerical Results with ℓ_2 Noise Budget

In the main text, we reported experimental results using an ℓ_∞ noise budget. Here, we present results evaluated under an ℓ_2 noise budget. We use 1,000 correctly classified ImageNet samples, randomly selecting one image

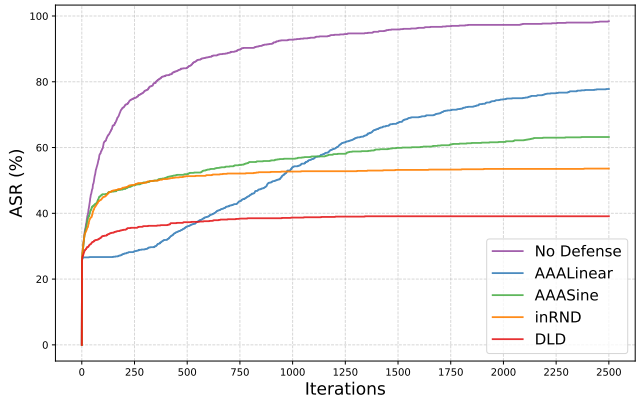


Figure 5: Attack success rate (ASR) over attack iterations.

Table 3: The test accuracy under attacks and defenses (%)

Model	Attack		Defense						
	generator G	tactic	None	RND 0.01	RND 0.02	AAA sine	AAA linear	DLD rand	DLD determine
WideResNet	None		100	97.4	95.8	100	100	100	100
	Square Attack	standard	8.1	46.8	57.1	70.7	87.4	84.6	85.9
		explore	50.6	59.5	62.1	69.3	81.8	72.2	73.2
		SA	69.8	68	66.6	67.8	71.5	70.0	70.6
		reverse	16.2	60.5	64.8	50.6	26.9	85.6	85.9
RegNet-Y	None		100	97.7	95.7	100	100	100	100
	Square Attack	standard	8.7	54.6	56.0	70.7	90.1	85.9	87.6
		explore	54.9	64.4	61.8	71.6	84.2	79.1	77.8
		SA	74.0	69.2	65.9	74.4	76.2	76.1	75.1
		reverse	15.5	64.2	64.5	51.9	27.3	87.0	87.7
MaxViT-T	None		100	98.6	97.5	100	100	100	100
	Square Attack	standard	18.5	60.9	65.5	80.6	92.2	89.2	90.5
		explore	70.0	73.7	71.6	79.9	88.5	83.6	83.3
		SA	81.6	79.3	74.7	80.2	83.7	81.7	81.5
		reverse	37.3	71.6	72.3	65.3	43.9	90.7	90.4

from each class. The experiments are conducted with the PyTorch-pretrained WideResNet-50, RegNet-Y 1.6GF, and MaxViT-T models. AEs are generated using Square Attack with an ℓ_2 noise budget of $\epsilon_{\text{n}} = 10$ and a query budget of $n = 1500$. The under-attack accuracy of different defenses is summarized in Table 3. For each defense, the lowest accuracy across all attack tactics is underlined to indicate the worst-case performance under adaptive attacks. Similar to the observations in Section 5.2, DLD achieves the best worst-case performance among all evaluated defenses.