DHBE: Data-free Holistic Backdoor Erasing in Deep Neural Networks via Restricted Adversarial Distillation

Zhicong Yan
zhicongy@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Shenghong Li*
shli@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Ruijie Zhao
ruijiezhao@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Yuan Tian
ee_tianyuan@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Yuanyuan Zhao
yyzhao04@163.com
Hangzhou Normal University
Zhejiang, China

ABSTRACT

Backdoor attacks have emerged as an urgent threat to Deep Neural Networks (DNNs), where victim DNNs are furtively implanted with malicious neurons that could be triggered by the adversary. To defend against backdoor attacks, many works establish a staged pipeline to remove backdoors from victim DNNs: inspecting, locating, and erasing. However, in a scenario where a few clean data can be accessible, such pipeline is fragile and cannot erase backdoors completely without sacrificing model accuracy. To address this issue, in this paper, we propose a novel data-free holistic backdoor erasing (DHBE) framework. Instead of the staged pipeline, the DHBE treats the backdoor erasing task as a unified adversarial procedure, which seeks equilibrium between two different competing processes: distillation and backdoor regularization. In distillation, the backdoored DNN is distilled into a proxy model, transferring its knowledge about clean data, yet backdoors are simultaneously transferred. In backdoor regularization, the proxy model is holistically regularized to prevent from infecting any possible backdoor transferred from distillation. These two processes jointly proceed with data-free adversarial optimization until a clean, high-accuracy proxy model is obtained. With the novel adversarial design, our framework demonstrates its superiority in three aspects: 1) minimal detriment to model accuracy, 2) high tolerance for hyperparameters, and 3) no demand for clean data. Extensive experiments on various backdoor attacks and datasets are performed to verify the effectiveness of the proposed framework. Code is available at https://github.com/yanzhicong/DHBE

CCS CONCEPTS

• **Computing methodologies**;

KEYWORDS

Deep Neural Networks; Backdoor Erasing; Restricted Adversarial Distillation; Data-free.

ACM Reference Format:
Zhicong Yan, Shenghong Li, Ruijie Zhao, Yuan Tian, and Yuanyuan Zhao. 2023. DHBE: Data-free Holistic Backdoor Erasing in Deep Neural Networks via Restricted Adversarial Distillation. In *ACM ASIA Conference on Computer and Communications Security (ASIA CCS '23), July 10–14, 2023, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3579856.3582822

1 INTRODUCTION

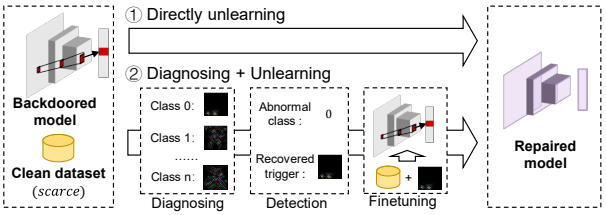
Despite the unprecedented success of DNNs in various machine learning tasks, their reliability has been challenged by various kinds of attacks as a consequence of lacking interpretability of its decisions. Recently, the neural backdoor attacks (a.k.a. trojaning attacks) against DNNs gained extensive attention due to the great threat they brought [26]. This kind of attack aims to construct a conditionally-triggered response between a specific input pattern and the target output desired by the adversary, so that the adversary could mislead the prediction of DNNs for subsequent malevolent activities.

Previous backdoor defense methods focus on filtering training data to thwart attacks via data-poisoning [5, 15, 39, 41]. However, as the supply chain of DNNs becomes increasingly complex, many corresponding defense methods that only focus on the training data are not effective anymore. Modern industrial DNN models are often delivered by third-party training platforms (such as Amazon SegeMaker, Huawei ModelArts, and Baidu PaddlePaddle, etc), the distributed model supply chain offers a new venue for other backdoor variants, such as poisoning the pre-trained models [18, 46, 52], neuron hijacking [31], and even code poisoning [2]. To provide a comprehensive defense scheme to those backdoor attacks, the defense techniques that target the final delivered models, also known as *model diagnosing and backdoor erasing*, provide more promising results and have drawn great research interest in recent years [6, 25, 29, 35, 45]. Many model diagnosing methods [6, 49] have been developed to identify hidden backdoors, but the victim DNN models where backdoors are identified still need to be repaired by backdoor erasing methods. However, as the malicious neurons are entangled with normal neurons in attacked DNN models, the backdoor erasing task is quite challenging.

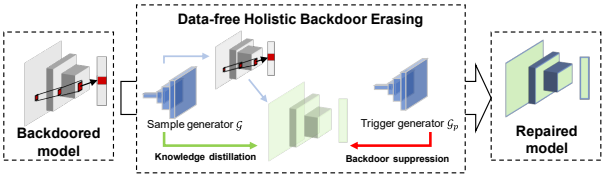
*Corresponding Author.

ASIA CCS '23, July 10–14, 2023, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0098-9/23/07.
https://doi.org/10.1145/3579856.3582822



(a) Unlearning-based Backdoor Erasing



(b) Our Backdoor Erasing Framework

Figure 1: Comparison of unlearning-based backdoor erasing methods and our backdoor erasing framework. We propose to combine data-free knowledge distillation and backdoor suppression in a single framework, so that we can 1) minimize detriment to model's functionality, 2) erase backdoors without access to clean data.

Existing backdoor erasing methods generally use the idea of unlearning where backdoored DNN models are finetuned [45], pruned [28], or distilled [25] to "forget" the backdoor. These methods seem to perform reasonably well in experimental settings, but are problematic in real-world defense scenarios. Two troubles are highlighted: 1) *Hard to find an appropriate setting.* During the unlearning process, the model not only forgets the injected backdoor but also compromises its functionality. As the intensity of unlearning increases (e.g. increasing learning rate, training epochs, etc.), the backdoor gets weaker, but the DNN model suffers from sharp performance degradation. However, different triggers have different resistance to unlearning, which may cause great trouble to the defender since it cannot measure the attack success rate of unknown triggers and can only use the most aggressive unlearning settings to minimize the probability of being attacked. Our evaluations show that Finepruning [28] and NAD [25] have unacceptable model accuracy degradation (10%) on Cifar10 dataset when all employed backdoors are wiped out. 2) *Too depend on high quality dataset.* Unlearning methods are more destructive when the training data is scarce. For example, the NAD [25] has about 5% accuracy degradation holding 4% of all clean data in the experimental scenario, but the accuracy degradation almost doubles when using 1% clean data. Someone may debate that the defender can collect more clean data to minimize the loss of model accuracy, but if the defender can collect enough clean data, why not train a new model from scratch?

To alleviate the negative effect of unlearning on model functionality, existing methods prefer to use a complex pipeline, where an optimization-based trigger reverse-engineering routine is performed first, then unlearning is executed to specifically mitigate the recovered trigger [14, 31, 35, 40, 45]. However, such a surgical pipeline increases the difficulty and complexity of backdoor erasing tasks, but only alleviates the aforementioned problems. What's more, these staged methods are easily compromised by newly-designed attacks with either trigger that is harder to be detected and recovered [21, 22], or composite trigger that contains multiple parts [27, 36, 38, 48].

Our work: In this work, we propose a novel *Data-free Holistic Backdoor Erasing* (DHBE) framework. Instead of unlearning backdoors, the DHBE treats the backdoor erasing task as a unified adversarial procedure, where a clean model is obtained by combining data-free knowledge distillation and backdoor suppression (Fig. 1). Given a backdoored model $\mathcal{T}$, a clean model $\mathcal{S}$ is obtained by solving an optimization problem with two conflicting objectives: 1) *functionality objective*, which minimizes the difference between the outputs of $\mathcal{T}$ and $\mathcal{S}$ over the entire input space, distilling the knowledge from $\mathcal{T}$ to $\mathcal{S}$, and 2) *backdoor suppression objective*, which restricts the expected output change of $\mathcal{S}$ w.r.t. ℓ_1-norm bounded trigger patterns. By jointly optimizing these two objectives based on data-free optimization strategies, the optimized model $\mathcal{S}^*$ finally reaches the desired equilibrium: it inherits the clean data accuracy of $\mathcal{T}$, and shows nearly no response to the trigger hidden in $\mathcal{T}$.

In DHBE, *adversarial distillation* [11] is employed to optimize the functionality objective. A sample generator $\mathcal{G}$ is designed to synthesize samples that have large discrepancies between the $\mathcal{T}$ and $\mathcal{S}$, then the model $\mathcal{S}$ is adversarially trained to minimize the discrepancy on generated samples. By dynamically updating $\mathcal{G}$, $\mathcal{S}$ is always convergent to $\mathcal{T}$ during the optimization, keeping the accuracy of $\mathcal{S}$ always close to $\mathcal{T}$. Along with distillation, we propose *adversarial backdoor regularization* (ABR) to accomplish the backdoor suppression objective. Since backdoor triggers are commonly designed to have a small ℓ_1 norm to avoid being noticed, we define the surrounding space of input x bounded by ℓ_1 distance threshold as the trigger space of x, and the regularization term is proposed as the expected output changes of $\mathcal{S}$ when input x changes within the trigger space. Another adversarial process is designed in ABR: A trigger generator $\mathcal{G}_p$ is optimized to synthesize the most sensitive trigger of $\mathcal{S}$ and $\mathcal{S}$ is adversarially optimized to erase the generated trigger.

Using DHBE, the defender could repair the backdoored model easily and effectively. First, the defender does not need to carefully adjust the hyperparameters. Since the DHBE is based on adversarial optimization rather than unlearning, the equilibrium state of $\mathcal{S}$ is so stable that a wide range of hyperparameters is feasible. Second, the defender does not need to prepare clean datasets. Existing backdoor erasing methods require access to datasets to finetune the neural network, which may lead to privacy leakage; Last, our framework offers a flexible plug-and-play service. For an application supported by multiple types of DNNs from Google, Tiktok, etc., deploying the DHBE is much more practical than deploying existing methods.

We conduct comprehensive evaluations involving four standard image datasets, attack methods with different backdoor injection mechanisms, and different sizes of patch triggers. Our experimental results on the CIFAR10 show that the DHBE can wipe out all patch triggers with less than 1% accuracy degradation, and its effectiveness is stable within a wide range of hyperparameters. In contrast, the unlearning methods cost about 5% model accuracy to erase a 2×2 square trigger and more to erase a 5×5 square trigger. Moreover, we quantified the models' internal response to injected triggers after erasing, and observed that: 1) Unlearning-produced models still have a sensitive response to the trigger and can be attacked by amplified triggers (i.e. inject the same trigger

multiple times to the input). 2) DHBE-produced models have nearly no sensitivity to the trigger, thus cannot be attacked by amplified triggers. Our results indicate that the DHBE not only has much less negative impact on the model's functionality, but is also more secure than unlearning methods.

Our contributions can be summarized as follows:

- We propose a novel Data-free Holistic Backdoor Erasing (DHBE) framework to cure the backdoored model and defend against backdoor attacks. To the best of our knowledge, this is the first backdoor erasing method that does not require extra training data.
- We propose adversarial backdoor regularization to prevent the student model in adversarial distillation from being infected by triggers with small ℓ_1 norms, where a clean model is achieved by reaching the equilibrium state between these two adversarial processes.
- Extensive experiments are performed on standard image datasets to demonstrate the superiority of the proposed defense method against various backdoor attacks.

2 RELATED WORK

2.1 Backdoor Attacks

The backdoor attacks need to temper the DNN model so that the tempered model is sensitive to a specific pattern that could be interpolated into any normal inputs [13, 50]. Attribute to the powerful perception ability of DNNs, various kinds of patterns (also known as "triggers" in backdoor attacks) had been shown to be capable of redirecting tempered models' predictions: 1) Small-size but intense triggers. Badnets [13] had shown that the trigger could be as small as a single pixel or a small square of pixels in a large image, then this kind of trigger was investigated in most subsequent studies [31, 37, 42, 48]. 2) Large-size but invisible triggers. Another line of studies employs a globally but subtle modification to original images as the backdoor triggers. Barni *et al.* [3] superimposed the images with a global sinusoidal signal, and Li *et al.* [22] leveraged steganography where the least significant bits (LSBs) of pixels are substituted. These triggers were implanted with invisible modification of the entire image, and demonstrated more robustness to the state-of-the-art defense methods.

2.2 Backdoor Defenses

Various backdoor defense methods were proposed, which could be categorized as follows according to their defense principle: poison data detection [5, 15, 41], robust learning [24, 51], model diagnosing and backdoor erasing [14, 25, 28, 30, 35, 45], post-deployment input inspecting and augmentation [9, 43].

In this work, we focus on backdoor erasing, which tries to erase the backdoors hidden in backdoored models and recover a clean model that immune to backdoor triggers. Two main kinds of backdoor erasing methods have been investigated:

(a) Blind Erasing Methods: In blind erasing methods, the defender directly employed finetuning, pruning [28], distillation [25] and so on, which could let the model unlearn the backdoor triggers. However, during the backdoor erasing process, useful visual clues were often erased along with the backdoor trigger, leading to an apparent descent of model accuracy.

(b) Targeted Erasing Methods: In targeted methods, the defender first performed backdoor detection routines to recover the possible trigger patterns and the intention of the adversary, then tried to eliminate the recovered trigger by model retraining or pruning. Neural Cleanse (NC) [45] proposed to generate a minimal ℓ_1-norm trigger for each output class, and consider the class with the minimal ℓ_1-norm trigger as the backdoor attacked class. Qiao *et al.* [35] improved NC by modeling the distribution of triggers with a staircase generative model, and Tabor [14] designed a new objective function to find an optimal trigger for each class.

In this work, we propose a unified backdoor erasing framework, and design a data-free optimizing approach with stable analysis and much less damage to the model's accuracy on normal inputs.

2.3 Data-free Knowledge Distillation

Knowledge distillation had been proposed to transfer the performance from a cumbersome model to a small and efficient model [17]. However, it employed a data-driven approach in the distilling process, which was difficult to practice when the training data is scarce or not accessible. To overcome this issue, authors in [7] and [54] tried to generate realistic synthesized images from the trained model that could be used for knowledge transfer. Fang *et al.* [11] designed a data-free adversarial distillation framework, where the training samples were crafted by a generator with the intention of maximizing the teacher-student discrepancy. Since the generator in [11] took a long time to convergent, in [10], a meta-learning method was designed to accelerate the knowledge distillation process.

Data-free knowledge distillation is fast evolving in recent years, and readers are recommended to refer to [33] for the latest advance. In this work, we choose [11] as one component of the proposed framework to show their potential in the field of backdoor erasing tasks due to their power capability of transferring knowledge between different networks.

3 THREAT MODEL AND DEFENSE SETTINGS

Instead of adopting the previous popular settings [6, 25, 35, 45] where a clean or synthetic dataset is available, we define a novel setting termed **Data-free Backdoor Erasing** where the defender holds a DNN model that has been diagnosed as being backdoor attacked with a high probability, but does not have access to clean data that could be used to finetune the model. Formally, the attacked DNN model is denoted as $\mathcal{T} : \mathcal{X} \mapsto \mathbb{R}^{n_c}$, which takes image x with size $H \times W \times C$ as inputs and output a class score vector $q \in \mathbb{R}^{n_c}$. Specifically, the attacked DNN model predicts the operated images as the attacker-desired category t. t is referred to as the target category while other categories are denoted as source categories $\{s_i\}$. The defender's goal is to transform the backdoored DNN model $\mathcal{T}$ into a clean DNN model $\mathcal{S}$ without access to any training data. However, what the defender knows is only the fact that an existing adversary could alter the model's predictions by making a specific operation $\mathcal{A} : \mathcal{X} \mapsto \mathcal{X}$ on the natural inputs $x \in \mathcal{X}$.

Although the only difference between our settings to previous works is the accessibility to clean datasets, we argue that this difference is critical in many real-world defense scenarios. Since people are increasingly concerned about data privacy, building artificial intelligent systems without enough training datasets or without

centralized datasets is more and more common in practice, such as 1) federated learning, where the training data are dispersedly controlled by participants. 2) few-shot learning [44], where each category in its test phase only has a few images that are far from enough for the unlearning process in backdoor erasing. and 3) zero-shot learning [47], where no training samples but auxiliary information is provided. In these scenarios, there is no access to the dataset that could be used for backdoor erasing, but they are potential victims of backdoor attacks. Besides, for many popular tasks such as face recognition, optical character recognition, pedestrian detection, and so on, the most convenient practice is to download models from open-source projects, however, their training datasets may also be unavailable for privacy concerns. What's more, under an adversarial environment, it is difficult to make sure all training data are clean, since more and more invisible backdoor attacks [22, 23, 37] are proposed in recent years, however, using poisoned datasets for backdoor erasing is definitely infeasible [25].

We emphasize here the advantages of the proposed novel settings. **1) Practical**: Our defense frees the defender from dealing with training datasets. Many real-world scenarios such as federated learning, face recognition, and medical image diagnosing are inaccessible to data because of the prohibitive data-collection cost or privacy concerns. **2) Stable**: Previous methods use clean datasets or patched datasets to finetune the backdoored model, where their performance will severely decrease if the dataset is of poor quality (too few samples or imbalanced classes). We believe our work will have more general applications in the foreseeable future.

4 DATA-FREE BACKDOOR ERASING FRAMEWORK

In this section, we first provide an overview of the proposed framework, then the detailed components design is introduced in the next. Finally, we present the overall DHBE framework.

4.1 Overview of Data-free Holistic Backdoor Erasing Framework

In this paper, we propose a novel data-free backdoor erasing framework, which could repair the backdoored model in a single procedure without access to clean data. In the proposed framework, the backdoored model $\mathcal{T}$ is distilled into a clean model $\mathcal{S}$, where a generalized objective function is formulated as:

$$\mathcal{S} = \underset{\mathcal{S}}{\operatorname{argmin}}\ \mathcal{L}(\mathcal{T}, \mathcal{S}) = \underset{\mathcal{S}}{\operatorname{argmin}}\ \mathcal{D}(\mathcal{T}, \mathcal{S}) + \lambda \mathcal{R}(\mathcal{S}). \tag{1}$$

The first term $\mathcal{D}(\mathcal{T}, \mathcal{S})$ is designed to measure the discrepancy between outputs of $\mathcal{T}$ and $\mathcal{S}$. Minimizing this discrepancy term is equivalent to transferring the backdoored model's knowledge to the student model. The second term $\mathcal{R}(\mathcal{S})$ is a regularization term that tries to mitigate the possible backdoors in the student model $\mathcal{S}$. By jointly minimizing these two terms using data-free adversarial optimization, we hope to obtain a student model that inherits the teacher's performance on clean data, but without backdoor reactions.

In DHBE, we design two coupled adversarial processes to simultaneously optimize these two terms ($\mathcal{D}$ and $\mathcal{S}$) in the objective function (Eq.1) over the entire data domain, which we denote as *adversarial distillation* (described in subsection 4.2) and *adversarial backdoor regularization* (described in subsection 4.3) respectively. Finally, we summarize the detailed training process of DHBE in subsection 4.4.

4.2 Adversarial Distillation (AD)

We employ data-free adversarial distillation to transfer the teacher's knowledge to the student, which does not suffer from accuracy degradation caused by incomplete datasets. Intuitively, as the common practice of knowledge distillation, the original backdoored model is fixed and employed as the teacher model $\mathcal{T}$, then a student model $\mathcal{S}$ is optimized to mimic the output of the teacher model. Instead of using fixed training data as inputs, we design a hard sample generator $\mathcal{G} : \mathbb{R}^n \mapsto \mathcal{X}$ to dynamically generate training samples that cause large discrepancies between $\mathcal{T}$ and $\mathcal{S}$ during the training process. In the meantime, the student model $\mathcal{S}$ is adversarially trained to minimize the discrepancy on the generated samples. In our framework, the discrepancy between $\mathcal{T}$ and $\mathcal{S}$ is designed as the expected Mean Absolute Error (MAE) of model's pre-softmax outputs over randomly generated inputs following [11]:

$$\mathcal{D}(\mathcal{T}, \mathcal{S}; \mathcal{G}) = \mathbb{E}_{z \sim p_z(z)} \left[\left\| \mathcal{T}(\mathcal{G}(z)) - \mathcal{S}(\mathcal{G}(z)) \right\|_1 \right], \tag{2}$$

where z is randomly sampled from normal distribution. While the teacher $\mathcal{T}$ is fixed, the sample generator $\mathcal{G}$ and the student $\mathcal{S}$ are iteratively trained to maximize and minimize the objective function respectively. Once the student catches up with the teacher over currently generated samples, the sample generator will move forward to the next confusing space. Finally, the student approaches the teacher over the entire input space, and inherits the teacher's accuracy on clean inputs.

If there is no regularization term to restrict the student model, the student model will learn the backdoor reactions from the teacher as well, since adversarial distillation forces it to comply with the teacher's behavior over the entire input space. In the next subsection, we characterize the backdoor reactions from a geometrical perspective, and propose a regularization term that could comprehensively erase all the backdoor reactions.

4.3 Adversarial Backdoor Regularization (ABR)

In this subsection, we describe the common characteristic of backdoor reactions in terms of how the output of the backdoored model changes when traveling through the input space. By characterizing the backdoor reactions without clean data distribution priori, we are able to distinguish the backdoor reactions from normal reactions in the backdoored model. Finally, we propose a backdoor regularization term and adversarially optimize it to erase the backdoor reactions.

4.3.1 Characteristics of Backdoor Reactions. Currently, backdoor attacks try to design the operation on clean images as subtle as possible to avoid being noticed, which means the backdoored model is forced to be extremely sensitive to triggers with small ℓ_1 distances designed by the adversary. This rule is implicitly abided by various backdoor attacks: attacks with small-size triggers try to modify as small number of pixels as possible, and attacks with large-size triggers try to change the value of pixels as small as possible. More precisely, we formulate this characteristic as follows:

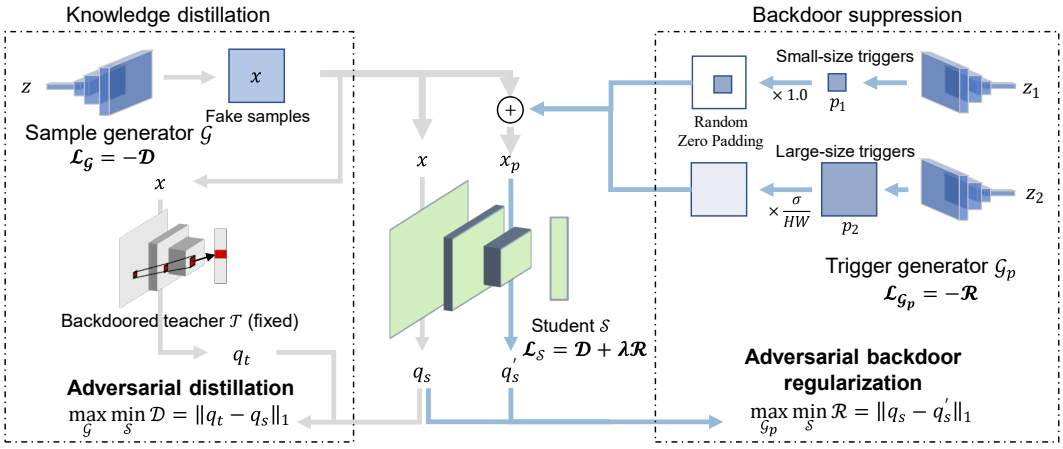


Figure 2: Overview of the Data-free Holistic Backdoor Erasing framework. Two adversarial processes are illustrated. Adversarial distillation is designed to transfer the backdoored model's reactions to the student model, including both the normal reactions and backdoor reactions. Adversarial backdoor regularization is designed to suppress the backdoor triggers in the student model. The student model tries to minimize a combined loss function with two adversarial objectives.

Proposition 4.1. *Let $\rho_s, \rho_t \in Prob(\mathcal{X})$ be the probability measures of source class and target class, and $\rho_{s'}$ be the probability measure of backdoored inputs. Then the Earth-Mover (EM) distance (a.k.a. Wasserstein distance) between ρ_s and $\rho_{s'}$ is far smaller than that between ρ_s and ρ_t:*

$$\mathbb{W}(\rho_s, \rho_{s'}) \ll \mathbb{W}(\rho_s, \rho_t), \tag{3}$$

where the EM distance [1] is defined as the minimum cost of all the possible transport plans ($\Pi(\rho_s, \rho_t)$) from one probability measure to another under ℓ_1 cost function:

$$\mathbb{W}(\rho_s, \rho_t) = \inf_{\gamma \in \Pi(\rho_s, \rho_t)} \mathbb{E}_{(x,y)\sim\gamma}\left[\|x - y\|_1\right]. \tag{4}$$

Table 1: The expected ℓ_1 EM distance between ρ_s and $\rho_{s'}$ in different kinds of backdoor attacks. Here, H,W and C are height, width and channels of input images, h and w are height and width of square trigger patterns.

Trigger Type	Expected ℓ_1 EM Distance
Pixel [13]	≤ 1
Square [31]	$\leq h \times w \times C$
Watermark / Refool [32]	$\leq Opacity \times H \times W \times C$
SIG [3]	$\leq \Delta/\sqrt{2} \times H \times W \times C$
Steganograph [22] (2 bits)	$\leq 3/255 \times H \times W \times C$

Table 2: We approximate the ℓ_1 EM distance between all the class pairs within each dataset using [19] and list the mean and minimal distances and the closest class pairs.

Dataset (Nb. dims)	Closest Class Pair : Dist.	Mean Dist.
MNIST (784)	"7" → "9" : 168.88	495.26
CIFAR10 (3072)	"cat" → "dog" : 282.76	869.83
CIFAR100 (3072)	"leopard" → "tiger" : 292.16	1060.27

The EM distance between ρ_s and ρ_t is hard to tackle, we approximate the distances within standard image datasets using Sliced Wasserstein Distances [19], and list the closest class pairs in Table 2. The EM distance between ρ_s and $\rho_{s'}$ can be calculated as the sum of the total pixel changes by the backdoor operation. We list the distance approximation method of common triggers in Table 1. It can be seen that $\mathbb{W}(\rho_s, \rho_{s'})$ is commonly designed to be far smaller than $\mathbb{W}(\rho_s, \rho_t)$ to keep stealthy. For example, a square trigger for CIFAR10 (32×32) may have up to the size of 10×10 to reach the distance of the closest class pair in CIFAR10, which is too obvious that can be easily discovered.

We note that assuming $\mathbb{W}(\rho_s, \rho_t)$ is large may not hold in some fine-grained classification tasks, such as face recognition, where the distance between two classes is relatively smaller than in other tasks. In the experiment section, we provide experimental results on fine-grained dataset (classification on a subset of classes in VGGFace2).

4.3.2 Backdoor Regularization. To get rid of the backdoor reactions, the student model should be smooth and robust to perturbations with small ℓ_1 distances. To this end, we design a regularization term that forces the student model to predict the same result for any input x and samples in its surrounding space $x' \in \Sigma(x, \varepsilon)$:

$$\mathcal{R}(\mathcal{S}) = \mathbb{E}_{x\sim\mathcal{X}, x'\sim\Sigma(x,\varepsilon)}\left[\left\|\mathcal{S}(x) - \mathcal{S}(x')\right\|_1\right], \tag{5}$$

where $\Sigma(x, \varepsilon)$ is a surrounding space of x bounded by ℓ_1 distance:

$$\Sigma(x, \varepsilon) = \{x' | \|x - x'\|_1 < \varepsilon\}. \tag{6}$$

4.3.3 Adversarial Optimization. Directly optimizing the regularization term is equivalent to training the student model on the production of the sample space $\mathcal{X}$ and the corresponding trigger space Σ, which will fail since the production of two spaces is too large to be optimized. Thus, we propose another coupled adversarial process to optimize this term: On the one hand, the input x is sampled from the sample generator $\mathcal{G}$ used in adversarial distillation. On the other hand, a trigger generator $\mathcal{G}_p : \mathbb{R}^n \mapsto \mathbb{R}^{h\times w\times C}$ is designed to generate triggers with size $h \times w$ that are most sensitive

Algorithm 1 Data-Free Blind Backdoor Erasing

Input: A backdoored teacher model $\mathcal{T}(\cdot, \theta_t)$, batch size BS, λ, α_{tv}, learning rates α_s, α_g, α_{gp}.
Output: A clean student model $\mathcal{S}(\cdot, \theta_s)$.
Initialize the student model's weights θ_s with θ_t.
Randomly initialize the sample generator $\mathcal{G}(\cdot, \theta_g)$ and the trigger generator $\mathcal{G}_p(\cdot, \theta_{gp})$.
for number of training iterations **do**
  for k steps **do**
    Randomly generate BS samples $\{x_i\}$ and BS triggers $\{p_i\}$ with $\mathcal{G}$ and $\mathcal{G}_p$;
    Randomly padding $\{p_i\}$ to the same size of $\{x_i\}$ with zeros;
    $\mathcal{L}_s = 1/BS \sum_i (\|\mathcal{T}(x_i) - \mathcal{S}(x_i)\|_1 + \lambda \|\mathcal{S}(x_i) - \mathcal{S}(x_i + p_i)\|_1)$;
    Update $\theta_s \leftarrow \theta_s - \alpha_s \nabla_{\theta_s} \mathcal{L}_s$;
  end for
  Randomly generate BS samples $\{x_i\}$ with $\mathcal{G}$;
  $\mathcal{L}_g = -1/BS \sum_i (\|\mathcal{T}(x_i) - \mathcal{S}(x_i)\|_1)$;
  Update $\theta_g \leftarrow \theta_g - \alpha_g \nabla_{\theta_g} \mathcal{L}_g$;
  Randomly generate BS samples $\{x_i\}$ and BS triggers $\{p_i\}$ with $\mathcal{G}$ and $\mathcal{G}_p$;
  Randomly padding $\{p_i\}$ to the same size of $\{x_i\}$ with zeros;
  $\mathcal{L}_{gp} = -1/BS \sum_i \|\mathcal{S}(x_i) - \mathcal{S}(x_i + p_i)\|_1$;
  Update $\theta_{gp} \leftarrow \theta_{gp} - \alpha_{gp} \nabla_{\theta_{gp}} \mathcal{L}_{gp}$;
end for

to $\mathcal{S}$, and enforce the following regularization term to mitigate the generated triggers:

$$\mathcal{R}(\mathcal{S}; \mathcal{G}, \mathcal{G}_p) = \mathbb{E}_{x\sim\mathcal{G}, z\sim p_z(z)} \left[\left\| \mathcal{S}(x) - \mathcal{S}\big(x + \mathcal{G}_p(z)\big) \right\|_1 \right]. \quad (7)$$

To constraint the ℓ_1-norm of the trigger generator's output below a given threshold ε, we design a trigger generator with size-fixed outputs ($h \times w$), and multiple the outputs with a constant scalar $s \in (0, 1]$, randomly pad to the same size of x with zeros before finally adding up to the fake sample x. We use $Tanh$ as the output activation layer of $\mathcal{G}_p$, so the max ℓ_1-norm of $\mathcal{G}_p$'s outputs is $h \times w \times s \times C$ (C is image channels).

4.4 Overall DHBE Framework

The overall DHBE framework is illustrated in Fig. 2. , where the student is adversarially optimized with two generators. We summarize the adversarial version of objective function (Eq. 1) as follows:

$$\max_{\mathcal{G}, \mathcal{G}_p} \min_{\mathcal{S}} \mathcal{L}(\mathcal{T}, \mathcal{S}; \mathcal{G}, \mathcal{G}_p) = \max_{\mathcal{G}, \mathcal{G}_p} \min_{\mathcal{S}} \mathcal{D}(\mathcal{T}, \mathcal{S}; \mathcal{G}) + \lambda \mathcal{R}(\mathcal{S}; \mathcal{G}, \mathcal{G}_p), \quad (8)$$

To accelerate the training process, we initialize the student model with the backdoored teacher model, then sequentially train the student and the generators like the training process of GANs [12]. In each iteration, we first update the student model $\mathcal{S}$ k times (same as [11], we set $k = 3$) to minimize the loss function that combines the discrepancy term and the regularization term, then the sample generator $\mathcal{G}$ and the trigger generator $\mathcal{G}_p$ are updated to maximize the discrepancy term and the regularization term respectively. This iteration step is repeated thousands of times and the equilibrium state is achieved by learning rate annealing. We summarize the training process of the proposed DHBE framework in Alg. 1.

5 EXPERIMENTS

In this section, we first describe our experiment settings, then we compare the effectiveness of DHBE with both targeted and blind erasing methods on several well-known backdoor attacks. Finally, we provide comprehensive ablation analyses.

5.1 Experimental Settings

5.1.1 Evaluation Datasets. Four standard image datasets are employed to evaluate the proposed framework, including three standard image datasets that are commonly used in various tasks: CIFAR10, CIFAR100 [20], Mini-Imagenet [8]. Besides, we employ another face recognition dataset, VGGFace2 [4], to show that our framework could perform well on fine-grained tasks.

5.1.2 Configurations for Backdoor Attacks. We employ three backdoor attacks with different backdoor injecting mechanism: Data-poisoning (Badnets[13], Clean-label [42]) and Neuron hijacking (Trojaning [31]). We use 3 different size of triggers for each attack method. For a fair comparison, we reimplement these attacks, and create backdoored models using the same Resnet-18 architecture [16] provided by PyTorch [34]. As a common practice for training small datasets with Resnet-18, the $conv1$ layer ($kernal\ size = 7, stride = 2$) is replaced by $conv$ ($kernal\ size = 3, stride = 1$) and the first *Pooling* layer is canceled to deal with inputs of size 32×32 (i.e. CIFAR10 and CIFAR100 in our experiments). For inputs of size 64×64 (i.e. Mini-Imagenet and VGGFace2 in our experiments), the $conv1$ layer is replaced by $conv$ ($kernal\ size = 5, stride = 2$).

5.1.3 Configurations for Backdoor Erasing Methods: Since our framework is the first attempt in data-free backdoor erasing, we compare its performance with existing data-driven backdoor erasing methods. Specially, we categorize those data-driven methods into blind and targeted erasing methods, and provide detailed comparison experiments with these two kinds of methods respectively.

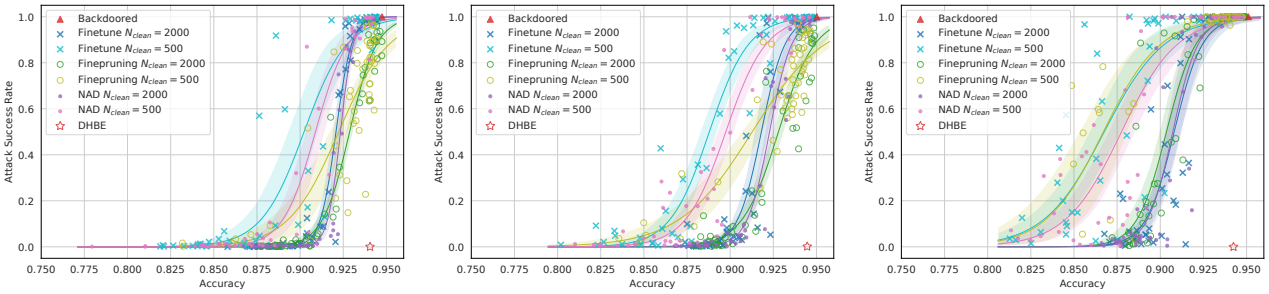
(a) Blind Erasing Methods: We compare our DHBE framework with four existing blind erasing methods: 1) finetuning, 2) finepruning [28], 3) mode connectivity repair (MCR) [55] and 4) neural attention distillation (NAD) [25]. For these data-driven methods, we assume that they all have access to the same 4% of clean training data (2000 samples).

(b) Targeted Erasing Methods: Targeted erasing methods need to determine the attacked class first, then perform the trigger recovery and backdoor erasing process. For a fair comparison on backdoor erasing effectiveness, we assume that those methods are already known the attacked class, then trigger recovery and backdoor erasing are performed on the attacked class. Two targeted erasing methods are employed: NC [45] and GDM [35].

(c) Proposed DHBE: We designed two generators in DHBE: the sample generator $\mathcal{G}$ and the trigger generator $\mathcal{G}_p$ ($h = w = 5, s = 1.0$). The detailed design of these two generators is described in Appendix. Only one hyperparameter is included in DHBE's loss functions : λ in Eq. 8, which controls the degree of adversarial backdoor regularization. We set λ to 0.1 in all experiments. Further analysis of this hyperparameter is included in ablation studies. For optimizers, we globally employ an SGD optimizer with initial learning rate of 0.1, momentum of 0.9, and weight decay of 5e-4 to update the student model, and use an Adam optimizer with initial

Table 3: Comparison results of DHBE to other blind erasing methods on CIFAR10 dataset against different backdoor attacks and different size of triggers. Numbers are displayed as percentages.

Attack Methods	Trigger Size	Backdoored t='truck'		Finetuning N_{clean} = 2000		Finepruning [28] N_{clean} = 2000		MCR [55] N_{clean} = 2000		NAD [25] N_{clean} = 2000		DHBE No data required	
		ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
Badnets [13]	2 × 2	94.75	100.00	90.69	2.64	91.91	12.92	88.70	0.18	90.75	0.38	**94.05**	**0.02**
	3 × 3	95.02	100.00	90.67	2.97	91.48	6.74	89.17	1.92	90.87	7.08	**94.46**	**0.12**
	5 × 5	95.11	100.00	91.47	1.10	92.30	69.07	87.96	1.78	90.41	3.26	**94.24**	**0.00**
Clean-label [42]	2 × 2	94.85	46.32	90.51	1.71	92.33	6.51	93.49	1.75	90.28	1.22	**94.19**	**0.14**
	3 × 3	95.04	93.45	90.48	6.61	90.99	38.95	93.39	3.79	90.21	9.98	**94.11**	**1.17**
	5 × 5	95.25	100.00	91.08	96.41	91.79	67.44	93.74	14.01	90.69	8.57	**94.45**	**4.48**
Trojaning [31]	2 × 2	94.56	56.20	89.75	0.09	91.98	0.90	88.65	0.16	91.24	**0.09**	**93.96**	4.39
	3 × 3	95.00	89.91	90.43	0.17	91.83	4.37	88.90	**0.08**	90.49	1.25	**94.40**	2.88
	5 × 5	94.93	99.91	90.05	17.05	92.03	68.82	87.79	0.84	90.67	**0.55**	**94.56**	1.13
Mean ACC/ASR		94.95	87.31	90.57	14.31	91.85	30.63	90.20	2.72	90.62	3.60	**94.27**	**1.59**



(a) Badnets (2 × 2) (b) Badnets (3 × 3) (c) Badnets (5 × 5)

Figure 3: Trade-off curves between accuracy and attack success rate of different blind erasing methods against different triggers. The curves are drawn by adjusting the learning rate of these methods (except DHBE) from 0.002 to 0.02.

learning rate of 1e-3 to update the generators. The student and generators are jointly optimized for 50 iterations × 300 epochs, where the student is updated by three times and generators are updated once in one iteration. 128 fake samples and triggers are generated in each iteration. The learning rates of these optimizers are simultaneously decayed by 0.1 at epoch 180 and 240. The whole backdoor erasing process takes about 6h on a GTX 2080TI gpu.

5.1.4 Evaluation Metrics. Two metrics are employed to evaluate the quality of backdoor erasing methods: ACC and ASR, For each backdoor erasing method, ACC and ASR are calculated using the cleaned model, and are compared with that of the original backdoored model to judge its performance. A good backdoor erasing method should not only mitigate the ASR metric, but also keep the ACC metric the same as the original backdoored model.

(a) Accuracy on Clean Data (ACC). The test accuracy on the benign test set, which shows the influence of the backdoor erasing process on the functionality of the model.

(b) Attack Success Rate (ASR). the ASR of model $\mathcal{S}$ is defined as the fraction of correctly classified inputs that are not labeled as the target class but misclassified to the target class after the backdoor trigger is injected:

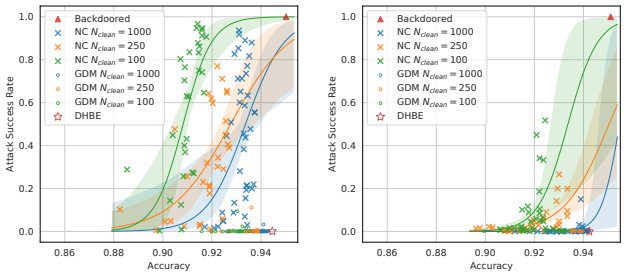
$$ASR(\mathcal{S}) = P\left(\mathcal{S}(x^{\delta}, \theta_s) = t \mid y \neq t, \mathcal{S}(x, \theta_s) = y\right), \quad (9)$$

where (x, y) is a sample from the test dataset and x^{δ} is the input that has been injected with a trigger pattern δ. t is the target class and θ_s is the weights of the model $\mathcal{S}$. Note that this metric only relies on the test data that is correctly classified by the backdoored model.

5.2 Experimental Results

Due to the experimental settings are already complicated by various attack methods and different triggers, in this section, we focus on the CIFAR10 classification task first, and present detailed comparison results and analyses using Resnet18 models trained on the CIFAR10 dataset and attacked by different methods and triggers. We first compare DHBE with existing blind erasing methods. Then, for those targeted erasing methods whose effectiveness are dependent on correctly discover the attacked class, we omit their diagnosing process, directly employ their trigger recovery routines and erasing routines on the backdoored model and report their backdoor erasing performance.

5.2.1 Comparison with Blind Erasing Methods. The comparison results of our framework with four blind erasing methods on different kinds of backdoor attacks are shown in Table 3. As shown in Table 3, our framework outperforms other methods by a large



(a) Badnets (3×3) (b) Badnets (5×5)

Figure 4: Trade-off curves between accuracy and attack success rate of different targeted erasing methods against different triggers. The curves are drawn by adjusting the learning rate of these methods (except DHBE) from 0.0002 to 0.002.

margin on all kinds of backdoor attacks: DHBE only sightly degrades the performance of the original model (less than 1%), and reduce the attack success rate of all triggers to nearly neglectable. In contrast, the results of Finetuning, Finepruning, and NAD has about 4% - 5% accuracy degradation when the learning rate is set to 0.01. Under this setting, the backdoor elimination effectiveness of Finetuning and Finepruning is unstable and failed to suppress ASR below 10% on some triggers. The results of NAD are relatively more stable, the ASRs of different triggers are all suppressed below 10%, but NAD still greatly degrades the model's accuracy. We note that NAD is claimed to have about 2.5% accuracy degradation in their experiments, however, they employ weak baselines where the reported accuracy is below the model's true capabilities. In our experiments, we use models that are trained with moderate data augmentations so that the backdoored model's accuracy reflects the true capabilities of the model. Therefore, the accuracy decay we report is closer to real-world scenarios.

Despite the weak performance of those blind erasing methods, their effectiveness seems to be extremely sensitive to hyperparameters and the quantity of the clean dataset. To clearly demonstrate how sensitive those blind erasing methods are to the aforementioned two factors, we show the ACC-ASR trade-off curves of those methods in Fig. 3: For each method, we run the method on the backdoored model multiple times with the experimental learning rate linearly increases from 0.002 to 0.02, and record the ACC and ASR of the resulted model for each run. We then draw the scatter plot of these results, and use a logistic model to fit the scattered points as the expected ACC-ASR trade-off curve of the evaluated method. Specifically, we evaluate the ACC-ASR trade-off curve of those methods two times with 4% of clean data and 1% of clean data. For each curve, five different random subsets of the clean dataset are used in evaluation. From Fig. 3, the effectiveness and drawbacks of existing blind erasing methods are clearly demonstrated:

(a) Blind erasing methods are extremely sensitive to the learning rate and the quantity of the clean dataset. The points of finetuning, finepruning, and NAD methods in figures are drawn with learning rates in $[0.002, 0.02]$. As shown in the figures, a large learning rate could mitigate ASR to nearly neglectable, but severe accuracy degradation is also observed. However, if a small learning rate is employed, the resulted model still can be attacked with a high ASR. The balancing of the trade-off is hard for the defender since the ASR cannot be measured in real-world applications. Another problem for these blind erasing methods is the quality dataset, when an adequate number of clean data is accessible ($N_{clean} = 2000$), those methods provide moderate backdoor erasing effectiveness at accuracy costs of about 7%. But when the number of clean data is more scarce ($N_{clean} = 500$), the accuracy cost is too much to afford (about 15%). In contrast, the DHBE framework is insensitive to hyperparameters (learning rate, λ) due to its adversarial design, which is demonstrated in ablation studies, and DHBE framework do not require any clean data for backdoor erasing.

(b) Blind erasing methods become less effective as the trigger size increases. When the trigger size increases from 2×2 to 5×5, the blind erasing methods achieve less model accuracy when the ASR is suppressed to the same level. This phenomenon is clearly demonstrated as we show the ACC-ASR curves of 2×2, 3×3 , 5×5 triggers in Figure 3 (a), (b), (c), respectively. For triggers of size 2×2, unlearning methods lost about 5% model accuracy, but for triggers of size 5×5, they lost 8% model accuracy. This may suggest that more neurons are influenced by triggers of large size, causing it hard to be erased by the unlearning methods. In contrast, the proposed DHBE framework appears to be equally effective to different size of triggers.

5.2.2 Comparison with Targeted Erasing Methods. We also evaluate NC [45] and GDM [35] in the same experimental settings. The results are shown in Table 5: DHBE achieves comparable results with Neural Cleanse and GDM against all backdoor attacks. We also show ACC-ASR trade-off curves of those targeted erasing methods in Fig. 4. As shown in those figures, NC and GDM are still sensitive to the employed learning rate, but they achieve much better results than blind erasing methods.

Comparison with Generative Distribution Modeling: Qiao *et al.* [35] discovers that a backdoored model can be triggered by a distribution of triggers, not only a single trigger. Thus, erasing a single recovered trigger using Neural Cleanse is not robust enough. In GDM [35], a sampling-free distribution modeling for valid triggers was proposed, then the backdoor erasing is performed by erasing all triggers within this distribution. GDM demonstrates its consistent effectiveness against all 3×3 triggers. In the DHBE framework, owing to the adversarial optimization, the trigger generator $\mathcal{G}_p$ only needs to recover the currently most sensitive trigger to the student model $\mathcal{S}$. As $\mathcal{S}$ and $\mathcal{G}_p$ are updated adversarially and simultaneously, any trigger that can be generated by $\mathcal{G}_p$ will be mitigated. In Fig. 6, we perform DHBE and GDM [35] on a large number of 3×3 square triggers, and show that the adversarial backdoor regularization has comparable performance with GDM, even it does not model the trigger distribution.

5.2.3 Evaluations on More Datasets. To show that the proposed framework could be easily extended to datasets with larger size and more categories, we run the DHBE on three other datasets using the same hyperparameters and learning rates : CIFAR100, Mini-Imagenet, and VGGFace2, and list the results in Table 4. Specifically, for VGGFace2, we randomly choose 100 identities and train a backdoored classification model on them. We also list the results of finetuning and NAD with learning rate of 0.001 and $N_{clean} = 2000$ in the table. The results show that our method still outperforms the blind erasing methods by a large margin.

Table 4: Results of DHBE and other methods on other datasets against Badnets attack with different triggers. Numbers are displayed as percentages. "†" represents the attacked class t is provided for the method to perform backdoor erase on.

Datasets	Trigger Size	Backdoored N_{poison} = 300		Finetuning N_{clean} = 2000		NAD [25] N_{clean} = 2000		NC† [45] N_{clean} = 1000		GDM† [35] N_{clean} = 1000		DHBE No data required	
		ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
CIFAR100	3 × 3	77.30	99.99	68.55	86.51	70.01	83.47	74.63	0.70	76.00	**0.12**	**76.06**	0.66
	5 × 5	77.29	99.83	68.49	91.48	69.54	81.39	75.39	34.70	**76.11**	**0.04**	75.17	2.58
Mini-Imagenet	3 × 3	68.61	98.56	58.36	11.91	60.13	42.26	66.55	0.08	**68.05**	**0.06**	66.61	1.18
	5 × 5	68.51	99.15	56.37	18.10	59.09	45.70	65.00	2.00	**68.31**	**0.10**	64.29	2.60
VGGFace2	3 × 3	96.85	99.97	91.49	4.35	91.91	3.35	95.91	0.01	**96.43**	**0.00**	96.32	6.49
	5 × 5	96.75	99.97	92.19	14.74	92.39	5.03	95.31	97.05	**96.06**	**0.00**	95.84	0.01

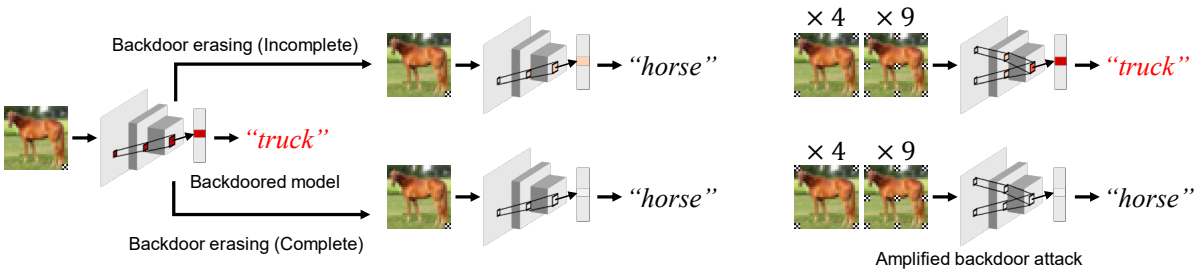


Figure 5: Illustration of how amplified backdoor attacks could still backdoor the models produced by backdoor erasing methods where injected backdoors are incompletely erased.

Table 5: Comparison results of DHBE to targeted erasing methods on CIFAR10 against patch triggers. Numbers are displayed as percentages. "†" represents the attacked class t is provided for the method to perform backdoor erase on.

Att. Meth.	Tri.	Backdoored t="truck"		NC† [45] $N_{cl.}$ = 1000		GDM† [35] $N_{cl.}$ = 1000		DHBE No data req.	
		ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
B. [13]	2 × 2	94.75	100.00	93.27	6.96	93.98	0.98	**94.05**	**0.02**
	3 × 3	95.02	100.00	93.47	17.83	94.23	**0.06**	**94.46**	0.12
	5 × 5	95.11	100.00	93.94	0.66	94.18	0.11	**94.24**	**0.00**
C. [42]	2 × 2	94.85	46.32	93.32	21.13	**94.44**	0.48	94.19	**0.14**
	3 × 3	95.04	93.45	93.35	54.45	**94.52**	**0.37**	94.11	1.17
	5 × 5	95.25	100.00	93.40	73.79	**94.58**	**0.13**	94.45	4.48
T. [31]	2 × 2	94.56	56.20	93.02	1.44	93.73	**0.14**	**93.96**	4.39
	3 × 3	95.00	89.91	92.91	0.58	93.93	**0.08**	**94.40**	2.88
	5 × 5	94.93	99.91	94.20	0.14	94.09	**0.11**	**94.56**	1.13
Mean		94.95	87.31	93.43	19.66	94.19	**0.27**	**94.27**	1.59

5.3 Incomplete Backdoor Erasing is Vulnerable under Amplified Backdoor Attack

Existing backdoor erasing methods default that the defense is sufficient if the attack success rate is suppressed below a given threshold, such as 10% [25]. However, in this work, we demonstrate that a simple attack strategy could easily trigger the cleaned model and obtain a high attack success rate, as shown in Fig. 5. The attack strategy is apparent and straight-forward: If the injected triggers are not wiped out completely, the cleaned models still have obvious responses to the triggers, then the attacker can inject the same trigger multiple times into the input to amplify the activation of backdoors. The activations of multiple triggers are accumulated in the model, finally, the model's prediction can still be altered by the attacker with a high attack success rate. Thus, it can be concluded that incomplete backdoor erasing is actually useless in real-world applications since it cannot defend against the intentional manipulation of unknown adversaries.

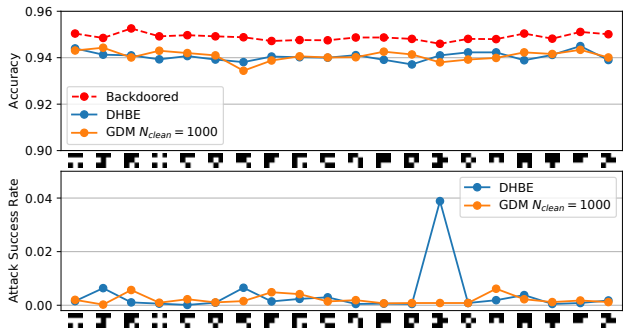


Figure 6: Comparison experiments of ABR and GDM on various triggers. ASRs of backdoored models are all close to 100% and omitted in the bottom figure.

Here, we conduct experiments where test inputs are injected with triggers 4 or 9 times, and the attack samples are shown in Fig. 5. To amplify the activations of backdoors in the cleaned model, We first inject triggers into four corners of the test inputs (×4), then we further inject more triggers into the center and four edges of the test inputs (×9). The ASR results are shown in Table 6. By injecting four triggers into the four corners of the test inputs, the attack remains highly stealthy. However, the cleaned models produced by Finetuning, Finepruning, and NAD show a high attack success rate in most cases, even though the model's ASR on a single trigger is suppressed below 10%. When more triggers are injected, those cleaned models could be attacked with nearly 100% ASR in many cases. Thus, we conclude that those unlearning methods are unsafe if the injected backdoors are not completely cleaned.

Table 6: Comparison results of DHBE to other backdoor erasing methods against amplified backdoor attacks. Numbers are displayed as percentages.

Trigger Size	Finetuning			Finepruning [28]			NAD [25]			NC [45]			GDM [35]			DHBE		
	x1	x4	x9	x1	x4	x9	x1	x4	x9	x1	x4	x9	x1	x4	x9	x1	x4	x9
2×2	2.64	50.68	93.08	12.92	92.88	99.62	0.38	7.30	12.92	6.96	44.04	54.85	0.98	9.15	16.86	0.02	0.51	1.94
3×3	2.97	37.98	66.32	6.74	63.47	98.34	7.08	84.34	100.00	17.83	52.95	83.43	0.06	0.18	1.89	0.12	0.22	1.00
5×5	1.10	4.75	22.11	69.07	100.00	100.00	3.26	28.70	73.81	1.20	21.33	78.08	0.18	0.63	16.15	0.00	0.09	0.66
Mean ASR	2.24	31.14	60.50	29.58	85.45	99.32	3.57	40.11	62.24	8.66	39.44	72.12	0.41	3.32	11.63	**0.05**	**0.28**	**1.20**

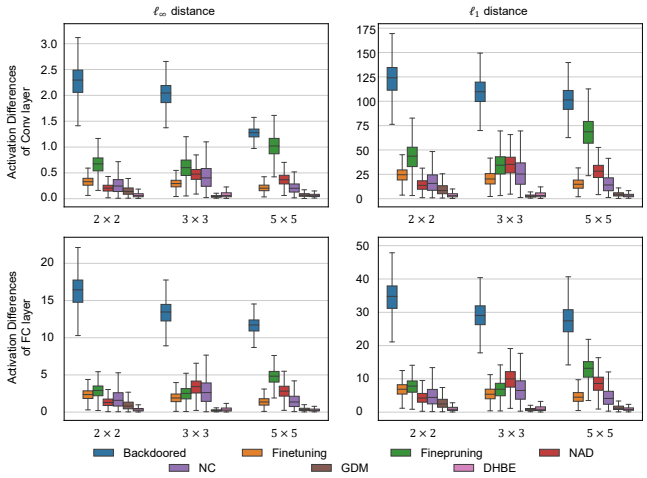


Figure 7: Activation differences between clean inputs and trigger-pasted inputs on different models. The activation differences of the last *conv* and fc layers are shown above and below, respectively. The left figures show ℓ_∞ distances and the right figures show ℓ_1 distances. The models produced by DHBE always has the minimal response to the triggers.

Since the attacker still can activate the backdoors with a high ASR, their backdoor erasing attempt is useless. In contrast, the DHBE framework mitigates the response of triggers in the resulted model completely using the proposed adversarial backdoor regularization, and demonstrates its robustness against amplified backdoor attacks.

To quantitatively measure the effectiveness of mitigating the model's backdoor reactions, we draw samples from the clean test dataset, and plot the distribution of activation differences between clean inputs and corresponding trigger-pasted inputs. The results are shown by boxplot in Fig. 7. The activation differences of the last *conv* layer and the final fc output layer are included, and differences are calculated by two measures : ℓ_1 distance and ℓ_∞ distance. As shown in Fig. 7, the model produced by DHBE demonstrates far less response to the triggers than other methods even when measured at an inner layer.

5.4 Ablation Studies

In this subsection, we show that the effectiveness of the proposed DHBE is insensitive to a wide range of choices of hyperparameters, and DHBE is able to deal with backdoor attacks with different size of triggers using a same set of hyperparameters. These ablation studies suggest that our backdoor erasing framework is robust enough and can be deployed in real-world applications with little trouble.

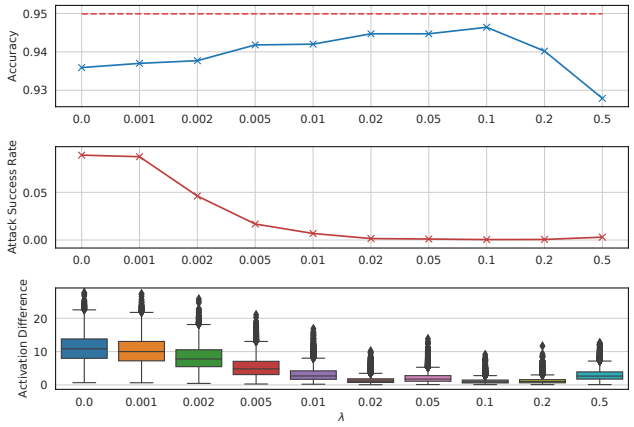


Figure 8: Ablation studies on ABR (controlled by λ) of DHBE framework (against Badnets attack with 3×3 square trigger). We show the changes of accuracy, attack success rate, and activation difference (Outputs of fc layer, ℓ_1 distance) w.r.t. different λ. Red dash line in top figure represents the accuracy of orignal backdoored model.

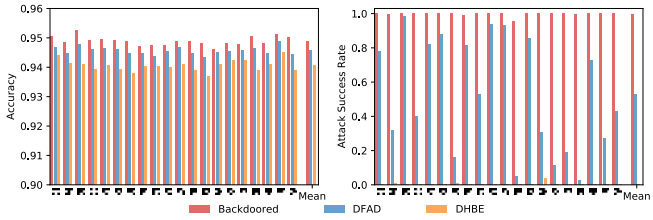


Figure 9: Comparison experiments of DHBE and pure adversarial distillation on different backdoor triggers.

5.4.1 Effectiveness of Adversarial Backdoor Regularization. First, we investigate the effectiveness of proposed adversarial backdoor regularization, whose magnitude is controlled by a single hyperparameter λ. Here, we perform experiments with different values of λ, and show the accuracy, attack success rate, and activation difference (ℓ_1 distance of the outputs of fc layer) in Fig. 8. As the value of λ increases, the ASR is decreased as well as the response to the trigger, until it becomes barely noticeable. It can be also found that this regularization term effectively suppresses the backdoor reactions and maintains the accuracy of the model across a wide range of λ (from 0.01 to 0.2).

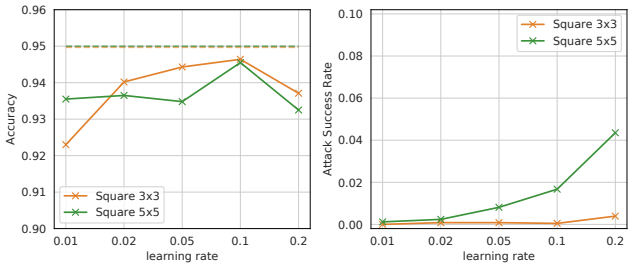


Figure 10: Ablation studies on the learning rate of DHBE framework. Dash lines in the left figure represent the accuracy of original backdoored model. ASRs of backdoored models are all close to 100% and omitted in the right figure.

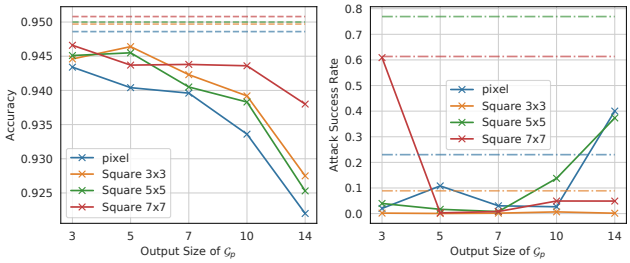


Figure 11: Ablation study on the output size of trigger generator $\mathcal{G}_p$ in DHBE framework against different size of triggers. Dash lines in the left figure represent the accuracy of original backdoored model, and dash lines in right figure represent ASRs of DHBE ($\lambda = 0$) (ASRs of backdoored models are all close to 100% and omitted in the right figure).

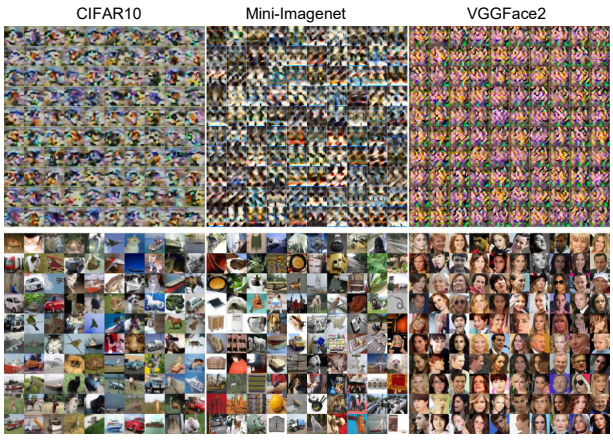


Figure 12: The first row are samples generated by $\mathcal{G}$. The images in the second row are sampled from original training data.

We note that the DHBE framework mitigates the ASR from 100% to 8.9% when $\lambda = 0.0$. This result does not weaken the actual effectiveness of ABR because 1) DHBE with $\lambda = 0.0$ does not equal pure adversarial distillation. Even if λ is set to 0.0 in DHBE, the generated trigger-pasted inputs are fed into the student model in the training mode, which influences its batchnorm layers' statistics and resulted in suppressed response to the triggers. 2) We also evaluate pure adversarial distillation with a large amount of triggers, and show the results in Fig. 9. It can be seen that different triggers have stochastic transferability. The evaluated triggers have up to 98% ASR and 50% average ASR after using pure adversarial distillation, but have nearly 0% ASR after using the proposed DHBE method.

5.4.2 Robustness w.r.t. Different Learning Rates. As shown in previous experiments in Fig. 3 and Fig. 4, both blind and targeted erasing methods are sensitive to the learning rate used in finetuning. In contrast, DHBE does not need to find a proper learning rate for each backdoor erasing attempt, thanks to its adversarial design. The results are shown in Fig. 10 where different learning rates from a wide range (from 0.01 to 0.2) are employed. The learning rates of the sample generator and trigger generator are synchronously adjusted for stabilized adversarial training. As shown in Fig. 10, the results of DHBE are consistent across a wide range of learning rates.

5.4.3 Trigger size v.s. Trigger Generator Size. Although various methods have been proposed to diagnose whether a model is backdoored or not, it is still difficult for defenders to determine the size, shape, and texture of the actual trigger. In Fig. 11, we show that the proposed DHBE framework still demonstrates effectiveness when the actual trigger size and the generation size of triggers generated by $\mathcal{G}_p$ do not well matched. Specifically, we train backdoored models implanted with different sizes of trigger, and employ DHBE with different output sizes of $\mathcal{G}_p$ to distill them. The results in Fig. 11 show that the DHBE with trigger generator size 7×7 is able to erase all those triggers with only about 1% accuracy degradation.

5.4.4 Compare to Model Inverting Techniques. A competitor for data-free adversarial distillation is distillation with inverted training dataset [53, 54]. However, it will cause privacy concerns in real-world applications, as the information of the original training data is inverted and visualized. We show that the generated samples in DHBE do not exhibit features of the original training dataset in Fig. 12. This enables the DHBE to be deployed in privacy-concerning scenarios involving federated learning and so on.

Besides, the dataset inverting techniques cannot be directly combined with blind or targeted erasing methods for data-free backdoor erasing. Our evaluations on distillation using inverted dataset [53, 54] show similar results with pure adversarial distillation [11] whose results are shown in Fig. 9, where backdoors could be distilled and transferred into student models with a high probability.

6 CONCLUSIONS

In this work, we propose a novel data-free holistic backdoor erasing framework (DHBE) to wipe out hidden triggers in attacked DNNs without access to clean data using the proposed restricted adversarial distillation paradigm. We empirically demonstrate that the proposed framework achieves superior performance in comparison to other backdoor erasing methods, even these methods are assisted by a number of clean data. We also quantify the models' internal response to show that the proposed method most effectively suppresses the model's backdoor reactions. On top of that, we explore the robustness of the proposed method, and show that it provides unprecedented stability in the tuning of hyperparameters, due to

its adversarial design. Overall, the proposed framework provides a practical, flexible, and effective solution for eliminating patch backdoors yet. We hope our work could inspire more comprehensive backdoor erasing efforts against evolving backdoor attacks.

ACKNOWLEDGMENTS

This research work is funded by the National Nature Science Foundation of China under Grant 61971283, 62202303, and Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein Generative Adversarial Networks. In *ICML*. 214–223.
[2] Eugene Bagdasaryan and Vitaly Shmatikov. 2020. Blind Backdoors in Deep Learning Models. *arXiv preprint arXiv:2005.03823* (2020).
[3] Mauro Barni, Kassem Kallas, and Benedetta Tondi. 2019. A New Backdoor Attack in CNNs by Training Set Corruption without Label Poisoning. In *ICIP*. 101–105.
[4] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. 2018. Vggface2: A Dataset for Recognising Faces across Pose and Age. In *FG*.
[5] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2019. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. In *AAAI*.
[6] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2019. DeepInspect: A Black-box Trojan Detection and Mitigation Framework for Deep Neural Networks.. In *IJCAI*.
[7] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. 2019. Data-free Learning of Student Networks. In *ICCV*. 3514–3522.
[8] Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. 2019. A Baseline for Few-Shot Image Classification. In *ICLR*.
[9] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. 2020. Februus: Input Purification Defense against Trojan Attacks on Deep Neural Network Systems. In *ACSAC*. 897–912.
[10] Gongfan Fang, Kanya Mo, Xinchao Wang, Jie Song, Shitao Bei, Haofei Zhang, and Mingli Song. 2022. Up to 100x Faster Data-Free Knowledge Distillation. In *AAAI*. 6597–6604.
[11] Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song. 2019. Data-free Adversarial Distillation. *arXiv preprint arXiv:1912.11006* (2019).
[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NeurIPS*.
[13] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. *arXiv preprint arXiv:1708.06733* (2017).
[14] Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. 2019. Tabor: A Highly Accurate Approach to Inspecting and Restoring Trojan Backdoors in Ai Systems. *arXiv preprint arXiv:1908.01763* (2019).
[15] Jonathan Hayase, Weihao Kong, Raghav Somani, and Sewoong Oh. 2021. SPECTRE: Defending Against Backdoor Attacks using Robust Statistics. In *ICML*.
[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.
[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531* (2015).
[18] Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. 2022. Badencoder: Backdoor Attacks to Pre-trained Encoders in Self-supervised Learning. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2043–2059.
[19] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and K Gustavo. 2019. Generalized Sliced Wasserstein Distances. In *NeurIPS*.
[20] Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images. *Master's thesis, University of Toronto* (2009).
[21] Gaolei Li, Kaoru Ota, Mianxiong Dong, Jun Wu, and Jianhua Li. 2020. DeSVig: Decentralized Swift Vigilance Against Adversarial Attacks in Industrial Artificial Intelligence Systems. *IEEE TII* 16, 5 (2020), 3267–3277.
[22] Shaofeng Li, Minhui Xue, Benjamin Zhao, Haojin Zhu, and Xinpeng Zhang. 2020. Invisible Backdoor Attacks on Deep Neural Networks via Steganography and Regularization. *IEEE TDSC* (2020).
[23] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. 2021. Invisible Backdoor Attack with Sample-specific Triggers. In *ICCV*. 16463–16472.
[24] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Anti-backdoor Learning: Training Clean Models on Poisoned Data. In *NeurIPS*.
[25] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks. In *ICLR*.
[26] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2020. Backdoor Learning: A Survey. *arXiv preprint arXiv:2007.08745* (2020).
[27] Junyu Lin, Lei Xu, Yingqi Liu, and Xiangyu Zhang. 2020. Composite Backdoor Attack for Deep Neural Network by Mixing Existing Benign Features. In *CCS*. 113–131.
[28] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-pruning: Defending against Backdooring Attacks on Deep Neural Networks. In *RAID*.
[29] Xuankai Liu, Fengting Li, Bihan Wen, and Qi Li. 2021. Removing Backdoor-based Watermarks in Neural Networks with Limited Data. In *ICPR*.
[30] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. Abs: Scanning Neural Networks for Back-doors by Artificial Brain Stimulation. In *CCS*. 1265–1282.
[31] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In *NDSS*.
[32] Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. 2020. Reflection Backdoor: A Natural Backdoor Attack on Deep Neural Networks. In *ECCV*.
[33] Yuang Liu, Wei Zhang, Jun Wang, and Jianyong Wang. 2021. Data-free Knowledge Transfer: A Survey. *arXiv preprint arXiv:2112.15278* (2021).
[34] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *NIPS-W*.
[35] Ximing Qiao, Yukun Yang, and Hai Li. 2019. Defending Neural Backdoors via Generative Distribution Modeling. In *NeurIPS*.
[36] Ge Ren, Jun Wu, Gaolei Li, Shenghong Li, and Mohsen Guizani. 2022. Protecting Intellectual Property with Reliable Availability of Learning Models in AI-based Cybersecurity Services. *IEEE TDSC* (2022), 1–18.
[37] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. 2020. Hidden Trigger Backdoor Attacks. In *AAAI*.
[38] Yuan Tian, Zhaohui Che, Wenbo Bao, Guangtao Zhai, and Zhiyong Gao. 2020. Self-supervised motion representation via scattering local motion cues. In *European Conference on Computer Vision*. Springer, 71–89.
[39] Yuan Tian, Guo Lu, Xiongkuo Min, Zhaohui Che, Guangtao Zhai, Guodong Guo, and Zhiyong Gao. 2021. Self-conditioned probabilistic learning of video rescaling. In *CVPR*. 4490–4499.
[40] Yuan Tian, Yichao Yan, Guangtao Zhai, Guodong Guo, and Zhiyong Gao. 2022. Ean: event adaptive network for enhanced action recognition. *IJCV* 130, 10 (2022), 2453–2471.
[41] Brandon Tran, Jerry Li, and Aleksander Madry. 2018. Spectral Signatures in Backdoor Attacks. In *NeurIPS*.
[42] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2019. Label-Consistent Backdoor Attacks. *arXiv preprint arXiv:1912.02771* (2019).
[43] Miguel Villarreal-Vasquez and Bharat Bhargava. 2020. Confoc: Content-focus Protection against Trojan Attacks on Neural Networks. *arXiv preprint arXiv:2007.00711* (2020).
[44] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In *NeurIPS*.
[45] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *SP*.
[46] Shuo Wang, Surya Nepal, Carsten Rudolph, Marthie Grobler, Shangyu Chen, and Tianle Chen. 2020. Backdoor Attacks against Transfer Learning with Pre-trained Deep Learning Models. *IEEE TSC* (2020).
[47] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. 2018. Zero-shot Recognition via Semantic Embeddings and Knowledge Graphs. In *CVPR*. 6857–6866.
[48] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2019. DBA: Distributed Backdoor Attacks against Federated Learning. In *ICLR*.
[49] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A Gunter, and Bo Li. 2021. Detecting AI Trojans using Meta Neural Analysis. In *SP*. 103–120.
[50] Zhicong Yan, Gaolei Li, Yuan TIan, Jun Wu, Shenghong Li, Mingzhe Chen, and H Vincent Poor. 2021. Dehib: Deep Hidden Backdoor Attack on Semi-supervised Learning via Adversarial Perturbation. In *AAAI*, Vol. 35. 10585–10593.
[51] Zhicong Yan, Jun Wu, Gaolei Li, Shenghong Li, and Mohsen Guizani. 2021. Deep Neural Backdoor in Semi-Supervised Learning: Threats and Countermeasures. *IEEE TIFS* 16 (2021), 4827–4842.
[52] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. 2019. Latent Backdoor Attacks on Deep Neural Networks. In *CCS*. 2041–2055.
[53] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. 2021. See Through Gradients: Image Batch Recovery via Gradinversion. In *CVPR*. 16337–16346.
[54] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. 2020. Dreaming to Distill: Data-free Knowledge Transfer via Deepinversion. In *CVPR*. 8715–8724.
[55] Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. 2019. Bridging Mode Connectivity in Loss Landscapes and Adversarial Robustness. In *ICLR*.

APPENDIX

A MORE ATTACK IMPLEMENTATION DETAILS

Detailed implementation of backdoor attacks are summarized as follows:

(a) Badnets: We perform Badnets attack by poisoning the training dataset, where we randomly select N_{poison} training images from the whole dataset (except images from the target class t), injecting triggers into them and flipping their labels to the target class t. Then the victim model is trained on the poisoned dataset using an SGD optimizer with initial learning rate of 0.1, momentum of 0.9, and weight decay of 5e-4. $N_{poison} = 300$ is enough to achieve a high attack success rate and we use this setting for all Badnets experiments. We finetune the model for 200 epochs with learning rate decay of 0.1 at epoch 100 and epoch 150. The training data augmentations are limited to Random Crop and Random Horizontal Flip only. In the experiments section, we always take the last class in the dataset as the target class (i.e. "truck" in CIFAR10 and "tractor" in CIFAR100). The attack and defense results of other classes is listed in Table 7.

(b) Clean-Label: Clean-label attack poisons the dataset without flipping the training labels, but it corrupts the salient object feature in the poisoned images and forces the model's decisions to be more dependent on the trigger pattern [42]. In Clean-label attacks, we randomly choose 300 training samples from the target category, use adversarial perturbations to corrupt their salient object feature, and stamp square triggers to them at the right-bottom location. Then we perform the normal learning procedure with the same training settings as the Badnets attack.

(c) Trojaning: In this work, we first randomly select 10 neurons of the *AveragePooling* layer in Resnet-18 model (shown in Table 9), reverse engineers a trigger pattern that can achieve maximum responses of selected neurons, then finetunes only the subsequent *FC* layer to strength the connection between the selected neurons and the target class score using a poisoned dataset with $N_{poison} = 300$ by inserting the reverse engineered trigger into them.

B MODEL ARCHITECTURES AND HYPERPARAMETERS

B.1 Architecture of Backdoored Teachers and Students

We list the employed architecture of Resnet-18 in Table 9, and use it as a unified model architecture for fair comparisons of various backdoor erasing methods. During the training or finetuning process of the Rsenet-18, we perform a normalization operation after randomized preprocessing on each sample according to the statistics of the employed dataset, so the model is fed with data that has zero means and unit variances.

B.2 Architecture of Sample Generators and Trigger Generators

We also list the architectures of employed sample generators $\mathcal{G}$ and trigger generators $\mathcal{G}_p$ that produce different size of outputs in Table 10, 11, 12 and 13. Specifically, the sample generators are ended with batch normalization layers, so that the produced samples should have zero means and unit variances, and can be directly fed into the teacher network and student network to perform knowledge distillation. The trigger generators are ended with $Tanh$ layers, so that their outputs are clamped within $(-1, 1)$. The slope of $LeakyReLU$ is set to 0.2 and all generators contains BN layers for stabilized training.

B.3 Mixing Generated Triggers to Fake Samples

In every step where a produced trigger δ is injected to a produced fake sample x, we first denormalize the fake sample, then add the produced trigger δ to the denormalized sample, and finally perform the normalization process.

Table 7: DHBE results against Badnets attack on other classes of CIFAR10 (3×3 square trigger is used). Numbers are displayed as percentages.

Target Class	Backdoored		DHBE	
	ACC	ASR	ACC	ASR
airplane	95.10	100.00	94.55	2.74
automobile	95.09	99.98	94.46	0.30
bird	94.83	100.00	94.16	0.15
cat	95.14	100.00	94.54	1.22
deer	94.74	100.00	93.80	0.14
dog	94.87	100.00	94.18	0.18
frog	94.69	100.00	94.59	0.66
horse	95.18	100.00	94.48	0.01
ship	94.70	100.00	94.21	0.08

Table 8: DHBE results against Badnets attack on other network architectures (3×3 square trigger is used). Numbers are displayed as percentages.

Target Class	Backdoored		DHBE	
	ACC	ASR	ACC	ASR
Resnet34	95.53	100.00	94.50	0.26
WRN40-1	92.99	99.99	92.50	0.45
WRN40-2	94.94	99.96	94.44	1.42

C DETAILED CONFIGURATION OF BLIND METHODS

The detailed configuration and training process of blind methods is summarized as follows:

(a) Finetuning, Finepruning [28], and Neural Attention Distillation (NAD) [25]. For these methods, we adapt the same optimizer and learning rate settings: An SGD optimizer with initial learning rate of 0.01, momentum of 0.9 is employed. the backdoored model is fully finetuned by 20 epochs and 2000 samples are loaded for each epoch with batch size of 64. When the number of input training samples is less than 2000, we duplicate them until there are 2000 samples. The learning rate is decayed by 0.1 at epoch 10 and epoch 15. For NAD, the attention transfer loss implemented

Table 9: Resnet-18 architectures used in this work.

Resnet-18	Output Size
Input	$32 \times 32 \times 3$ / $64 \times 64 \times 3$
3×3 64 *S1 Conv* / 5×5 64 *S2 Conv*	$32 \times 32 \times 64$
BN, *ReLU*	$32 \times 32 \times 64$
$\begin{bmatrix} 3 \times 3 \text{ 64 } Conv, BN, ReLU \\ 3 \times 3 \text{ 64 } Conv, BN, ReLU \end{bmatrix} \times 2$	$32 \times 32 \times 64$
$\begin{bmatrix} 3 \times 3 \text{ 128 } Conv, BN, ReLU \\ 3 \times 3 \text{ 128 } Conv, BN, ReLU \end{bmatrix} \times 2$	$16 \times 16 \times 128$
$\begin{bmatrix} 3 \times 3 \text{ 256 } Conv, BN, ReLU \\ 3 \times 3 \text{ 256 } Conv, BN, ReLU \end{bmatrix} \times 2$	$8 \times 8 \times 256$
$\begin{bmatrix} 3 \times 3 \text{ 512 } Conv, BN, ReLU \\ 3 \times 3 \text{ 512 } Conv, BN, ReLU \end{bmatrix} \times 2$	$4 \times 4 \times 512$
Avg Pool	512
FC	Num. of Classes

Table 10: Architectures of trigger generators $\mathcal{G}_p$ of output size 3×3, 5×5 and 7×7.

Trigger Generator $\mathcal{G}_p$	Output Size		
Input Dim. of z	256		
FC, *Reshape*, *BN*	$3 \times 3 \times 64$	$5 \times 5 \times 64$	$7 \times 7 \times 64$
3×3 3 *Conv*, *BN*, *Tanh*	$3 \times 3 \times 3$	$5 \times 5 \times 3$	$7 \times 7 \times 3$

Table 11: Architectures of trigger generators $\mathcal{G}_p$ of output size 10×10 and 14×14.

Trigger Generator $\mathcal{G}_p$	Output Size	
Input Dim. of z	256	
FC, *Reshape*, *BN*	$5 \times 5 \times 128$	$7 \times 7 \times 128$
Upsample $2\times$	$10 \times 10 \times 128$	$14 \times 14 \times 128$
3×3 64 *Conv* *BN*, *LeakyReLU* 0.2	$10 \times 10 \times 64$	$14 \times 14 \times 64$
3×3 3 *Conv*, *BN*, *Tanh*	$10 \times 10 \times 3$	$14 \times 14 \times 3$

in open-source code [1] is used. Between the teacher and student models, we add four attention transfer losses between the outputs of their four residual blocks, and each loss is multiplied by $beta = 5000$ as suggested in [25] before finally adding up with the cross-entropy loss and performing back-propagation optimization.

(b) Model Connectivity Repair (MCR) [55]. For MCR, we imitate the open-source code [2] and create a Curve net version of employed Resnet-18 (Table 9), where the *conv*, *fc*, *BN* layers are modified. We set the endpoint of Resnet-18 curve net ($t = 0.0$ and $t = 1.0$) with the same backdoorded Resnet-18 model, and train the connection path for 200 epochs with SGD optimizer ($lr = 0.01$, $momentum = 0.9$, $lrdecay = 0.1$ at epoch 100 and epoch 150). As suggested in [55], when evaluating the curve model at its middle point, we first let the model go through the whole test dataset in training mode to correct its running statistics of batch normalization layers, then we fix the batch normalization layers and report its ACC and ASR.

[1] https://github.com/bboylyg/NAD
[2] https://github.com/IBM/model-sanitization

Table 12: Architectures of trigger generators $\mathcal{G}_p$ of output size 32×32.

Trigger Generator $\mathcal{G}_p$	Output Size
Input Dim. of z	256
FC, *Reshape*, *BN*	$8 \times 8 \times 128$
Upsample $2\times$	$16 \times 16 \times 128$
3×3 128 *Conv* *BN*, *LeakyReLU* 0.2	$16 \times 16 \times 128$
Upsample $2\times$	$32 \times 32 \times 128$
3×3 64 *Conv* *BN*, *LeakyReLU* 0.2	$32 \times 32 \times 64$
3×3 3 *Conv*, *BN*, *Tanh*	$32 \times 32 \times 3$

Table 13: Architectures of sample generators $\mathcal{G}$ of different output size.

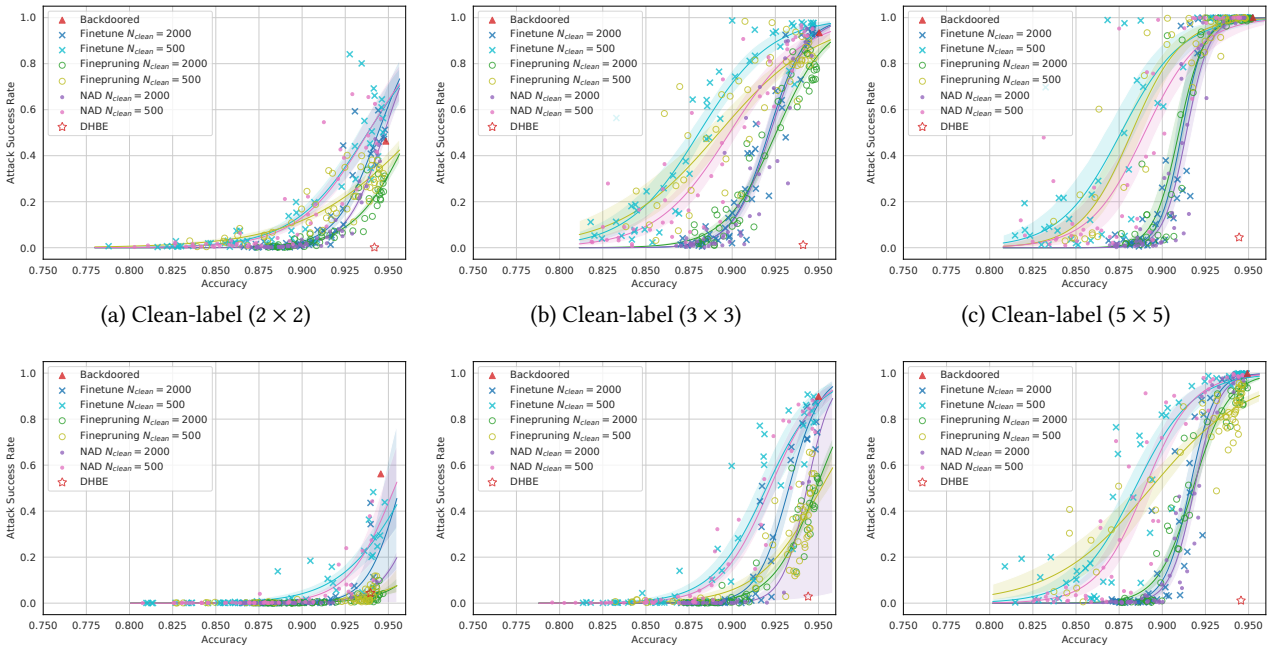
Sample Generator $\mathcal{G}$	Output Size	
Input Dim. of z	256	
FC, *Reshape*, *BN*	$8 \times 8 \times 128$	
Upsample $2\times$	$16 \times 16 \times 128$	
3×3 128 *Conv* *BN*, *LeakyReLU* 0.2	$16 \times 16 \times 128$	
Upsample $2\times$	$32 \times 32 \times 128$	
3×3 128 *Conv* *BN*, *LeakyReLU* 0.2	-	$32 \times 32 \times 128$
Upsample $2\times$	-	$64 \times 64 \times 128$
3×3 64 *Conv* *BN*, *LeakyReLU* 0.2	$32 \times 32 \times 64$	$64 \times 64 \times 64$
3×3 3 *Conv*, *Sigmoid*, *BN*	$32 \times 32 \times 3$	$64 \times 64 \times 3$

D DETAILED CONFIGURATION OF TARGETED METHODS

The detailed configuration and training process of targeted methods is summarized as follows:

(a) Neural Cleanse [45]. Using the open-source code [3], we generate a reverse-engineered trigger pattern Δ and trigger mask m for the target class t. Then, 10% of 2000 clean training samples (samples are duplicated if their number is less than 2000) are randomly selected and injected with Δ. Finally, the backdoored model is finetuned using the same set of hyperparameters as the finetuning process in blind methods, except the initial learning rate is 0.001.

[3] https://github.com/Abhishikta-codes/neural_cleanse



(a) Clean-label (2×2) (b) Clean-label (3×3) (c) Clean-label (5×5)

(d) Trojaning (2×2) (e) Trojaning (3×3) (f) Trojaning (5×5)

Figure 13: Trade-off curves between accuracy and attack success rate of different blind erasing methods against different triggers. The curves are drawn by adjusting the learning rate of these methods (except DHBE) from 0.002 to 0.02.

(2) Generative Distribution Modeling (GDM) [35]. Using the open-source code [4], we train a generative model for the trigger distribution, using hyperparameters ($\alpha = 0.1$, $\beta = 0.9$). Then we use the same finetuning process and the same learning rate schedule as Neural Cleanse to finetune backdoored models. For each training sample in each iteration, it is stamped by a randomly sampled trigger with probability 0.1.

[4] https://github.com/superrrpotato/Defending%2DNeural%2DBackdoors%2Dvia%2DGenerative%2DDistribution%2DModeling