The Tale of Evil Twins: Adversarial Inputs versus Backdoored Models

Ren Pang*  Xinyang Zhang*  Shouling Ji*  Yevgeniy Vorobeychik†
Xiapu Luo‡  Ting Wang*
*Pennsylvania State University  *Zhejiang University  †Washington University in St. Louis  ‡Hong Kong Polytechnic University

Abstract

Despite their tremendous success in a wide range of applications, deep neural network (DNN) models are inherently vulnerable to two types of malicious manipulations: adversarial inputs, which are crafted samples that deceive target DNNs, and backdoored models, which are forged DNNs that misbehave on trigger-embedded inputs. While prior work has intensively studied the two attack vectors in parallel, there is still a lack of understanding about their fundamental connection, which is critical for assessing the holistic vulnerability of DNNs deployed in realistic settings.

In this paper, we bridge this gap by conducting the first systematic study of the two attack vectors within a unified framework. More specifically, (i) we develop a new attack model that integrates both adversarial inputs and backdoored models; (ii) with both analytical and empirical evidence, we reveal that there exists an intricate “mutual reinforcement” effect between the two attack vectors; (iii) we demonstrate that this effect enables a large spectrum for the adversary to optimize the attack strategies, such as maximizing attack evasiveness with respect to various defenses and designing trigger patterns satisfying multiple desiderata; (iv) finally, we discuss potential countermeasures against this unified attack and their technical challenges, which lead to several promising research directions.

1 Introduction

The abrupt advances in deep learning [23] have led to breakthroughs in a number of long-standing machine learning tasks (e.g., image classification [11], natural language processing [36], and even playing Go [38]), enabling scenarios previously considered strictly experimental. However, it is well known that deep neural networks (DNNs) are inherently vulnerable to adversarial manipulations, which significantly hinders their use in security-critical domains.

As illustrated in Figure 1, two primary attack vectors have been considered in the literature. (i) Adversarial inputs – through modifying a benign input $x$, the adversary crafts an adversarial copy $x^*$ which deceives the target DNN $f$ during inference [6, 14, 34, 41]. (ii) Backdoored models – through polluting the training data, the adversary embeds a malicious backdoor in $f$ during training, which causes this backdoored model $f_s$ to misbehave on one (or more) predefined input $x$ [18, 19, 37, 40]. There are also certain attacks [15, 27] that leverage the two attack vectors simultaneously: the adversary modifies $f$ to be sensitive to trigger patterns (e.g., specific watermarks) during training and then generates inputs with such patterns during inference to cause the backdoored model $f_s$ to malfunction.

Despite their apparent difference, the two attack vectors share the common objective of forcing target DNNs to misbehave on predefined inputs. Thus both can result in consequential damages. For example, autonomous vehicles can be misled to crashing [45]; video surveillance can be maneuvered to miss illegal activities [8]; phishing pages can bypass web content filtering [24]; while biometric authentication can be manipulated to allow improper access [3].

Figure 1: Illustration of the “duality” of adversarial inputs and backdoored models.

Existing work has intensively studied the two attack vectors
in parallel; yet, there is still a lack of holistic understanding about their connection. First, it is unclear how the two vectors are inherently related and what the vulnerability to one vector may imply for the other. Revealing such connection is essential for building defenses effective against both attacks. Further, the adversary may leverage the two vectors simultaneously, or colluding adversaries may perform coordinated attacks. It is unclear how the two vectors may potentially interact with each other and how their interaction may influence the attack dynamics. Understanding such interaction is critical for building defenses effective against integrated or coordinated attacks. Finally, studying the two vectors within a unified framework is essential for understanding the holistic vulnerability of DNNs deployed in practical settings, wherein multiple, varied attacks may be launched simultaneously.

More specifically, we seek to answer the following set of research questions.

- **RQ1** – What is the fundamental connection between adversarial inputs and backdoored models?
- **RQ2** – What is the potential interaction between the two vectors if they are applied together?
- **RQ3** – What is the impact of external factors (e.g., model and data complexity) on such interaction?
- **RQ4** – What is the adversary’s optimal strategy if she is able to leverage both vectors simultaneously?
- **RQ5** – What is the possible mitigation to defend against such integrated attacks?

**Our Work**

This work represents a solid step towards answering these key questions. We cast adversarial inputs and backdoored models into a unified attack framework, and conduct both analytical and empirical studies of their fundamental connection and their implication for the vulnerability of DNNs, which leads to a set of interesting findings.

**A1** – We develop an attack model that integrates both attack vectors. Within this unified framework, we show that there exists a “duality” relationship between the two vectors. Specifically, to achieve the same objective (i.e., misclassification of predefined inputs), one modifies inputs while the other modifies DNN models. This leads to the tradeoff between “fidelity” (i.e., whether the attack retains the original input’s perceptual quality) and “specificity” (i.e., whether the attack influences the target input only).

**A2** – We reveal that there exists an intricate “mutual-reinforcement” (MR) effect between fidelity and specificity. When launching the unified attack, by slightly sacrificing fidelity (or specificity), the adversary is able to drastically reduce the loss of specificity (or fidelity). This opens a large spectrum for the adversary to optimize the attack strategies.

**A3** – We observe that several external factors (e.g., data dimensionality and DNN model complexity) significantly influence this MR effect. For instance, the MR effect is manifested more evidently for higher dimensional data.

**A4** – We demonstrate that by leveraging this MR effect, the adversary is able to optimally balance the fidelity and specificity losses, make the attack evasive with respect to various detection mechanisms, and search for trigger patterns satisfying multiple desiderata.

**A5** – We show that to effectively defend against this unified attack, it is critical to investigate the attacks from multiple perspectives (i.e., fidelity and specificity) and carefully account for the MR effect in applying the mitigation solutions.

To our best knowledge, this work represents the first systematic study of adversarial inputs and backdoored models within a unified framework. We believe our findings deepen the holistic understanding about the vulnerability of DNNs in practical settings and shed light on developing effective countermeasures.

**Roadmap**

The remainder of the paper proceeds as follows. § 2 introduces a set of fundamental concepts; § 3 presents a unified framework that integrates adversarial inputs and backdoored models; § 4 empirically studies the fundamental connection between the two attack vectors; § 5 provides analytical justification for the empirical observations in § 4; § 6 discusses potential countermeasures against this unified attack; § 7 surveys relevant literature; the paper is concluded in § 8.

## 2 Preliminaries

In this section, we introduce a set of fundamental concepts and assumptions. The important notations used in the paper are summarized in Table 1.

### 2.1 Deep Neural Networks

Deep neural networks (DNNs) represent a class of machine learning models to learn high-level abstractions of complex data using multiple processing layers in conjunction with nonlinear transformations. In this paper, we primarily consider the predictive setting, in which a DNN \( f \) (parameterized by
\( \theta \) essentially encodes a function \( f: X \to Y \). Given an input \( x \in X \), \( f \) predicts a nominal variable \( f(x; \theta) \) ranging over a set of classes \( Y \). For instance, \( f \) classifies each image in the CIFAR10 dataset [21] into 10 pre-defined classes (e.g., “bird” and “ship”).

We consider DNNs obtained via supervised learning. To train a DNN \( f \), the training algorithm takes as input a training set \( D \), of which each instance \((x, y) \in D \subset X \times Y\) comprises an input \( x \) and its ground-truth class \( y \). The algorithm determines the best parameter configuration \( \theta \) for \( f \) via optimizing a loss function \( \ell(f(x; \theta), y) \) (e.g., the cross entropy of \( y \) and \( f(x; \theta) \)). This is typically achieved using stochastic gradient descent or its variants [50].

Once trained, during inference, \( f \) is given a testing set \( \mathcal{R} \), which does not include ground-truth class information. For each input \( x \in \mathcal{R} \), \( f \) predicts its class as \( f(x; \theta) \).

### 2.2 Threat Models

It is known that DNNs are inherently vulnerable to malicious manipulations in many ways. In particular, two primary attack vectors have been considered in the literature, namely, adversarial inputs and backdoored models.

#### Adversarial Inputs

Adversarial inputs are maliciously crafted samples to force target DNNs to misbehave during inference. An adversarial input \( x \) is typically generated by modifying (e.g., pixel perturbation [30] or spatial transformation [1]) a benign input \( x \) to change its classification to a target class \( t \) desired by the adversary. Formally, with \( r \) being the perturbation vector, the attack is represented as \( x_s = x + r \). Meanwhile, the adversary often strives to maximize \( x_s \)’s perceptual similarity to \( x \). Thus the attack can be formulated as the following optimization objective:

\[
\min_{r} \mathcal{L}_e(x + r; \theta) + \lambda \mathcal{L}_t(r) \tag{1}
\]

where \( \mathcal{L}_e \), referred to as the \emph{efficacy} loss, measures the difference (e.g., cross entropy) between the model prediction \( f(x + r; \theta) \) and the classification \( t \) desired by the adversary; \( \mathcal{L}_t \), referred to as the \emph{fidelity} loss, quantifies the perceptual similarity of \( x \) and \( x_s \) (e.g., \( L^p \) norm of \( r \)); while the hyper-parameter \( \lambda \) balances the two factors.

Different adversarial input-based attacks (“adversarial attacks” for short) differ in their solvers of Eqn (1): FGSM [14] solves Eqn (1) using one-step descent along the direction of \( \mathcal{L}_e \)’s gradient sign, PGD [30] solves Eqn (1) using a sequence of projected gradient descent on \( \mathcal{L}_e \), while C&W [6] solves Eqn (1) with iterative optimization.

#### Backdoored Models

Backdoored models are adversely forged DNNs which are embedded with hidden functions (i.e., “backdoors”) during training. A backdoored DNN is typically generated by polluting its training data [15, 37, 40] or perturbing a benign DNN [18, 27] directly. It then malfunctions on adversary-chosen inputs (i.e., “triggers”) during inference.

Most backdoored model-based attacks (“backdoor attacks” for short) keep DNN architectures intact while only modifying model parameters. Therefore, this attack can be formulated as perturbing a benign model configuration \( \theta_0 \) to a backdoored version \( \theta_s \), where \( r_{\theta} = \theta_s - \theta \) is referred to as the model perturbation vector. To generate the backdoored model \( \theta_s \), the adversary also strives to minimize the influence on other non-trigger inputs. Essentially, the adversary attempts to optimize the following objective function:

\[
\min_{r_{\theta}} \mathcal{L}_e(x, t; \theta + r_{\theta}) + \nu \mathcal{L}_s(r_{\theta}) \tag{2}
\]

where \( \mathcal{L}_s \), referred to as the \emph{specificity loss}, quantifies the difference of the benign and backdoored models on non-trigger inputs. For example, it can be measured by the average discrepancy between the predictions of \( \theta \) and \( \theta_s \), regarding non-trigger inputs in a reference set \( \mathcal{R} \). The hyper-parameter \( \nu \) balances the two losses.

The formulation above subsumes a number of backdoor attacks in the literature. For instance, StingRay [40] generates poisoning inputs by perturbing benign inputs close to \( x \) in the feature space but labeled as \( t \); PoisonFrog [37] synthesizes poisoning inputs that are close to \( x \) in the feature space but perceptually belonging to \( t \) in the input space; while MODELEUSE [18] directly perturbs the DNN parameters to minimize \( x_s \)’s distance to a representative input from \( t \) in the feature space.

#### Trojaning Attacks

The trojanning attacks (e.g., [15, 27]) leverage the two attack vectors simultaneously. During training, the adversary modifies the DNN to make it sensitive to a trigger pattern (e.g., a specific watermark) embedded in inputs, such that any inputs with this pattern tend to be misclassified to a target class by the DNN during inference.

Conceptually one may regard the trigger pattern as a \emph{universal} perturbation vector \( r \). To train the backdoored model \( \theta_s \), the adversary samples a set of inputs \( T \) from the training set \( D \) and enforces each pattern-embedded input \( x + r \) for \( x \in T \) to be misclassified to \( t \). Formally, the adversary optimizes the following objective function:

\[
\min_{r, r_{\theta}} \mathcal{L}_e(x + r; \theta + r_{\theta}) + \nu \mathcal{L}_s(r_{\theta}) \tag{3}
\]

where the specificity loss \( \mathcal{L}_s \) is defined similarly as in the case of backdoor attacks.

The existing trojanning attacks differ in their means of generating trigger patterns. For instance, in BADNNet [15], the trigger pattern is specified by the adversary; in TrojanNet [27],

\footnote{With a little abuse of notation, in the following we use \( \theta \) to refer to both a DNN and its parameter configuration.}
the adversary specifies the pattern shape (e.g., Apple logo) and determines its pixels in a preprocessing step. Both attacks then re-train the malicious DNN $\theta$, to solve Eqn (3). Note that compared with the backdoor attacks, the trojansing attacks make stronger assumptions about the adversary’s capability (i.e., modifying inputs during inference).

3 A Unified Attack Model

Despite their apparent variation, the adversarial, backdoor, and trojansing attacks share the common objective of forcing target DNNs to misclassify predefined inputs. To this end, the adversary may craft adversarial inputs, forge backdoored models, or perform both. While intensive research has been conducted on the two attack vectors in parallel, thus far little is known about their fundamental connection.

3.1 Objectives

To bridge this gap, next we study the two attack vectors within a unified framework. Intuitively, in this framework, the adversary is allowed to modify a set of predefined inputs $x \in T$ as well as the target DNN $\theta$, with the goal of forcing each perturbed input $x_t$, of $x \in T$ to be misclassified to a target class $t'$, by the backdoored DNN $\theta = \theta + r_0$.

Formally, we define a unified attack model by integrating the objectives of Eqn (1), Eqn (2), and Eqn (3):

$$\min_{(r_1), r_0} E_{x \in T} L_e(x + r_1, t; \theta + r_0) + \lambda E_{x \in T} L_f(r_1) + v L_s(r_0)$$

(4)

Here the loss terms define the adversary’s multiple attack objectives, while the hyper-parameters $\lambda$ and $v$ balance the importance of these objectives. Specifically,

- $L_e$ quantifies the difference between the model prediction and the classification desired by the adversary, which represents the attack efficacy – whether the attack successfully forces the DNN to misclassify each given input $x$ to its target class $t$.
- $L_f$ quantifies the influence of the input perturbation on each given input, which represents the attack fidelity – whether the attack faithfully retains each perturbed input $x_t$’s perceptual similarity to its benign counterpart $x$.
- $L_s$ quantifies the influence of the model perturbation on non-trigger inputs, which represents the attack specificity – whether the attack precisely directs its influence to the trigger inputs $T$ only.

Below we use $L_e$, $L_f$, and $L_s$ to denote the attack efficacy, fidelity, and specificity losses respectively.

Note that this formulation subsumes the existing adversarial, backdoor, and trojansing attacks. Specifically, (i) in the case of $\lambda \ll v$, by maximizing the attack specificity, this attack approximates an adversarial attack, which can be either a single target ($|T| = 1$) or multiple targets ($|T| > 1$); (ii) in the case of $\lambda \gg v$, by maximizing the attack fidelity, this attack approximates a backdoor attack, which can be either a single trigger ($|T| = 1$) or multiple triggers ($|T| > 1$); and (iii) in the case of all $\{r_1\}$ are fixed as $r$ and all $\{r_0\}$ are fixed as $t$, this is instantiated as a trojansing attack.

Also note that this formulation does not make any assumption regarding the adversary’s capability or resource (e.g., access to the training or inference data), while it is solely defined in terms of the adversary’s objectives.

3.2 Design Spectrum

Interestingly, the objectives above are tightly intertwined, forming a triangle structure, as illustrated in Figure 2. We have the following observations.

- It is impossible to achieve the three objectives simultaneously. For instance, to attain attack efficacy (i.e., launching a successful attack), it requires to either perturb the input (i.e., at the cost of fidelity) or modify the DNN (i.e., at the cost of specificity).
- It is feasible to achieve two out of the three objectives at the same time. For instance, it is trivial to achieve both attack efficacy and fidelity by setting $\lambda \gg v$ (i.e., it is only allowed to modify the DNN).
- With one objective fixed, it is possible to balance the other two. For instance, with fixed attack efficacy, it allows to trade between attack fidelity and specificity.

In the following, we reveal the fundamental connection between the attack vectors of adversarial inputs and backdoored models, and explore the interaction among the attack efficacy, fidelity, and specificity.

3.3 Attack Implementation

Recall that the unified attack is formulated as optimizing the objective function over both the input and model perturbation vectors $\{r_1\}$ and $r_0$. While it is often expensive to solve Eqn (4) exactly due to its non-convexity and non-linearity, numerical solutions are still possible. In our implementation, we solve
Eqn (4) by updating \( \{ r_i \} \) and \( r_0 \) in an interleaving manner. To simplify the discussion, in the following, we assume the case of a single target input \( x \) in the trigger set \( T \) (i.e., \(|T| = 1\)), while the generalization to the case of multiple targets is fairly straightforward.

The unified attack finds the optimal perturbation vectors by updating them alternatively, as illustrated in Figure 3. Specifically, let \( x^{(k)} \) and \( \theta^{(k)} \) be the perturbed input and model respectively after the \( k \)-th iteration of the update. The \((k+1)\)-th iteration comprises two steps.

In the input perturbation step, with the model \( \theta^{(k)} \) fixed, it finds the incremental input perturbation \( \Delta_\text{x} \) by optimizing the following objective:

\[
\Delta_\text{x} = \arg\min_{r_\text{x}} L_\text{x}(x^{(k)} + r_\text{x}, t; \theta^{(k)}) + \lambda L_\text{f}(x^{(k)} + r_\text{x} - x)
\]

where in the second term \((x^{(k)} + r_\text{x} - x)\) corresponds to the overall perturbation. It then updates \( x^{(k+1)} = x^{(k)} + \Delta_\text{x} \).

In the model perturbation step, with the input \( x^{(k+1)} \) fixed, it searches for the incremental model perturbation \( \Delta_\theta \) by optimizing the following objective:

\[
\Delta_\theta = \arg\min_{r_\theta} L_\text{f}(x^{(k+1)}, t; \theta^{(k)} + r_\theta) + \nu L_\text{s}(\theta^{(k)} + r_\theta - \theta)
\]

It then updates \( \theta^{(k+1)} = \theta^{(k)} + \Delta_\theta \).

The complete algorithm is sketched in Algorithm 1.

Note that by their definitions, the fidelity loss \( L_\text{f} \) and specificity loss \( L_\text{s} \) only depend on the input and model perturbation vectors. It is thus possible to balance \( L_\text{f} \) and \( L_\text{s} \) by properly setting \( \lambda \) and \( \nu \).

### 3.4 Analysis

Next we provide analytical justification for Algorithm 1. Consider the attack objective in Eqn (4). The optimization theory [5] indicates that specifying the hyper-parameters \( \lambda \) and \( \nu \) (which balance different losses) amounts to specifying the perturbation bounds \( \varepsilon \) and \( \delta \) on the input and model perturbation vectors \( r_\text{x} \) and \( r_\theta \) respectively. We thus introduce the concept of feasible sets.

**Definition 1 (Feasible Set).** Let \( F_\text{x}(x) \) be the feasible set with respect to a given input \( x \), such that any \( r_\text{x} \in F_\text{x}(x) \) satisfies the constraint of \( L_\text{f}(r_\text{x}) \leq \varepsilon \). Similarly, let \( F_\theta(\theta) \) be the feasible set with respect to a given DNN \( \theta \), such that any \( r_\theta \in F_\theta(\theta) \) satisfies the constraint of \( L_\text{s}(r_\theta) \leq \delta \).

In the following, when the context is clear, for given \( x, \theta \), and \( t \), we use the following short notation for the efficacy loss:

\[
L(r_\text{x}, r_\theta) \triangleq L(x + r_\text{x}, t; \theta + r_\theta)
\]

Then the unified attack in Eqn (4) can be reformulated as:

\[
\min_{r_\text{x} \in F_\text{x}(x), r_\theta \in F_\theta(\theta)} L(r_\text{x}, r_\theta)
\]

To implement Algorithm 1, solving Eqn (6) alternates between (i) input perturbation – it searches for \( r_\text{x}^* = \arg\min_{r_\text{x} \in F_\text{x}(x)} L(r_\text{x}, r_\theta) \) and (ii) model perturbation – it finds \( r_\theta^* = \arg\min_{r_\theta \in F_\theta(\theta)} L(r_\text{x}^*, r_\theta) \). We now show that this implementation effectively solves Eqn (6) (proof deferred to Appendix A1).

**Proposition 1.** Let \( r_\text{x}^* \in F_\text{x}(x) \) be a minimizer of the function \( \min_{r_\text{x}} L(r_\text{x}, r_\theta) \). If \( r_\text{x}^* \) is non-zero, then \( \nabla_{r_\text{x}} L(r_\text{x}^*, r_\theta) \) is a proper descent direction for \( \min_{r_\text{x} \in F_\text{x}(x)} L(r_\text{x}, r_\theta) \).

Therefore, we conclude that Algorithm 1 is an effective implementation of the unified attack.

### 4 An Empirical Study

Next we conduct an empirical study of the inherent connection between adversarial inputs and backdoored models, and discuss its implication for the adversary’s strategies. We begin with introducing the setting of our study.

#### 4.1 Study Setting

**Datasets and DNNs**

To factor out the influence of specific models or datasets, we consider a range of datasets and DNNs.
We primarily use two datasets: (i) CIFAR10 [21], which consists of 32 × 32 color images drawn from 10 classes (e.g., ‘airplane’); (ii) ImageNet [11] dog-vs-fish, which consists of 256 × 256 color images drawn from 2 classes (‘dog’ and ‘fish’), with the setting identical to [20]. All the images are normalized to [−1, 1].

For CIFAR10, we apply VGG16 and VGG13 [39] as the reference DNNs, which respectively attain 88.5% and 89.0% accuracy on the testing set; for ImageNet, we use ResNet101 and ResNet50 [17] as the reference DNNs, which respectively achieve 98.5% and 99.0% accuracy on the testing set.

To implement the unified attack, we instantiate the input and model optimizers (line 3 and 5 in Algorithm 1) with the update steps in concrete adversarial and backdoor attacks. For example, with PGD and StingRay as the underlying input and model optimizers, at the k-th iteration, we compute the incremental input perturbation as:

\[ \Delta_k = \Pi_{B(x)}(x^k) - \alpha \text{sgn}(\nabla_c(x^k; \theta^{(k)}) - x^k) \]

where \( \Pi \) is the projection operator, \( B(x) \) is the allowed set of inputs (e.g., \( \{x_i \mid \|x_i - x\| \leq \epsilon\} \)), and \( \alpha \) is the learning rate. We then update the model \( \theta^{(k)} \) by performing a re-training step with poisoning inputs generated using StingRay based on \( x^{k+1} = x^k + \Delta_k \). In the following, we use \( \mathcal{A}_t\mathcal{A}_m \) to denote a unified attack that uses \( \mathcal{A}_t \) and \( \mathcal{A}_m \) as its input and model optimizers respectively.

Figure 4 shows a set of sample inputs and their decision boundaries (each in a random 2D subspace surrounding the inputs) under various attacks on ImageNet. Observe that the PGD-StingRay attack attains the same attack efficacy but with less input and model distortion compared with the adversarial (PGD) and backdoor (StingRay) attacks (more samples are shown in Appendix C1).

### Loss Measures

We quantify the fidelity loss by the \( L^p \) norm of the input perturbation vector \( r; \) \( \mathcal{L}_f(r) \triangleq \|r\|_p \) (\( p = \infty \) for PGD and 2 for C&W). Further, we measure the specificity loss using the difference of the benign and backdoored models on classifying a reference set \( \mathcal{R} \). Formally, \( \mathcal{L}_s(r) \triangleq \sum_{x \in \mathcal{R}} I(f(x; \theta) \neq f(x; \theta + r)) / |\mathcal{R}| \) (7)

where \( I \) denotes an indicator function, which returns 1 if \( z \) is true and 0 otherwise. Finally, we measure the attack efficacy as the misclassification confidence, \( f_i(x + r; \theta + r_0) \), which is the probability that the adversarial input \( (x + r) \) belongs to the target class \( t \) as predicted by the backdoored DNN \( (\theta + r_0) \). We consider an attack successful if the misclassification confidence exceeds a threshold \( \kappa \).

### 4.2 Mutual Reinforcement Effect

Now we dive into the fundamental connection between adversarial inputs and backdoored models. At a high level, we reveal that there exists an intricate “mutual reinforcement” (MR) effect between them: by leveraging the two attack vectors simultaneously, it is possible to (i) trade for fidelity (or specificity) with a disproportionately small amount of specificity (or fidelity), and (ii) attain attack efficacy beyond using each attack vector alone.

#### Fidelity-Specificity Tradeoff

In the first set of experiments, we demonstrate that with slight cost of fidelity, it is feasible to significantly reduce the specificity loss, and vice versa.
For each dataset, we apply the adversarial, backdoor, and unified attacks over 1,000 inputs randomly sampled from the testing set, and use the rest as the reference set $R_\mathcal{C}$ to evaluate the specificity loss. For each input, we randomly select its target class $t$ and fix the misclassification confidence ($\kappa$ = 0.5 or 0.9). For the unified attack, by varying the hyper-parameters (i.e., $\lambda$ and $\nu$), we control the relative importance of fidelity and specificity losses.

We then measure the fidelity and specificity losses for the cases of successful attacks. Specifically, for each input $x$ and DNN $\theta$, let $r_x$, $\bar{r}_x$, and $(r_{\mathcal{A}}^x, r_{\mathcal{B}}^x)$ be the perturbation vectors generated by the adversarial, backdoor, and unified attacks respectively. Further, we define the relative fidelity and specificity losses as: $L_f(r_{\mathcal{A}}^x)/L_f(\bar{r}_x)$ and $L_s(r_{\mathcal{B}}^x)/L_s(\bar{r}_x)$, which quantify the losses of the unified attack with respect to the adversarial (with only fidelity loss) and backdoor (with only specificity loss) attacks. Note that the relative losses are bound to the interval of [0, 1] (details in § 5).

Figure 5 illustrates how the unified attack balances the fidelity and specificity losses under different settings of datasets, DNNs, and attacks. We have the following observations. First, there exists an intricate tradeoff between fidelity and specificity: with fixed attack efficacy (i.e., misclassification confidence $\kappa$), by slightly sacrificing the attack specificity, the adversary is able to significantly improve the attack fidelity (compared with that required by the adversarial attack) and vice versa. For example, in the case of C&W-PoisonFrog against VGG (Figure 5 (b)), when $\kappa$ = 0.9, as the specificity loss increases from 0 to 0.2, the fidelity loss drops by about 0.82. Second, this effect varies across different attack implementation, showing higher intensity in C&W-PoisonFrog than PGD-StingRay. Third, this effect seems insensitive to the setting of attack efficacy, which shows similar patterns under $\kappa$ = 0.5 and 0.9. The analytical explanations for these observations are deferred to § 5. We have the following conclusion.

**Observation 1**

There exists an intricate tradeoff between fidelity and specificity. At slight cost of fidelity, it is possible to significantly reduce the specificity loss, and vice versa.

**Enhancement of Attack Efficacy**

In this set of experiments, we show that leveraging the two attack vectors gives rise to higher attack efficacy compared with using each vector alone.

![Enhancement of Attack Efficacy](image)

Figure 6: Average misclassification confidence ($\kappa$) as a function of the fidelity and specificity losses. (a) PGD-StingRay against VGG on CIFAR10; (b) C&W-PoisonFrog against VGG on CIFAR10; (c) PGD-StingRay against ResNet on ImageNet; (d) C&W-PoisonFrog against ResNet on ImageNet.

We measure the attack efficacy (i.e., average misclassification confidence) attainable by the unified attack under varied fidelity and specificity losses. It is observed in Figure 6 that in all the cases the misclassification confidence grows sharply as the fidelity or specificity loss increases, while the unified attack achieves higher efficacy than the adversarial and backdoor attacks. For instance, in the case of C&W-PoisonFrog on CIFAR10 (Figure 6 (b)), the adversarial and backdoor attacks respectively attain 0.80 and 0.61 average misclassification confidence, while the unified attack steadily attains confidence close to 1. We therefore have the conclusion as:
Impact of Model Complexity

We further measure the impact of external factors on the MR effect. Here we consider the impact of DNN model complexity and defer the discussion of the data dimensionality to Appendix A5. To compare DNNs of different complexity, we apply VGG16 and VGG13 on CIFAR10, and ResNet101 and ResNet50 on ImageNet. Each pair of DNNs share similar primitive constructs (e.g., residual blocks) but use different numbers of constructs. Note that each pair of DNNs achieve similar performance on the same dataset (details in § 4.1).

We apply VGG16 and VGG13 on CIFAR10, and ResNet101 and ResNet50 on ImageNet. Each pair of DNNs share similar primitive constructs (e.g., residual blocks) but use different numbers of constructs. Note that each pair of DNNs achieve similar performance on the same dataset (details in § 4.1).

Figure 7: Impact of model complexity on fidelity-specificity tradeoff. (a) PGD-StingRay against VGG on CIFAR10; (b) C&W-PoisonFrog against VGG on CIFAR10; (c) PGD-StingRay against ResNet on ImageNet; (d) C&W-PoisonFrog against ResNet on ImageNet.

We compare the fidelity-specificity tradeoff for each pair of DNNs, where the misclassification confidence is fixed as $x = 0.75$. The measurement is plotted in Figure 7. Observe that the tradeoff shows a strong MR effect across all the cases. Further, the effect seems marginally more significant for simpler DNNs. For instance, in the case of C&W-PoisonFrog on CIFAR10 (Figure 7(b)), as the specificity loss increases from 0 to 0.1, the fidelity loss drops by 0.39 and 0.50 for VGG16 and VGG13 respectively. Therefore,

4.3 Adversary’s Optimal Strategies

The MR effect also has profound implication for the adversary’s attack strategies. Here we consider two scenarios. First, this effect allows the adversary to maximize the attack evasiveness with respect to various defenses simultaneously. Second, the effect allows the adversary to design trigger patterns with multiple desiderata in the trojaning attacks.

Optimization of Attack Evasiveness

To assess the attack evasiveness, we consider a range of representative detection methods (detectors).

Detection of Adversarial Inputs – To detect adversarial inputs, we apply feature squeezing (FS) [49], local intrinsic dimensionality (LID) [29], MagNet [31], and high-level representation guided denoiser (HGD) [25]. At a high level, FS reduces the adversary’s search space by coalescing inputs corresponding to different feature vectors into a single input, and detects adversarial inputs by comparing their predictions under the original and squeezed settings; LID detects adversarial inputs by assessing the space-filling capability of the regions surrounding each reference input, based on the distance distribution of the input to its neighbors; MagNet employs a reformer network to move adversarial inputs towards the manifold spanned by normal inputs and a detector network to differentiate normal and adversarial inputs; HGD uses the errors caused by adversarial perturbation at a DNN’s intermediate layers to train a denoiser network, which is then applied to detect adversarial inputs.

Detection of Backdoored Models – To our best knowledge, there is still a lack of effective defenses against backdoored models without access to the contaminated training data [46]. Thus the detector proposed here is interesting in its own right. We apply the concept of influence function (IF) [7, 20] to detect backdoored models. Rooted in robust statistics, the influence function is a type of Gateaux derivative, which measures the impact of an input on a model’s predictions. Formally, with respect to a given model $\theta$, the influence of up-weighting an input $x$ in the training data on the prediction of itself is given as:

$$IF(x) = \nabla_\theta \ell(x; \theta)^\top H^{-1}_\theta \nabla_\theta \ell(x; \theta)$$

where $\nabla_\theta \ell(x; \theta)$ is the gradient of the loss function with respect to $\theta$, while $H_\theta = \nabla^2_\theta \ell(x; \theta)$ is the Hessian matrix.

Recall that the backdoor attack alters the decision boundary for the trigger $x$ but with little impact on other inputs. Thus adding $x$ to the training data would significantly influence the decision boundary surrounding $x$. IF naturally measures this
effect. It first estimates the impact of up-weighting $x$ on the model parameters $\theta$, which is measured by the direct influence on the loss function $\nabla_{\theta} L(x; \theta)$ and the “resistance” from other training data $H_{\theta}^{-1}$, and then estimates the impact of changing $\theta$ on the loss evaluated on $x$. The computational implementation of this detector is given in Appendix B2.

We evaluate the effectiveness of IF to detect trigger inputs. Specifically, Figure 8 (a) plots the distribution of influence scores of trigger and non-trigger inputs in the case of applying StingRay on CIFAR10 (with other cases shown in Appendix C3). Observe that the trigger inputs tend to have much higher influence scores than non-trigger inputs, which suggests the utility of IF. Figure 8 (b) further shows the ROC of using IF to defend against the backdoor attack. Note that IF is fairly effective to detect triggers. For instance, with small false positive rate (e.g., 0.18), IF is able to detect trigger inputs (correspondingly, backdoored models) with high accuracy (e.g., 0.99).

**Detection of Unified Attacks** — We apply all the detectors above to the inputs generated by the unified attack under varying fidelity-specificity tradeoff (with misclassification confidence $\kappa$ fixed as 0.75). Figure 9 compares their attack detection rates with respect to different datasets, DNNs, and attacks. We have the following observations.

First, the success rate of detecting malicious inputs (IF) gradually increases with the specificity loss (i.e., more significant model perturbation); on the contrary, the success rate of detecting malicious inputs (FS, LID, MagNet, and HGD) drops sharply with the specificity loss (i.e., decreasing input perturbation due to the MR effect). For instance, in the case of C&W-PoisonFrog on CIFAR10 (Figure 9 (b)), as the specificity loss grows from 0 to 0.5, the detection rate of IF increases from 0 to 0.17, while the detection rates of FS, LID, and MagNet drop by 0.73, 0.39, and 0.48 respectively.

Second, across all the cases, the adversary is able to optimally balance the fidelity and specificity losses, which leads to evasiveness with respect to multiple detectors simultaneously. For instance, in the case of C&W-PoisonFrog on ImageNet (Figure 9 (d)), with the specificity loss set as 0.31, the detection rates of IF, FS, LID, and MagNet reduce to 0.29, 0.29, 0.26, and 0.0 respectively.

**Optimization of Trigger Patterns**

In the second case, we exploit the MR effect to optimize trigger patterns in trojanning attacks. In the existing trojanning attacks, trigger patterns (e.g., watermarks) are often defined in an ad hoc manner. For instance, in BADNET [15], the watermark is defined by the adversary; in TrojanNet [27]), the watermark shape (e.g., Apple logo) is predefined, while its pixel values are computed in a preprocessing step. Next we show that the unified attack framework allows to optimize trigger patterns with respect to multiple metrics.

We consider TrojanNet [27] as a concrete case. We use a watermark generated by TrojanNet as the initial trigger pattern and optimize it in terms of both opacity and size within the unified attack framework. Specifically, we introduce a mask $m$ for each input $x$. For each dimension (pixel) $i$, $m[i] = 1 - p$ ($p$ is the opacity setting) if $i$ is covered by the watermark and $m[i] = 0$ otherwise. Thus the perturbation operation is defined as $x_t = \psi(x, r; m) = x \odot (1 - m) + r \odot m$, where $\odot$ denotes element-wise product. We reformulate the trojanning attack in Eqn (3) as follows:

$$\min_{r_\theta, r} E_{x \in T} L(x, r; m; \theta + r_\theta) + \nu L_{\theta}(r_\theta)$$  \hspace{1cm} (9)$$

We compare the attack success rate (ASR) (i.e., inputs with watermarks are misclassified) and model accuracy (ACC) (i.e.,
inputs without watermarks are correctly classified) of TrojanNet and the unified attack under varying setting of watermark size and opacity, with results shown in Table 2. Observe that across all the settings, the unified attack outperforms TrojanNet in terms of both ASR and ACC, indicating that the unified attack is able to optimize the trigger patterns with respect to both metrics.

| Size | Opacity | 30% | 70% |
|------|---------|-----|-----|
| 0.1  | ASR     | 0.79| 0.88|
|      | ACC     | 0.85| 0.88|
| 0.2  | ASR     | 0.92| 0.88|
|      | ACC     | 0.90| 0.89|

Table 2. Performance of TrojanNet and unified attacks under varied settings of watermark opacity and size on CIFAR10.

Meanwhile, it is also possible to exploit the unified attack to optimize watermark opacity and size under fixed ASR and ACC. Figure 10 illustrates sample watermarks generated by TrojanNet and the unified attacks optimizing opacity and size, where the ASR and ACC of all the attacks differ by less than 0.01 and 0.02 respectively.

Figure 10: Watermarks generated by TrojanNet (a), unified attacks optimizing opacity (b) and optimizing size (c).

Overall, combining the studies above, we can conclude:

**Observation 4**

The mutual reinforcement effect allows the adversary to optimize the attack strategies with respect to multiple metrics simultaneously.

5 An Analytical Study

In this section, we provide analytical justification for the empirical observations in § 4. To facilitate the analysis, we first introduce a set of fundamental concepts and assumptions.

5.1 Fundamentals

Without loss of generality, we consider a binary classification setting (i.e., \( \mathcal{Y} = \{0, 1\} \)), with \((1 - r) \) and \( r \) being the input \( x \)’s ground-truth class and the adversary’s target class respectively. Let \( f_t(x; \theta) \) be the model \( \theta \)’s predicted probability that \( x \) belongs to the class \( r \).

**Loss Measures**

Under this setting, we quantify the set of attack objectives as follows.

**Efficacy** – The attack succeeds only if the input and model perturbation vectors \( r_x, r_\theta \) make \( f_t(x + r_x; \theta + r_\theta) > 0.5 \) i.e., the input crosses the decision boundary. We thus use \( \kappa = f_t(x; \theta) - 0.5 \) to measure the gap between \( \theta \)’s prediction and the adversary’s target prediction regarding \( x \).

**Fidelity** – We quantify the fidelity loss of the input perturbation vector \( r_x \) using its \( L^p \) norm: \( \mathcal{L}_f(r_x) = \|r_x\|_p \). For two perturbation vectors \( r_x, r_{x'} \), we say \( r_x < r_{x'} \) if \( \mathcal{L}_f(r_x) < \mathcal{L}_f(r_{x'}) \). By default, we use \( p = 2 \), while the analysis generalizes to other norms as well.

As shown in Figure 11 (a), in a successful adversarial attack, if the perturbation magnitude is small enough, we can approximate the fidelity loss as \( x \)’s distance to the decision boundary \( [32] \): \( \mathcal{L}_f(r_x) \approx \kappa / \|\nabla f_t(x; \theta)\|_2 \), where a linear approximation is applied to the loss function. In the following, we denote \( h \triangleq \mathcal{L}_f(r_x) \).

**Specificity** – In the backdoor attack, the model perturbation vector \( r_\theta \) modifies the decision boundary surrounding \( x \), as shown in Figure 11 (b). While it is difficult to precisely describe the decision boundaries encoded by DNNs \([12]\), we approximate \( x \)'s local boundary with the surface of a \( d \)-dimensional sphere, where \( d \) is the input dimensionality.

This approximation is justified as follows. First, it uses a quadratic form, which is more precise than a linear form to describe decision boundaries \([32]\). Second, it reflects the impact of the model complexity on the decision boundary. Note that the maximum possible curvature of the decision boundary is often determined by the model’s inherent complexity \([12]\). For instance, the curvature of a linear model is 0, while a one hidden-layer neural network with an infinite number of neurons is able to model arbitrary decision boundaries \([9]\). Using the surface of a \( d \)-dimensional sphere to approximate the decision boundary, we relate the model complexity to the maximum curvature, which corresponds to the minimum radius of the sphere. Thus the decision boundaries before and after the attacks are described by two hyper-spherical caps. As the boundary before the attack is fixed, without loss of generality, we assume it to be flat to simplify the analysis.

Now, according to Eqn (7), the specificity loss is quantified by the number of inputs whose classification is changed due to \( r_\theta \). Following the assumptions, such inputs reside in a \( d \)-dimensional hyper-spherical cap, as shown in Figure 11(b). Due to its minuscule scale, the probability density \( p_{data} \) in this cap is roughly constant. Minimizing the specificity loss is thus equivalent to minimizing the cap volume \([35]\), which amounts to maximizing the curvature of the sphere (or minimizing its radius). Let \( r \) be the minimum radius induced by the model.
We quantify the specificity loss as:

\[
L_s(r_\theta) = p_{\text{data}} \frac{d}{1 + d} \int_0^\infty \frac{\sin^d(t)}{\Gamma(d+1)} \, dt
\]

where \( \Gamma(z) \triangleq \int_0^\infty e^{-t}t^{z-1} \, dt \) is the Gamma function.

**Mutual Reinforcement Effect**

Recall the unified attack aims to optimize the following objective function:

\[
(r^*_s, r^*_\theta) = \arg\min_{r_s, r_\theta} L_f(r_s, r_\theta) + \lambda L_s(r_s) + \nu L_s(r_\theta)
\]

where \( \lambda \) and \( \nu (\lambda, \nu \geq 0) \) control the balance of the two losses.

Let \( \bar{r}_s, \bar{r}_\theta \) be the input and model perturbation vectors given by the adversarial and backdoor attacks respectively. Note that for fixed attack efficacy, \( r^*_s = \bar{r}_s \) if \( r^*_\theta = 0 \) and \( r^*_\theta = \bar{r}_\theta \) if \( r^*_s = 0 \).

In the following, we normalize the fidelity and specificity losses as \( L_f(r_s^*) / L_f(\bar{r}_s) \) and \( L_s(r^*_\theta) / L_s(\bar{r}_\theta) \) respectively. Both relative losses are bounded to \([0, 1]\).

We now quantify the MR effect in the case of trading fidelity for specificity, while the alternative case can be derived similarly. Specifically, the MR effect is measured by the ratio of specificity “saving” and fidelity “cost”:

\[
\phi(r^*_s, r^*_\theta) \triangleq \frac{1 - L_s(r^*_\theta)/L_s(\bar{r}_\theta)}{L_f(r^*_s)/L_f(\bar{r}_s)}
\]

Intuitively, the numerator is the specificity saving, while the denominator represents the fidelity cost. We say that the MR effect is significant, if \( \phi(r^*_s, r^*_\theta) \gg 1 \), i.e., the saving dwarfs the cost. It is trivial to verify that if \( \phi(r^*_s, r^*_\theta) \approx 1 \) then the MR effect of trading fidelity for specificity is also significant \( \phi(r^*_s, r^*_\theta) \approx 1^3 \).

Consider the unified attack as shown in Figure 11 (c). The input perturbation vector \( r^*_s \) moves \( x \) towards the decision boundary and reduces the loss by \( \kappa / \| \nabla_\mathcal{L}(0,0) \|_2 \). The relative fidelity loss is given by:

\[
L_f(r^*_s)/L_f(\bar{r}_s) = \kappa / \kappa
\]

Following we use \( \varepsilon = \kappa^2 / \kappa \) for simplicity.

Meanwhile, it is straightforward to derive that the height of the hyper-spherical cap is \((1 - \varepsilon)h\). The relative specificity loss is thus given by:

\[
L_s(r^*_\theta)/L_s(\bar{r}_\theta) \leq \frac{\arccos(1 - \varepsilon / \kappa)}{\arccos(1 - 1 / \kappa)} \sin^d(t)dt
\]

Instantiating Eqn (12) with Eqn (13) and Eqn (14), the MR effect of trading fidelity for specificity is defined as:

\[
\phi(r^*_s, r^*_\theta) = \frac{\arccos(1 - \varepsilon / \kappa)}{\arccos(1 - 1 / \kappa)} \sin^d(t)dt
\]

**5.2 Analytical Justification**

Next we provide analytical justification for all the empirical observations in § 4. The proofs are deferred to Appendix A.

**RQ1: Why is the tradeoff between fidelity and specificity disproportional?**

We have the following proposition for the MR effect of trading fidelity for specificity. A similar argument can be derived for the case of trading specificity for fidelity.

**Proposition 2.** The MR effect defined in Eqn (15) is strictly greater than 1 for any \( 0 < \varepsilon < 1 \).

Intuitively, to achieve fixed attack efficacy \( (\kappa) \), with a slight increase of fidelity loss \( L_f(r^*_s) \), the specificity loss \( L_s(r^*_\theta) \) is reduced super-linearly.

Figure 12 evaluates the MR effect as a function of relative fidelity loss under varying \( h / \kappa \) setting. Observe that the MR effect is larger than 1 by a large margin, especially for small fidelity loss \( \kappa / \kappa \), which is consistent with our empirical observation: with little fidelity cost, it may significantly reduce the specificity loss.

**RQ2: Why is leveraging two attack vectors more effective than using each one alone?**

Let \( \bar{r}_s \) be the input perturbation vector given by the adversarial attack: \( \bar{r}_s = \arg\min_{r_s} L(r_s, 0) + \lambda L_f(r_s) \). Recall the unified attack model defined in Eqn (11). We have the following derivation:

\[
\min_{r_s, r_\theta} L(r_s, r_\theta) + \lambda L_f(r_s) + \nu L_s(r_\theta)
\]

\[
\leq \min_{r_s} L(r_s, 0) + \lambda L_f(r_s) \quad \text{(cf. Eqn (12))}
\]

\[
= L(\bar{r}_s, 0) + \lambda L_f(\bar{r}_s)
\]
Given that $\mathcal{L}_{\lambda}(r_0) \geq 0$ for $r_0 \neq 0$, it follows that $\mathcal{L}(r^*_s, r^*_d) + \lambda \mathcal{L}_{\lambda}(r^*_s) \leq \mathcal{L}(\vec{r}, 0) + \lambda \mathcal{L}_{\lambda}(\vec{r})$. We conclude that the unified attack is more effective than the adversarial attack, as it incurs less efficacy (or fidelity) loss if the fidelity (or efficacy) loss is fixed. A similar argument can be made regarding attacks using backdoored models only.

Also note that $\vec{r}$ is the optimal input perturbation vector under the condition of $r_0 = 0$. Therefore, $\mathcal{L}(r^*_s, 0) + \lambda \mathcal{L}_{\lambda}(r^*_s) \geq \mathcal{L}(\vec{r}, 0) + \lambda \mathcal{L}_{\lambda}(\vec{r})$. That is, the input perturbation vector $\vec{r}$ by the unified attack alone is not superior to its counterpart by the adversarial attack.

RQ3: Why is the MR effect negatively correlated with the DNN model complexity?

In § 4, it is empirically observed that the MR effect is marginally more evident for simpler DNN models. Here we provide analytical explanations for this phenomenon.

Recall that a model’s inherent complexity is reflected in the maximum possible curvature (or the minimum radius $r$) induced by this model. We measure the impact of model complexity on the MR effect by the influence of minimum radius $r$ on $\phi(r^*_s, r^*_d)$. We have the following proposition.

Proposition 3. For fixed relative fidelity loss, the MR effect increases with the minimum radius $r$.

Intuitively, for a simpler model, the adversary needs to perform more significant perturbation (in terms of specificity loss) to the benign model in order to achieve the same amount of efficacy loss reduction. Figure 12 illustrates the MR effect under varying setting of $h/r$ (with $h$ fixed). Observe that the MR effect is more evident for larger $r$ (e.g., a linear model), which is consistent with our empirical observations.

Indeed, we can derive the asymptotic limit of the MR effect as $h/r$ approaches 0 (i.e., the simplest model), which is given by the following proposition.

Proposition 4. As the minimum radius approaches infinity, the asymptotic limit of the MR effect is specified by:

$$\lim_{r \to +\infty} \phi(r^*_s, r^*_d) = \frac{1 - (1 - z)(d + 1)/2}{z}.$$ 

In Appendix A5, we also discuss the correlation between MR effect and data dimensionality.

RQ4: Why is the MR effect exploitable to optimize the attack with respect to multiple metrics?

It is shown in § 4 that the unified attack is evasive with respect to the detectors designed for the adversarial and backdoor attacks. This is intuitively explained by that in contrast with the adversarial attack ($\vec{r}$) or the backdoor attack ($r_0$) alone, the MR effect allows the unified attack to use much less perturbation, $\mathcal{L}(r^*_s) < \mathcal{L}(\vec{r})$ and $\mathcal{L}(r^*_d) < \mathcal{L}(r_0)$ to attain the same efficacy.

Therefore, if the detectors heavily rely on the input or model perturbation magnitude, they tend to be less effective against the unified attack, which flies under the radar of each detector. For instance, the effectiveness of feature squeezing (FS) is strongly correlated with the input perturbation magnitude [49]; while the effectiveness of influence function (IF) depends on the “singularity” of trigger inputs, which is often related to the mode perturbation magnitude (§ 4).

A similar argument applies to the case of optimizing trigger patterns. Rather than using predefined trigger patterns as in the existing trojaining attacks, considering the trigger pattern as an additional variable $r$ in Eqn (9) enables a larger search spectrum to optimize both efficacy and specificity. Thanks to the MR effect, properly adjusting the perturbation vector $r$ (even though there is no loss defined on $r$!) is expected to improve both efficacy and specificity.

6 Discussion

Next we discuss two potential countermeasures against the unified attack and their technical challenges.

6.1 Ensemble Detection

It is shown in § 4 and § 5 that it is insufficient to use detectors for adversarial inputs or backdoored models independently to defend against the unified attack.

One potential solution is to build an ensemble detector that integrates multiple individual ones and detects the unified attack based on both input and model perturbation. To this end, we build a neural network-based ensemble detector that aggregates the measures from individual detectors (FS, LID, MagNet for adversarial inputs and IF for backdoored models) as features to detect suspicious inputs (i.e., trigger or adversarial inputs). We consider two ensemble models: (i) one-layer network with linear transformation and (ii) one hidden-layer network with sigmoid activation.
Their performance of detecting the PGD-StingRay-based unified attack against VGG on CIFAR10 (including 200 positive and 200 negative cases) is summarized in Table 3, where the relative fidelity loss is fixed as 0.25 and the misclassification confidence as 0.75.

| D | PS | LID | MagNet | IF | ENS(L) | ENS(N) |
|---|----|-----|--------|----|--------|--------|
| + | + | 0.05 | 0.15 | 0.35 | 0.05 | 0.44 | 0.17 | 0.33 | 0.16 |
| - | - | 0.34 | 0.48 | 0.44 | 0.11 | 0.39 | 0.48 | 0.34 |

Table 3. Confusion matrix of detecting the PGD-StingRay unified attack on CIFAR10 by individual and ensemble detectors (G: ground truth, D: detector, Ens: ensemble).

Observe that the ensemble detectors attain much higher accuracy compared with individual ones, especially in terms of true positive rate. Further, the nonlinear ensemble detector (N) outperforms the linear counterpart (L). This may be explained by that the intricate MR effect is fundamentally non-linear. Note that despite the apparent effectiveness of the ensemble detectors, the adversary may adapt the attack with respect to such detection (e.g., optimization with extra loss terms accounting for the ensemble detection). We consider developing adaptive attacks and refining ensemble detection mechanisms as our ongoing work.

### 6.2 Adversarial Re-Training

Besides attempting to detect the unified attack, another potential mitigation is to perform adversarial training [30,44] before reusing untrusted DNNs (i.e., adversarial “re-training”). By considering worst-case perturbation during training DNNs, adversarial training is regarded as one state-of-the-art defense against adversarial attacks. Table 4 summarizes the effectiveness of using PGD to adversarially re-train backdoored VGG models on CIFAR10.

| Unified Attack | % Data for Re-Training |
|---------------|------------------------|
| PGD-StingRay  | 1.0 0.94 0.68          |
| C&W-StingRay  | 1.0 0.99 0.96          |

Table 4. Attack success rate of the unified attack again adversarially re-trained DNNs (with PGD as the reference attack).

Observe that adversarial re-training significantly improves the robustness against the PGD-StingRay attack, reducing the attack success rate from 100% to 68.2%. However, this robustness is specific to the target attack. For example, it is much less effective against the C&W-StingRay attack. Further, this robustness heavily relies on the available training data, differing by 26.5% in the cases of using 25% and 50% training data for re-training. Unfortunately, in transfer learning scenarios, the available training data is often limited, which hinders the practicality of this mitigation. Also intensive adversarial re-training defeats the benefit of transfer learning (i.e., reusing DNNs without costly re-training).

### 7 Related Work

With their increasing use in security-critical domains, DNNs are becoming the new targets of malicious manipulations [4]. Two primary attack vectors are considered in the literature: adversarial inputs and backdoored models.

#### Adversarial Inputs

The existing research on adversarial inputs is divided in two campaigns. One line of work focuses on developing new attacks against DNNs [6, 14, 34, 41], with the objective of crafting adversarial samples to force DNNs to misbehave. The existing attacks can be categorized as untargeted (in which the adversary desires to simply force misclassification) and targeted (in which the adversary desires to force the inputs to be misclassified into specific classes).

Another line of work attempts to improve DNN resilience against the existing adversarial attacks by developing new training strategies (e.g., adversarial training) [16,22,33,42] or detection mechanisms [13,28,31,49]. However, it is observed that existing defenses are often penetrated or circumvented by even stronger attacks [2, 26], resulting in a constant arms race between the attackers and defenders.

#### Backdoored Models

The backdoored model-based attacks can be categorized according to their trigger types. In the input-as-trigger attacks, the triggers are defined based on specific inputs, while the adversary’s goal is to force such inputs to be misclassified by target DNNs [18,19,37,40,47]. In the pattern-as-trigger attacks (i.e., the trojaning attacks), the triggers are defined as specific patterns embedded in inputs (e.g., a particular watermark), while the adversary’s goal is to force any inputs with such patterns to be misclassified by target DNNs [15,27]. Note that compared with the pattern-as-trigger attacks, the trojaning attacks leverages the attack vectors of both adversarial inputs and backdoored models.

The existing defense methods against backdoored models mostly focus on the pattern-as-trigger attacks [43,46], which identify potential poisoning inputs by detecting abnormal distributions in the feature space due to injected trigger patterns and re-train the DNNs using cleansed training data.

Despite the intensive research on adversarial inputs and backdoored models in parallel, there is still a significant lack of understanding about their fundamental connection. This work bridges this gap by studying the two attack vectors within a unified framework and providing a holistic view of the vulnerability of DNN models deployed in practice.

### 8 Conclusion

This work represents a solid step towards understanding adversarial inputs and backdoored models in a holistic manner. We demonstrate both empirically and analytically that (i) there exists an intricate mutual reinforcement effect between the two
attack vectors, (ii) the adversary is able to exploit this effect
to optimize the attacks with respect to multiple metrics, and
(iii) it requires to carefully account for this effect to design
effective defenses against attacks leveraging both vectors. We
believe our findings shed light on the holistic vulnerability of
deep neural network models.

This work also opens a few avenues for further investi-
gation. First, besides the targeted, white-box attack setting
considered in this paper, it is interesting to study the con-
nection of the two attack vectors under alternative settings
(e.g., untargeted, black-box attacks). Second, integrating the
two attack vectors with other types of security threats (e.g.,
membership attacks) is also a direction worthy of exploration.
Finally, devising a unified robustness metric accounting for
both attack vectors may serve as a promising starting point
for developing effective countermeasures.
References

[1] Rima Alaifari, Giovanni S. Alberti, and Tandri Gauksson. ADef: an Iterative Algorithm to Construct Adversarial Deformations. In Proceedings of International Conference on Learning Representations (ICLR), 2019.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In Proceedings of IEEE Conference on Machine Learning (ICML), 2018.

[3] Battista Biggio, Giorgio Fumera, Fabio Roli, and Luca Didaci. Poisoning Adaptive Biometric Systems. In Proceedings of Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition (SSPR&SPR), 2012.

[4] Battista Biggio and Fabio Roli. Wild Patterns: Ten Years after The Rise of Adversarial Machine Learning. Pattern Recognition, 84:317–331, 2018.

[5] Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, 2004.

[6] Nicholas Carlini and David A. Wagner. Towards Evaluating the Robustness of Neural Networks. In Proceedings of IEEE Symposium on Security and Privacy (S&P), 2017.

[7] R.D. Cook and S. Weisberg. Residuals and Influence in Regression. Chapman & Hall, 1982.

[8] Paul Cooper. Meet AISight: The scary CCTV network completely run by AI. http://www.itproportal.com/, 2014.

[9] G. Cybenko. Approximation by Superpositions of A Sigmoidal Function. Mathematics of Control, Signals, and Systems (MCSS), 2(4):303–314, 1989.

[10] J.M. Danskin. The Theory of Max-Min and Its Application to Weapons Allocation Problems. Springer-Verlag, 1967.

[11] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-scale Hierarchical Image Database. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[12] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. Classification Regions of Deep Neural Networks. ArXiv e-prints, 2017.

[13] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation. In Proceedings of IEEE Symposium on Security and Privacy (S&P), 2018.

[14] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In Proceedings of International Conference on Learning Representations (ICLR), 2015.

[15] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. ArXiv e-prints, 2017.

[16] Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering Adversarial Images Using Input Transformations. In Proceedings of International Conference on Learning Representations (ICLR), 2018.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[18] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. Model-Reuse Attacks on Deep Learning Systems. In Proceedings of ACM SAC Conference on Computer and Communications (CSS), 2018.

[19] Yujie Ji, Xinyang Zhang, and Ting Wang. Backdoor Attacks against Learning Systems. In Proceedings of IEEE Conference on Communications and Network Security (CNS), 2017.

[20] Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In Proceedings of IEEE Conference on Machine Learning (ICML), 2017.

[21] Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

[22] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. In Proceedings of International Conference on Learning Representations (ICLR), 2017.

[23] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[24] Bin Liang, Miaoaquing Su, Wei You, Wenchang Shi, and Gang Yang. Cracking Classifiers for Evasion: A Case Study on the Google’s Phishing Pages Filter. In Proceedings of International Conference on World Wide Web (WWW), 2016.

[25] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against Adversarial Attacks Using High-Level Representation Guided Denoiser. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[26] X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, B. Li, and T. Wang. DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model. In Proceedings of IEEE Symposium on Security and Privacy (S&P), 2019.

[27] Yingqi Liu, Shiqing Ma, Yousa Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In Proceedings of Network and Distributed System Security Symposium (NDSS), 2019.

[28] Shiqing Ma, Yingqi Liu, Guanhong Tao, Wen-Chuan Lee, and Xiangyu Zhang. NIC: Detecting Adversarial Samples with Neural Network Invariant Checking. In Proceedings of Network and Distributed System Security Symposium (NDSS), 2018.

[29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In Proceedings of International Conference on Learning Representations (ICLR), 2018.
[31] Dongyu Meng and Hao Chen. MagNet: A Two-Pronged Defense Against Adversarial Examples. In Proceedings of ACM SAC Conference on Computer and Communications ( CCS), 2017.

[32] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Analysis of Universal Adversarial Perturbations. ArXiv e-prints, 2017.

[33] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In Proceedings of IEEE Symposium on Security and Privacy (S&P), 2016.

[34] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. In Proceedings of IEEE European Symposium on Security and Privacy (Euro S&P), 2016.

[35] A.D. Polyanin and A.V. Manzhirov. Handbook of Mathematics for Engineers and Scientists. Taylor & Francis, 2006.

[36] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016.

[37] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitraș, and Tom Goldstein. Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. In Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2018.

[38] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature, (7587):484–489, 2016.

[39] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of International Conference on Learning Representations (ICLR), 2014.

[40] Octavian Suciu, Radu Mărginean, Yiğitcan Kaya, Hal Daumé, III, and Tudor Dumitraş. When Does Machine Learning FAIL? Generalized Transferability for Evasion and Poisoning Attacks. In Proceedings of USENIX Security Symposium (SEC), 2018.

[41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. In Proceedings of International Conference on Learning Representations (ICLR), 2014.

[42] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble Adversarial Training: Attacks and Defenses. In Proceedings of International Conference on Learning Representations (ICLR), 2018.

[43] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral Signatures in Backdoor Attacks. In Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2018.

[44] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness May Be at Odds with Accuracy. In Proceedings of International Conference on Learning Representations (ICLR), 2019.

[45] Allyson Versprille. Researchers Hack Into Driverless Car System, Take Control of Vehicle. http://www.nationaldefensemagazine.org/, 2015.

[46] B. Wang, Y. Yao, S. Han, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In Proceedings of IEEE Symposium on Security and Privacy (S&P), 2019.

[47] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal Security Analysis of Neural Networks Using Symbolic Intervals. In Proceedings of USENIX Security Symposium (SEC), 2018.

[48] Robert A. Wijsman. A Useful Inequality on Ratios of Integrals, with Application to Maximum Likelihood Estimation. Journal of the American Statistical Association, 80(390):472–475, 1985.

[49] W. Xu, D. Evans, and Y. Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In Proceedings of Network and Distributed System Security Symposium (NDSS), 2018.

[50] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J. Smola. Parallelized Stochastic Gradient Descent. In Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2010.

Appendix

A. Proofs

A0. Preliminaries

In the following proofs, we use the following definitions for notational simplicity:

\[ \alpha \triangleq \frac{h}{r} \]

\[ y \triangleq 1 - z \]

Further we have the following result.

\[ \int_{0}^{\arccos(x)} \sin^{d}(t) \,dt = \int_{x}^{1} (1 - t^{2})^{\frac{d-1}{2}} \,dt \]

(16)

We also use the following theorem [48].

**Theorem 1.** Let \( L \) be a measure on the real line \( \mathbb{R} \) and \( f_{i}, g_{j}(i = 1, 2) \) be 4 Borel-measurable functions: \( \mathbb{R} \rightarrow \mathbb{R} \) such that \( f_{2}, g_{2} \geq 0 \) and \( \int |f_{i}g_{j}| \,dt < \infty \) \((i, j = 1, 2)\). If \( f_{1}/f_{2} \) and \( g_{1}/g_{2} \) are monotonic in the same direction, then

\[ \int f_{1}g_{1} \,dt \int f_{2}g_{2} \,dt > \int f_{1}g_{2} \,dt \int f_{2}g_{1} \,dt \]

If \( f_{1}/f_{2} \) and \( g_{1}/g_{2} \) are monotonic in opposite directions, then the inequality is reversed.
A1. Proof of Proposition 1  

Proof. Recall that \( \mathcal{F} \) represents a non-empty compact set, \( \mathcal{L}(r, \cdot) \) is differentiable for \( r \in \mathcal{F} \), and \( \nabla r_a \mathcal{L}(r_a, r) \) is continuous over \( \mathcal{F} \times \mathbb{R}^n \).

Let \( \mathcal{F}^* = \{ \arg \min_{r_a \in \mathcal{F}} \mathcal{L}(r_a, r) \} \) be the set of minimizers and \( \ell(r) \triangleq \min_{r_a \in \mathcal{F}} \mathcal{L}(r_a, r) \). The Dvanski’s theorem \([10]\) states that \( \ell(r) \) is locally continuous and directionally differentiable. The derivative of \( \ell(r_a) \) along the direction \( d \) is given by \( D_d \ell(r_a) = \min_{r_a \in \mathcal{F}} D_d \nabla r_a \mathcal{L}(r_a, r) \).

We apply the Dvanski’s theorem to our case. Let \( r_a^* \in \mathcal{F} \) be a minimizer of \( \min_{r_a} \mathcal{L}(r_a, r) \). Consider the direction \( d = -\nabla r_a \mathcal{L}(r_a^*, r) \). Then we have:

\[
D_d \ell(r_a^*) = \min_{r_a \in \mathcal{F}} \nabla r_a \mathcal{L}(r_a^*, r) \leq -\| \nabla r_a \mathcal{L}(r_a^*, r) \|_2^2 \leq 0
\]

Thus, it follows that \( \nabla r_a \mathcal{L}(r_a^*, r) \) is a proper descent direction of \( \min_{r_a \in \mathcal{F}} \mathcal{L}(r_a, r) \).

Note that in the proof above, we ignore the constraint of \( r_a \in \mathcal{F}(\theta) \). Nevertheless, the conclusion is still valid. With this constraint, instead of considering the global optimum of \( r_a \), we essentially consider its local optimum within \( \mathcal{F}(\theta) \). Further, for DNNs that use constructs such as ReLU, the loss function is not necessarily continuously differentiable. However, since the set of discontinuities has measure zero, this is not an issue in practice.

A2. Proof of Proposition 2  

Proof. Proving \( \phi(r_a^*, r_a^0) > 1 \) is equivalent to showing the following inequality:

\[
\int_0^{\arccos(1-\gamma \alpha)} \sin^d(t) dt < \int_0^{\arccos(1-\alpha)} \sin^d(t) dt
\]

We define \( f(y) = \frac{1}{y} \int_0^{\arccos(1-\gamma \alpha)} \sin^d(t) dt \). This inequality is equivalent to \( f(y) < f(1) \) for \( y \in (0, 1) \). It thus suffices to prove \( f'(y) > 0 \).

Considering Eqn (16), we have \( f(y) = \frac{1}{y} \int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt \) and \( f'(y) = g(y)/y^2 \) where \( g(y) = y \alpha \left(1 - (1-\gamma \alpha)^2 \right)^{-\frac{d-1}{2}} - \int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt \)

Denote \( x = 1 - \gamma \alpha \). We have

\[
g(x) = (1+x)^{-\frac{d-1}{2}} - \int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt
\]

Note that \( g(1) = 0 \). With \( d > 1 \), we have

\[
g'(x) = -(d-1) x^{-\frac{d-3}{2}} (1-x)^{-\frac{d-1}{2}} < 0
\]

Therefore, \( g(x) > 0 \) for \( x \in (0, 1) \), which in turn implies \( f'(y) > 0 \) for \( y \in (0, 1) \).

A3. Proof of Proposition 3  

Proof. To show that the MR effect decreases with \( \alpha = h/r \), it suffices to show \( \frac{\partial \mathcal{L}(r_a)/\partial \alpha}{\partial \mathcal{L}(r_a)/\partial r} > 0 \), which is equivalent to

\[
\frac{(\gamma \alpha(2-\gamma \alpha))^{-\frac{d-1}{2}}}{((\alpha(2-\alpha))^{-\frac{d-1}{2}} > \frac{\int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}{\int_{1-\alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}
\]

where we have used Eqn (16).

We can rewrite the above inequality as

\[
\frac{\int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}{\int_{1-\alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt} > \frac{\int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}{\int_{1-\alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}
\]

(17)

We define the following functions (\( I_i \) represents an indicator function, which returns 1 if \( i \) is true and 0 otherwise).

\[
f_1(t) = I_{t \in [1-\alpha, 1]} (t(1-t)^{-\frac{d-1}{2}})
\]

\[
f_2(t) = I_{t \in [1-\alpha, 1]} (1-t)^{-\frac{d-1}{2}}
\]

\[
g_1(t) = I_{t \in [1-\gamma \alpha, 1]} 1
\]

\[
g_2(t) = I_{t \in [1-\alpha, 1]} 1
\]

It can be easily verified that \( f_2, g_2 \geq 0 \), \( \int f_2 g_i \, dt < \infty \) (\( i, j = 1, 2 \)), and \( f_i / f_2 \) and \( g_1 / g_2 \) are both monotonically increasing over \([1-\alpha, 1] \). By applying Theorem 1, we conclude that the inequality of Eqn (17) holds.

A4. Proof of Proposition 4  

Proof. Denote \( f(y) = \int_0^{\arccos(1-\gamma \alpha)} \sin^d(t) dt \).

\[
\lim_{\alpha \to 0} \phi(r_a^*, r_a^0) = \frac{\arccos(1-\alpha)}{\arccos(1-\alpha + \gamma \alpha)} \frac{\sin^d(t) dt}{\int_0^{\arccos(1-\alpha)} \sin^d(t) dt} = \frac{1}{1-y} \left( 1 - \lim_{\alpha \to 0} \frac{f(y)}{f(1)} \right)
\]

Using Eqn (16), \( f(y) = \int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt \). Thus,

\[
\lim_{\alpha \to 0} \frac{f(y)}{f(1)} = \lim_{\alpha \to 0} \frac{\int_{1-\gamma \alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}{\int_{1-\alpha}^1 (1-t^2)^{-\frac{d-1}{2}} dt}
\]

\[
= \frac{1}{1-y} \left( 1 - (1-\gamma \alpha)^2 \right)^{\frac{d-1}{2}}
\]

Therefore,

\[
\lim_{\alpha \to 0} \phi(r_a^*, r_a^0) = \frac{1}{1-y} \frac{1}{1-y} = \frac{1-(1-x)^{\frac{d-1}{2}}}{x}
\]

□
A5. MR Effect versus Data Dimensionality

**Proposition 5.** For fixed relative fidelity loss, the MR effect increases with the data dimensionality $d$.

**Proof.** Let $a = \arccos(1 - \alpha + z\alpha), b = \arccos(1 - \alpha)$. Note that $0 < a < b < \pi/2$.

To show that the leverage effect is an increasing function of $d$ ($d > 1$), it suffices to show $\frac{\partial L_2(f_0)/\partial L_3(r^2)}{\partial L_2(f_0)/\partial L_3(r^2)} < 0$, which is equivalent to showing that

$$f_0^a - \sin^2(t) \ln(\sin(t))dt > f_1^b - \sin^2(t) \ln(\sin(t))dt$$

We define the following functions:

$$f_1(t) = \int_{t_0}^{t} \sin^2(t)dt$$

$$f_2(t) = \int_{t_0}^{t} \cos^2(t)dt$$

$$g_1(t) = \int_{t_0}^{t} \sin^2(t) - \ln(\sin(t))dt$$

$$g_2(t) = \int_{t_0}^{t} \cos^2(t) - 1$$

Eqn (18) can be re-written as:

$$\int f_1(t)g_1(t)dt > \int f_2(t)g_1(t)dt$$

In our context, it can be verified that

$$f_1/f_2 = \int_{t_0}^{t} \sin^2(t)dt$$

$$g_1/g_2 = \int_{t_0}^{t} \sin^2(t) - \ln(\sin(t))dt$$

Thus both $f_1/f_2$ and $g_1/g_2$ are monotonically decreasing over $[0, b]$. We can apply Theorem 1 and conclude that the inequality in Eqn (18) holds. 

Figure 13 evaluates the MR effect as a function of relative fidelity loss with varying setting of data dimensionality $d$.

![Figure 13: MR effect with respect to relative fidelity loss and data dimensionality (default setting: $h/r = 0.2$).](image)

B. Implementation Details

Here we elaborate on the implementation of attacks and defenses in this paper.

**B1. Parameter Setting**

Table 5 summarizes the default parameter setting in our empirical evaluation ($\S$ 4).

| Attack  | Parameter                      | Setting                      |
|---------|--------------------------------|------------------------------|
| PGD     | confidence threshold           | $\epsilon = 0.031$           |
|         | maximum iterations             | $n_{max} = 100$              |
| C&W     | initial tradeoff               | $\kappa = 0$                 |
|         | learning rate                  | $c = 0.1$                    |
|         | binary search steps            | $\alpha = 0.01$              |
|         | maximum iterations             | $n_{step} = 5$               |
|         | maximum retraining epochs      | $n_{retrain} = 10,000$       |
| StingRay| base instance search radius    | $r = 600$                    |
| PoisonFrog | Gaussian noise mean, variance | $\mu = 0, \sigma^2 = 0.01$  |
|         | maximum retraining epochs      | $n_{retrain} = 100$          |
| TrojanNet | number of target neurons       | $n_{neurons} = 2$            |
|         | maximum preprocessing steps    | $n_{step} = 1,000$           |
|         | learning rate                  | $\alpha = 0.001$             |
|         | maximum retraining epochs      | $n_{retrain} = 100$          |

| Table 5. Default setting of attack parameters. |

**B2. Influence Function Estimation**

The complexity of computing the influence function stems from the inverse Hessian $H^{-1}$. Following [20], we apply the conjugate gradient (CG) method to transform it to an optimization problem. Given $H_\theta \approx 0, H_\theta^{-1} = \arg\min_{\theta} \{ t^THd - v^tt \}$. We solve this with the CG method typically in a few iterations. To further reduce the computational cost, rather than using the input layers directly, we use the penultimate layers of the DNNs (which typically correspond to high-level features) to estimate the influence function.

C. Additional Experiments

Here we provide experiment results additional to $\S$ 4.

**C1. Decision Boundaries under Attacks**

Figure 14 shows a set of sample inputs and their decision boundaries (each in a 2D subspace surrounding the input) under varied attacks (C&W, PoisonFrog, C&W-PoisonFrog) on CIFAR10, which complements the results in Figure 4.

**C2. Utility of Influence Function**

Figure 15(a) illustrates the distribution of influence scores of the trigger and non-trigger inputs in the case of applying PoisonFrog on ImageNet. Observe that the trigger inputs tend to have much higher influence scores than non-trigger inputs, which suggests the utility of IF. Figure 15(b) further shows the ROC of using IF to defend against backdoor attacks.
Figure 14: Inputs and decision boundaries (in a 2D subspace surrounding the inputs) under the adversarial, backdoor, and unified attacks on CIFAR10. (a) “horse” misclassified as “deer”; (b) “airplane” misclassified as “bird”. The x-axis and y-axis respectively represent two randomly selected orthogonal directions.

Figure 15: (a) Probability density distribution of IF of the trigger and non-trigger inputs with respect to the PoisonFrog attack on ImageNet. (b) ROC of IF in terms of detecting trigger inputs.