REStore: Exploring a Black-Box Defense against DNN Backdoors using Rare Event Simulation

Quentin Le Roux
Thales DIS & INRIA
La Ciotat & Rennes, France
quentin.le-roux@thalesgroup.com
quentin.le-roux@inria.fr

Kassem Kallas
INRIA
Rennes, France
kassem.kallas@inria.fr

Teddy Furon
INRIA
Rennes, France
teddy.furon@inria.fr

***Abstract*—Backdoor attacks pose a significant threat to deep neural networks as they allow an adversary to inject a malicious behavior in a victim model during training. This paper addresses the challenge of defending against backdoor attacks in a black-box setting where the defender has a limited access to a suspicious model. In this paper, we introduce Importance Splitting, a Sequential Monte-Carlo method previously used in neural network robustness certification, as an off-the-shelf tool for defending against backdoors. We demonstrate that a black-box defender can leverage rare event simulation to assess the presence of a backdoor, reconstruct its trigger, and finally purify test-time input data in real-time. So-called REStore, our input purification defense proves effective in black-box scenarios because it uses triggers recovered with a query access to a model (only observing its logit, probit, or top-1 label outputs). We test our method on MNIST, CIFAR-10, and CASIA-Webface. We believe we are the first to demonstrate that backdoors may be considered under the lens of rare event simulation. Moreover, REStore is the first one-stage, black-box input purification defense that approaches the performance of more complex comparables. REStore avoids gradient estimation, model reconstruction, or the vulnerable training of additional models.**

***Index Terms*—deep neural networks, backdoor defense, black-box, trigger reconstruction, input purification**

I. Introduction

The rise of Deep Neural Networks (DNN) over the past decade and their growing use in critical applications raise significant security concerns. As the driving force behind Machine Learning as a Service (MLaaS), DNN models are commonly outsourced for rapid and economical deployment [1]. However, this third-party reliance introduces vulnerabilities that malicious agents may hijack.

The exploration of this new attack surface focused first on test-time adversarial examples [2]. However, attention has since shifted to a broader range of integrity risks, with backdoor attacks taking center stage. Backdoors involve seemingly-benign DNNs that nonetheless conceal a stealthy and malicious behavior. Once the model is deployed by a victim, an adversary may activate the backdoor at any time [3]–[5].

The development of backdoor defenses followed, with a distinction based on which stage of a machine learning pipeline they protect: data-based or model-based defenses. In the first approach, a defender sifts through DNN inputs during training or inference. In the latter, the defender directly detects a backdoor within a model. If anomalies are found, the defender may filter the inputs accordingly [6] or clean the DNN [7].

While effective, many defenses require access to DNN internals. This white-box approach is a strong requirement that many real-world settings prohibit. DNN providers safeguard their models against intellectual property theft by restricting users to Application Programming Interfaces (API) [8], [9]. This leads to a black-box scenario for defenders, thus making black-box defenses a pertinent area of research [6], [10]–[12].

In this paper, we demonstrate the first use of rare event simulation (RES) to defend DNN classifiers against backdoors in a black-box setting. Our workhorse is Importance Splitting (IS), a procedure that estimates the probability of increasingly rarer inputs. IS provides three complementary tools: backdoor diagnosis, trigger reconstruction, and test-time data purification (see illustration in Fig. 1). IS reveals a backdoor within a DNN by identifying suspiciously fragile classes. IS concurrently outputs a set of rare input perturbations that have a strong likelihood of reconstructing backdoor triggers. Finally, these reconstructions enable a real-time and lightweight, albeit not state-of-the-art, test-time input purification defense, dubbed REStore. This paper thus expands the use of RES in the DNN literature beyond robustness certification [13].

This paper is organized as follows: Section II positions our work in the backdoor and RES literature. Section III outlines our threat model and how IS is used against backdoors. Our results are reported in Section IV. We cover the case of adaptive adversaries in Section V. Section VI discusses the application, complexity, and suggested lines for future directions. Section VII concludes the paper.

II. Background and Related Work

Backdoor attacks inject stealthy behaviors in DNNs by hijacking their training. Consequently, a range of defenses have emerged to mitigate these attacks and safeguard trust between DNN providers and users. This Section provides an overview of both attacks and defenses, outlines the relevance of RES to our method, and covers comparables for context.

A. Backdoor Attacks

Data poisoning. Data poisoning is a prevalent backdoor attack strategy [14]–[17] where an attacker has gained access

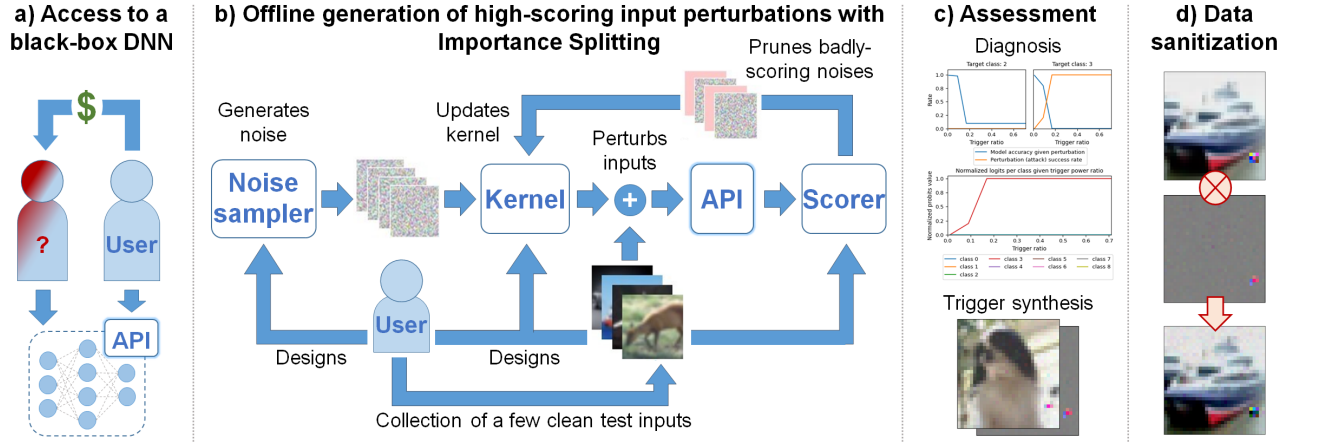


Fig. (1) Overview of REStore: (a) The defender interacts with a black-box DNN, e.g. via an API. (b) The defender collects a few clean inputs and runs Importance Splitting to generate input perturbations yielding outlier scores. (c) IS enables a model assessment, which points to possibly backdoored classes, and the reconstructions of potential triggers. (d) The defender purifies the triggers from incoming inputs.

to a victim's training dataset (either directly or by hijacking data collection). Data poisoning leads to DNNs learning to associate malicious patterns with specific and erroneous outcomes. For instance, BadNets [18] alters images with small pixel patches. We note that other frameworks exist, such as weight attacks [19]. However, they fall outside the scope of this paper and its comparables (see Section II-D).

Clean-label vs. poison-label. Backdoors based on data poisoning are split between poison-label and clean-label strategies [4]. Poison-label attacks work by altering datapoints from different source classes while flipping their ground truth labels to a malicious target. Though these attacks achieve high Attack Success Rates (ASR), they inject conspicuously misclassified data into a dataset. Clean-label attacks poison the inputs of the target class instead, thus eschewing the label-flipping step [14]. Doing so prioritizes stealth at the expense of a lower ASR or needing a greater number of poisoned samples.

Patch-based versus watermark-based triggers. Backdoor triggers typically fall under two categories: patches and watermarks. That is, triggers that are either localized in an input or blended in its entirety [14]. BadNets [18] illustrates the former, while Chen et al. [20] demonstrates the latter by blending a cartoon logo into benign images. More recent attacks have explored moving away from this binary framework. For instance, WaNet [21] proposes a backdoor trigger based on image warping that fits neither categories.

Other categories. Backdoors are designed to fit a range of stealth requirements, resulting in many variants. For instance, dynamic backdoors [22] decouple location from being a necessary component of a trigger's activation. If an attacker typically crafts a trigger independently of input semantics, as with BadNets or SIG [4], they may also create a trigger through an optimization process (e.g. Liu et al. [23] designs a pattern to target specific neurons in a DNN). Others also exploit Universal Adversarial Perturbations (UAP) [24].

Takeaway. Backdoor attacks typically fit several categories. For instance, SIG [4] uses a watermark-style sine wave to backdoor a DNN with clean-label data poisoning. In essence, this diverse range of attacks underscores the key issue of developing robust defenses against backdoors.

B. Backdoor Defenses

Training-time data filtering. A defender may defeat data poisoning by unmasking and excluding abnormal samples during training. Such process revolves around scrutinizing the interactions between a DNN and its training data. Methods like Spectral Signatures [25] or SCAn [26] observe that poisoned samples do not activate the same neurons as benign inputs, or rely on shortcuts in a DNN's intermediary layers. A DNN is then retrained using the cleaned training dataset.

Test-time data filtering. We note that similar defenses can be implemented at test-time to reject incoming, suspicious inputs. For instance, Neural Cleanse [27] uses an outlier detection scheme (an anomaly index) to filter test-time inputs.

Test-time input purification. Prior work also explores the case where data filtering is not reliable. Instead, a defender may choose to pre-process all inputs forwarded to a DNN with a purification strategy. For example, NEO [28] searches images for aberrant patterns, e.g. unusual colors, that do not fit some expected data distribution. NEO then erases these anomalies whether or not they trigger a backdoor. Other defenses offer similar strategies, leveraging clever data augmentation techniques to cause a trigger-backdoor mismatch. For instance, Februus [29] studies neuron activations to find suspicious areas in DNN inputs and, using a generative process, erases then reconstructs cleaned versions of these areas.

Backdoor diagnosis. Detecting a backdoor relies on using meticulously-crafted perturbations to reveal the stealthy behavior [30]. For instance, Universal Litmus Patterns [31] devises a binary test based on feeding specific, fixed patterns (referred to as "litmus tests") to a DNN to expose that it is compromised. Diagnosis may also involve examining a DNN's neuron activations using explainability techniques [32], feature clustering [33], or topological analysis [34].

Trigger reconstruction. A defender may try to disclose a trigger embedded in a suspicious model. This process often serves as the initial step to other backdoor defenses. For instance, DeepInspect [35] uses a generative model trained on a suspicious DNN to learn the distribution of possibly-backdoored inputs. Afterwards, DeepInspect uses the distribution to patch the DNN. However, the faithfulness of recovered triggers is not always guaranteed and remains an ongoing challenge, as pointed out by Veldanda et al. [36].

Backdoor suppression. As with DeepInspect [35], a defender may retrain a suspicious DNN to erase its backdoor. For instance, NNoculation [36] relies on a three-model ensemble strategy. It distills a secondary DNN from the suspicious one using augmented validation data. Disagreement between the predictions of these DNNs then helps isolate abnormal inputs sent to the first model. Afterwards, a third, generative DNN is trained on the quarantined data to reverse-engineer a possible trigger. The defender finally uses the third DNN to generate a new training dataset made of poisoned but correctly-labeled inputs with which to retrain the first model. Alternatively, methods like Fine-Pruning [7] directly modify suspicious DNNs by pruning and/or fine-tuning their neurons.

White-box vs. black-box. White-box defenses assume access to a DNN's training data or its architecture and weights. Such access allows the defender to either retrain or fine-tune a compromised DNN [7], [25] or access its neuron activations as in Februus [29]. While most defenses fall in the white-box case, there is a growing shift towards black-box defenses. In a black-box setup, defenders operate under a more challenging and realistic threat model, assuming only a query access to a DNN as in BDMAE [11].

Offline vs. online. Offline defenses are used before a DNN's deployment. Meanwhile, online defenses proactively monitor and adjust user interactions with a DNN to thwart attackers. For example, NNoculation [36] is an online defense that detects and quarantines abnormal inputs in real-time.

Single-stage vs. multi-stage. Finally, backdoor defenses differ in the number of stages they involve. For instance, a defender may require sourcing or training additional DNNs, as illustrated by NNoculation [36] or BDMAE [11].

Takeaway. As with backdoor attacks, defenses may fit multiple categories. It is important to note that no one-size-fits-all defense has been proposed so far, especially in the context of black-box defenses where we set our work.

C. Rare Event Simulation

Accurately estimating rare event probabilities is crucial, particularly in fields like civil engineering where failures may be catastrophic. However, the study of rare events faces numerous hurdles as generic approaches often prove impractical. For example, the crude Monte Carlo method tends to underestimate rare event probabilities unless an intractably large number of samples is used [37], [38].

To overcome these limitations, efficient methods (i.e. needing fewer samples) have emerged, including Sequential Monte Carlo (SMC) methods like RESTART [37] or Importance Splitting [38]. They expedite the convergence to a low variance estimate of a rare event probability by partitioning an event space into nested, progressively rarer regions (the rare event is the intersection of all regions within the nested structure, i.e. the bottom-most region). Moreover, a valuable output of SMC procedures is a set of observations of the rare event.

RES recently found its way to the DNN literature due to their intricate nature and the challenge of assessing DNN robustness. For instance, Importance Sampling [39] and Importance Splitting [13] have been proposed to certify a DNN's probability of misclassification under noisy input corruption.

D. Prior Works

This paper introduces Importance Splitting [13] (IS) as a tool for **offline backdoor diagnosis** and **trigger recovery** that power **REStore**, a fast **online test-time input purification defense**. It is a **single-stage**, **black-box** defense that eschews training additional DNNs and access to a DNN's internals.

With IS and REStore, we aim to explore the applicability of RES to defending against DNN backdoors, including whether simpler input purification methods are achievable. Current purification comparables typically adopt a DNN-based, multi-stage generative approach that removes and regenerates sections of an input to mitigate *all-to-one*, *input-agnostic* backdoor threats at test time. These methods operate independently of whether the protected DNN is backdoored, being built to run on all incoming inputs. Here, we select two relevant prior input purification defenses: Februus [29] and BDMAE [11].

Februus [29] is a white-box purification defense that leverages explainability methods (e.g. GradCam [40]) to detect a backdoor in an input. Once a suspicious input region is found, Februus uses a Generative Adversarial Network (GAN) to inpaint the compromised area. BDMAE [11] follows a similar but black-box approach, using a Masked Autoencoder (MAE) to (i) sample masks that select portions of an incoming input to reconstruct, (ii) score these reconstructions given their top-1 label predictions, and (iii) generate a final reconstruction that erases a potential trigger with a high likelihood.

These methods are effective but come with downsides. Februus needs a white-box access to the underlying DNN, which may not be possible in real-life scenarios. Additionally, Februus and BDMAE both require the use of supplemental generative DNNs that are typically sourced from a third party. This only diverts a defender's concerns about trust from the underlying, suspicious DNN to the inpainting supplements.

With the need to push forward the black-box capabilities of defenders against backdoors, this paper ports IS to assess DNN *security*, expanding its use beyond and independently from DNN *robustness* certification [13], [39]. We use IS due to its effectiveness [38], prior use in the DNN literature [13], and our access to a GPU-enabled implementation. We believe that we are the first to design a SMC-based method for black-box backdoor defense, specifically in the context of input purification. The only exception may be found in a backdoor diagnosis tool, AEVA [12], which relies on a gradient estimation stage via a crude Monte Carlo simulation. Crucially, our method

eliminates the need to learn supplemental DNNs, engage in gradient estimation, optimize a trigger generator, or gain access to a DNN's internals. Finally, we also believe we are first to try input purification beyond standard input-agnostic, patch-based backdoors: we additionally cover two watermark or warping-based attacks (SIG [4] and WaNet [21]) in Section IV and the case of adaptive adversaries in Section V.

III. Methodology

A. Threat Model

1) General setup: This paper investigates a two-party setup between an attacker and a defender. The attacker hides a backdoor in a DNN. The defender must prevent its use.

On the attacker's side, we adopt the attack surface from BadNets [18], SIG [4], and WaNet [21] (see App. A for detailed descriptions). This is reflected in Section IV where the attacker relies on a data poisoning strategy (as is also typical in input purification [29]) to backdoor a target DNN, unbeknownst to the defender. We expand this attack surface in Section V by exploring two adaptive attacker settings.

On the defender's side, we adopt a black-box scenario. The defender has no information about the training dataset and parameters, or the architecture and weights of the scrutinized DNN. The defender only observes its inputs and outputs. This paper explores three access types: whether the DNN's outputs are (i) its logits, (ii) its probits, or (iii) its top-1 predicted label, also called a "hard-label" case in BDMAE [11].

2) Formalization: The attacker has access to a clean, labeled training set $\mathcal{D}^{\mathsf{cl}}_{\mathsf{train}}$, a backdoor injection function $poison : \mathcal{X} \to \mathcal{X}$, and the opportunity to alter the labels of $poison$ed datapoints with a flip function $c : [\kappa] \to [\kappa]$ such that:

$$\mathcal{D}^{\mathsf{cl}}_{\mathsf{train}} = \{(x_i^{\mathsf{cl}}, y_i^{\mathsf{cl}})\}_{i=1}^{n} \subset \mathcal{X} \times [\kappa] \quad (1)$$

$$x_i^{\mathsf{po}} = poison(x_i^{\mathsf{cl}}) \quad (2)$$

$$y_i^{\mathsf{po}} = c(y_i^{\mathsf{cl}}) = \begin{cases} y_i \in [\kappa],\ y_i \neq y_i^{\mathsf{cl}} & \text{(Poison-label)} \\ y_i^{\mathsf{cl}} & \text{(Clean-label)} \end{cases} \quad (3)$$

where $^{\mathsf{cl}}$ and $^{\mathsf{po}}$ denote *clean* and *poisoned* data, $\mathcal{X}$ is the input space of the DNN f_θ with parameters θ, $[\kappa] = \{1, \ldots, \kappa\}$ is the set of classes predicted by f_θ, x^{po} is an input altered with $poison$, and y^{po} is the attacker's target label specified by c.

As stated in Section II-A, data poisoning works in either a poison-label or clean-label fashion. In the former, an attacker poisons inputs from different *source* classes with $poison$ and flips their ground truth labels to a single *target* class y^{po} with c. In the latter case, an attacker poisons inputs from the target class itself. In both cases, an attacker poisons the available inputs up to some proportion, or poisoning rate, $\beta \in (0, 1]$.

Both cases result in a clean dataset $\mathcal{C} = \{(x_i^{\mathsf{cl}}, y_i^{\mathsf{cl}})\}_{i=1}^{n-m}$ and a poisoned dataset $\mathcal{P} = \{(x_i^{\mathsf{po}}, y_i^{\mathsf{po}})\}_{i=1}^{m}$ that the attacker merges to create a backdoored dataset $\mathcal{D}^{\mathsf{po}}_{\mathsf{train}} = \mathcal{C} \cup \mathcal{P}$. The attacker is then free to choose training hyperparameters to craft a stealthy and effective backdoored DNN f_θ^{po} that performs well on some hold-out set. A backdoored model must achieve a clean accuracy similar to a benign model so as to be used by a victim, while also having a high attack success rate.

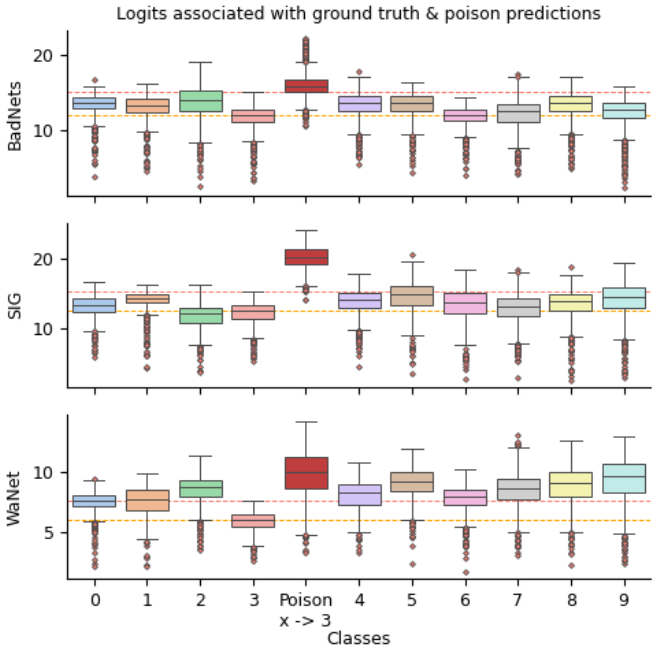


Fig. (2) Comparison of the distribution of the class logits of ResNet-18 DNNs trained on MNIST and backdoored with BadNets [18], SIG [4], or WaNet [21]. Dotted lines: median and maximum (outliers excl.) of the clean inputs of target class 3.

Meanwhile, the defender has a black-box access to f_θ^{po} and a limited set of clean-labeled samples denoted $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$. These samples may be collected in-the-wild and manually labeled.

3) Metrics: We gauge f_θ^{po} on benign inputs with Clean Data Accuracy (CDA), on poisoned inputs with Attack Success Rate (ASR), and on purified inputs, i.e. processed with a purification defense, with Sanitized Data Accuracy (SDA).

4) Backdoor attacks: Section IV focuses on three ubiquitous *all-to-one* attacks, central in the backdoor literature on image classification: (i) BadNets [18], a local pattern, (ii) SIG [4], a watermark signal, and (iii) WaNet [21], and image warping (See App. A for the detailed attack descriptions). Section V explores BadNets in two attacker-adaptive settings.

B. Main idea - intuition

1) Observation: This paper's core insight is illustrated in Fig. 2. When examining a DNN's outputs over a set of clean inputs, we find that the distributions for each predicted class are relatively similar. However, the introduction of the trigger t results in a significant increase of the scores of the backdoored class. This difference leads us to consider backdoored inputs as rare event occurrences (i.e. inputs with abnormally large scores in f_θ) that we can sample with the right tool.

2) Backdoors as rare events: Given a suspicious DNN f_θ that performs well on a classification task and a search distribution $\mathcal{L}_\mathcal{X}$, we want to model $\rho_\tau = \mathbb{P}(f_\theta(X)_\mathsf{c} > \tau)$. The quantity ρ_τ is the probability of occurrence of the following rare event: a random sample $X \sim \mathcal{L}_\mathcal{X}$ that yields an output score from f_θ for a class c greater than an arbitrarily-large threshold τ. We see three types of such rare occurrences:

(A) Inputs with a backdoor trigger t,
(B) Inputs that are purely random,
(C) Inputs with high-scoring adversarial perturbations.

We assume each type defines a disjoint subset in the input space of f_θ, allowing ρ_τ, the probability of the union of the subsets, to be written as $\rho_\tau = \mathbb{P}(X \in A) + \mathbb{P}(X \in B) + \mathbb{P}(X \in C)$ (see App. D for our rationale). As such, we expect to only observe type B and C occurrences in benign DNNs as opposed to backdoored DNNs. This implies that the probability ρ_τ equals a higher value for backdoored models, particularly when $\mathbb{P}(X \in A) \gg \max(\mathbb{P}(X \in B), \mathbb{P}(X \in C))$. *This rationale grounds our method's backdoor diagnosis step*.

Then, one may sample rare event realisations given a cleverly-designed search distribution $\mathcal{L}_\mathcal{X}$, such that the proportion of samples in A equals c. $\mathbb{P}(X \in A)/\rho_\tau$. A majority of the realisations should then contain observations that reveal the backdoor trigger t with a high likelihood ($\mathbb{P}(X \in A) \gg \max(\mathbb{P}(X \in B), \mathbb{P}(X \in C))$). Analyzing a set of rare event realisations may therefore reveal t. *This rationale grounds our method's backdoor trigger recovery step*.

Finally, as the set of rare event realisations contains the trigger t with a high likelihood, a defender may leverage the recovered pattern to design a test-time input purification scheme on the model f_θ. For instance, the defender may leverage simple data augmentation techniques as a pre-processing step to subtract t from incoming inputs. *This rationale grounds this paper's input purification defense*.

3) Limitations: Our intuition may not always hold. An attacker aware of a backdoor attack's outlier logits, and who is in control of a victim DNN's training environment, may train a model to display benign logits. Section IV focuses first on a threat model where the attacker only uses data poisoning. Such an *adaptive* adversary is covered in Section V.

C. Main idea - implementation

The main idea is to consider backdoored inputs as occurrences of a rare event that a defender is able to sample. We are looking for an algorithm that, given a suspicious DNN f_θ, can efficiently sample inputs or input perturbations Δ that yield increasingly high scores (i.e. logits, probits, or a metric derived from top-1 labels). Such procedure would enable checking via random guesses each class of f_θ for a type-A event.

1) Choice of search distribution $\mathcal{L}_\mathcal{X}$: A key ingredient to this formulation is the search space with distribution $\mathcal{L}_\mathcal{X}$. The defender must carefully choose it such that

$$\mathbb{P}(X \in A) \gg \max(\mathbb{P}(X \in B), \mathbb{P}(X \in C)).$$

This not only eases the backdoor diagnosis step but also enables reconstructing t from the realizations Δ of the estimated rare event. We explain in further details in App. F why choosing (i) pure noise or (ii) a perturbation space over 1 clean test sample for $\mathcal{L}_\mathcal{X}$ respectively favor sampling realizations of type B and C, i.e. the types of realizations to avoid.

Instead, we follow an approach that considers a search distribution $\mathcal{L}_\mathcal{X}$ such that a single perturbation $\Delta \sim \mathcal{L}_\mathcal{X}$ (e.g. $\mathcal{L}_\mathcal{X} = \mathcal{N}(0_d, \sigma^2 I_d)$) is applied to *several* clean test inputs not classified as c by f_θ. We denote this set $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ and the perturbation function over all samples as g (e.g. an add-and-clip operation). The score associated with Δ is an aggregate of the $|\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}|$ scores of the perturbed inputs. We thus use the empirical mean as our $scorer$ function:

$$scorer(\Delta) = \frac{1}{|\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}|} \sum_{x \in \mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}} f_\theta\left(g(x, \Delta)\right)_{\mathsf{c}}. \tag{4}$$

This redefines the rare event probability as:

$$\rho_\tau = \mathbb{P}(scorer(\Delta) > \tau). \tag{5}$$

Simulating a rare event thus amounts to finding a perturbation that jointly deludes the model f_θ over the set $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$. This approach has a higher likelihood of success because it samples inputs close to the manifold of typical images while avoiding pure noise (type B) and adversarial examples (type C).

A final advantage of this formulation is that it enables using the hard-label outputs of f_θ and not just its logits or probits. The score associated with Δ becomes the number of label flips to the target class c among the perturbed elements of $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$.

2) Importance Splitting as our RES algorithm: Even if this randomness favors the realization of a type-A event compared to B and C, the probability ρ_τ may be minuscule. The accurate reconstruction of the trigger t roughly needs a hundred realizations of the event A. The Crude Monte Carlo needs approximately $100/\rho_\tau$ random samples to sieve these hundred realizations. In our experiments, finding a backdoor occurs when targeting an extremely rare event with probability ρ_τ below 10^{-11}. Monte Carlo is thus intractable.

We choose IS as our workhorse thanks to (i) **its efficiency for estimating a rare event probability** [38], with a complexity only needing $O(\log(1/\rho_\tau))$ samples, (ii) its prior use in the DNN robustness certification[1] literature [13], and (iii) our access to a GPU-enabled implementation. As an iterative algorithm, IS enables us to:

(i) compute low-variance estimates $(\hat{\rho}_{\tau_j})_{j=1:J}$ for a series of increasing score thresholds $(\tau_j)_{j=1:J}$ (with $\tau_J = \tau$, and J the number of iterations afforded to IS),
(ii) generate a set of f_θ input perturbations Δ typical of the rarest event associated with $\tau_J = \tau$.

Thanks to (i), our backdoor diagnosis and reconstruction steps are not just based on a single probability estimation, but on the shape of the estimated map $\tau \to \rho_\tau$. Outputs (ii) enable our backdoor trigger reconstruction step.

D. Technical details

To keep this paper self-contained, the full implementation details of IS are covered in App. E-F.

1) Defender's standpoint: We consider a DNN classifier model f_θ that outputs either (i) logits s.t. $f_\theta : \mathbb{R}^d \to \mathbb{R}^\kappa$, (ii) probits after the application of the Softmax function s.t. $f_\theta : \mathbb{R}^d \to [0,1]^\kappa$, or (iii) its top-1 label s.t. $f_\theta : \mathbb{R}^d \to [\kappa]$.

[1]We note that this paper explores the use of IS in the context of DNN security independently from its prior application in DNN certification.

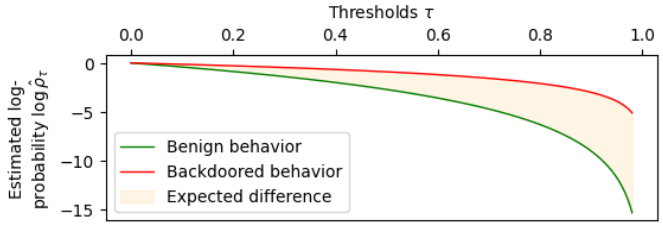


Fig. (3) Expected difference between benign and backdoored $\tau \to \rho_\tau$ mappings. The x-axis represents score thresholds and the y-axis the log-probabilities.

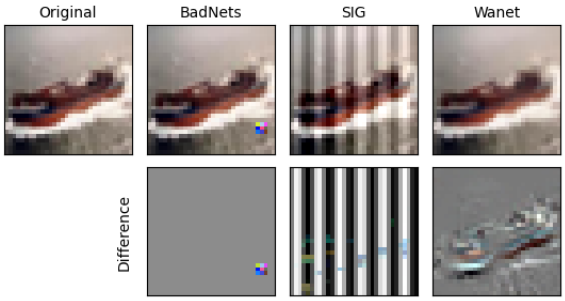


Fig. (4) Benign CIFAR-10 sample (left), poisoned with BadNets [18] (middle left), SIG [4] (right), or WaNet [21] (right).

To run the IS procedure (with the $scorer$ (4) function), the defender owns a small, fixed set of clean, correctly-labeled elements $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$ from which to create the set $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$. To test whether the DNN is backdoored for a given class c, and to stay on the manifold of typical images, the defender only has to pick a few inputs in $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$ not classified as c by f_θ. For example, $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ may contain $\kappa - 1$ clean-label, test images taken from $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$ (i.e. one input for each non-target class). The defender then runs IS to compute a score for a perturbation Δ over the $\kappa-1$ elements in $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$. In practice, we cap the number of elements in $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ when κ is large ($|\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}| = \min(\kappa - 1, 9)$), without observed loss in effectiveness.

2) Expected observations: Given that benign DNNs only suffer from rare events B or C and backdoored DNNs from rare events A, B, and C, IS should estimate a higher probability map $\tau \to \rho_\tau$ when the scrutinized class c is backdoored, as illustrated in Fig. 3. That is, high scores are achieved at a higher probability when a DNN is backdoored.

IS also outputs k perturbations Δ for a given class whose associated scores cross the final rare event threshold τ_J. These perturbations can be tested as potential triggers on the elements of $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$. A drop in a DNN's CDA and, conversely, an increase in ASR then validate the quality of the reconstructed trigger.

Additionally, purifying the recovered patterns from possibly backdoored test-time inputs may lead to a poison-backdoor mismatch (i.e. SDA tends towards CDA while ASR drops), yielding an online defense against backdoor attacks.

3) Defender's Input Purification defense: We finally explore a possible single-stage, lightweight input purification defense, dubbed **REStore**. REStore works by removing the IS-reconstructed patterns from all test-time inputs. In practice, REStore is a set of transformations designed such that the DNN's CDA does not degrade beyond some threshold.

The defender therefore proceeds as such: (i) they design a list of pre-processing steps related to their task at hand, (ii) using their clean test data $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$, they find purification transformations that does not degrade the model's CDA beyond some threshold, (iii) they deploy the defense. In this paper, we focus on two basic data transformations[2]:

- subtracting the mean of the IS patterns weighted by their signal-to-noise ratio (SNR),
- masking pixel values proportionally to the additive inverse of the IS perturbations' SNR.

Because IS is run offline, it allows a computational trade-off for REStore. At test-time, REStore is fast, having next to no overhead cost compared to Februus [29] or BDMAE [11], which both require multiple queries to supplemental DNNs.

[2]Another approach is briefly covered in Section IV-G.

IV. Experiments

A. Datasets and model setups

Datasets. We focus on three classification datasets: *MNIST* [41], *CIFAR-10* [42], and a custom face recognition task derived from *CASIA-Webface* [43]. Further details, including the construction of the latter dataset, are found in App. B.

Models. Following the backdoor literature for MNIST [4], [27], we select two models: a shallow LeNet-5 as in [30], and a deeper ResNet-18 model as in [11], [34]. For CIFAR-10, we use a ResNet-18 as in [10], [11]. Finally, for CASIA-Webface, we use a ResNet-50 pre-trained with ArcFace [44].

Backdoors. We backdoor each dataset with the following three *all-to-one* attacks: (i) *BadNets* [18], (ii) *SIG* [4], and (iii) *WaNet* [21] (see Fig. 4 for CIFAR-10 examples). The target class for all three is the 4^{th} one (e.g. digit 3 for MNIST).

Poison parameters. For BadNets [18], we use a static 3-by-3-pixel checkerboard that is stamped over the original pixels of an image (MNIST: 3-by-3 grayscale checkerboard located at the bottom left of an image; CIFAR-10 and CASIA-Webface: we reuse the 3-by-3 color pattern found in BDMAE [11], respectively located in the bottom right and top right quadrants of a given image). For SIG [4], we use a sinusoidal signal (8) with frequencies $(F_x, F_y) = (6/28, 0)$, $(6/32, 0)$ or $(6/112, 0)$, and blending ratios $\alpha_{\mathsf{atk}} = 60/255$, $40/255$, or $40/255$ for MNIST, CIFAR-10, and CASIA-Webface respectively. For WaNet, we use the original implementation [21]. All backdoors use the poisoning ratio $\beta = 0.05$. Visual illustrations of the triggers are found in Fig. 10 in App. J.

Training and checkpoint policies. Training regimens, hyperparameters, and our model checkpoint policies are found in App. C. The test CDA and ASR of our resulting models are reproduced in Table I.

B. Importance Splitting setup

IS parameters. Expanding on previous uses of IS [13], [38], we set the 6 IS hyperparameters described in App. E as

TABLE (I) Benign & backdoored models' CDA & ASR

Dataset	Model	Backdoor	CDA	ASR
MNIST	LeNet-5	Benign	99.1%	-
		BadNets	99.1%	100%
		SIG	99.1%	100%
		WaNet	98.9%	90.4%
	ResNet-18	Benign	99.5%	-
		BadNets	99.5%	100%
		SIG	99.4%	100%
		WaNet	99.3%	96.9%
CIFAR-10	ResNet-18	Benign	93.3%	-
		BadNets	92.7%	100%
		SIG	93.0%	100%
		WaNet	94.0%	94.3%
CASIA-Webface	ResNet-50	Benign	93.7%	-
		BadNets	94.2%	100%
		SIG	93.7%	100%
		WaNet	94.6%	95.1%

follows: $n = 500$, $k = 250$, $\mathsf{T} = 30$, the maximum number of iterations $J_{max} = 40$, $s = 1$, and $decay$ is set s.t., if s decreases at each IS iteration until T, the last pass will only modify a single pixel in an image if perturbations are generated sparsely. For a given DNN, *IS is run over each of its predicted classes*. $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ is comprised of 9 elements from $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$.

Importance Splitting setups. We define an IS setup as the set of $generator$, $kernel$, and $scorer$ functions needed to run IS, described in App. G. This paper focuses on 2 setups:

1) PI: samples perturbation Δ according to an Independent and Identical distribution in the pixel-space,
2) FI: samples perturbation Δ according to an Independent and Identical distribution in the frequency domain.

The PI and FI setups are complementary in that they each search for either pixel or frequency-based patterns, doing so by generating sparse perturbations in replacement of original pixels or frequency coefficients (using the Discrete Cosine Transform (DCT)) in inputs from $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$. We set the corresponding sparsity ratio as the proportion of elements that, once replaced by random noise, reduce a model's CDA by half (see our rationale in App. G). We also limit the search space for the FI setup following the rationale found in [45], [46] (see our rationale in App. G). To avoid erasing input semantics, we also freeze the lowest 5% of the DCT coefficients.

More details on the $generator$, $kernel$, and $scorer$ (incl. g) functions associated with PI and FI are found in App. G and H.

C. Test-time input purification defense comparables

1) Methods: We compare REStore (see Section III-D3) to Februus [29] and BDMAE [11]. To do so, we use the implementations provided by the latter. We use the mask ratios $\{0.6, 0.8\}$ for the XGradCam and GradCam++ methods for Februus, which extracts the *layer.2* activations of LeNet-5 DNNs and the *layer.3* activations of ResNet-18/50 DNNs. With regards to the BDMAE defense, we reuse the provided $base$ and $large$ MAE models, pre-trained on ImageNet (see full results for Februus, BDMAE in App. J, Fig. V).

2) Metrics: We assess the three input purification methods given: whether (i) the ASR decreases, (ii) that the accuracy of sanitized poisoned inputs (SDA) increases, and (iii) that the DNN's CDA is not damaged in the scenario where we apply the defenses to all inputs regardless of their content.

D. Backdoor diagnosis results

As defined in Section III-B, IS backdoor assessment relies on observing the estimated maps $\tau \to \rho_\tau$ *for each DNN class*.

1) Observations on BadNets [18] and WaNet [21] backdoors: The IS-yielded $\tau \to \rho_\tau$ maps demonstrably highlight the presence of a backdoor in the DNNs poisoned with BadNets or WaNet triggers, each outlined using the PI and FI setups respectively. Results using the probits-based $scorer$ functions are found (i) for MNIST in Fig. 5a and 5b (For conciseness, results on LeNet-5 models are found in App. J, Fig. 14), (ii) for CIFAR-10 in Fig. 6a and 6b and (iii) for CASIA-Webface in Fig. 7a and 7b.

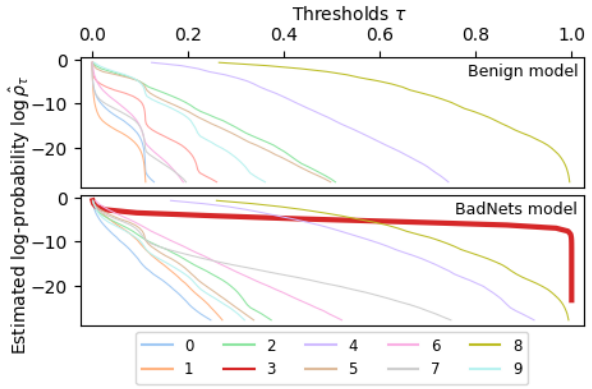
Additionally, the results demonstrate that benign classes, whether of a benign model or backdoored model, generally display similar curves. We surmise it is because they do not suffer from type-A events (see Section III-B).

2) Observations on SIG [4] backdoors: SIG is not as easily identified by assessing $\tau \to \rho_\tau$ maps. Besides MNIST (see Fig. 5b and 14b), the map of the backdoored class does not properly separate it from other classes (see in App. J, Fig. 15 for failed examples on the CIFAR-10 and CASIA datasets).

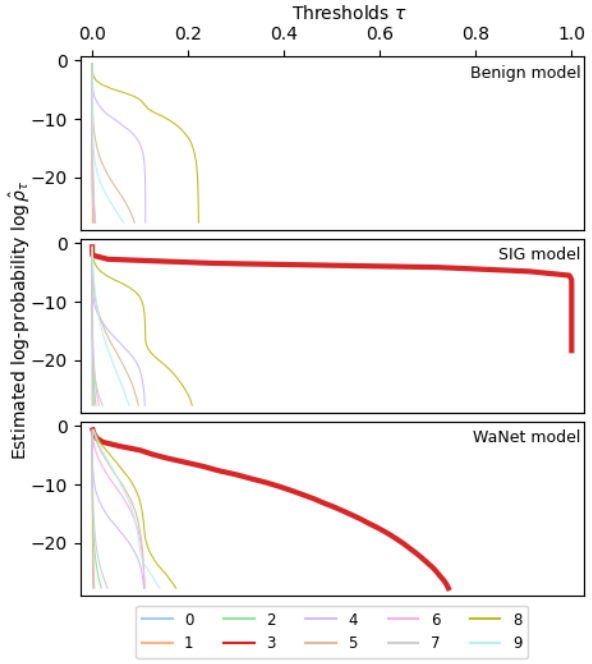
3) Interpretation: Of note, IS highlights suspiciously brittle classes whether it uses logits, probits, or hard-labels DNN outputs. The only exception may concern the hard-label case on MNIST (LeNet-5) on BadNets [18], where IS succeeds in identifying the backdoored class as suspiciously brittle but by a smaller margin compared to other IS maps in this paper. For conciseness, results with the logits and labels-based $scorer$ functions are found in App. J, respectively for (i) MNIST in Fig. 16, 17, 20, and 21, (i) for CIFAR-10 in Fig. 18 and 22, and (iii) for CASIA-Webface in Fig. 19 and 23.

We note the IS maps with the most conclusive visuals are generated with PI for BadNets [18] and FI for SIG [4] and WaNet [21]. This is expected given that BadNets is a local pattern, while SIG and WaNet are diffuse and may be easily captured by seeking perturbations in the frequency space. However, the characteristics of the triggers do not preclude a specific setup as we will see in the next Section IV-E.

4) Takeaway: The observed maps $\tau \to \rho_\tau$ fit the theoretical expectation set out in Section III-B and confirm the hypothesis made in Section III-C2 and derived from its point (i). IS is a useful early warning tool for discovering suspicious classes in a classifier DNN. This is especially relevant in the case of BadNets [18] and WaNet [21] but less so with SIG [4]. Overall, our observations underline the applicability of RES against backdoors for future work.

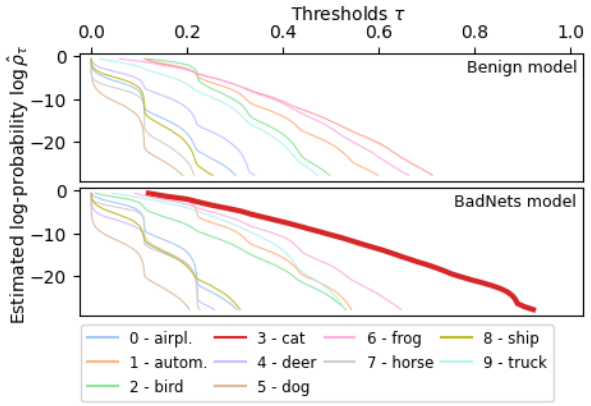


(a) IS PI results on ResNet-18 models (benign and BadNets [18]).

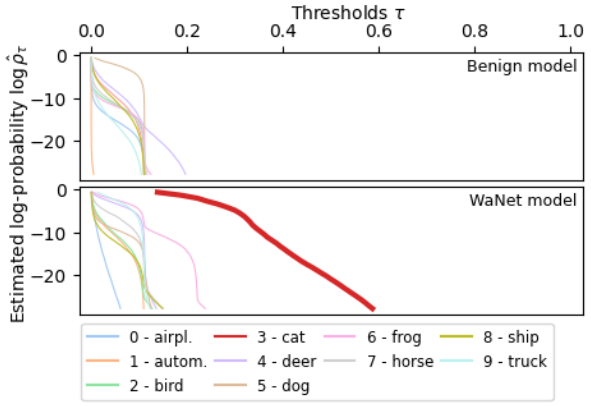


(b) IS FI results on ResNet-18 models (benign, SIG [18] or WaNet [21]).

Fig. (5) Estimated $\tau \rightarrow \rho_\tau$ IS maps (probit-based $scorer$) on MNIST, either benign or backdoored (class 3).



(a) IS PI results on ResNet-18 models (benign and BadNets [18]).



(b) IS FI results on ResNet-18 models (benign and WaNet [21]).

Fig. (6) Estimated $\tau \rightarrow \rho_\tau$ IS maps (probit-based $scorer$) on CIFAR-10, either benign or backdoored (class 3).

TABLE (II) Recovered triggers' ASR on the backdoored class (probit-based $scorer$). Bold indicates the best recovered ASR.

Dataset	Model	Backdoors	PI	FI
MNIST	LeNet-5	BadNets	**99.9%**	74.5%
		SIG	**100%**	97.9%
		WaNet	11.2%	**34.0%**
	ResNet-18	BadNets	**100%**	3.19%
		SIG	**100%**	**100%**
		WaNet	**66.1%**	9.5%
CIFAR-10	ResNet-18	BadNets	**65.7%**	4.4%
		SIG	**51.5%**	3.8%
		WaNet	14.1%	**19.5%**
CASIA-Webface	ResNet-50	BadNets	**70.2%**	0.0%
		SIG	**2.3%**	0.0%
		WaNet	42.6%	**50.3%**

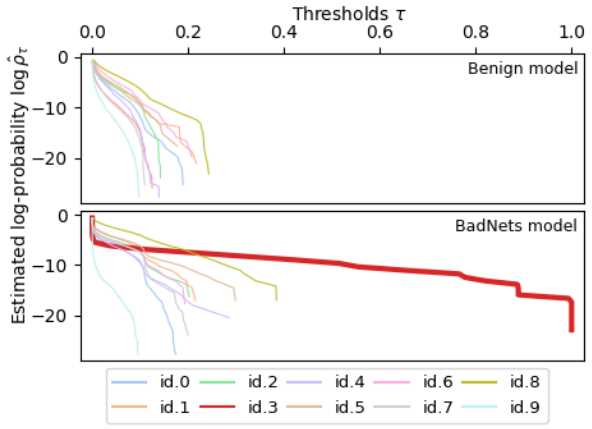
E. Trigger recovery results

To check that we found a backdoor, we now look at the second output of IS: typical realizations of the rare event yielded by it, which we surmise should reconstruct a trigger.

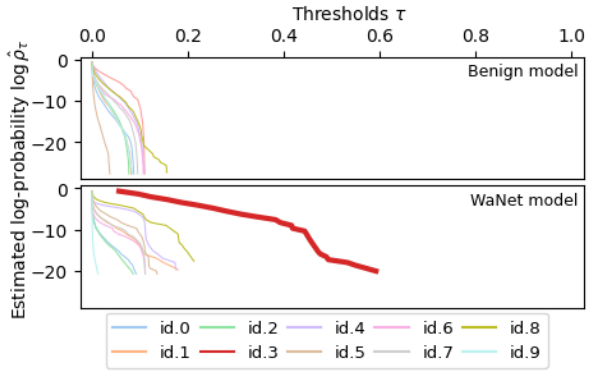
IS outputs perturbations that contain high-scoring patterns for brittle classes in a DNN f_θ. Here, we evidence that these patterns enable the recovery of a backdoor trigger $\mathtt{t}$ *independently* from the visual assessment done in Section IV-D. As such, we study the performance of the IS-recovered patterns as backdoor triggers. The patterns reconstructed with the two IS setups are stamped or blended-in on test inputs to assess their ASR performance. That is, we perform an attack with the recovered patterns on their respective backdoored DNNs.

1) Assessing the ASR of recovered patterns: Besides SIG [4] for CASIA-Webface, IS patterns yield a relatively high ASR on $\mathcal{D}^{\mathsf{cl}}_{\mathsf{test}}$ for the backdoored class on all the models *and* IS setups (see Table II). WaNet does not favor either PI or FI IS setups while BadNets and SIG favor the PI setup.

2) Faithful pattern recovery: We note that the BadNets [18] reconstructions are faithful in location but also color (see Fig. 9). However, as IS does not perform image warping, we cannot assert that the recovered trigger is faithful in

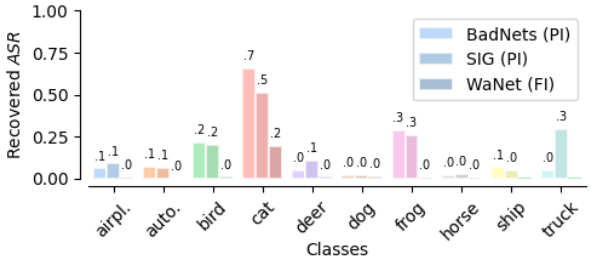


(a) IS PI results on ResNet-50 models (benign and BadNets [18]).

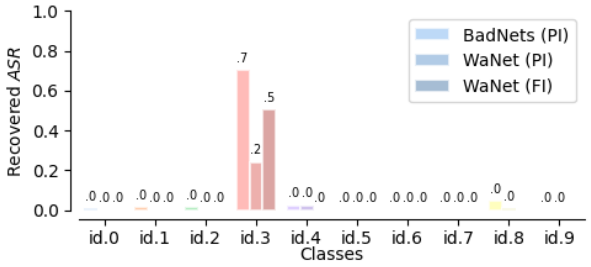


(b) IS FI results on ResNet-50 models (benign and WaNet [21]).

Fig. (7) Estimated $\tau \rightarrow \rho_\tau$ IS maps (probit-based *scorer*) on CASIA-Webface, either benign or backdoored (id. 3). Only the first 10 identities are displayed.



(a) Backdoored ResNet-18 trained on CIFAR-10.



(b) Backdoored ResNet-50 trained on CASIA-Webface.

Fig. (8) Illustration of the distinctiveness of the recovered backdoor triggers (only the first ten identities are shown for CASIA-Webface).

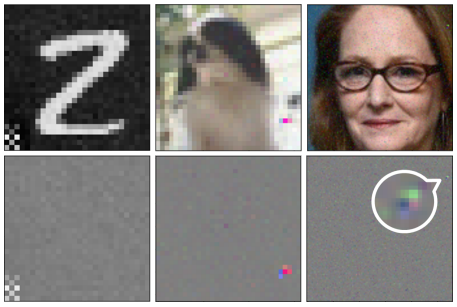


Fig. (9) Reconstructed BadNets [18] on MNIST (left), CIFAR-10 (center), and CASIA-Webface (right) with the IS PI setup on probit scores (top: stamped on an image, bottom: reconstruted trigger alone)

the WaNet [21] case (see an example in App. J, Fig. 13). Nevertheless, IS outputs strong perturbations for the WaNet DNN, indicative that the malicious warping leaves recoverable traces, identifiable in a black-box setting.

3) Recovery holds for all black-box access: We observe that trigger recovery holds whether IS uses a *scorer* function based on logits, probits, or hard-labels (see in App. J, Tables VIII and IX for the former and latter results respectively).

4) Interpretation: We surmise that IS is effective in recovering perturbations that set apart a backdoored class from benign ones as hypothesized in Section III-B. IS highlights events of type A (backdoors) instead of types B (random inputs) or C (adversarial perturbations on a single input). The ASR of the recovered patterns for backdoored DNNs trained on CIFAR-10 for instance is two to three times higher than the ASR of patterns found on other classes (see Fig. 8a). This demonstrates the capacity of IS to set apart A-type events. This effect is even starker on the ResNet-50 trained on CASIA-Webface for the BadNets and WaNet backdoors (see Fig. 8b).

5) Takeaway: IS outputs suspiciously strong perturbations for backdoored classes, which should alert a defender. They reconstruct the underlying trigger t, at least for BadNets [18]. Thus, we provide evidence that RES is useful to assess backdoors and recover their triggers. This confirms our second hypothesis made in III-C2 and derived from its point (ii).

F. Input purification via REStore results

We arrive at the crux of this paper: whether these reconstructions can be used for a defensive purpose as part of REStore.

1) Defending against BadNets [18] and WaNet [21]: With MNIST-trained DNNs (LeNet-5 and ResNet-18), we use the IS reconstructions that yield the highest recovered ASR (see Tab. II) as part of our REStore scheme (see Section III-D3). Here we observe a strong decrease of the ASR using naive data pre-processing strategies (see Table III and reconstruction examples in Fig. 11). This holds whether we use the logit, probits, or hard-label *scorer* function with IS. Moreover, REStore beats its comparables for the SIG and WaNet attacks.

For the ResNet-18 DNNs trained on CIFAR-10, we manage to effectively neutralize the BadNets [18] trigger for the logit and probit cases. Only the results for the hard-label *scorer* is underwhelming compared to the rest. We find that the IS-based defense also performs well on the WaNet attack [21],

TABLE (III) REStore Input Purification Defense and 2 comparables: Februus [29] and BDMAE [11]. **Bold** indicates the best reduced ASR over all defense types. Attack abbreviations: BN (BadNets), WN (WaNet).

Dataset	Network	Trigger	**Februus** (XGradCam) mask=0.8			**Februus** (GradCAM++) mask=0.8			**BDMAE** (base)			**BDMAE** (large)			**IS** with Logits-based *scorer*			**IS** with Probits-based *scorer*			**IS** with Hard-Label -based *scorer*		
			CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR
MNIST	LeNet-5	BN	95.0	1.5	98.5	96.0	52.0	47.4	98.1	98.3	**0.2**	98.4	97.9	**0.2**	99.1	90.3	9.1	98.9	84.9	14.5	98.6	70.8	28.7
		SIG	90.6	0.0	100	93.2	0.0	100	98.3	0.0	100	98.5	0.0	100	98.9	95.2	2.7	98.6	96.5	**0.8**	98.4	94.9	2.4
		WN	83.1	7.3	92.0	89.0	9.3	90.3	77.8	8.9	91.0	92.3	11.6	88.2	98.6	50.9	47.9	98.9	64.7	34.3	97.9	97.3	**0.2**
	ResNet-18	BN	98.8	0.0	100	98.6	93.6	2.5	99.0	98.9	**0.0**	99.0	99.0	0.1	99.3	99.4	0.1	99.4	99.4	0.1	99.4	99.3	0.1
		SIG	98.3	0.0	100	98.5	25.0	74.3	99.0	0.0	100	99.0	0.0	100	97.9	85.7	5.4	99.2	99.0	**0.1**	99.3	98.5	0.5
		WN	98.2	2.8	97.2	98.0	3.1	96.9	86.1	3.6	96.4	95.0	3.7	96.3	99.2	97.4	**1.8**	97.7	93.0	5.1	99.3	69.0	30.6
CIFAR-10	ResNet-18	BN	89.9	0.1	99.9	90.0	42.6	56.3	92.2	92.6	**1.3**	92.8	93.1	1.4	92.7	90.9	4.2	93.2	79.4	17.8	78.9	42.0	55.1
		SIG	90.5	1.8	98.1	91.8	1.4	98.5	92.5	1.1	98.9	92.6	1.1	98.9	89.4	1.0	98.9	81.9	1.8	98.0	91.7	1.0	98.9
		WN	92.4	20.4	78.5	89.2	27.2	71.5	94.1	78.2	18.2	94.0	79.7	16.0	89.4	86.3	4.4	91.1	89.1	**3.9**	88.0	85.5	4.5
CASIA	ResNet-50	BN	92.4	0.0	100	93.9	0.0	100	92.6	92.9	**0.0**	91.8	93.2	**0.0**	93.4	61.9	37.0	88.8	83.1	11.4	87.6	81.6	13.9
		SIG	91.5	0.0	100	92.6	0.0	100	93.0	0.0	100	91.6	0.0	100	93.4	0.0	100	93.8	0.0	100	93.2	0.0	100
		WN	91.4	7.9	92.1	91.7	8.7	91.2	85.4	6.5	93.4	93.2	40.3	58.0	93.9	71.3	24.1	93.8	74.2	21.2	93.1	82.8	**12.0**

TABLE (IV) Defense speed comparison (seconds per image)

Dataset	Network	**Februus** (all cases)	**BDMAE** (normal)	**BDMAE** (large)	**REStore** (our)
MNIST	LeNet-5	0.03	0.16	0.38	0.00003
	ResNet-18	0.04	0.16	0.39	0.00008
CIFAR-10	ResNet-18	0.04	0.17	0.39	0.0002
CASIA	ResNet-50	0.06	0.20	0.42	0.0008

beating the performance achieved by the BDMAE-*large* [11] defense in terms of cleaned ASR and SDA.

On the ResNet-50 DNNs trained on CASIA-Webface, there is strong evidence that REStore is able to purify face inputs, even in the hard-label case. Moreover, REStore purifies inputs backdoored with WaNet [21] where BDMAE struggles.

The most successful purifications are achieved by subtracting the recovered patterns weighted by the SNR of the PI setup's k perturbations (this happens in 80% of the cases).

2) Defending against SIG [4] backdoors: In all comparable defenses, the diffuse trigger evades purification for CIFAR-10 and CASIA-Webface. This issue outlines the difficulty of performing test-time data purification on watermark-style backdoors. IS is the only one to succeed on MNIST.

3) Takeaway: REStore rarely surpasses the performance of the state-of-the-art, multi-stage BDMAE [11] defense, typically registering a $3-6\%$ drop in CDA. However, our method shines in its speed as aimed for in Section II-D. We demonstrate that, using RES with IS and accepting a trade-off between offline and online search (running IS on all classes of a ResNet-18 takes under 2 hours on a V100), we can discard using supplemental DNNs for input purification. REStore is faster than its comparables by at least two orders of magnitude at test-time (see Table IV).

G. Other experiments with BadNets, SIG, and WaNet

This Section refers to supplemental experiments carried out to give additional insights on the applicability of REStore.

1) Clean-label versus poison-label poisoning: We trained ResNet-18 DNNs on CIFAR-10 and backdoored with *clean-label* BadNets [18] and SIG [4] (WaNet is poison-label only). We followed the same training and data augmentation procedures, except for a poisoning ratio $\beta = 0.3$ instead of 0.05.

Applying the previous IS workflow, we find strong results when computing the ASR of the PI-recovered triggers (see in App. J, Tables XI, XII, and XIII). Additionally, like the poison-label cases, we observe that the recovered triggers can be used to perform test-time data purification with REStore on the BadNets [18] DNN *even* in the hard-label case, albeit resulting in a failure for SIG [4] (see in App. J, Table VI).

2) Dynamic and optimized backdoors: To further test IS, we trained a ResNet-18 DNN on CIFAR-10 using two, more complex triggers: (i) a dynamic BadNets [18] trigger, (ii) a TrojanNN backdoor [23] (either a 3-by-3 or a larger 7-by-7 pattern). The dynamic BadNets is randomly placed in a poisoned image during training instead of a static location. Meanwhile, TrojanNN is a more sophisticated backdoor than BadNets or SIG [4] where the attacker proceeds in three steps: (i) trigger optimization to target specific neurons in a victim DNN, (ii) generation of synthetic, poisoned training data, (iii) backdoor embedding via model retraining. In all cases, the backdoor attack reaches a ASR of c. 99.9%.

We find that the dynamic BadNets is discoverable like its static variant, yielding a distinctive estimated $\tau \rightarrow \rho_\tau$ map (see in App. J, Fig. 24). Comparatively, TrojaNN does not yield a differentiating estimated map (see in App. J, Fig. 24) with respect to other classes. We surmise that this is due to the attack targeting specific neuron activations during training compared to a more generic data poisoning process, which teaches a DNN to maximize a specific class output.

Secondly, we find that, regardless of the estimated $\tau \rightarrow \rho_\tau$ map, the reconstructed trigger yields a strong ASR on the backdoored class 3 for both backdoors (see in App. J, Fig. 25a for dynamic BadNets and Fig. 25b for TrojanNN).

Finally, REStore works to defend against TrojanNN while expectedly failing against the dynamic BadNets because of the naive nature of the defense as-is (see in App. J, Fig. VI).

3) Other IS kernel: We also explore a different variation of the IS kernel, denoted as setup PG. It relies on generating pixel-based Δ perturbations with a Gaussian $kernel$ (see more details in App. G). These perturbations act as watermarks and are blended in the inputs from $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ following a blending ratio α_{def}. The defender must choose this α_{def} ratio without knowing the attacker's own α_{atk}. Here, we proceed similarly to how we designed the sparsity ratio for the PI and FI setups, setting α_{def} as the ratio of uniform noise that, when blended in a DNN's inputs, reduce its CDA by half.

We observe that the PG setup has a harder time outlining backdoors in a given model, besides the easy success on the LeNet-5 and ResNet-18 DNNs trained on MNIST. The output of the PG setup is more unwieldy, as exemplified by the results on WaNet, which fail for CIFAR-10 but succeed on CASIA-Webface (see in App. J, Fig. X).

For reference, we also tested a fourth setup FG, equivalent to FI, but with a Gaussian $kernel$ as PG. However, the results do not provide additional value and were left out of this paper.

4) Black-and-white search space: We also tested reducing the search space of the IS algorithm to more effectively find backdoor patterns. To do so, we reduce the search space to grayscale perturbations on CIFAR-10 and CASIA-Webface, de facto dividing the search space from 3 to 1 input channel.

We observe that CIFAR-10 and CASIA-Webface DNNs, backdoored with WaNet [21], can be defended against using REStore when the IS reconstruction occurs only in grayscale using at least the logit-based scorer (see in App. J, Fig. VII).

5) Breaking the SIG backdoor in CIFAR-10: Another possible data transformation for input purification is a Gram-Schmidt construction, where an input image x is orthogonalized with respect to the k IS perturbations: $x^{\mathsf{cl}} = x - \left(\frac{\mathsf{t}^T \cdot x}{\|\mathsf{t}\|^2}\right)\mathsf{t}$. We observe that using Gram-Schmidt against a SIG-backdoored ResNet-18 trained on CIFAR-10 (with FI in a grayscale search space), REStore achieves a CDA of 85.9%, SDA of 18.8%, and a ASR of 12.1%. In a nutshell, REStore conserves clean inputs while destroying incoming backdoors.

6) Low-Pass filter and purification transforms: Finally, we observe that applying a low-pass filter (e.g. Gaussian blur with kernel size 3) on the SNR used in the two purification transforms tested in this paper provides some marginal gains against BadNets [18] and WaNet [21].

V. The case of adaptive adversaries

Input purification defenses like Februus [29], BDMAE [11], and ours focus on defeating *all-to-one* backdoors. In this context, assessing how a stronger attacker may look to defeat these methods matters. Here, we explore how an *adaptive* attacker (i.e. they know about the underlying defense) tries to defeat IS and REStore, along with the resulting effect on REStore's comparables.

A. Dataset, model, and backdoor setups

We use CIFAR-10 along with the same ResNet-18 architecture as described in App. B and Section IV-A. We use BadNets [18] as our backdoor attack (see Section IV-A).

We follow the same training and checkpoint procedures as described in Section IV-A. CDA and ASR are reported in Table XIV in App. J.

Finally, we focus on the PI IS setup used previously in Section III-B, restricted to the logit-based $scorer$ case as it was the most effective against BadNets [18] for CIFAR-10. The defender is oblivious to the attacker's adaptive setup.

B. Backdooring every classes

So far, we have only backdoored a single class in target DNNs. This translates to a visible outlier pattern in IS-generates maps in Section IV. However, an adaptive attacker may instead backdoor all the classes of a DNN to confound our method's diagnosis step. To do so, we backdoor each class of CIFAR-10 with unique 3-by-3 BadNets [18] triggers (i.e. their patterns and locations have been randomly generated).

1) Observations: As expected, the first step of our method (IS-generated $\tau \to \rho_\tau$ maps, see Section III-D) does not work in this case given that all classes are backdoored. No single class will provide a comparably distinct pattern.

Nonetheless, the second step of our method (IS-generated trigger reconstructions, see Section III-D) is independent from the first step. Therefore, we can recover perturbations with a high ASR for all classes (see Table XV in App. J).

Finally, we find that our final step, REStore, is still effective (see Table XVIin App. J). While BDMAE remains state-of-the-art, we beat Februus in terms of ASR and SDA as the white-box method fails to fully erase backdoor patterns, which triggers misclassifications towards other classes.

2) Takeaway: Our method manages an ASR drop from c. 99% to 12%. However, we note that CDA drops by c. 20%. This highlight a limitation of our paper in this adaptive setup. We surmise the drop is due to noisier reconstructed patterns that, once removed using simple input preprocessing (see Section III-D3), move inputs further away from the manifold of natural images on which a model has been trained.

C. Logit-obfuscated BadNets [18] attack

So far in this paper, our threat model focused on an attacker using data poisoning to inject a backdoor (e.g. they hijack the data collection step). We show in Section III-B that it leads victim DNNs to yield outlier logits.

However, an attacker under a more favorable threat model may be able to provide a backdoored DNN directly to a user instead of relying on data poisoning. The attacker is thus free to train and/or fine-tune the DNN such that evidence of the backdoor in the DNN's outputs is reduced if not erased.

Here we consider an attacker that trains a victim DNN such that the logits of backdoored samples mimick the logits of benign samples from the target class. The attacker thus aims to generate a stealthier backdoor that defeats IS specifically. To do so, we set an attacker who performs data poisoning along

with the manipulation of the training loss. We use the Mean Square Error to anchor the logits of backdoored and benign samples (see details in App. A).

We importantly note here that stronger backdoor attacks than the ones covered in this paper have emerged in the literature (e.g. WB [47]). We do not claim our method is effective against them as no input purification method has yet to be demonstrated against those to the best of our knowledge.

1) Observations: As in the first adaptive setup, the attack defeats the first step of our method by anchoring the logits of backdoored and benign inputs. However, our IS method's $scorer$ function does not maximize the score of samples from the target class. Instead, IS generates patterns that misclassify (with high scores) the samples from different, source classes. This leads IS to succeed in reconstructing patterns that yields a high ASR (see Table XV in App. J).

Finally, REStore is able to purify backdoored inputs under a logit-obfuscated attack (see Table XVIin App. J). As in the first adaptive setup, BDMAE remains state-of-the-art, while REStore beats Februus again.

2) Takeaway: An adaptive adversary who anchors the logits of backdoored samples may succeed against defenses that model a suspicious DNN's logit distribution. However, IS avoids this trap by looking for misclassification patterns. In practice, this results in a drop from c. 91% to 18% in ASR, at the cost of a 10% decrease in CDA (better than in the all-class adaptive setup).

VI. Limitations & Future Work

Matching the performance of more complex defenses. In our experiments with non-adaptive adversaries, we typically observe an average $3-6\%$ drop in CDA for our single-stage defense (see Table III in Section IV). In the adaptive settings that we also cover, the CDA drops as much as 20% (see Section V-B). This is a current limitation of our work. Consequently, future explorations of RES for defending against backdoors should strive to match the clean-data performance of state-of-the-art methods like BDMAE [11].

Stronger backdoors. Existing input purification defenses are typically not effective against non-patch attacks (e.g. SIG [4]). Moreover, such defenses typically target backdoors that only rely on data poisoning and not stronger, sometimes adaptive, attackers with more a favorable threat model. Future work must explore better reconstruction and removal of triggers different from BadNets [18], such as diffuse attacks like SIG or even imperceptible attacks like WB [47].

Backdoors and imbalanced datasets. Using a custom CASIA-Webface dataset questions a potential effect of class imbalance on backdoor attacks and defenses. To the best of our knowledge, this is a new topic that warrants future work.

Smarter $scorer$ **functions**. We show RES can be used to identify backdoors and reconstruct their trigger. However, IS requires a high, but non-prohibitive number of DNN queries. IS takes under two hours to run on a ResNet-18 using a V100 GPU, using c. 2.5m queries (using 5m did not yield empirical improvements). There are likely more parsimonious implementations, e.g. using the score difference between clean and perturbed inputs. Some preliminary tests on MNIST for example show that maximizing a score over all classes at once reconstructs a backdoor trigger (see in App. J, Fig. 12). This cuts the runtime by the class number κ.

Beyond classification. The positive results on local patterns in terms of detection, recovery, and purification may be of use to defend against patch attacks on different tasks such as object detection, or face recognition based on contrastive losses for instance. Future works in this direction have to explore adapting novel $scorer$ functions to such tasks.

Importance Splitting for other defenses. This paper focused on a simple one-stage tool for defending against backdoors. Here, IS reconstructs backdoor triggers with some level of faithfulness. Because of this, we envision that IS may be of use to power other backdoor defenses. IS may enable the trigger reconstruction of methods like Neural Cleanse [27]. Moreover, IS may work as a noise pre-processor for a multi-stage defense (such as BDMAE [11]) where a denoising model is taught to remove IS-generated perturbations from incoming inputs before being sent to a suspicious DNN.

Importance Splitting as a stand-alone. Given its parallel use in DNN robustness certification [13], future works may also explore the use of Importance Splitting alone to the field of backdoor certification defenses, as illustrated by [48].

Detection of UAP. In the case of a CIFAR-10 ResNet-18 backdoored via TrojanNN [23], we find that a benign class shows IS reconstructions with a relatively higher ASR than other benign classes (see in App. J, Fig. 25b). This behavior is likely a UAP. This is supporting evidence that RES may be of use for diagnosing and defending against (high-scoring) UAPs, which are in a sense natural backdoors to DNNs.

VII. Conclusion

In this paper, we demonstrate for the first time that rare event simulation can be used to defend against backdoors in deep neural networks, expanding its use beyond robustness certification [13]. We use **Importance Splitting** as the cornerstone of (i) a novel single-stage, offline, black-box backdoor assessment tool, (ii) a trigger recovery method, and (iii) a lightweight test-time input purification defense, named **REStore**. Importance Splitting shines in its versatility, offering defenders an array of means to help them evidence possibly compromised DNNs in their pipelines, being effective against multiple backdoor setups (e.g. poison-label *and* clean-label poisoning). It is self-contained and does not rely on supplemental DNNs like [11], [29], which move the goalpost of security. Moreover, though it does not match previous state-of-the-art, multi-stage methods, REStore demonstrates that rare event simulation makes lightning-fast input purification against backdoors possible.

Acknowledgment

This publication is supported by the resources of ANR/AID under the Chaire SAIDA ANR-20-CHIA-0011. In addition, we thank the reviewers for their detailed and fruitful comments.

REFERENCES

[1] HuggingFace, "Pytorch image models, scripts, pretrained weights." https://github.com/huggingface/pytorch-image-models, 2023. Accessed: 2023-06-13.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[3] Y. Liu, Y. Xie, and A. Srivastava, "Neural trojans," 2017.

[4] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in cnns by training set corruption without label poisoning," 2019.

[5] A. Bhalerao, K. Kallas, B. Tondi, and M. Barni, "Luminance-based video backdoor attack against anti-spoofing rebroadcast detection," in *IEEE Int. Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, 2019.

[6] Y. Li, Y. Jiang, Z. Li, and S.-T. Xia, "Backdoor learning: A survey," 2022.

[7] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," 2018.

[8] A. AWS, "Aws rekognition." https://aws.amazon.com/rekognition/, 2023. Accessed: 2023-08-22.

[9] Microsoft, "Microsoft azure face api." https://azure.microsoft.com/en-gb/products/cognitive-services/face, 2023. Accessed: 2023-08-22.

[10] Y. Dong, X. Yang, Z. Deng, T. Pang, Z. Xiao, H. Su, and J. Zhu, "Black-box detection of backdoor attacks with limited information and data," in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision*, October 2021.

[11] T. Sun, L. Pang, C. Chen, and H. Ling, "Mask and restore: Blind backdoor defense at test time with masked autoencoder," 2023.

[12] J. Guo, A. Li, and C. Liu, "AEVA: Black-box backdoor detection using adversarial extreme value analysis," in *Int. Conf. on Learning Representations*, 2022.

[13] K. Tit, T. Furon, and M. Rousset, "Efficient statistical assessment of neural network corruption robustness," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 9253–9263, 2021.

[14] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," 2018.

[15] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," 2019.

[16] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," 2020.

[17] Y. Liu, A. Mondal, A. Chakraborty, M. Zuzak, N. L. Jacobsen, D. Xing, and A. Srivastava, "A survey on neural trojans," *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 33–39, 2020.

[18] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," 2019.

[19] B. Wu, L. Liu, Z. Zhu, Q. Liu, Z. He, and S. Lyu, "Adversarial machine learning: A systematic survey of backdoor attack, weight attack and adversarial example," 2023.

[20] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017.

[21] A. Nguyen and A. Tran, "Wanet – imperceptible warping-based backdoor attack," 2021.

[22] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang, "Dynamic backdoor attacks against machine learning models," *IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 703–718, 2020.

[23] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *Network and Distributed System Security Symposium*, 2018.

[24] Q. Zhang, Y. Ding, Y. Tian, J. Guo, M. Yuan, and Y. Jiang, "Advdoor: adversarial backdoor attack of deep learning system," *Proc. of the ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2021.

[25] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," 2018.

[26] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of dnns for robust backdoor contamination detection," 2019.

[27] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE Symposium on Security and Privacy*, 2019.

[28] S. Udeshi, S. Peng, G. Woo, L. Loh, L. Rawshan, and S. Chattopadhyay, "Model agnostic defence against backdoor attacks in machine learning," 2022.

[29] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Annual Computer Security Applications Conference*, dec 2020.

[30] N. B. Erichson, D. Taylor, Q. Wu, and M. W. Mahoney, "Noise-response analysis of deep neural networks quantifies robustness and fingerprints structural malware," in *SDM*, 2021.

[31] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, "Universal litmus patterns: Revealing backdoor attacks in cnns," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[32] X. Huang, M. Alzantot, and M. Srivastava, "Neuroninspect: Detecting backdoors in neural networks via output explanations," 2019.

[33] E. O. Soremekun, S. Udeshi, S. Chattopadhyay, and A. Zeller, "Exposing backdoors in robust machine learning models," *ArXiv*, vol. abs/2003.00865, 2020.

[34] S. Zheng, Y. Zhang, H. Wagner, M. Goswami, and C. Chen, "Topological detection of trojaned neural networks," 2021.

[35] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks," in *International Joint Conference on Artificial Intelligence*, 2019.

[36] A. K. Veldanda, K. Liu, B. Tan, P. Krishnamurthy, F. Khorrami, R. Karri, B. Dolan-Gavitt, and S. Garg, "Nnoculation: Catching badnets in the wild," in *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, CCS '21, ACM, Nov. 2021.

[37] A. Lagnoux, "Rare event simulation," *Probability in the Engineering and Informational Sciences*, vol. 20, pp. 45 – 66, 2005.

[38] F. Cérou, P. Del Moral, T. Furon, and A. Guyader, "Sequential monte carlo for rare event estimation," *Statistics and Computing*, vol. 22, pp. 795–808, 05 2012.

[39] Y. Bai, Z. Huang, H. Lam, and D. Zhao, "Rare-event simulation for neural network and random forest predictors," 2020.

[40] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, pp. 336–359, oct 2019.

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[42] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009.

[43] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," 2014.

[44] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, pp. 5962–5979, oct 2022.

[45] D. Yin, R. G. Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A fourier perspective on model robustness in computer vision," 2020.

[46] Y. Zeng, W. Park, Z. M. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," 2022.

[47] K. Doan, Y. Lao, and P. Li, "Backdoor attack with imperceptible input and latent modification," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 18944–18957, Curran Associates, Inc., 2021.

[48] Z. Xiang, Z. Xiong, and B. Li, "CBD: A certified backdoor detector based on local dominant probability," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[49] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," 2014.

[50] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," 2017.

[51] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," 2021.

APPENDIX

A. Backdoor attacks

1) BadNets – local pattern [18]: BadNets stamps a small, localized pattern in an image. Formally, given the space of vectorized images $\mathcal{X} = [0,1]^d$ with $d = C \times H \times W$, BadNets

replaces the content of $x^{\mathsf{cl}} \in \mathcal{X}$ with a pattern $\mathtt{t}$ at the location indicated by a Boolean mask M such that:

$$x^{\mathsf{po}} = (1 - M) \circ x^{\mathsf{cl}} + M \circ \mathtt{t}, \tag{6}$$

where $\circ$ is the component-wise multiplication operator.

2) SIG – diffuse signal [4]: SIG blends a global pattern $\mathtt{t}$ in an image x^{cl}, given a blending ratio $\alpha_{\mathsf{atk}} \in (0, 1]$. The pixel values are then clipped to the original x^{cl} range:

$$x^{\mathsf{po}} = \max(\min(x^{\mathsf{cl}} + \alpha_{\mathsf{atk}} \cdot \mathtt{t}, 1), 0). \tag{7}$$

An example trigger pattern $\mathtt{t}$ is the SIG sine wave [4]:

$$\mathtt{t}(i(x, y)) = \sin\left(2\pi(F_x x + F_y y)\right), \tag{8}$$

where $i : [M]^2 \to [d]$ maps the pixel indices of a square image of size M to the indices of a vector of size $d = M^2$, and (F_x, F_y) are the signal's width and height-wise frequencies.

3) WaNet – image warping [21]: WaNet is an attack based on a function $\mathcal{W}$ that warps an image given a warping field M:

$$x^{\mathsf{po}} = \mathcal{W}(x^{\mathsf{cl}}, M). \tag{9}$$

In order for the warping field M to trigger the backdoored DNN f_θ^{po}, the attacker trains f_θ^{po} through three distinct modes: clean, attack, and noise. The clean and attack modes correspond to a standard poison-label backdoor setup. The additional noise mode introduces Gaussian noise to M and trains f_θ^{po} to classify $\mathcal{W}(x^{\mathsf{cl}}, M + \mathcal{N}(0, 1))$ as a clean input of class y^{cl}. This mode guarantees that the field M remains a unique trigger.

4) Logit obfuscation: An attacker may adopt a logit regularization step (which we explore in Section V) to anchor the logits of training samples (backdoored with some trigger $\mathtt{t}$) to the logits of clean samples of the target class such that:

$$\begin{aligned}\mathcal{L}_{total}((x^{\mathsf{po}}, y^{\mathsf{po}})) = &(1 - \lambda) \cdot \mathcal{L}_{CE}((x^{\mathsf{po}}, y^{\mathsf{po}})) \\ &+ \lambda \cdot MSE(f_\theta^{\mathsf{po}}(x^{\mathsf{po}}), f_\theta^{\mathsf{po}}(\hat{x})),\end{aligned} \tag{10}$$

where $(x^{\mathsf{po}}, y^{\mathsf{po}})$ is a backdoored datapoint, $\hat{x}$ is a benign datapoint of the target class y_i^{po}, $\mathcal{L}_{CE}$ is the cross-entropy loss, MSE is the mean square error of the logits of x^{po} and $\hat{x}$ forwarded through the network f_θ^{po}, and $\lambda \in [0, 1]$ is a weighting parameter. In practice, we set $\lambda = 0.05$. The datapoint $\hat{x}$ is chosen randomly within the same minibatch as x^{po}. If no $\hat{x}$ exists, we use the previously chosen one (it is memoized).

B. Datasets used in this paper

Description. MNIST contains $70,000$ grayscale 28-by-28 handwritten digits and is split between $60,000$ training samples and $10,000$ testing samples. CIFAR-10 contains $60,000$ RGB-colored 32-by-32 images of vehicles and animals, and is split between $50,000$ training samples and $10,000$ testing samples. Both datasets are equally divided between 10 classes. CASIA-Webface is a face recognition dataset containing about $10,000$ subjects and $500,000$ RGB-colored 112-by-112 images. We select the subjects with the most samples in CASIA-Webface and build a 200-class custom dataset of c. $52,000$ elements. The resulting dataset is imbalanced with the minimum/maximum/mean/median number of samples per class being: 170, 308, 225, and 216.

Hold-out splits. For MNIST and CIFAR-10 respectively, $10,000$ and $5,000$ training datapoints are randomly set aside for validation. For the custom CASIA-Webface dataset, $45,000$ samples are set aside for training and the rest is split between validation and testing. For all datasets, the pixel values are re-ranged to $[0, 1]$.

C. Training regimens and checkpoint policies

Training policies. Benign DNNs and DNNs backdoored with BadNets [18] or SIG [4] are trained for 100 epochs, with an initial learning rate 0.001 (divided by 10 at epochs $\{50, 75\}$) using the Adam optimizer, and a batch size of 128. Models backdoored with WaNet [21] are trained for 500 epochs with an initial learning rate 0.01 (divided by 10 at epochs $\{100, 200, 300, 400\}$) using the SGD optimizer, and a batch size of 32.

For data augmentation, We normalize MNIST inputs. For CIFAR-10, we use the following: (i) Random crop ($pad = 4$), (ii) random horizontal flip ($p = 0.5$), (iii) random rotation of (≤ 10 degrees), (iv) random affine transformation with shear ≤ 10 degrees and scaling interval $[0.8, 1.2]$, (v) color jitter with brightness, contrast, and saturation set to 0.2, and (vi) normalization. For CASIA-Webface, we use the following: (i) random horizontal flip ($p = 0.5$), and (ii) normalization.

Backdoored model checkpoint policy. We keep the backdoored DNNs that maximizes a trigger's ASR, given a validation CDA no lower than 2% versus their benign counterparts. The test CDA and ASR are reproduced in Table I.

Hardware & software. We use 1 NVidia V100 GPU for our experiments, relying on the PyTorch v1.13.1 and NumPy v1.24.1 libraries.

D. Rationale for events A, B, and C being disjoints

This choice is underlined, first, by noting that a DNN that learns onto the manifold of typical inputs (i.e. the manifold of natural images) may exhibit uncontrolled behaviors outside of it (Nguyen et al. [49] confirm the existence of high-scoring noise). Thus, type B events stand outside of the manifold of images classified by f_θ and their alterations, which types A and C events cover. Type B events are therefore disjoint from A and C. Secondly, we surmise that type A and C events are also disjoint by definition as the latter include natural defects in a DNN rather than crafted and maliciously-injected behaviors [2], [18], [19]. We subsume the case of UAPs [50] in type C (we discuss in Section VI how IS may extend to discovering some of them).

E. Overview of Importance Splitting (IS)

Definition of a rare event. Given a random vector $X \in \mathbb{R}^d$ following a distribution $\mathcal{L}$, a score function $scorer : \mathbb{R}^d \to \mathbb{R}$ and a given threshold $\tau \in \mathbb{R}$, we define a *rare event* $\mathcal{E}$ as:

$$\mathcal{E} = \left\{x | scorer(x) > \tau\right\} \subset \mathbb{R}^d \tag{11}$$

Importance Splitting. The Sequential Monte-Carlo sampling procedure named Importance Splitting (IS) [38] defines a series of regions $\mathcal{E}_j \subset \mathbb{R}^d$ until the region $\mathcal{E}$ is reached s.t.:

$$\forall j \in \{0, \dots, J\},\ \mathcal{E}_j = \big\{x | scorer(x) > \tau_j\big\} \tag{12}$$

with $-\infty = \tau_0 < \tau_1 < \ldots < \tau_J = \tau$. That is, the series is composed of nested events: $\mathbb{R}^d = \mathcal{E}_0 \supset \mathcal{E}_1 \supset \cdots \supset \mathcal{E}_J = A$. We define $P_j = \mathbb{P}(X \in \mathcal{E}_j)$ and $P_0 = 1$ such that $\forall j \in [J]$:

$$P_j = \mathbb{P}(\mathcal{E}_j \cap \mathcal{E}_{j-1}) = \mathbb{P}(\mathcal{E}_j | \mathcal{E}_{j-1}) \cdot \mathbb{P}(\mathcal{E}_{j-1}) \tag{13}$$

$$= \prod_{i=1}^{j} \mathbb{P}(\mathcal{E}_i | \mathcal{E}_{i-1}) \tag{14}$$

Initialization. IS works by iteratively updating a pool of n elements drawn from a distribution $\mathcal{L}$ (accessible through a $generator$ function). This pool is initially given by a standard Monte-Carlo sampling. Once drawn, the n elements are sorted by their respective score computed with $scorer$. The $n-k$ lower score becomes the initial estimated threshold $\hat{\tau}_1$ defining the region $\mathcal{E}_1$ such that $P_1 = \frac{k}{n}$ (k samples have a score strictly higher than $\hat{\tau}_1$). The $n-k$ lowest scoring samples are discarded.

Refresh procedure. The missing $n-k$ elements are refreshed by sampling the distribution $\mathcal{L}$ conditioned on $\mathcal{E}_1$. This procedure is done via a reversible transition (mixing) function $kernel : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}^d$ with respect to the distribution $\mathcal{L}$. This function is parametrized by a strength parameter $s > 0$ and holds two properties: (i) $kernel(X, s) \sim \mathcal{L}$, and (ii) the transition distribution from X to $kernel(X, s)$ is identical to that from $kernel(X, s)$ to X, i.e., $kernel$ is $\mathcal{L}$-invariant.

A single pass (at a given step j) of the refresh procedure is reproduced in Alg. 1. For each of the $n-k$ elements to be refreshed, the algorithm samples uniformly-at-random one candidate from the k survivors (line 3). The candidate is refreshed through the kernel function T times, each time computing its corresponding score through the $scorer$ function (lines 5 and 6). The refreshed candidate follows the distribution $\mathcal{L}$ conditioned on $\mathcal{E}_{j-1}$. The refreshed candidate is saved if its score is above $\hat{\tau}_{j-1}$ (line 8), otherwise it is rejected.

Once the $n-k$ elements have been refreshed, scores are computed for the n elements. They are then sorted in descending order and the $n-k$ lowest score sets the new estimated threshold $\hat{\tau}_j$. This threshold now defines the region $\mathcal{E}_j$ with probability $(\frac{k}{n})^j$ and the $n-k$ lowest scoring samples are discarded. Finally, the algorithm starts the next iteration, now conditioned on $\mathcal{E}_j$.

Setting a rejection rate. The rejection rate depends on the kernel's strength s. A large s leads to highly-diverse refreshed elements, with an accompanying high rate of rejection, while a small s lowers both the diversity and rejection of the refreshed samples. The defender strikes a balance by performing a clever control of s between different passes of IS using a $decay \in (0, 1)$ rate (line 10): s is increased to $\frac{s}{decay}$ if the rejection rate is too low or decreased to $decay \cdot s$ if the rate is too high.

Algorithm 1: Importance Splitting, iteration at step j

Input: Set $\mathcal{K}$ of k survivors in $\mathcal{E}_{j-1}$, threshold $\tau_{j-1} \in \mathbb{R}$, $kernel : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ with strength $s \in \mathbb{R}_+$, and $scorer : \mathbb{R}^d \to \mathbb{R}$

Output: Set $\mathcal{R}$ of n refreshed elements in $\mathcal{E}_{j-1}$

1 $\mathcal{R} \leftarrow \mathcal{K}$;
2 **for** $i = 1, 2 \dots, n-k$ **do**
3 $x \xleftarrow{U} \mathcal{K}$;
4 **for** $j = 1, 2 \dots, \mathsf{T}$ **do**
5 $c \leftarrow kernel(x, s)$;
6 $sc \leftarrow scorer(c)$;
7 **if** $sc > \hat{\tau}_{j-1}$ **then**
8 $x \leftarrow c$;
9 $\mathcal{R} \leftarrow \mathcal{R} \cup \{x\}$;
10 update s;
11 **return** $\mathcal{R}$

Iteration and terminal conditions. IS iterates J_{max} times or until the target rare event threshold τ is reached (i.e. $\mathcal{E} = \mathcal{E}_J$).

Notes. In order to minimize the variance of the procedure, the estimators for each nested region $\mathcal{E}_j$ should have the same probability such that $\forall j \in \{1, \dots, J\}$, $\hat{\mathbb{P}}(\mathcal{E}_j | \mathcal{E}_{j-1}) = p$ [38]. This is done by fixing k and n from the starting point $\hat{\mathbb{P}}(\mathcal{E}_1 | \mathcal{E}_0) = \frac{k}{n}$ to the end $\hat{\mathbb{P}}(\mathcal{E}) = (\frac{k}{n})^J$. In our experiments (see Sections IV and V), we set $k = \lfloor \frac{n}{2} \rfloor$.

F. Notes on choosing $\mathcal{L}_X$ carefully in order to reconstruct backdoor triggers

To check whether the class c of a DNN f_θ is backdoored, a key ingredient to sampling backdoored inputs with RES is the search space with distribution $\mathcal{L}_\mathcal{X}$. As it impacts the probability of occurrence of types A, B, and C events, the defender must carefully choose $\mathcal{L}_\mathcal{X}$ such that $\mathbb{P}(X \in A) \gg \max(\mathbb{P}(X \in B), \mathbb{P}(X \in C))$. This eases both the backdoor diagnosis and trigger reconstruction steps.

A first attempt makes $\mathcal{L}_\mathcal{X}$ a purely random distribution. That is, the defender probes f_θ with samples from the uniform distribution $\mathcal{L}_\mathcal{X} = \mathcal{U}([0, 1]^d)$. However, this search distribution amounts to discovering the backdoor trigger t via random guesses, which favors sampling realizations of the type B event, which are uncorrelated with t.

A second attempt selects a search distribution whose support follows the manifold of typical inputs. One possible setup is to pick 1 clean test sample $x \in \mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ not classified as the target class c and to consider the search distribution $\mathcal{L}_\mathcal{X}$ as the space of perturbations of x (e.g. $\mathcal{L}_\mathcal{X} = \mathcal{N}(x, \sigma^2 I_d)$ with a small variance σ^2). That is, the defender generates a sample X that is a noisy version of x. The defender's procedure then amounts to finding a perturbation Δ such that the perturbed sample $X = x + \Delta$ is misclassified as class c with a large f_θ score. However, this choice does not work well either because it favors sampling realizations of the type C event. That is,

we discover adversarial perturbations specific to the input x before encountering the backdoor trigger t.

Therefore, choosing pure noise or a perturbation space over 1 clean test sample respectively favor sampling realizations of type B and C events, i.e. the types of realizations to avoid.

G. Choosing $generator$ and $kernel$

Choosing $generator$ and $kernel$ is free with only requirement that $kernel$ be distribution-invariant.

Generators. We define the function $generator : \varnothing \to \mathcal{L}$, which takes no input and draws one sample from a distribution $\mathcal{L}$. In this paper, we cover two $generator$s: Gaussian or Independent and Identically Distributed (IID).

In the Gaussian case $\mathcal{L} = \mathcal{N}(0_d, I_d)$, and in the IID distribution $\mathcal{L} = \Lambda^d$ (i.e. $X_i \overset{i.i.d.}{\sim} \Lambda$). Here, Λ is a hybrid distribution with a $\alpha_{\mathsf{def}} \in (0, 1)$ chance to be drawn from a distribution $\mathcal{V}$ or otherwise be set to an arbitrary value δ_0. Using a IID distribution set as such helps enforce the sparseness of IS-generated perturbations.

The rationale for sparseness is that the defender limits the search space of IS to perturbations close to the manifold of natural images, limiting the realisation of event B (see App. III-C2). In practice, we draw random Boolean masks of the same dimension as inputs from $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ with an increasing proportion of 1s. We then replace the elements of $\mathcal{X}^{\mathsf{cl}}_{\mathsf{test}}$ at the masked indexes with IID uniform draws over the input range $[0, 1]$. The proportion α_{def} is determined such that, when inputs to a DNN are perturbed with sparse random uniform noise, the DNN's CDA is halved. The resulting Boolean mask sparsity ratio is kept for our experiments (see Sections IV and V). This yields a sparsity ratio of 0.28 and 0.27 for the LeNet-5 and ResNet-18 models trained on MNIST, 0.07 for the ResNet-18 trained on CIFAR-10, and 0.08 for the ResNet-50 trained on CASIA-Webface.

Gaussian $kernel$. We define a $\mathcal{L}$-invariant kernel such that $\forall s > 0$:

$$kernel(x, s) = \frac{x + s \cdot N}{\sqrt{1 + s^2}} \tag{15}$$

where N is drawn from the Gaussian $generator$ and $x \perp N$ (independence).

IID $kernel$. We draw $M \sim \{Bernoulli(s)\}^d$ and N from the IID $generator$. Then, $\forall s \in (0, 1)$ we have:

$$\begin{aligned} kernel(x, s) = & \, x \circ (1 - M) \\ & + N \circ (M \wedge (N \neq \delta_0)) \\ & + x \circ (M \wedge (N = \delta_0)) \end{aligned} \tag{16}$$

An average of $s \cdot d \cdot \alpha_{\mathsf{def}}$ sparse components are redrawn given $\mathcal{V}$ accessible via $generator$ and its distribution $\mathcal{L}$.

Note/Rationale on FI **setup**. The search space found in the FI importance splitting setup accounts for the observations found in [45], [46]. That is, high frequencies in an image are characteristic of adversarial perturbations (type-C event) and sophisticated backdoor attackers will typically avoid leaving high-frequency artifacts in their trigger payload. We therefore remove the one-third highest DCT coefficients from the FI search space. In our experiments, we also find that FI regularly blanks out input images by perturbing their lowest DCT coefficients.

H. Perturbation functions g used in the IS $scorer$ functions

Replacement in the pixel-space (found in the PI **setup)**. PI performs pixel replacements in an image given a perturbation Δ drawn from the IID $generator$ with $\mathcal{V} = \mathcal{U}([0, 1])$ and $\delta_0 = -\infty$:

$$g_{PI}(x, \Delta)_i = \begin{cases} x_i & \text{if } \Delta_i = \delta_0 \\ \Delta_i & \text{otherwise} \end{cases} \tag{17}$$

Replacement in the DCT-space (found in the FI **setup)**. FI performs the same as with PI but in the frequency space with $\mathcal{V}$ being the Laplacian distribution with scaling factor b_i corresponding to the given coefficient to be replaced. The function then clips the result of the inverse DCT to the range $[0, 1]$.

Addition in the pixel-space (found in the PG **setup)**. Given an amplitude $\lambda \in [0, 1]$, PG blends a perturbation $\Delta \sim \mathcal{N}(0_d, I_d)$ in a clean image such that:

$$g_{PG}(x, \Delta) = \max\left(\min\left(x + 2\lambda \cdot (\Phi(\Delta) - 1/2), 1\right), 0\right) \tag{18}$$

where $\Phi(.)$ is the element-wise, cumulative distribution function (CDF) of the standard normal distribution.

I. Cost of Running IS

In practice, with the parameters listed in Section IV-B and noted in Section VI, the maximum number of queries per class for a given model amounts to c.$2.5m$ with an average of c. $1.7m$ queries.

This is a relatively low, and economical cost compared to training a Masked Auto-encoder as in BDMAE [11], which are typically trained for several hundred epochs over datasets of up to millions of samples [51].

J. Supplemental Tables & Figures

TABLE (V) All Februus and BDMAE setups ran in this paper, based on the implementation found in [21] (Abbreviations. LN5: LeNet-5; RN18/50: ResNet-18/50; BN: BadNets; WN: WaNet; TRJ3: TrojanNN 3x3-sized trigger; TRJ7: TrojanNN 7x7-sized trigger; BN DYN: Dynamic BadNets).

Dataset	Network	Trigger	**Februus** (XGradCam) mask=0.6			**Februus** (XGradCam) mask=0.8			**Februus** (GradCAM++) mask=0.6			**Februus** (GradCAM++) mask=0.8			**BDMAE** (base)			**BDMAE** (large)		
			CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR
MNIST	LN5	BN	80.4	1.5	98.5	95.0	1.5	98.5	80.4	60.3	39.1	96.0	52.0	47.4	98.1	98.3	0.2	98.4	97.9	0.2
		SIG	64.4	0.0	100	90.6	0.0	100	69.5	0.0	100	93.2	0.0	100	98.3	0.0	100	98.5	0.0	100
		WN	28.7	1.1	98.4	83.1	7.3	92.0	39.3	4.8	94.4	89.0	9.3	90.3	77.8	8.9	91.0	92.3	11.6	88.2
	RN18	BN	97.2	0.0	100	98.8	0.0	100	95.9	80.4	5.0	98.6	93.6	2.5	99	98.9	0.0	99.0	99.0	0.1
		SIG	96.9	9.0	100	98.3	0.0	100	93.9	65.6	27.4	98.5	25.0	74.3	99.0	0.0	100	99.0	0.0	100
		WN	96.2	4.1	95.6	98.2	2.8	97.2	93.3	5.0	94.5	98.0	3.1	96.9	86.1	3.6	96.4	95.0	3.7	96.3
CIFAR-10	RN18	BN	78.7	0.9	99.0	89.9	0.14	99.9	82.0	48.1	50.4	90.0	42.6	56.3	92.2	92.6	1.3	92.8	93.1	1.4
		BN CL	80.2	75.3	18.2	88.9	78.5	15.0	75.8	66.7	29.3	88.1	64.8	31.3	90.3	81.2	10.4	90.5	82.3	9.9
		SIG	81.0	4.9	94.1	90.5	1.8	98.1	83.3	4.8	94.6	91.8	1.4	98.5	92.5	1.1	98.9	92.6	1.1	98.9
		SIG CL	82.9	22.6	69.4	90.7	16.5	80.4	79.9	20.5	69.3	89.2	17.2	78.4	91.9	16.5	80.4	91.7	16.5	81.0
		WN	82.9	34.8	60.9	92.4	20.4	78.5	72.4	45.0	43.6	89.2	27.2	71.5	94.1	78.2	18.2	94.0	79.7	16.0
		TRJ3	77.5	74.2	20.4	86.0	2.4	97.6	61.6	55.7	41.8	81.9	2.2	97.8	86.9	87.1	2.4	87.0	86.0	2.3
		TRJ7	77.4	67.1	16.0	84.8	72.2	16.9	61.1	85.6	2.8	81.9	81.8	7.7	86.7	75.6	12.0	86.5	75.8	12.4
		BN DYN	82.4	70.7	24.2	90.0	47.4	49.2	79.0	3.4	96.6	90.4	2.2	97.7	91.8	90.4	1.9	92.1	91.2	1.4
CASIA	RN50	BN	83.7	0.0	100	92.4	0.0	100	89.7	3.9	96.1	93.9	0.0	100	92.6	92.9	0.0	91.8	93.2	0.0
		SIG	75.0	0.1	99.7	91.5	0.0	100	90.7	0.1	99.9	92.6	0.0	100	93.0	0.0	100	91.6	0.0	100
		WN	72.9	11.4	88.4	91.4	7.9	92.1	80.2	18.1	81.5	91.7	8.7	91.2	85.4	6.5	93.4	93.2	40.3	58.0

TABLE (VI) REStore Input Purification Defense and 2 comparables: Februus [29] and BDMAE [11]. (**Results on backdoored ResNet-18 models trained on CIFAR-10**). **Bold** indicates the best reduced ASR over all defense types.

Trigger	**Februus** (XGradCam) mask=0.8			**Februus** (GradCAM++) mask=0.8			**BDMAE** (base)			**BDMAE** (large)			**IS** with Logits-based $scorer$			**IS** with Probits-based $scorer$			**IS** with Hard-Label -based $scorer$		
	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR
BadNets (clean-label)	88.9	78.5	15.0	88.1	64.8	31.3	90.3	81.2	10.4	90.5	82.3	**9.9**	88.1	34.2	64.4	91.9	78.0	17.9	91.8	77.7	18.0
SIG (clean-label)	90.7	16.5	80.4	89.2	17.2	78.4	91.9	16.5	80.4	91.7	16.5	81.0	83.1	39.2	**48.8**	65.5	38.4	37.7	92.8	19.3	77.1
TrojanNN (3x3 pattern)	86.0	2.4	97.6	81.9	2.2	97.8	86.9	87.1	2.4	87.0	86.0	**2.3**	87.6	84.0	8.5	87.6	84.6	8.0	87.5	85.0	7.4
TrojanNN (7x7 pattern)	84.8	72.2	16.9	81.9	81.8	**7.7**	86.7	75.6	12.0	86.5	75.8	12.4	87.6	56.5	35.4	88.8	56.7	35.9	87.8	57.2	33.3
Dynamic BadNets	90.0	47.4	49.2	90.4	2.2	97.7	91.8	90.4	1.9	92.1	91.2	**1.4**	93.2	0.0	100.0	93.2	0.0	100.0	92.3	0.0	100.0

TABLE (VII) REStore Input Purification Defense and 2 comparables: Februus [29] and BDMAE [11]. (**Results on backdoored RN18/50 models trained on CIFAR-10/CASIA-Webface** using a **grayscale** PI **setup**). **Bold** indicates the best reduced ASR over all defense types.

Trigger	**Februus** (XGradCam) mask=0.8			**Februus** (GradCAM++) mask=0.8			**BDMAE** (base)			**BDMAE** (large)			**IS** with Logits-based $scorer$			**IS** with Probits-based $scorer$			**IS** with Hard-Label -based $scorer$		
	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR
WaNet (CIFAR-10)	92.4	20.4	78.5	89.2	27.2	71.5	94.1	78.2	18.2	94.0	79.7	**16.0**	89.2	68.2	25.1	93.4	13.8	85.9	93.3	23.1	75.9
WaNet (CASIA-W.)	91.4	7.9	92.1	91.7	8.7	91.2	85.4	6.5	93.4	93.2	40.3	58.0	93.6	83.1	11.4	93.0	86.0	**7.5**	94.2	6.3	93.6

TABLE (VIII) Recovered triggers' ASR on the backdoored class (**logit-based** *scorer*). Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PI	FI
MNIST	LeNet-5	BadNets	**100%**	98.85%
		SIG	**100%**	99.81%
		WaNet	5.7%	**30.9%**
	ResNet-18	BadNets	**100%**	3.4%
		SIG	**100%**	**100%**
		WaNet	**62.1%**	18.4%
CIFAR-10	ResNet-18	BadNets	**72.8%**	4.9%
		SIG	**55.7%**	3.2%
		WaNet	13.3%	**21.4%**
CASIA-Webface	ResNet-50	BadNets	**69.3%**	0.0%
		SIG	**1.6%**	0.0%
		WaNet	34.4%	**38.8%**

TABLE (IX) Recovered triggers' ASR on the backdoored class (**hard-label-based** *scorer*). Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PI	FI
MNIST	LeNet-5	BadNets	**97.0%**	1.1%
		SIG	**97.5%**	89.8%
		WaNet	0.0%	**24.5%**
	ResNet-18	BadNets	**94.8%**	0.0%
		SIG	**100%**	**100%**
		WaNet	**65.7%**	1.7%
CIFAR-10	ResNet-18	BadNets	**63.0%**	3.0%
		SIG	**41.0%**	2.2%
		WaNet	11.2%	**15.0%**
CASIA-Webface	ResNet-50	BadNets	**71.7%**	0.0%
		SIG	**1.5%**	0.0%
		WaNet	21.7%	**41.5%**

TABLE (X) Recovered triggers' ASR on the backdoored class **using the supplemental PG setup**. Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PG (logits)	PG (probits)	PG (hard-labels)
MNIST	LeNet-5	BadNets	**100%**	99.9%	96.6%
		SIG	**100%**	**100%**	**100%**
		WaNet	16.08%	**25.4%**	2.7%
	ResNet-18	BadNets	**100%**	**100%**	91.3%
		SIG	**100%**	**100%**	**100%**
		WaNet	45.6%	**57.1%**	49.2%
CIFAR-10	ResNet-18	BadNets	**24.1%**	19.2%	16.7%
		SIG	**40.1%**	37.2%	26.6%
		WaNet	**58.2%**	57.4%	51.4%
CASIA-Webface	ResNet-50	BadNets	0.0%	**49.6%**	0.8%
		SIG	3.1%	**3.9%**	2.8%
		WaNet	7.8%	9.3%	**9.4%**

TABLE (XI) Recovered triggers' ASR on the backdoored class *poisoned using a* **clean-label** *strategy* (**logit-based** *scorer*). Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PI	FI
CIFAR-10	ResNet-18	BadNets	**53.1%**	2.9%
		SIG	**34.4%**	1.8%

TABLE (XII) Recovered triggers' ASR on the backdoored class *poisoned using a* **clean-label** *strategy* (**probit-based** *scorer*). Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PI	FI
CIFAR-10	ResNet-18	BadNets	**48.5%**	3.1%
		SIG	**29.6%**	1.9%

TABLE (XIII) Recovered triggers' ASR on the backdoored class *poisoned using a* **clean-label** *strategy* (**hard-label-based** *scorer*). Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	PI	FI
CIFAR-10	ResNet-18	BadNets	**45.3%**	1.9%
		SIG	**29.9%**	1.1%

TABLE (XIV) Backdoored models' CDA & ASR in an attacker-adaptive setting

Dataset	Model	Backdoor	CDA	ASR
CIFAR-10	ResNet-18	All classes backdoored	93.9%	99.9%
		Logit-obfuscated	93.5%	91.2%

TABLE (XV) Recovered triggers' ASR in an attacker-adaptive setting. Bold indicates the best recovered ASR for a given model.

Dataset	Model	Backdoor	IS case	PI
CIFAR-10	ResNet-18	All classes backdoored (mean ASR)	Logits	**93.7%**
		Logit-obfuscated	Logits	**82.5%**

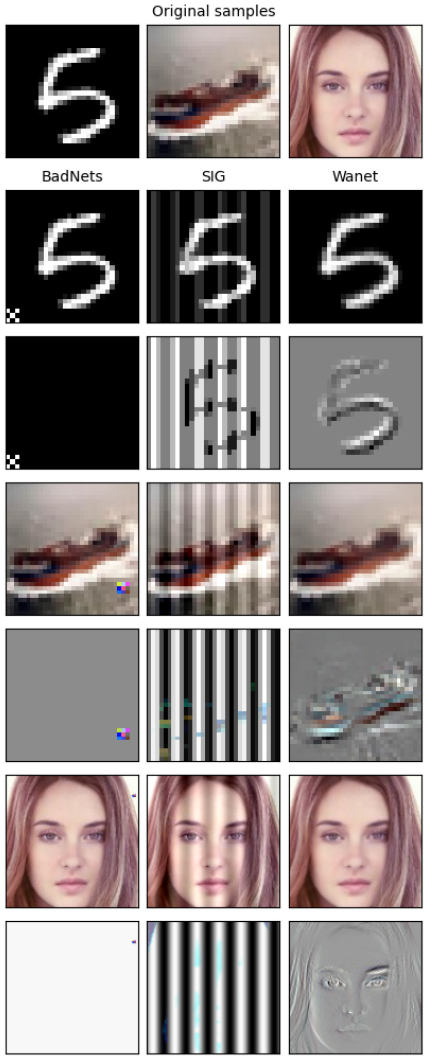


Fig. (10) Samples from the MNIST, CIFAR-10, and CASIA-Webface datasets, when manipulated with a BadNets, SIG, or WaNet backdoor trigger – along with the difference compared to the original.

TABLE (XVI) REStore Input Purification Defense and 2 comparables: Februus [29] and BDMAE [11] (**Results on backdoored ResNet-18 models trained on CIFAR-10 using adaptive attacks**). **Bold** indicates the best reduced ASR over all defense types. Abbreviations: CI, CIFAR-10; R18, ResNet-18; ACB, all classes backdoored; LO, Logit-obfuscation

Dataset	Network	Trigger	**Februus** (XGradCam) mask=0.8			**Februus** (GradCAM++) mask=0.8			**BDMAE** (base)			**BDMAE** (large)			**IS** with Logits-based *scorer*		
			CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR	CDA	SDA	ASR
CIFAR-10	ResNet-18	All classes backdoored	92.0	20.5	8.3	79.8	19.0	8.7	93.4	91.2	1.3	93.3	92.0	**1.0**	73.4	64.8	12.0
		Logit-obfuscation	89.3	52.9	43.7	88.6	76.6	19.5	92.5	85.5	7.4	92.6	85.2	**6.6**	82.6	77.7	17.7

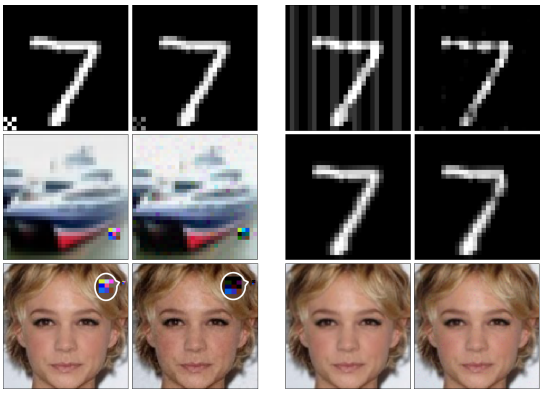
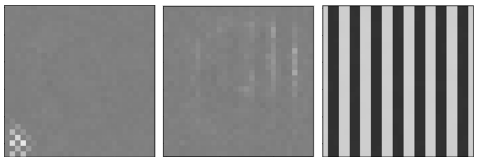
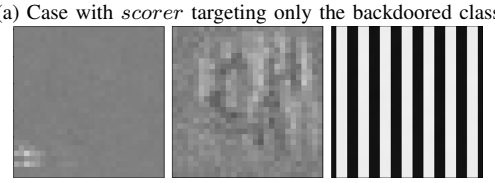


Fig. (11) Example purifications of BadNets (left), SIG on MNIST (top right), and WaNet (center/bottom right) on MNIST and CASIA.



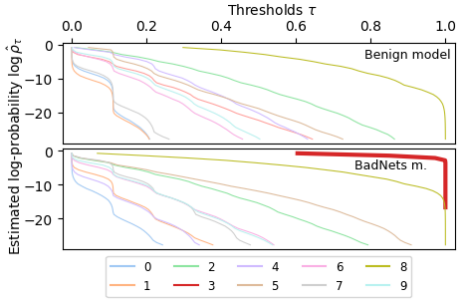
(a) Case with *scorer* targeting only the backdoored class



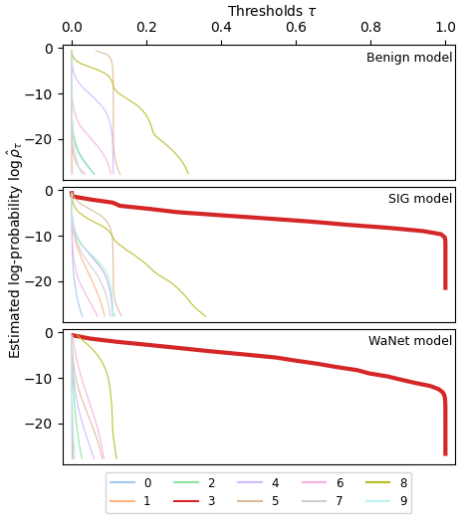
(b) Case with *scorer* maximizing a score over all classes

Fig. (12) Recovered triggers on a LeNet-5 model trained on MNIST, using a probit-based *scorer* function that computes a score for each class or over all classes at once for BadNets [18] (left) and SIG [4] (center, right) using the IS setups (from left to right): PI, PI, FI.

Fig. (13) WaNet [21] synthesis on CIFAR-10 using the IS FI setup with probit-based *scorer* (images in order from left to right: original, backdoored, trigger, reconstructed trigger).

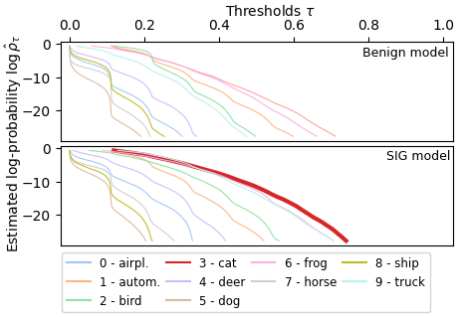


(a) IS PI setup results on LeNet-5 models (benign and BadNets [18]).

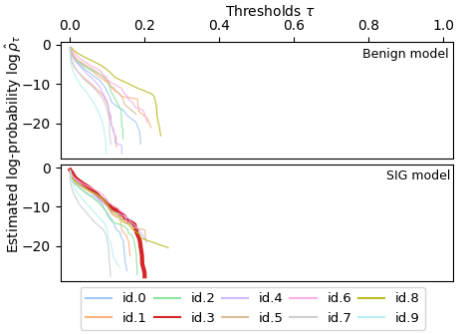


(b) IS FI setup results on LeNet-5 models (benign, SIG [18] or WaNet [21]).

Fig. (14) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**probit-based** *scorer*) on LeNet-5 models trained on MNIST, either benign or backdoored (class 3) with BadNets [18], SIG [4], or WaNet [21].

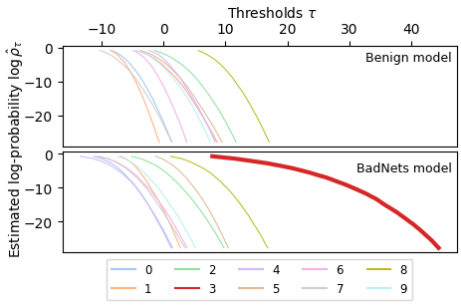


(a) IS PI setup results on ResNet-18 models (benign and SIG [4]).

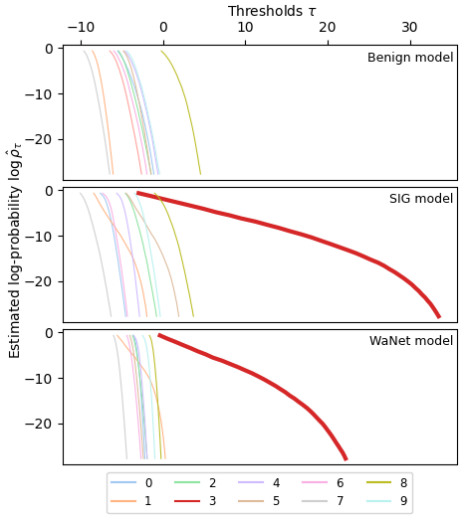


(b) IS PI setup results on ResNet-50 models (benign and SIG [4]).

Fig. (15) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**probit-based** $scorer$) on ResNet-18 and ResNet-50 models respectively trained on CIFAR-10 and CASIA-Webface, either benign or backdoored with SIG [4].

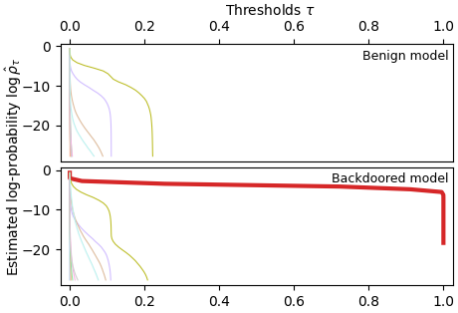


(a) IS PI setup results on LeNet-5 models (benign and BadNets [18]).

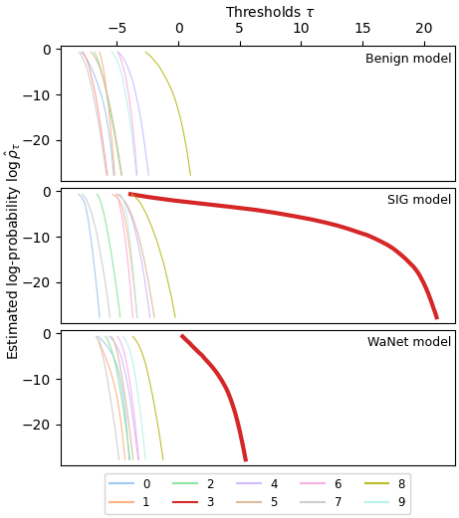


(b) IS FI setup results on LeNet-5 models (benign, SIG [18] or WaNet [21]).

Fig. (16) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**logit-based** $scorer$) on LeNet-5 models trained on MNIST, either benign or backdoored (class 3) with BadNets [18], SIG [4], or WaNet [21].

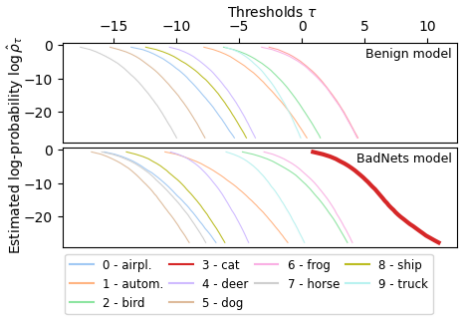


(a) IS PI setup results on ResNet-18 models (benign and BadNets [18]).

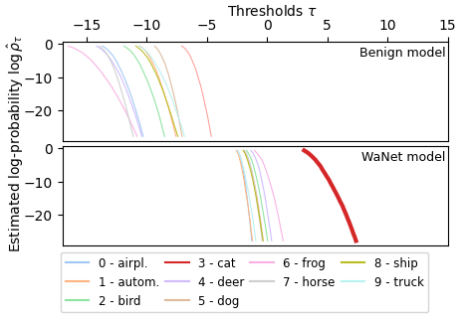


(b) IS FI setup results on ResNet-18 models (benign, SIG [18] or WaNet [21]).

Fig. (17) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**logit-based** *scorer*) on ResNet-18 models trained on MNIST, either benign or backdoored (class 3) with BadNets [18], SIG [4], or WaNet [21].

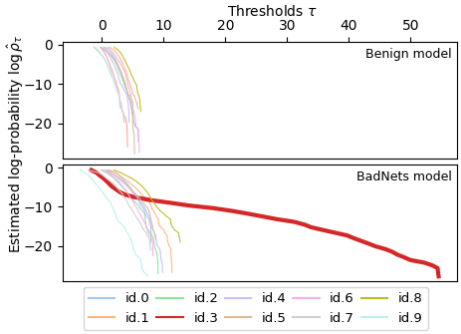


(a) IS PI setup results on ResNet-18 models (benign and BadNets [18]).

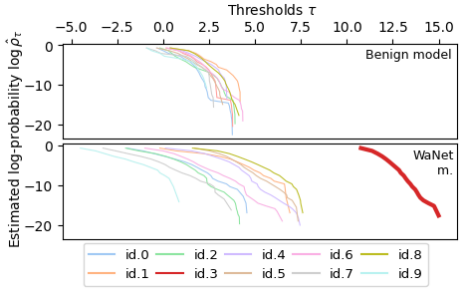


(b) IS FI setup results on ResNet-18 models (benign and WaNet [21]).

Fig. (18) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**logit-based** *scorer*) on ResNet-18 models trained on CIFAR-10, either benign or backdoored (class 3) with BadNets [18] or WaNet [21].

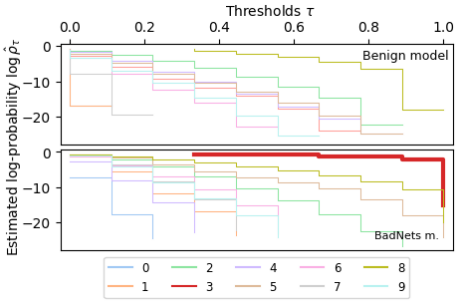


(a) IS PI setup results on ResNet-50 models (benign and BadNets [18]).

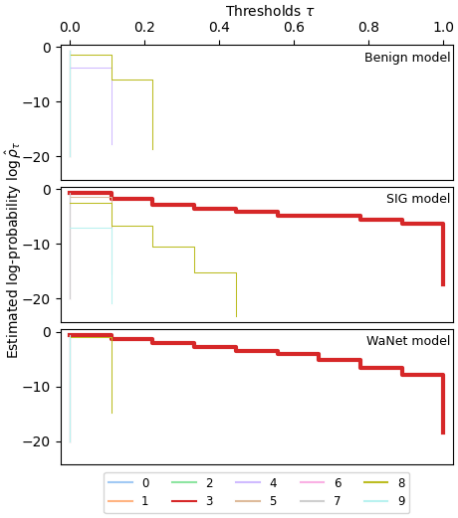


(b) IS FI setup results on ResNet-50 models (benign and WaNet [21]).

Fig. (19) Estimated $\tau \rightarrow \rho_\tau$ IS maps (**logit-based** *scorer*) on ResNet-50 models trained on CASIA-Webface, either benign or backdoored (id. 3) with BadNets [18] or WaNet [21]. Only the first ten classes/identities are displayed.

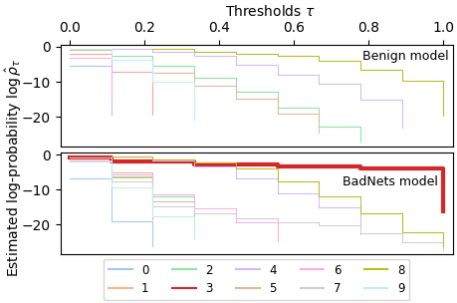


(a) IS PI setup results on LeNet-5 models (benign and BadNets [18]).

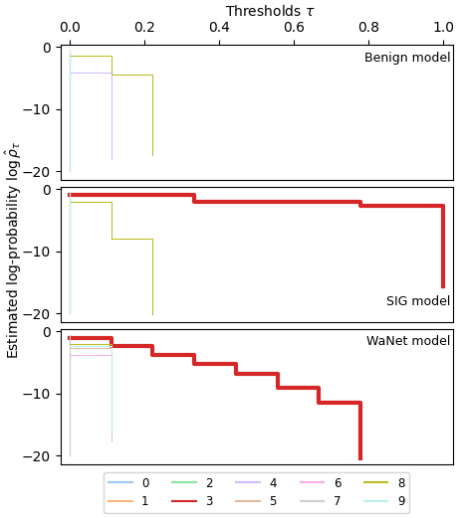


(b) IS FI setup results on LeNet-5 models (benign, SIG [18] or WaNet [21]).

Fig. (20) Estimated $\tau \to \rho_\tau$ IS maps (**hard-label-based** *scorer*) on LeNet-5 models trained on MNIST, either benign or backdoored (class 3) with BadNets [18], SIG [4], or WaNet [21].

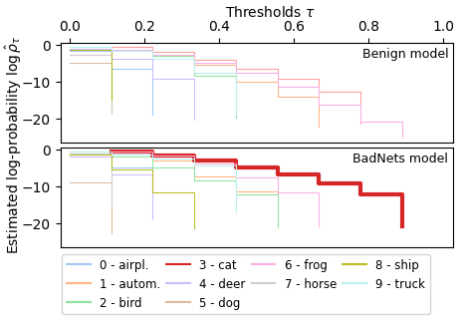


(a) IS PI setup results on ResNet-18 models (benign and BadNets [18]).

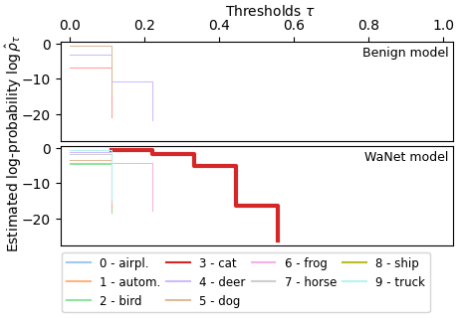


(b) IS FI setup results on ResNet-18 models (benign, SIG [18] or WaNet [21]).

Fig. (21) Estimated $\tau \to \rho_\tau$ IS maps (**hard-label-based** *scorer*) on ResNet-18 models trained on MNIST, either benign or backdoored (class 3) with BadNets [18], SIG [4], or WaNet [21].

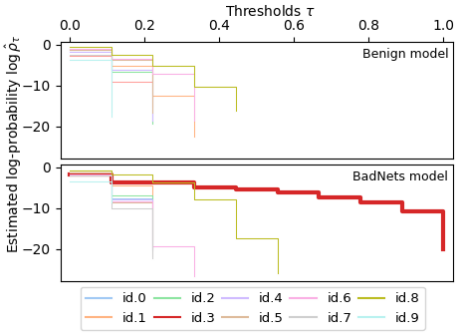


(a) IS PI setup results on ResNet-18 models (benign and BadNets [18]).

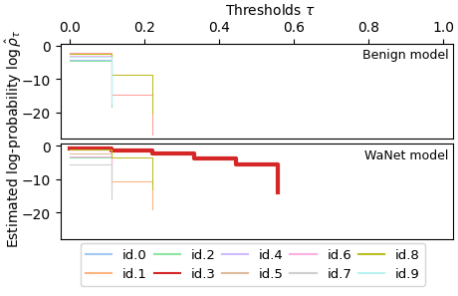


(b) IS FI setup results on ResNet-18 models (benign and WaNet [21]).

Fig. (22) Estimated $\tau \to \rho_\tau$ IS maps (**hard-label-based** *scorer*) on ResNet-18 models trained on CIFAR-10, either benign or backdoored (class 3) with BadNets [18] or WaNet [21].



(a) IS PI setup results on ResNet-50 models (benign and BadNets [18]).



(b) IS FI setup results on ResNet-50 models (benign and WaNet [21]).

Fig. (23) Estimated $\tau \to \rho_\tau$ IS maps (**hard-label-based** *scorer*) on ResNet-50 models trained on CASIA-Webface, either benign or backdoored (id. 3) with BadNets [18] or WaNet [21]. Only the first ten classes/identities are displayed.

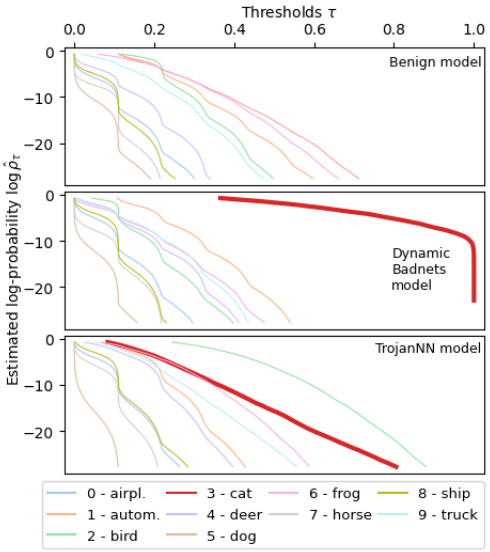
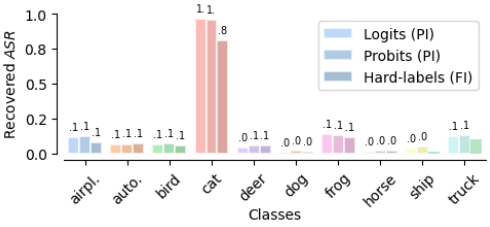
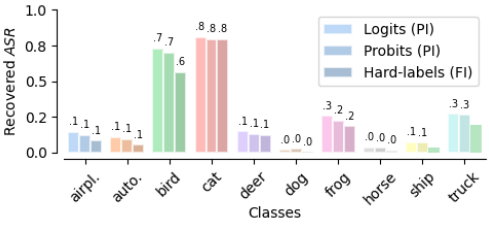


Fig. (24) Estimated $\tau \to \rho_\tau$ IS maps (**probit-based** *scorer*) on ResNet-18 models trained on CIFAR-10, either benign or backdoored (class 3) with a dynamic BadNets [18] or a TrojanNN [23] trigger.



(a) ResNet-18 trained on CIFAR-10 backdoored with the dynamic BadNets



(b) ResNet-18 trained on CIFAR-10 backdoored with TrojanNN.

Fig. (25) Illustration of the distinctiveness of the recovered triggers for the backdoored class 3 on backdoored DNNs trained on CIFAR-10 using either a dynamic BadNets [18] or a TrojanNN [23] trigger (here the 3-by-3 pattern result).