MM-BD: Post-Training Detection of Backdoor Attacks with Arbitrary Backdoor Pattern Types Using a Maximum Margin Statistic

Hang Wang*, Zhen Xiang*, David J. Miller George Kesidis
Anomalee Inc., State College, PA, USA Pennsylvania State University, University Park, PA, USA

2024 IEEE Symposium on Security and Privacy (SP) | 979-8-3503-3130-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/SP54263.2024.00015

***Abstract*—Backdoor attacks are an important type of adversarial threat against deep neural network classifiers, wherein test samples from one or more source classes will be (mis)classified to the attacker's target class when a backdoor pattern is embedded. In this paper, we focus on the *post-training* backdoor defense scenario commonly considered in the literature, where the defender aims to detect whether a trained classifier was backdoor-attacked without any access to the training set. Many post-training detectors are designed to detect attacks that use either one or a few specific backdoor embedding functions (e.g., patch-replacement or additive attacks). These detectors may fail when the backdoor embedding function used by the attacker (unknown to the defender) is different from the backdoor embedding function assumed by the defender. In contrast, we propose a post-training defense that detects backdoor attacks with *arbitrary* types of backdoor embeddings, without making any assumptions about the backdoor embedding type. Our detector leverages the influence of the backdoor attack, *independent of the backdoor embedding mechanism*, on the landscape of the classifier's outputs prior to the softmax layer. For each class, a *maximum margin* statistic is estimated. Detection inference is then performed by applying an *unsupervised* anomaly detector to these statistics. Thus, our detector does not need any legitimate clean samples, and can efficiently detect backdoor attacks with *arbitrary* numbers of source classes. These advantages over several state-of-the-art methods are demonstrated on four datasets, for three different types of backdoor patterns, and for a variety of attack configurations. Finally, we propose a novel, general approach for backdoor mitigation once a detection is made. The mitigation approach was the runner-up at the first IEEE Trojan Removal Competition. The code is online available.**

***Index Terms*—Backdoor attack; Trojan; backdoor defense**

1. Introduction

Although deep neural networks (DNNs) are successful in many research areas, they are vulnerable to attacks [1]. A backdoor attack or Trojan is an important type of attack under which a DNN classifier will predict to the attacker's target class when a test sample from one or more source classes is embedded with the attacker's backdoor pattern [2]–[4]. A backdoor attack is typically launched by poisoning the classifier's training set with samples originally from the source classes, embedded with the same backdoor pattern that will be used during inference, and labeled to the target class [5]. Since successful backdoor attacks do not degrade the classifier's accuracy on clean test samples, they cannot be easily detected, e.g., using validation set accuracy [6].

Defenses against backdoor attacks are sometimes deployed during the classifier's training stage [7]–[14], but this is also often infeasible (e.g., considering proprietary and legacy systems). Here, we consider a practical *post-training* scenario, where the defender is, e.g., a downstream user of a downloaded pre-trained classifier who aims to detect whether the classifier has been attacked (e.g. Model Zoo that provides pre-trained models [15].), without access to the classifier's training set. The defender also has no access to any unattacked classifiers for reference (e.g. for setting a detection threshold).

Many post-training defenses assume the defender independently possesses a small set of clean, legitimate samples from every class. These samples may be used: i) to reverse-engineer putative backdoor patterns, which are the basis for anomaly detection [16]–[22]; or ii) to train shadow neural networks with and without (known) backdoor attacks – based upon which a binary "meta-classifier" is trained to predict whether the classifier under inspection was backdoor-attacked [23]–[25]. However, these methods assume the mechanism used by the attacker for embedding a backdoor pattern is known. For reverse-engineering-based defenses, the embedding function of the backdoor pattern is explicitly involved in the reverse-engineering problem, e.g., [16]; thus, these methods may not be able to effectively detect backdoor attacks with pattern types different from that assumed for the reverse-engineering. For the meta-classification approach, a pool of backdoor patterns is assumed for training shadow neural networks with backdoor attacks [24]; these methods may not generalize well to backdoor patterns not seen in the training pool.

Here, we propose a **M**aximum-**M**argin-based **B**ackdoor **D**etection method (MM-BD) which does not make any assumptions about the backdoor pattern type used by the attacker. Our detection method is based on a novel approach to capture the atypicality of the landscape of the classifier's outputs (before the softmax) induced by backdoor attacks. In particular, we propose a novel *maximum margin* (MM) detection statistic. This statistic is obtained for each class by solving a margin maximization problem starting from multiple random input patterns, with the solution with largest margin then chosen. We will show that use of these

*. Equal contribution

DOI 10.1109/SP54263.2024.00015

statistics is discriminative between the backdoor target class and non-backdoor classes, irrespective of the type of backdoor pattern/embedding function used. *No clean samples* are required for MM-based detection. A fully *unsupervised* anomaly detector is applied to the maximum margin statistics to perform the detection inference. Notably, the design of our method allows it to detect backdoor attacks with an *arbitrary* number of source classes, and also with better *computational efficiency* than many existing methods.

We also propose a **M**aximum-**M**argin-based **B**ackdoor **M**itigation method (MM-BM) that does require use of a few clean samples. This method suppresses the maximum possible neuron activations using a set of optimized upper bounds (one per neuron), without reducing the classifier's accuracy on clean samples. Unlike existing methods, we do not modify the DNN architecture or any trained parameters.

Our contributions are summarized as follows:

- We reveal that the maximum margin (MM) can be used as a signature of the attack (for attacks with a sufficiently high attack success rate) in the landscape of the victim classifier's output function, irrespective of the backdoor pattern type. Based on this, we propose MM-BD, a post-training backdoor detection method without any assumptions about the backdoor pattern type used by the attacker.
- MM-BD does not require any clean samples for detection. Moreover, as shown experimentally, it accurately, efficiently detects backdoor attacks irrespective of the number of source classes used by the attacker.
- We show superior performance of MM-BD compared with many recent post-training detectors for standard attack settings considered by these works. The experiments involve four datasets, three different types of backdoor patterns, and various DNN architectures.
- We evaluate MM-BD against many emerging backdoor attacks not considered by previous detectors. These attacks involve two additional types of backdoor patterns and six attack settings. Most of these attacks assume the attacker has strong control over the training process; by contrast, very few assumptions are made for MM-BD.
- We propose a novel backdoor mitigation approach (MM-BM) by applying an optimized upper bound to each neuron activation. This approach does not modify the DNN architecture or any trainable parameters.

2. Background

This section provides background on backdoor attacks and defenses. We start with high-level descriptions of emerging threats to machine learning (ML) systems in Sec. 2.1. In Sec. 2.2, we introduce the classical backdoor attack, including its goals and launching strategies. In Sec. 2.3 and Sec. 2.4, we give a taxonomy of backdoor defenses and advanced backdoor attacks, respectively.

2.1. Threats to Machine Learning Systems

Machine learning (ML) systems, especially deep neural networks, are starting to be adopted in various safety-critical applications, such as fraud detection, e.g., [26], [27], and healthcare, e.g., [28], [29]. However, ML systems are threatened by adversarial attacks [30]. Typically, ML systems involve a training stage and an inference stage [31]. Accordingly, adversarial attacks can also be divided into *training phase* attacks and *inference phase* attacks [32].

Inference phase attacks aim to either cause ML systems to produce adversary-selected outputs or collect evidence about the model characteristics [32]. For example, an *adversarial evasion attack* uses carefully crafted adversarial examples (i.e., inputs with adversarial perturbations) to mislead ML models to make incorrect predictions [1], [33]–[37]. As another example, *model extraction* attacks replicate the functional mapping rule of a victim ML model, an important asset of the model owner, by (e.g.) querying the victim model [38]–[41].

On the other hand, training phase attacks aim to subvert the model by modifying its training set. For example, data poisoning attacks degrade the prediction accuracy of a victim model by poisoning its training set with, e.g., mislabeled samples [42]–[46]. The backdoor attack focused on in this paper is also a type of training phase attack since the victim model will be planted with a backdoor.

2.2. Classical Backdoor Attacks

A backdoor (Trojan) attack is an important type of training phase adversarial attack mainly targeting deep neural network classifiers. For a classification domain with sample space $\mathcal{X}$ and label space $\mathcal{Y}$, a classical backdoor attack aims to have the victim classifier $f : \mathcal{X} \to \mathcal{Y}$: (a) learn to classify to the attacker's target class $t \in \mathcal{Y}$, when a test sample $\mathbf{x} \in \mathcal{X}$ from any of the source classes $\mathcal{S} \subset \mathcal{Y}$ is embedded with the backdoor pattern; and (b) correctly classify clean test samples (without the backdoor pattern) [2]. For images (which is the most common domain of focus in the backdoor literature), common backdoor pattern types include: 1) an additive perturbation $\mathbf{v}$ embedded by $\tilde{\mathbf{x}} = [\mathbf{x} + \mathbf{v}]_c$, where $||\mathbf{v}||_2$ is small (for imperceptibility) and $[\cdot]_c$ is a domain-dependent clipping function, e.g., [5], [18], [47]; 2) a local patch $\mathbf{u}$ embedded using an image-wide binary mask $\mathbf{m}$ by $\tilde{\mathbf{x}} = (1-\mathbf{m}) \odot \mathbf{x} + \mathbf{m} \odot \mathbf{u}$, where $||\mathbf{m}||_1$ is small for imperceptibility and $\odot$ represents element-wise multiplication, e.g., [2], [16]; and 3) a local or global pattern $\mathbf{u}$ "blended" using an image-wide binary mask $\mathbf{m}$ and a blending factor $\alpha \in (0, 1)$ by $\tilde{\mathbf{x}} = (1-\alpha \cdot \mathbf{m}) \odot \mathbf{x} + \alpha \cdot \mathbf{m} \odot \mathbf{u}$, where α is close to 0 for imperceptibility [5].

The classical way to launch a backdoor attack, following the protocol of BadNet [2], is by poisoning the training set of the classifier[1]. The attacker first collects a small number of source class samples, embedded with the backdoor pattern, and then (re)labels them to the target class. Then, these created samples are inserted into the training set of the victim classifier for poisoning [2]. This classical backdoor attack is considered by most works on backdoor defense, including this paper. But unlike most prior works, our defense is also effective against more recent, advanced backdoor attacks (introduced shortly) launched by more capable attackers.

1. Alternatively, backdoors may be planted by first intercepting the trained model and then directly modifying the model parameters [48]–[50].

2.3. Backdoor Defenses

Backdoor defenses can be deployed during the classifier's training phase, post-training, or during inference. Each of these scenarios assumes a different defender role and capabilities.

Backdoor defenses during training aim to produce a backdoor-free classifier from the possibly poisoned training set [7]–[14]. Existing defenses detect and remove suspicious samples from the training set [7]–[9], identify outliers, and train only on non-outlier, trustworthy samples [10]–[12], or modify the training loss or training procedure for better robustness against data poisoning [13], [14].

Backdoor defenses deployed during the inference stage aim to detect test samples embedded with the backdoor pattern [51]–[56] and may also seek to correct the decisions on these samples. One type of method perturbs an input sample by overlapping a large number of benign samples or random noises and then uses the ensemble prediction results for detection [51], [52]. In [53], [54], suspicious regions of input space are identified. In [55], [57], the latent representation of clean samples is modeled, such that test-time inputs with the backdoor pattern will be detected as outliers.

In this paper, we focus on the *post-training* scenario, where the defender has the vantage point of a user of the classifier, with no access to the training set. A primary post-training defense task is backdoor detection, where the defender aims to detect if a given classifier is backdoor attacked [25], [58]. Existing defenses either i) reverse-engineer putative backdoor patterns for all classes and detect if some of these reverse-engineered patterns are anomalous with respect to the others (*e.g.*, with an unusually small pattern size) [16]–[22]; or ii) train a "meta-classifier" to recognize features extracted from the classifier being inspected [23], [24]. Another post-training defense task is backdoor mitigation, which aims to remove the learned backdoor mapping from a victim classifier [59]. Mainstream approaches either fine-tune the classifier to "unlearn" the backdoor mapping [16], [60], [61] or prune neurons possibly activated by the backdoor pattern [62].

2.4. Advanced Backdoor Attacks

Recently, advanced backdoor attacks have been proposed to achieve better stealthiness against human inspectors. Clean-label attacks poison the training set of the classifier using samples collected from only the target class [63], [64]. These samples are perturbed to remove the original class-discriminating features and/or to embed features associated with the backdoor pattern. There is no relabeling for these samples, which helps to avoid human suspicion during training. In [47], [65], [66], the backdoor pattern is optimized (e.g. using a surrogate classifier) to be imperceptible to humans at test time. Other strategies for achieving such test-time imperceptibility include embedding the backdoor pattern in the frequency domain [67] or leveraging a warping tragrantormation on the image to be embedded [68]. Alternatively, visible backdoor patterns can be embedded subtly, e.g., as a scene-plausible physical object [69] or mimicking physical reflection on smooth surfaces [70].

There are also advanced adaptive attacks proposed to evade particular types of backdoor defenses. For example, the "all-to-all" attack in [2], can bypass backdoor defenses that assume a single target class, e.g. [16], [17]. The label-smoothed attack [71] is designed to bypass defenses based on backdoor pattern reverse-engineering [16], [18]. The Wasserstein-based backdoor attack in [72] is designed for clustering-based defenses deployed during the training phase, such as [7], [8], [73]. The sample-specific backdoor attack proposed in [74] (a.k.a. input-aware backdoor [75]) can bypass defenses that assume a common backdoor pattern (in the input space), such as [51]. However, most of these advanced backdoor attacks require strong adversary capabilities, as will be discussed next in Sec. 3. Even so, in our experiments, we will evaluate the detection capability of the proposed MM-BD against some representative advanced attacks mentioned above.

3. Threat Model

3.1. Attacker's Goals and Capabilities

In this paper, we focus on backdoor attacks against image classifiers. The extension of our method for other domains will be discussed in Sec. 6.5. In particular, we consider three types of attackers with different levels of goals and capabilities to evaluate our proposed defense thoroughly.

Basic attacker: An attack launched by a "basic" attacker is a classical backdoor attack discussed in Sec. 2.2. The attack is associated with one backdoor target class, an arbitrary number of source classes, and a common backdoor pattern, such that samples from the source class are supposed to be misclassified to the target class when the backdoor pattern is present, with backdoor-free samples correctly classified. The attacker has the ability to poison the training set [76]. But the attacker has neither access to the samples originally in the classifier's training set, nor to the training process itself. In this paper, we compare our proposed MM-BD and our mitigation approach with other methods that address basic attackers.

Advanced attacker: These attackers are motivated by goals in addition to those for a basic attacker, such as the stealthiness of the attack against human inspection, and the evasiveness of the attack against backdoor defenses. Most of these additional goals cannot be achieved by a basic attacker. Thus, an advanced attacker is endowed with additional capabilities such as: (a) the capability to collect sufficient data and to train a surrogate classifier, and/or even (b) full control of the victim's training process. The latter is a particularly significant assumption which is valid only for a few scenarios, e.g., where there is an insider or training is outsourced to a third party which happens to be an attacker. In our experiments, MM-BD will be evaluated against several advanced backdoor attacks mentioned in Sec. 6.3.

Adaptive attacker: The strongest attacker considered in this

paper is an adaptive attacker who, in addition to the goals of a basic attacker, also aims to defeat mounted defenses. An adaptive attacker is stronger than the previous two types of attackers, with full control of the training process and full knowledge of the defense. In Sec. 6.6.1, we create a strong, optimized adaptive attack based on these capabilities. We show that to bypass MM-BD, the attacker needs to solve a complex min-max optimization problem, which is time-consuming.

3.2. Defender's Goals and Assumptions

In this paper, we consider the practical post-training defense scenario where the defense is applied after the classifier has been trained, and without access to the training set [15]. The most important goal for post-training defense is to detect if the classifier was backdoor-attacked. If an attack is detected, the user could choose a replacement classifier, *if* one is available. Otherwise, the backdoor attack should be mitigated, so that: a) the classifier predicts to the original source class when a test sample is embedded with the backdoor pattern, and b) the classification accuracy on clean, backdoor-free test samples is not significantly degraded.

The assumptions associated with the post-training defense scenario are summarized as follows:
(1) *The defender does not know a priori if there is an attack.* This is the fundamental assumption for the post-training backdoor detection problem to be meaningful.
(2) *No information about the backdoor pattern is available to the defender.* This assumption is relaxed by many existing post-training detectors – THEY make assumptions about the backdoor pattern type, or how human-imperceptibility is achieved (e.g., a small ℓ_2 norm for additive perturbation backdoor patterns in Sec. 2.2). However, MM-BD strictly *makes no assumptions* about the backdoor pattern.
(3) *The defender has no access to the classifier's training set.* The defender is a user of the classifier, or the user of a legacy system. In the former (proprietary) case, the training set of the classifier may not be publicly available, while in the latter (legacy) case it has long been forgotten.
(4) *There are no clean classifiers trained for the same domain.* Otherwise, the defender may directly use the clean classifier (possibly after some fine-tuning) in replacement of the classifier to be inspected. More importantly, post-training detection will be an easier "semi-supervised" problem if clean classifiers are available, since these classifiers can be used (e.g.) to set a conformal threshold for detection inference [77].
(5) *The defender is able to independently collect a small, clean data set containing samples from all classes in the domain.* This assumption is used by most post-training detectors, e.g., [16]–[21]. However, MM-BD *does not need* any clean samples for detection. This makes MM-BD applicable to scenarios where clean samples are very rare or expensive.

4. Related Work

Post-training backdoor detection methods. Existing methods include a family of "meta-learning" approaches, where a large number of shadow neural networks, labeled "attack" or "no attack", are trained – features extracted from these labeled networks are treated as supervised examples, input to a binary "meta-classifier" trained to discriminate "attack" from "no attack" [23], [24]. However, these methods employ a pool of backdoor pattern types, and may fail when the actual type used by the attacker is not in the pool. Moreover, training the shadow networks requires a relatively large number of clean samples and is heavily computational. Another family of detectors trial reverse-engineer the backdoor pattern for each putative target class [16]–[20], [22], [78]. Such reverse-engineering is performed using a small set of clean samples [16], or using simulated samples obtained by model inversion [79], [80]. Detection inference is then based on statistics obtained from the estimated backdoor patterns (e.g. the ℓ_1 norm of the estimated mask for patch replacement backdoor patterns). However, reverse-engineering relies on knowledge of the backdoor pattern type [16], [18], [24]. Also, most of these reverse-engineering-based methods, except [79], [80], require some clean samples, which are not always available as we have discussed in Sec. 3.2. In comparison, MM-BD does not rely on knowledge of the backdoor pattern type and does not need any clean samples. Moreover, some reverse-engineering-based methods fail when the backdoor attack involves only a few source classes, as shown in [18], [21]. [18] addressed this issue with a significant increment in computational complexity. While [21] estimates the source classes and the target class via a complicated optimization procedure, our method can accurately detect backdoor attacks with an arbitrary number of source classes and is computationally efficient, as will be shown in our experiments.

Post-training backdoor mitigation methods. In [16], [60], [61], a detected classifier is fine-tuned on a large number of clean samples to "unlearn" the backdoor mapping. In [62], [81], neurons possibly associated with the backdoor pattern are pruned based on their activation or modulus of (Lipschitz) continuity. Although the main focus of our paper is backdoor detection, in Sec. 5.3, we also propose a method that mitigates backdoors, for most types of backdoor patterns, using very few clean samples, and without fine-tuning any of the classifier's original parameters. Our method can also be combined with existing ones to achieve even better performance.

Backdoor defenses addressing various backdoor pattern types. There are many backdoor defenses capable of addressing various backdoor pattern types. In particular, most training phase backdoor defenses, such as [7]–[14], and some inference-stage backdoor defenses, such as [51], [55], [57], are not designed with any particular type of backdoor pattern in mind (i.e., they are backdoor agnostic). However, for the post-training backdoor detection problem focused on in this paper, the defender has no access to any actual backdoor patterns, which is different from the previous two defense scenarios. Thus, some post-training detectors tend to rely on assumptions about the backdoor pattern incorporation mechanism. For example, [16] addresses patch-based

patterns, [18] assumes imperceptible perturbation patterns, and [82] considers attacks that do not even involve a backdoor pattern. In contrast to these, our post-training detector makes no assumptions about the incorporation mechanism of the backdoor pattern.

5. Method

In this section, we first discuss the key ideas behind our detector in Sec. 5.1. In Sec. 5.2 we present our detection procedure, which consists of an estimation step to obtain a novel *maximum margin* (MM) statistic for each class, followed by unsupervised detection inference. We then propose a mitigation method in Sec. 5.3.

5.1. Key Ideas

Unlike existing detectors based on the small patch size ([16]) or the small perturbation size ([18]) of the backdoor pattern, our detector is based on the influence of a backdoor attack on the landscape of the classifier's logits $\{g_c(\cdot)|c \in \mathcal{Y}\}$ (i.e., the activations directly before the softmax layer), independent of the backdoor pattern type. Consider a backdoor attack with target class $t \in \mathcal{Y}$ and denote the classifier's logit function associated with any class $c \in \mathcal{Y}$ as $g_c : \mathcal{X} \to \mathbb{R}$ (assuming $\mathcal{X}$ compact without loss of generality). For all $c \in \mathcal{Y} \setminus t$, regardless of the backdoor pattern type, we will likely observe:

$$\max_{\mathbf{x}\in\mathcal{X}} \left[g_t(\mathbf{x}) - \max_{k\in\mathcal{Y}\setminus t} g_k(\mathbf{x})\right] \gg \max_{\mathbf{x}\in\mathcal{X}} \left[g_c(\mathbf{x}) - \max_{k'\in\mathcal{Y}\setminus c} g_{k'}(\mathbf{x})\right] \quad (1)$$

I.e., the *maximum margin statistic* for the true backdoor attack target class (defined as the left hand side of (1)) will tend to be *much larger* than the maximum margin statistics for all other classes (right hand side of (1)).

Why a backdoor attack causes the above phenomenon? Note that a backdoor pattern is a common pattern[2] embedded in all samples used for poisoning the classifier's training set; and the same backdoor pattern will be embedded in test samples to induce test-time misclassifications. By contrast, class-discriminating features not associated with the backdoor pattern typically exhibit high variability[3] – e.g., a 'bird' category with multiple bird species, or even the same object captured under different views, range, or lighting conditions. The commonality of a backdoor pattern is critical for it to override class-discriminating features that will favor deciding to the source class and so that the backdoor pattern can be easily learned (at a low poisoning rate [4]) by the victim classifier during training. However, the repetition[4] of the common backdoor pattern in the training set also induces an inevitable overfitting which: a) boosts the target class logit (by causing abnormally large activations from neurons positively correlated with this logit), and b) suppresses the logits of all other classes (as will be shown empirically in Apdx. D). Consequently, an abnormally large margin between the target class logit and logits of all other classes will be created, due to both the "boosting" and the "suppression" effects.

It is worth mentioning that the concurrent work [84] also reveals that the internal feature representation of backdoor samples will be dominated by the backdoor pattern rather than by the benign class-discriminative features, and such domination is mainly caused by overfitting. However, [84] focuses on during-training defense, with the training set available, while our work is post-training, with the training set unavailable.

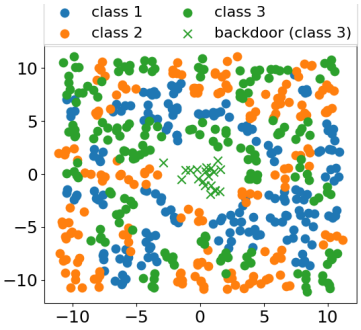
The above reasoning is illustrated in Fig. 1 using a toy, visualizable example. We consider two-dimensional inputs from three classes, with the sample distribution for each class specified by a Gaussian mixture model (with a large number of components to mimic the high variability of class-discriminating features). We generated 500 training samples per class, and launched two backdoor attacks (both with target class 3), respectively with 10 (BA-10) and 100 (BA-100) "backdoor samples" inserted for poisoning. For each backdoor attack, we trained a multilayer perceptron with three hidden layers, which achieves nearly 91% accuracy on clean test samples for both backdoor attacks. The attack success rates of the trained classifiers on test "backdoor samples" are 92% and 99% for BA-10 and BA-100, respectively. In Fig. 1, for each backdoor attack and each class, we plot the margin between a class's logit and the largest logit among the other two classes as a function of the input space (only positive values are kept for better visualization). We observe that the backdoor target class (Fig. 1f and 1g) has abnormally large maximum margin for both backdoor attacks (which agrees with Eq. (1)). Moreover, compared with BA-10, the atypicality of the maximum margin for the target class is more obvious for BA-100 (note that the higher poisoning rate may be preferred by the attacker since this yields a higher attack success rate).

To give another perspective on our hypothesis, consider a simple linear model $g(\cdot) : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ where the output related to class $c \in \mathcal{Y}$ is given by the logit g_c formed by the column-vector dot-product, $g_c(x) = \mathbf{w}_c'\mathbf{x}$, where: $\mathbf{w}_c$ represents the weight vector related to class c, which has the same dimension as input $\mathbf{x}$; and the class decision c of the model for $\mathbf{x}$ has largest associated logit $g_c(\mathbf{x})$. Also assume that, after training, the model can correctly classify all the training samples with confidence higher than τ, assessed by the margin of the correct class (which is the logit for that

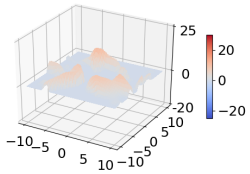
2. Some recently proposed advanced backdoor attacks use backdoor patterns that are sample-specific in the input space [74], [75] (Sec. 2.4). However, these backdoor patterns embedded in different samples still share common semantic features and are similar to each other in some latent embedding space. Thus, these backdoor attacks are still detectable by our method, as will be shown in Sec. 6.3.

3. In extreme cases, class-discriminating features can be highly common across samples from the same class, which creates an "intrinsic" backdoor that is hardly distinguishable from backdoors planted by an attacker [83]. A general solution to rule out such intrinsic backdoors is still an open problem.

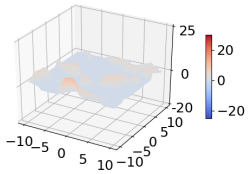
4. We acknowledge a severely imbalanced training set may cause a similar overfitting phenomenon, which may lead to false detections. However, an imbalanced training set itself is harmful to the classifier, as will be discussed in detail in Sec. 7.2.1.



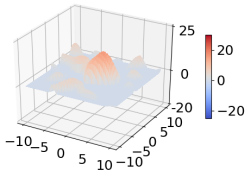
(a) Illustration of the distribution of samples from all three classes.



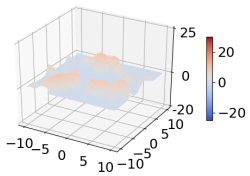
(b) BA10-class1



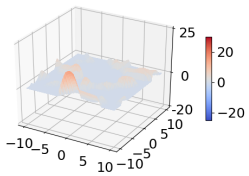
(d) BA10-class2



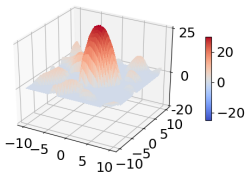
(f) BA10-class3



(c) BA100-class1



(e) BA100-class2



(g) BA100-class3

Figure 1: For two-dimensional input and three classes following the sample distribution in (a), consider two backdoor attacks (BA-10 and BA-100) with different poisoning rates but the same target class 3. For both backdoor attacks, the maximum margin for target class 3 ((f) and (g)) is abnormally large compared with that for the non-target classes ((b), (c), (d), (e)).

class minus the maximum logit among all other classes):

$$g_y(\mathbf{x}) - \max_{c \neq y} g_c(\mathbf{x}) = \mathbf{w}_y'\mathbf{x} - \max_{c \neq y} \mathbf{w}_c'\mathbf{x} \geq \tau, \quad \forall (\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}. \tag{2}$$

Suppose a backdoor pattern $\mathbf{v}$ is additively incorporated into a clean training image $\mathbf{x}_s$, which is originally from source-class $s \neq t$ and relabeled to target class t. Eq. (2) implies:

$$g_t(\mathbf{x}_s+\mathbf{v}) - g_s(\mathbf{x}_s+\mathbf{v}) = \mathbf{w}_t'(\mathbf{x}_s+\mathbf{v}) - \mathbf{w}_s'(\mathbf{x}_s+\mathbf{v}) \geq \tau. \tag{3}$$

Based on the confident margin assumption in Eq. (2), for the clean x_s we have

$$\mathbf{w}_s'\mathbf{x}_s - \mathbf{w}_t'\mathbf{x}_s \geq \tau, \tag{4}$$

So, adding Eq. (3) and Eq. (4) gives:

$$g_t(\mathbf{v}) - g_s(\mathbf{v}) = \mathbf{w}_t'\mathbf{v} - \mathbf{w}_s'\mathbf{v} \geq 2\tau. \tag{5}$$

That is, the lower bound of the maximum margin for the backdoor target class is at least 2τ, *larger* than the lower bound for legitimate samples from all classes.

5.2. Detection Procedure

Estimation step. For each class $c \in \mathcal{Y}$, we estimate a maximum margin statistic by solving:

$$\underset{\mathbf{x} \in \mathcal{X}}{\text{maximize}} \quad g_c(\mathbf{x}) - \max_{k \in \mathcal{Y} \setminus c} g_k(\mathbf{x}) \tag{6}$$

using gradient ascent (common for model inversion [80] and related applications, e.g., generating adversarial examples [85]) with projection onto $\mathcal{X}$ (e.g. $\mathcal{X} = [0,1]^{H \times W \times C}$ for color images with height H, width W, and C channels).

Note that $\mathcal{X}$ is a closed convex set; thus, the continuous logit function on $\mathcal{X}$ is bounded and Lipschitz. In other words, there exist closed balls in which the logit function is convex and with a local maximum. Then, according to Theorem 3.2 in [86], convergence is guaranteed for gradient ascent from any random initialization in $\mathcal{X}$ with projection onto $\mathcal{X}$. As is common practice, we perform multiple random initializations in $\mathcal{X}$ (e.g., for images, pixel values are uniformly randomly initialized in the interval $[0, 1]$), and pick the largest local optimal solution. Compared with reverse-engineering-based defenses that may suffer from mismatch between the assumed backdoor pattern embedding type and the true attack backdoor pattern type, our optimization problem does not require assuming a backdoor pattern embedding type. Moreover, reverse-engineering-based defenses' backdoor pattern estimation using clean samples from *all* non-target classes has been found experimentally to fail when the majority of these classes are not source classes [21]. By contrast, our method can detect backdoor attacks with an arbitrary number of source classes and does not require knowledge of any legitimate samples from the domain.

Detection inference step. We propose an unsupervised anomaly detector as given in [18]. Denote the estimated maximum margin statistic for each class $c \in \mathcal{Y}$ as r_c and the largest statistic as $r_{\max} = \max_{c \in \mathcal{Y}} r_c$. We hypothesize that, when there is a backdoor attack, $r_{\max}$ will be associated with the target class and will be an outlier to the distribution of maximum margin statistics for non-target classes. Thus, we estimate a null distribution H_0 using all statistics excluding $r_{\max}$. Given that the maximum margin is strictly positive (both theoretically and empirically for the estimated maximum margin in our experiments), we choose single-tailed density forms for the null distribution, e.g. Gamma distribution, in our experiments. In order to evaluate the atypicality of $r_{\max}$ under the estimated null, we compute an order-statistic p-value:

$$\text{pv} = 1 - H_0(r_{\max})^{K-1}, \tag{7}$$

where $K = |\mathcal{Y}|$ is the total number of classes in the domain. It can easily be shown that pv follows a uniform distribution on $[0, 1]$ under the null hypothesis of "no attack". Thus, we claim a detection with confidence $1 - \theta$ (e.g. with the classical $\theta = 0.05$) if $\text{pv} < \theta$. If a backdoor attack is detected, the class associated with $r_{\max}$ is inferred as the backdoor target class.

5.3. Mitigation of Backdoor Attacks

When a backdoor attack is detected, predictions made by the victim classifier to classes other than the detected target class, or predictions made on the test examples that are not embedded with backdoor trigger might still be trustworthy. An option is to *mitigate* the backdoor attack.

Our mitigation approach is based on the observation that a backdoor attack induces a subset of neurons in each layer to have abnormally large activations. Such "large activation" phenomenon is also the basis of the backdoor *detection* approach in [19], though several hyperparameters are required, e.g., to identify the neurons responsible for the large activations. In contrast, we apply to *every neuron* a specific optimized upper bound in order to suppress any possible large activations caused by a backdoor attack, without significant degradation in the classifier's accuracy on clean samples. Let $\sigma_l : \mathbb{R}^{n_{l-1}} \rightarrow \mathbb{R}^{n_l}$ be the activations of the l-th layer (for $l = 1, \cdots, L$) of the victim classifier (as a function of the activations of the previous layer). Here, we do not explicitly represent the parameters in each layer for brevity, since none of them will be modified during our *mitigation* process. Then, the logit function for any class $c \in \mathcal{Y}$ and any input $\mathbf{x} \in \mathcal{X}(= \mathbb{R}^{n_0})$ can be written as:

$$g_c(\mathbf{x}) = \mathbf{w}_c^T(\sigma_L \circ \cdots \circ \sigma_1(\mathbf{x})) + b_c, \tag{8}$$

where $\mathbf{w}_c \in \mathbb{R}^{n_L}$ and $b_c \in \mathbb{R}$ are the weight vector and bias associated with class c respectively. For each layer $l = 2, \cdots, L$, we also denote a bounding vector $\mathbf{z}_l \in \mathbb{R}^{n_l}$, such that the logit function, *with bounded activation*, for each class $c \in \mathcal{Y}$ and any input $\mathbf{x}$ can be represented by:

$$\begin{aligned}\bar{g}_c(\mathbf{x}; \mathbf{Z}) = \mathbf{w}_c^T(\bar{\sigma}_L(\bar{\sigma}_{L-1}(\cdots \bar{\sigma}_2(\sigma_1(\mathbf{x}) \\ ; \mathbf{z}_2) \cdots ; \mathbf{z}_{L-1}); \mathbf{z}_L)) + b_c,\end{aligned} \tag{9}$$

where $\mathbf{Z} = \{\mathbf{z}_2, \cdots, \mathbf{z}_L\}$ and $\bar{\sigma}_l(\cdot; \mathbf{z}_l) = \min\{\sigma_l(\cdot), \mathbf{z}_l\}$ for any $l = 2, \cdots, L$ (and where the 'min' operator is applied to each component of the vector). To find the minimum activation upper bound for each neuron without affecting the classifier's performance on clean test samples, we propose to solve the following problem on a small set $\mathcal{D}$ of clean samples:

$$\begin{aligned}&\min_{\mathbf{Z}=\{\mathbf{z}_2, \cdots, \mathbf{z}_L\}} \sum_{l=2}^{L} ||\mathbf{z}_l||_2 \\ &\text{subject to } \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}, y) \in \mathcal{D}} \mathbb{1}[y = \arg\max_{c \in \mathcal{Y}} \bar{g}_c(\mathbf{x}; \mathbf{Z})] \geq \pi,\end{aligned} \tag{10}$$

where $\mathbb{1}[\cdot]$ represents the indicator function, and π is the minimum accuracy benchmark (e.g., set $\pi = 0.95$). Here, we minimize the ℓ_2 norm of the bounding vectors to penalize activations with overly large absolute values in each layer. To practically solve the above problem, we propose to minimize the following Lagrangian using gradient descent:

$$\begin{aligned}L(\mathbf{Z}, \lambda; \mathcal{D}) = \frac{1}{|\mathcal{D}| \times |\mathcal{Y}|} \sum_{(\mathbf{x}, y) \in \mathcal{D}} \sum_{c \in \mathcal{Y}} [\bar{g}_c(\mathbf{x}; \mathbf{Z}) \\ - g_c(\mathbf{x})]^2 + \lambda \sum_{l=2}^{L} ||\mathbf{z}_l||_2,\end{aligned} \tag{11}$$

where $\mathbf{Z}$ is initialized large. The first term of Eq. (11) aims to keep the classifier's logits for the samples in $\mathcal{D}$ unchanged. Such a design not only helps satisfy the accuracy constraint in problem (10), but also avoids having the logit associated with the class label for a given sample to further grow (i.e. overfit), allowing mitigation to be achieved with limited samples[5]. λ is updated automatically to fulfill the constraint of problem (10). This process is summarized in Alg. 1. Note that we initialize the values in $\mathbf{Z}$ large enough such that no activation bounding/saturation is initially performed. This can be easily achieved by feeding in clean samples to get a rough range for the activations, and then setting the initial upper bound to a magnitude larger than typical activations.

Finally, a class posterior with the backdoor mitigated is obtained by applying a softmax to the logits $\{\bar{g}_c(\mathbf{x}; \mathbf{Z}^*)\}_{c \in \mathcal{Y}}$.

6. Experiments

Experiments are conducted mainly on four benchmark datasets with different image resolutions, image sizes and numbers of classes: CIFAR-10, CIFAR-100 [87], TinyImageNet, and GTSRB [88] (with more details in Apdx. A.1).

6.1. Main Experiments for Backdoor Detection

In the following, we show the effectiveness of our backdoor detector MM-BD in terms of *detection accuracy* and *computational efficiency* compared with several state-of-the-art post-training detectors for a variety of backdoor attack configurations (consisting of the backdoor pattern type and the number of source classes). While here we focus on classical backdoor attacks commonly considered by existing backdoor detection works and recent competitions like [58], the success of MM-BD against many other advanced backdoor attack settings and adaptive backdoor attacks is shown in Sec. 6.3.

6.1.1. Settings. We consider three common backdoor embedding types from the backdoor attack literature: *additive*, *patch replacement*, and *blending* (with associated embedding functions in Sec. 2.2). Effectiveness of MM-BD on warping-based backdoor patterns and sample-specific backdoor patterns is shown in Sec. 6.3. In particular, we consider the following five respective backdoor patterns. For the additive backdoor pattern, we consider a *global* "chessboard" pattern from [18] and a *local* "1-pixel" perturbation from [7]. For the patch replacement backdoor pattern type, we consider a BadNet pattern from [2] and a "*unicolor*"

5. Other choices, such as a differentiable surrogate for correct classification count, might lead to overfitting.

patch from [16]. For the blended backdoor pattern type, we consider a "blended" *noisy* patch from [18]. Examples of these backdoor patterns and more details are in Apdx. A.2.

Like most existing works (e.g. [16]–[19]), we consider backdoor attacks with one target class (randomly selected for each attack). However, we allow one or multiple source classes – for brevity, these two source class settings are denoted as 'S' (for "single") and 'M' (for "multiple") respectively. For each backdoor attack with the 'S' setting, the source class is randomly selected. For TinyImageNet, we randomly select ten source classes for each backdoor attack with the 'M' setting; while for the other three datasets, all classes other than the target class are selected as source classes (which is assumed by the detectors in [16], [17], [19]) for the 'M' setting. With the above notations, a backdoor attack with "chessboard" backdoor pattern and multiple source classes is denoted 'chessboard-M'.

For CIFAR-10, we created ten ensembles of backdoor attacks for all ten combinations of 5 different backdoor patterns and 'S'/'M' respectively. For CIFAR-100 and GTSRB, we created five backdoor attack ensembles for 5 backdoor patterns respectively, all with the 'M' setting. We did not create backdoor attacks with a single source class for these two datasets because we cannot generate sufficient backdoor training images from a single class to launch a successful backdoor attack, given the limited number of images in each class. For TinyImageNet, we only generated one backdoor attack ensemble with the setting 'BadNet-M' (which is arbitrarily selected) since the amount of time just to train a classifier on this dataset is extraordinarily high. For each ensemble, ten different attacks were generated independently according to the specified settings using the classical "data poisoning" protocol in [2]. Other configurations, including the number of backdoor training images created for backdoor attacks in each ensemble, are deferred to Apdx. A.3.

We trained one classifier for each backdoor attack. The DNN architectures for backdoor attacks on CIFAR-10, CIFAR-100, TinyImageNet, and GTSRB were ResNet-18 [89], VGG-16 [90], ResNet-34 [89], and MobileNet [91], respectively. For each dataset, we also created an ensemble of clean classifiers to evaluate the false detection rate. All the backdoor attacks we created are successful with high attack success rate (ASR) and negligible degradation in clean test accuracy (ACC)[6], compared with the clean classifiers trained for the same dataset. More details are shown in Apdx. A.4.

6.1.2. Detection Performance. We compare MM-BD with six state-of-the-art post-training detection methods: NC [16], TABOR [17], ABS [19], PT-RED [18], META [24], and TND [22]. We followed the original implementations of these methods with only modest changes (e.g. choosing the best detection threshold to maximize their performance). Especially for META, we used the official code to train the "meta-classifier" for detection. For MM-BD, we solved problem (6) using gradient ascent with convergence criterion[7] $\epsilon = 10^{-5}$ and 30 random initializations. These choices are not critical to the detection accuracy. For the inference stage, the detection threshold was set to $\theta = 0.05$, i.e., 0.95 detection confidence, which is the classical threshold for statistical hypothesis tests and was kept fixed in all experiments in this paper.

In Tab. 1, we show detection accuracy of MM-BD compared with the other methods on the backdoor attack ensembles we created and also report the number of legitimate images per class, N_{img}, used by each method. A successful detection requires both the backdoor attack to be detected and the target class to be correctly inferred. We also show the proportion of clean classifiers deemed to be not attacked by each detector. We only evaluated META on CIFAR-10 since the computational cost for applying META to backdoor attacks on the other datasets is overly high (over 24 hours). For PT-RED, which estimates a backdoor pattern for each (source, target) class pair, we reduced its complexity by estimating a backdoor pattern only for each putative target class on CIFAR-100, TinyImageNet, and GTSRB – otherwise, repeated experiments on these datasets will be infeasibly complex due to the large number of classes. Also due to the time constraint (and space limitations), we include detection results for only a few arbitrarily selected backdoor attack ensembles for CIFAR-100, TinyImageNet, and GTSRB, respectively. Complete results for MM-BD for *all* backdoor attack ensembles are in Apdx. B.1.

As we have discussed in Sec. 4, existing post-training detectors presume one or several backdoor pattern types. For example, NC shows strong capability in detecting backdoor attacks with patch replacement backdoor patterns (BadNet&unicolor), for which it is designed; but NC fails to detect backdoor attacks with local additive backdoor patterns (1-pixel)[8]. Similar results are reported for ABS and META, which are designed for patch replacement/blending backdoor patterns (BadNet, unicolor, and blend), while not addressing additive backdoor patterns (chessboard&1-pixel). In contrast, PT-RED performs well for additive backdoor patterns (chessboard&1-pixel), for which it is designed, but is generally ineffective for the other backdoor patterns (BadNet-blend). TABOR and TND do not show competitive performance compared with the other methods, since they adopt additional constraints on the shape or color of the backdoor pattern. Different from all these methods, MM-BD achieves *high detection accuracy* for all backdoor patterns with a *low false detection rate* for all datasets; *i.e.*, its performance is largely **invariant to the backdoor pattern type**. Even if ABS (effective for patch backdoor patterns BadNet, unicolor and blend) and PT-RED (effective for additive backdoor patterns chessboard-1-pixel) are jointly deployed (detecting if either one detects), the detection accuracy on classifiers with backdoor attacks is just comparable to MM-BD, but with more false detections and a

6. ASR and ACC are commonly used for assessing the effectiveness of backdoor attacks; see definitions in [18], [22].

7. Stopping when the relative change of the objective function between any two consecutive iterations is less than ϵ.

8. NC does detect global additive backdoor patterns when there are multiple source classes (similar results are reported in [18]).

	N_{img}	CIFAR-10 clean	chessboard-S	chessboard-M	1-pixel-S	1-pixel-M	BadNet-S	BadNet-M	unicolor-S	unicolor-M	blend-S	blend-M
NC [16]	10	56.7±49.6	66.7±47.1	100.0±0.0	56.7±49.6	36.7±48.2	93.3±24.9	96.7 ± 18.0	90.0±30.0	76.7±42.3	36.7±48.2	33.3±47.1
TABOR [17]	10	70.0±45.8	70.0±45.8	93.3±24.9	73.3±44.2	93.3±24.9	63.3±48.2	73.3±44.2	26.7±44.2	66.7±47.1	73.3±44.2	86.7±34.0
ABS [19]	1	93.3±24.9	26.7±44.2	66.7±47.1	66.7±47.1	80.0±40.0	46.7±49.9	96.7±18.0	73.3±44.2	46.7±49.9	80.0±40.0	90.0±30.0
META [24]	10k	73.3±44.2	76.7±42.3	60.0±49.0	10.0±30.0	6.7±24.9	83.3±37.3	83.3±37.3	30.0±45.8	30.0±45.8	76.7±42.3	86.7±34.0
TND [22]	5	50.0±50.0	10.0±30.0	26.7±44.2	46.7±49.9	83.3±37.3	20.0±40.0	33.3±47.1	6.7±24.9	3.3±18.0	40.0±49.0	56.7±49.6
PT-RED [18]	100	76.7±42.3	100.0±0.0	96.7±18.0	93.3±24.9	100.0±0.0	23.3±42.3	13.3±34.0	16.7±37.3	23.3±42.3	46.7±49.9	63.3±48.2
PT-RED+ABS	100	66.7±47.1	100.0±0.0	100.0±0.0	93.3±24.9	100.0±0.0	56.7±49.6	96.7±18.0	76.7±42.3	56.7±49.6	86.7±34.0	96.7±18.0
MM-BD(ours)	0	86.7±34.0	86.7±34.0	90.0±30.0	83.3±37.3	100.0±0.0	90.0±30.0	100.0±0.0	76.7±42.3	100.0±0.0	86.7±34.0	100.0±0.0
		CIFAR-100				TinyImageNet		GTSRB				
		clean	1-pixel-M	BadNet-M	unicolor-M	clean	BadNet-M	clean	chessboard-M	1-pixel-M	unicolor-M	blend-M
NC	1	53.3±49.9	40.0±49.0	93.3±24.9	90.0±30.0	30.0±45.8	80.0±40.0	66.7±47.1	90.0±30.0	66.7±47.1	20.0±40.0	63.3±48.2
TABOR	1	43.3±49.6	70.0±45.8	60.0±49.0	56.7±49.6	40.0±49.0	70.0±45.8	56.7±49.6	90.0±30.0	80.0±40.0	20.0±40.0	33.3±47.1
ABS	1	96.7±18.0	20.0±40.0	86.7±34.0	90.0±30.0	90.0±30.0	20.0±40.0	83.3±37.3	30.0±45.8	76.7±42.3	56.7±49.6	60.0±49.0
TND	1	26.7±44.2	26.7±44.2	23.3±42.3	20.0±40.0	50.0±50.0	30.0±45.8	60.0±49.0	20.0±40.0	30.0±45.8	0.0±0.0	10.0±30.0
PT-RED	2	90.0±30.0	93.3±24.9	33.3±47.1	20.0±40.0	100.0±0.0	0.0±0.0	66.7 ±47.1	100.0±0.0	80.0±40.0	60.0±49.0	56.7±49.6
MM-BD(ours)	0	100.0±0.0	100.0±0.0	100.0±0.0	90.0±30.0	100.0±0.0	90.0±30.0	90.0±30.0	73.3±44.2	100.0±0.0	86.7±34.0	100.0±0.0

TABLE 1: Detection accuracies (%). For CIFAR-10, CIFAR-100, and GTSRB, 30 clean models and 30 models for each attack are trained. For TinyImageNet 10 clean models and 10 attack models are trained. Accuracy on clean models equals one minus the false positive rate.

	CIFAR10	CIFAR100	TinyImageNet	GTSRB
NC	308.4±50.1s	800.4±147.5s	11227.0±1537.8s	412.5±51.4s
TABOR	57.7±4.3s	341.1±25.2s	10792.2±824.6s	138.7±6.5s
ABS	50.3±3.2s	234.6±14.5s	819.1±91.7s	92.8±7.1s
META	32h	-	-	-
TND	591.9±16.6s	8207.3±257.2s	53530.8±1035.1s	1161.0±33.8s
PT-RED	342.5±37.2s	-	-	-
MM-BD (ours)	**27.2±3.4s**	**114.8±10.3s**	**503.4±42.1s**	**37.1±4.2s**

TABLE 2: Average execution times (with standard deviation).

significant increment in computational cost. Moreover, NC, TABOR, ABS, and TND assume backdoor attacks have multiple source classes (the 'M' setting); thus they may easily fail for backdoor attacks with a single source class (the 'S' setting). However, MM-BD is generally effective in detecting backdoor attacks with **arbitrary number of source classes**, as shown in Tab. 1, as it does not make any assumptions about the number of source classes. Finally, different from all the other methods, MM-BD **does not need any clean images for detection**.

In Tab. 2, we show average execution times (in seconds). All experiments were conducted on a NVIDIA RTX-3090 GPU. Clearly, MM-BD is more efficient than most other post-training detectors. STRIP [51] is also known for its high efficiency, but it is an inference-time method, which detects if a test sample is embedded with a backdoor trigger, while our method detects if the *model* was attacked. The inference time for META [24] to inspect a single model is almost negligible. But for the unsupervised detection setting, in practice, the model to be inspected is typically associated with a new domain – thus, META will need to train a large number of shadow models as well as the binary meta-classifier. This is very time-consuming.

6.2. Experiments for Backdoor Attack Mitigation

We evaluate our backdoor mitigation approach, dubbed "MM-BM" in the following, using the ten backdoor attack ensembles created in Sec. 6.1.1 for CIFAR-10, compared with 3 other backdoor mitigation approaches: NC-M (the mitigation approach associated with NC) [16], fine-pruning (FP) [62], and NAD [61] (Notably, backdoor detection and mitigation are two different tasks; here we only consider mitigation methods). NC-M embeds the backdoor pattern reverse-engineered for the target class detected by NC into clean images from all source classes, and then fine-tunes the classifier deemed to be attacked on these images to "unlearn" the backdoor mapping. Here, we evaluate NC-M regardless of whether the backdoor attack is successfully detected by NC – we apply NC-M to each backdoor attack in each ensemble using the backdoor pattern estimated by NC for the ground-truth target class. FP, on the other hand, removes neurons with low activations on clean images (these neurons are hypothesized to be "reserved" for backdoor patterns) subject to a prescribed budget of accuracy degradation; and then fine-tunes the classifier to recover its accuracy. Since FP does not perform backdoor attack detection, we directly apply it to all backdoor attack ensembles on CIFAR-10. NAD [61] erases the backdoor attack by knowledge distillation. A *teacher* is first obtained by fine-tuning the attacked model on a small clean data set; then the attacked model is fine-tuned under the guidance of the teacher model's activations. MM-BM is implemented following the description in Sec. 5.3, with the accuracy constraint set to $\pi = 0.95$ and 20 clean images per class used for mitigation. Again, these settings are not critical to the performance of MM-BM. For the convolutional neural networks used in our experiments, we apply a common activation upper bound to all neurons associated with the same convolutional filter, since all the neuron activations in the feature map produced by a given convolutional filter should have the same dynamic range. More implementation details are deferred to Apdx. B.2 due to space limitations.

In Tab. 3, we show the average ASR and average ACC of classifiers over each of the ten backdoor attack ensembles created on CIFAR-10, and after applying each of NC-M, FP, and MM-BM. We also show the number of clean images per class used by each method. Compared with the baseline without backdoor mitigation, NC-M significantly reduces ASR for most backdoor attack ensembles, although

	N_{img}		chessboard-S	chessboard-M	1-pixel-S	1-pixel-M	BadNet-S	BadNet-M	unicolor-S	unicolor-M	blend-S	blend-M
Without Mitigation		ASR	99.9±0.1	99.9±0.1	91.1±7.8	91.3±6.9	99.4±0.3	99.9±0.1	97.8±1.3	98.4±1.6	96.4±2.8	96.9±3.4
		ACC	91.3±0.9	91.1±0.8	91.8±0.4	91.1±0.3	91.6±0.4	91.6±0.3	91.3±0.6	91.4±0.7	91.4±0.4	91.5±0.4
NC-M [16]	500	ASR	39.5±37.9	38.5±27.6	26.9±14.4	61.2±25.0	21.8±15.3	28.3±19.3	55.9±33.3	93.2±6.4	13.4±12.5	47.5±43.2
		ACC	87.0±1.9	87.7±0.9	90.8±0.5	86.0±4.2	84.9±4.9	76.7±6.5	86.0±1.1	85.1±5.4	88.4±2.1	88.2±1.6
Fine-Pruning [62]	500	ASR	31.9±44.1	52.4±45.2	61.1±36.2	71.6±33.8	86.7±21.9	89.5±17.2	89.2±11.6	86.9±13.9	65.4±32.3	75.5±27.9
		ACC	90.7±1.0	90.6±0.7	91.2±1.0	91.5±0.6	91.2±1.0	91.6±0.5	91.3±0.2	90.9±1.8	91.6±0.7	91.6±0.6
NAD	250	ASR	3.2±2.5	3.0±2.0	3.6±4.1	2.2±1.6	**8.8**±6.7	**4.0**±3.6	3.9±3.2	1.8±0.3	**1.7** ±0.7	**2.1**±1.8
		ACC	82.7±2.4	81.1±2.5	83.5±3.0	84.3±4.3	**87.5**±1.2	**88.1**±1.8	82.5±3.1	82.8±2.5	**87.0**±1.1	**88.4**±1.6
MM-BM(ours)	20	ASR	99.4±0.3	99.5±0.2	**7.8**±4.9	**9.0**±3.2	**3.1**±5.0	**1.8**±0.9	12.2±16.9	**5.0**±8.6	10.8±9.2	**9.9**±8.2
		ACC	90.4±1.5	90.7±1.4	**87.4**±1.2	**88.3**±1.4	**87.2**±2.8	**90.6**±1.1	89.8±0.4	**89.8**±3.2	88.7±1.2	**90.8**±1.1
MM-BM(ours) +Fine-Pruning	500	ASR	55.2±40.1	53.1±36.3	**2.4**±2.8	**2.4**±4.2	**1.2**±0.5	**1.7**±1.0	**1.2**±2.6	**1.2**±0.8	**2.3**±0.7	**2.3**±3.6
		ACC	90.1±0.7	90.2±0.7	**90.1**±0.6	**89.7**±0.9	**90.0**±2.4	**90.2**±0.8	**90.5**±0.6	**90.2**±0.6	**89.6**±0.5	**89.9**±1.2

TABLE 3: Average ASR(%) and average ACC(%) of classifiers for each of the ten backdoor attack ensembles created for CIFAR-10, after each of NC-M, FP, and MM-BM is applied. Results for which (jointly) ACC$\geq$87% and ASR$\leq$10% are in bold. (All the test examples from the source classes with the backdoor trigger incorporated are used to measure the ASR).

	NC-M	fine-pruning	NAD	MM-BM(ours)
time (s)	40.7±0.5	42.8±0.2	34.21±2.5	25.3±0.7

TABLE 4: Average execution times of MM-BD compared with other mitigation approaches on CIFAR-10.

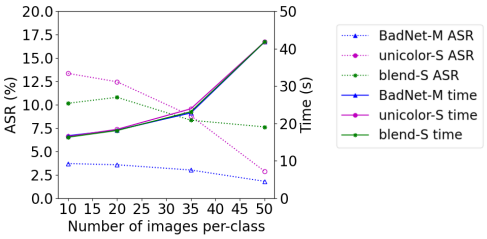


Figure 2: ASR and execution time for MM-BM versus the number of clean images per class used for mitigation, for three backdoor attack ensembles.

while suffering non-negligible degradation in ACC for some ensembles (notably for BadNet-M). In fact, the performance of NC-M largely relies on the effectiveness of the NC detector in backdoor pattern reverse-engineering and the hyper-parameter choices such as the step size for (un)learning. FP does not perform well in removing the backdoor mapping, possibly because the FP hypothesis that a subset of neurons are reserved solely for triggering the backdoor mapping does not hold. However, FP yields the highest ACC among the three mitigation approaches. The NAD method is generally effective for all types of backdoor patterns considered here (while MM-BM is not effective on chessboard-M and chessboard-S). However, the fine-tuning step of NAD (on a small clean dataset) causes large degradation in clean test accuracy. In comparison, with only 20 images per class (much fewer than for the other two methods), MM-BM reduces ASR to low levels for backdoor patterns 1-pixel-blend, with only moderate degradation in ACC. In Sec. 6.2.1, we will show that even better performance can be achieved by MM-BM for these backdoor patterns when a few more clean images are used. Unfortunately, MM-BM fails to mitigate backdoor attacks with the chessboard pattern (Notably, our detection method detects the chessboard pattern well, as shown in Table 1). However, as shown in Apdx. B.2 and Fig. 7b the internal activations caused by the chessboard pattern are just slightly larger than those induced by clean samples; thus, MM-BM, which places upper bounds on the activations of internal layers, cannot mitigate such a backdoor. But, given that MM-BM *does not* change the architecture or any trained parameters of the classifier, it can be naturally deployed together with other tuning-based mitigation methods. For example, as shown in Tab. 3, if MM-BM is deployed followed by FP, the ASR of all backdoor attacks will be largely reduced (and close to zero for backdoor patterns 1-pixel-blend), and with generally less degradation in ACC.

6.2.1. Mitigation Design Choices. In Fig. 2, for ensembles BadNet-M, unicolor-S, and blend-S, we show ASRs and execution times versus the number of clean images used by MM-BM for backdoor mitigation. For 50 images per class, we observe clear drops in ASR, especially for the unicolor-S ensemble, with some (acceptable) increments in execution time. Even with as few as 10 images per class, MM-BM achieves decently low ASRs for all three ensembles.

6.3. Detecting Advanced Backdoor Attacks

Here we consider advanced backdoor attacks including a clean-label/label-consistent backdoor attack [63], an input aware (IA) backdoor attack [75], WaNet [68], and a label-smoothed backdoor attack [71].

	clean label				IA	WaNet	label smooth		
PR (%)	0.4	1.5	6	25	10	10	0.16	0.4	1
ACC	89.34	89.20	88.34	88.68	90.73	91.02	91.41	91.14	90.97
ASR	11.27	65.13	88.26	94.13	99.42	96.45	81.50	97.03	99.93
MM-BD	1/10	6/10	8/10	10/10	8/10	10/10	1/1	1/1	1/1
MM-BM ACC	86.77	86.88	86.71	86.06	87.21	86.71	90.10	89.88	89.74
MM-BM ASR	1.68	1.65	1.64	1.82	12.30	8.74	2.32	3.50	1.13

TABLE 5: MM-BD and MM-BM performance on advanced attacks. PR represents the poisoning rate, the first and second rows show the ACC and ASR after attack, the third row shows the detection performance of MM-BD, the last two rows show the ACC and ASR after mitigation.

Unlike classical backdoor poisoning, the clean-label backdoor attack [63] embeds the backdoor pattern into *target* class samples, and *without* label flipping. To ensure the backdoor pattern, not the original class-discriminating features of the target class, is learned, [63] proposed to "destroy" those discriminating features using adversarial perturbations.

In [75], an input-aware(IA)/sample-specific backdoor pattern was proposed. The backdoor pattern for each sample is obtained using a generator given the sample as the input.

The generator and the DNN classifier are trained jointly such that: 1) the accuracy of the classifier on clean inputs is maximized; 2) the backdoor pattern generated for each sample induces the image to be misclassified to the designated target class; and 3) the backdoor pattern generated for one sample does not induce other samples to be misclassified. Based on this design, existing methods such as NC – which reverse-engineers a common backdoor pattern in the input domain that induces a group of samples to be misclassified – easily fail [75].

In [68], the backdoor pattern is generated for each sample using a 'WaNet', and is smooth and largely imperceptible to humans. Also in [68], a 'noise' training mode is proposed to help the attack evade some backdoor detectors. In particular, images with the backdoor pattern plus additive noise will not be (mis)classified to the designated target class. Thus, to detect backdoor attacks with such warping-based backdoor patterns using a reverse-engineering-based defense, the backdoor pattern used by the attacker should be precisely estimated, which is almost impossible in practice.

A label-smoothed backdoor attack was proposed in [71], with more than one backdoor pattern associated with each attack. A similar embedding function as in [2] and [68] was used, though the label flipping probability for each sample $\mathbf{x}_i \in \mathcal{X}$ is now some $p_n(\mathbf{x}_i) \in [0, 1]$ instead of one. The label flipping probability can be determined using a surrogate classifier trained on an independent dataset possessed by the attacker. The purpose here is to prevent a single backdoor pattern from causing a high ASR when it appears in a test instance. However, when multiple backdoor patterns used for poisoning are simultaneously embedded in a test instance, the test instance will be (mis)classified to the backdoor target class with high confidence, leading to a high ASR.

The evaluation of MM-BD is conducted on CIFAR-10. For IA and WaNet, following the authors' suggestion, we used the PreActResNet-18 architecture; the ResNet-18 architecture was used for the other attacks. For clean label and label smooth attacks, we create multiple groups of attacked models with different poisoning rates. As shown in Table 5, for attacks with high ASR (higher than 80%), MM-BD achieves detection accuracy higher than 8/10; also, for all attacks, MM-BM reduces ASR to a low value with moderate degradation in ACC. Again, such good performance is based on the fact that MM-BD does not perform backdoor reverse-engineering and does not make assumptions about the backdoor pattern type.

6.4. Detecting Attacked Models Trained with Differential Privacy

The "landscape" of the classifier's function will largely be altered when the model is trained with differential privacy. While differentially private training (e.g. DPSGD [92]) may reduce overfitting to the exact backdoor pattern (and the target class margin accordingly), sufficient overfitting to the key features of the backdoor pattern is still required to ensure the backdoor pattern overrides the original class-discriminating features when embedded in any input (so that the backdoor pattern will be recognized). Thus, the margin for the backdoor target class will still be large enough to be detected by MM-BD.

Specifically, following the standard implementation of DPSGD (with noise scale 0.1, and maximum gradient norm 2.0 — a larger noise scale will cause a severe drop in ASR), we trained 5 models on CIFAR-10 with the BadNet-M attack, resulting in ACC 58.06, ASR 87.61. We observe that the margins for all classes are significantly reduced compared to those for models trained without differential privacy. However, MM-BD still detects all 5 attacks.

6.5. Detection Performance on Other Domains

While we mainly focus on image classification tasks in this paper, backdoor attacks (and defenses) have been extended to other domains such as natural language processing [93], [94], speech recognition [95], video [96], and point clouds [97]. Here, we show the effectiveness of MM-BD against backdoor attacks on speeches and point clouds for example.

6.5.1. Speech Command Classification. We evaluate MM-BD on a speech domain using 10 clean and 10 backdoor ensembles of M5 [98] models trained on the Speech Commands dataset [99]. The dimension of the backdoor pattern is 40 (the dimension of the speech signal is 8000). The pattern is embedded using the BadNet [2] embedding function, with a poisoning rate of 10%. As shown in Tab. 6, our method has good detection performance on this speech domain.

6.5.2. Point Cloud Classification. On the point cloud domain, we use the attack proposed in [97] to evaluate our detection method. The backdoor pattern is a small set of inserted points (e.g.) to mimic real objects, such as a ball carried by a pedestrian. 10 good and 10 attacked DGCNN [100] models were trained. Tab. 6 shows that the detection performance for this domain is not as good as for speech. As revealed in [97], *intrinsic backdoors* exist in models trained on point cloud datasets, which means that, for a clean model, there will be a universal pattern that can induce misclassifications when embedded into test examples. This may explain why the detection results are not as strong as for other domains.

Gradient-based MM-BD may not perform well on categorical datasets depending on how their features are mapped to continuous numerical ones (e.g., natural language). In these cases, alternative search strategies could be used by MM-BD, e.g., based on genetic algorithms.

6.6. Robustness Against Adaptive Attacks

6.6.1. Adaptive Attack via Minimization of the Maximum Margin Statistic. Here, we consider a strong adaptive attack where the attacker has full knowledge of MM-BD. Moreover, we allow the attacker to have full control of the training process in addition to its standard poisoning capability. The purpose is to evaluate MM-BD more thoroughly,

	speech		point cloud	
	clean	attacked	clean	attacked
ACC	81.4±0.5	80.3±1.0	91.0±0.3	87.5±0.5
ASR	0	99.4±0.2	0	99.0±0.8
MM-BD	10/10	10/10	8/10	7/10

TABLE 6: Detection results for speech and point cloud.

under extreme conditions, to show its superior power of post-training backdoor detection.

Note that a successful detection will be made by MM-BD if the maximum margin statistic associated with the backdoor target class is abnormally large. This abnormality can be easily observed for most backdoor settings and for a large range of backdoor pattern types as shown by our main experiments. Thus, to defeat MM-BD, the strong, adaptive attacker can fine-tune the classifier's parameters on the same poisoned training set (to keep both a high ASR and a high ACC) while *minimizing* the maximum margin for the backdoor target class. To do so, the attacker minimizes the following training loss:

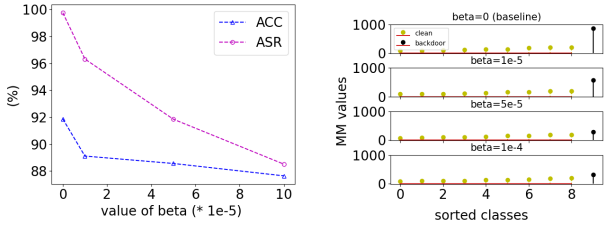
$$\begin{aligned}\min_{\phi} \quad & \beta_{\mathrm{T}} \times \frac{1}{|\mathcal{D}_{\mathrm{T}}|} \sum_{(\mathbf{x},y)\in\mathcal{D}_{\mathrm{T}}} \mathcal{L}_{\mathrm{E}}(f(\mathbf{x};\phi),y) \\ & + \beta_{\mathrm{B}} \times \frac{1}{|\mathcal{D}_{\mathrm{B}}|} \sum_{(\tilde{\mathbf{x}},t)\in\mathcal{D}_{\mathrm{B}}} \mathcal{L}_{\mathrm{E}}(f(\tilde{\mathbf{x}},\phi),t) \\ & + \beta_{\mathrm{M}} \times \mathcal{L}_{\mathrm{M}}(t;\phi),\end{aligned} \tag{12}$$

where

$$\mathcal{L}_{\mathrm{M}}(t;\phi) = \max_{\mathbf{x}\in\mathcal{X}}[g_t(\mathbf{x};\phi) - \max_{k\in\mathcal{Y}\setminus t} g_k(\mathbf{x};\phi)] \tag{13}$$

is the maximum margin for class $t \in \mathcal{Y}$. Slightly different from the notations used previously, here, for both the classifier $f : \mathcal{X} \to \mathcal{Y}$ and the logit function $g_c : \mathcal{X} \to \mathbb{R}$ for class $c \in \mathcal{Y}$, we explicitly include the model parameters denoted by ϕ. $\mathcal{D}_T$ is the set of clean training samples, with multiplier β_{T}. $\mathcal{D}_B$ is the set of backdoor poisoned training samples, with multiplier β_{B}. β_M is the multiplier on the maximum margin term. $\mathcal{L}_{\mathrm{E}}$ denotes the cross-entropy loss. $\tilde{\mathbf{x}}$ and $t \in \mathcal{Y}$ are the backdoor-poisoned sample and the backdoor target class, respectively.

To solve this min-max problem, the attacker can follow the classical strategy for adversarial training [85] by maximizing Eq. (13) with the model parameters ϕ fixed, and then minimizing Eq. (12) to update the model. Note that the first two terms in Eq. (12) are associated with traditional training on the backdoor poisoned training set to launch an attack. In our experiments, we consider $\beta_{\mathrm{M}} \in \{10^{-5}, 2 \times 10^{-5}, 10^{-4}\}$. As shown in Fig. 3a, as β_{M} increases, the abnormality of the maximum margin statistic associated with the backdoor target class decreases, such that attacks with $\beta_{\mathrm{M}} = 10^{-4}$ are not detectable by MM-BD. However, there is a degradation in ASR of the classifier as β_{M} increases, with also some drop in the ACC. There will be moderate drop in ASR and ACC for an adaptive attacker to bypass our detection method. Moreover, the conventional backdoor simply requires modifying a subset of the training data, while the adaptive attack requires the attacker to have



(a) ASR and ACC for attacks with different β_{M}.

(b) maximum margin statistics for attacks with different β_{M}.

Figure 3: Results for adaptive attack.

full control of the training process. Also to defeat MM-BD, an attacker needs to solve the min-max optimization, which is time-consuming. Conventional training (60 epochs) takes 35 minutes while fine-tuning an attacked model based on the objective (12) (for 1 epoch, which can make the attacked model undetectable) takes 62 minutes. Finally, all these adaptive attacks are successfully mitigated by MM-BM, with average ACC of 85.8% and average ASR of 2.3% post-mitigation.

6.6.2. A Stronger Adaptive Attack. The adaptive attack proposed above can be further enhanced by also maximizing the maximum margin of other classes. $\mathcal{L}_{\mathrm{M}}(t;\phi)$ in the last term of Eq. 12 now becomes:

$$\begin{aligned}\mathcal{L}_{\mathrm{M}}(t;\phi) = & \max_{\mathbf{x}\in\mathcal{X}}[g_t(\mathbf{x};\phi) - \max_{k\in\mathcal{Y}\setminus t} g_k(\mathbf{x};\phi)] \\ & - \sum_{s\in\mathcal{Y}\setminus t} \{\max_{\mathbf{x}\in\mathcal{X}}[g_s(\mathbf{x};\phi) - \max_{k\in\mathcal{Y}\setminus s} g_k(\mathbf{x};\phi)]\}.\end{aligned} \tag{14}$$

For $\beta_{\mathrm{M}} = 10^{-5}$, the learned model has ACC 88.75% and ASR 96.04% (with a similar drop in ACC but less drop in ASR than for the basic adaptive attack). This model is not detectable by MM-BD, with p-value 0.34.

While the adaptive term defined in Eq. 14 leads to a stronger adaptive attack, it also makes the attack $|\mathcal{Y}|$ times more computationally expensive than the basic adaptive attack defined by Eq. 13 and Eq. 12 (since the attacker needs to perform maximum margin optimization for all the classes). It takes 642 minutes to fine-tune one epoch.

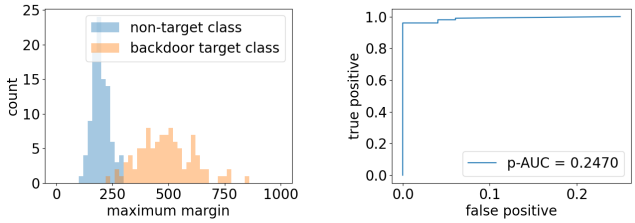
7. Limitations and discussion

7.1. All-to-All Attacks

For all-to-all attacks where every class is a backdoor target class with a specific[9] source class [2], all the detection statistics are induced by the backdoor attack. So, there will be no statistics to provide any information about the null distribution for non-backdoor classes; unsupervised anomaly detection is clearly an ill-posed problem for this scenario.

Despite this, our maximum margin (MM) statistic can still be used for detection with a calibrated threshold leveraging some domain knowledge to set a detection threshold, *e.g.* as in [19], [21], [25], [58]. Here, we created ten all-to-all attacks on CIFAR-10. For each attack, every class is

9. The very first all-to-all attack assumes that the backdoor target class for source class i is class $(i+1) \bmod K$, where $K = |\mathcal{Y}|$ is the total number of classes. The scenario considered here is more general.



(a) (b)

Figure 4: Analysis of maximum margin statistics for all-to-all attacks. (a) Histogram of maximum margin statistics for backdoor target classes and for non-target classes. (b) The ROC curve and the partial area under the curve (p-AUC) when the false positive rate is less than 0.25.

a backdoor target class with a unique source class that is randomly assigned. In other words, each of the ten classes will be the source class for one backdoor class pair, and the target class for another. For each backdoor class pair, we randomly generated a backdoor pattern from type BadNet. In Fig. 4a, we show the distribution of the maximum margin statistics obtained for each target class for each attack, compared with the maximum margin statistics obtained from ten clean classifiers. In Fig. 4b, we show the receiver operating characteristic (ROC) curve when the false positive rate is less than 0.25 in Fig. 4a. The area under the curve (AUC) is 0.2470, showing that our MM statistic is very effective at distinguishing backdoor target classes from non-target classes regardless of the attack setting.

Moreover, we applied MM-BM to mitigate the ten attacks we created. Notably, MM-BM does not depend on the number of backdoor target classes by its design. It achieves 89.3% average ACC and 1.7% average ASR after mitigation – all the all-to-all attacks are successfully mitigated.

7.2. Potential False Detection: Case Studies

MM-BD possesses neither clean models nor clean samples for adjusting the detection threshold. We use a detection threshold $\theta = 0.05$ for all the datasets, which *theoretically* induces 5% false positives. The same significance level is also adopted by NC, TABOR, and PT-RED. However, the theoretical false positive rates may not always be accurate due to inaccuracy of the estimated "null" model used for hypothesis testing. Especially, on CIFAR-10, there are only 9 MM statistics available for estimating the "null" model, with some false detections observed in the experiment shown in Table 1. Notably, our method is generally better than the baseline methods in false positive control. Moreover, to achieve a lower false positive rate (FPR), one can always adopt a more conservative significance level to achieve a desired FPR, with only a moderate drop in the true positive rate (TPR). Tab. 7 gives the FPR and TPR over all models trained on CIFAR-10 under different significance levels.

On other datasets like CIFAR-100 and TinyImageNet, with more classes, more MM statistics are available for estimating the "null" model and false detections tend to be lower. However, GTSRB is an exception. Some false detections are still observed even though the number of classes in GTSRB is larger than CIFAR-10 – this could be attributable to class imbalance in GTSRB, which is discussed below.

θ	0.05	0.02	0.01	0.005
FPR (%)	11.3	11.3	6.7	0
TPR (%)	91.3	88.0	85.3	82.3

TABLE 7: TPR and FPR under different significance levels.

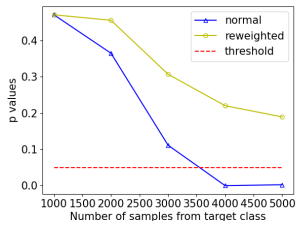


Figure 5: Detection p-values of normally trained models (denoted as "normal" in the figure) and models trained with re-weighted data sampler (denoted as "reweighted") under different numbers of examples from the target class. The number of samples from classes other than the target class is fixed as 1000.

7.2.1. Clean model trained on imbalanced data. In MM-BD, the abnormally large maximum margin is used as evidence of a backdoor attack. Notably, clean models trained on imbalanced data can also be overconfident on a particular class's decision, resulting in false detections by MM-BD. However, in practice, data re-weighted sampling is commonly applied to "balance out" imbalanced data. Here we evaluate the influence of class imbalance on MM-BD. We use a subset of CIFAR-10. First a "target" class is chosen and we make the number of samples from the "target" class larger than that from other classes. In all the experiments we set class "9: truck" as the "target" class, with the number of samples from all other classes fixed at 1000. We trained the model two ways – both with or without data re-weighted sampling. The detection p-values with different number of examples from the target class are given in Fig. 5, showing that MM-BD is robust against a moderate level of class imbalance (which is also verified in the experiment on GTSRB shown in Tab. 1.), but makes false detections when the class imbalance becomes severe.

Notes: Backdoor poisoning may also cause class imbalance. However, the abnormally larger MM statistic for the backdoor target class is due to the backdoor attack, not to class imbalance. To see this, we conduct an experiment where the poisoned training set is class-balanced (4000 samples per class). Our MM-BD achieves a p-value of 2.7×10^{-7} against this attack, much smaller than the detection threshold.

7.2.2. Regularized Clean Model with Abnormally Large Maximum Margin. Other than class imbalance, a "regularized" clean model also can have an abnormally large maximum margin. To evaluate the influence of such regularization, we added a term to the (clean) training loss, such that the resulting model will have an abnormally large

maximum margin for a certain class. The training objective is as follows:

$$\min_{\phi} \quad \frac{1}{|\mathcal{D}_{\mathrm{T}}|} \sum_{(\mathbf{x},y)\in\mathcal{D}_{\mathrm{T}}} \mathcal{L}_{\mathrm{E}}(f(\mathbf{x};\phi),y) - \beta \times \max_{\mathbf{x}\in\mathcal{X}} [g_t(\mathbf{x};\phi) - \max_{k\in\mathcal{Y}\setminus t} g_k(\mathbf{x};\phi)], \tag{15}$$

where the first term is the cross entropy loss, and the second term is the maximum margin of a predefined *target class* t. Class "9" was chosen to be the target class and β was set to 10^{-5}. The trained model has test accuracy of 91.42% (similar to the clean unregularized models trained solely on the cross entropy loss), and we did observe an abnormally large maximum margin for the target class. The model is falsely detected, with a p-value of 1.2×10^{-11}. However, in practice, an honest training authority would not use this "regularized" objective function for training, unless this training authority is an insider seeking to bypass MM-BD.

8. Conclusions

In this paper, we revealed a maximum margin statistic that substantially discriminates between clean and backdoor-attacked DNNs and, based upon which, we proposed a post-training backdoor detection method that makes no assumptions about the backdoor pattern type or method of incorporation. MM-BD is based on a novel maximum margin statistic that can be estimated without using any legitimate (clean) samples. It accurately and efficiently detects backdoor attacks with arbitrary numbers of source classes, as shown by our substantial experiments. Additionally, we proposed a backdoor mitigation approach which effectively removes most backdoor mappings. This method does leverage a small number of legitimate samples.

Ethics Statement

The purpose of this research is to understand the behavior of deep learning systems facing malicious activities, and enhance their safety level. The backdoor attack considered in this paper is well-known, with open-sourced implementation code. Thus, publication of this paper will be beneficial to the community in defending against backdoor attacks.

Acknowledgement

This work was supported by NSF SBIR grant 2132294.

References

[1] C. Szegedy, W. Zaremba, I Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[2] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.

[3] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, and J Zhai, "Trojaning attack on neural networks," in *NDSS*, San Diego, CA, 2018.

[4] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia, "Backdoor learning: A survey," *TNNLS*, 2022.

[5] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," https://arxiv.org/abs/1712.05526v1, 2017.

[6] B. Nelson, B. Barreno, and al., "Misleading learners: Co-opting your spam filter," in *Machine Learning in Cyber Trust: Security, Privacy, and Reliability*, 2009.

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

[8] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," http://arxiv.org/abs/1811.03728, Nov 2018.

[9] Shawn Shan, Arjun Nitin Bhagoji, Haitao Zheng, and Ben Y Zhao, "Poison Forensics: Traceback of Data Poisoning Attacks in Neural Networks," in *Proc. of USENIX Security*, 2022.

[10] M. Du, R. Jia, and D. Song, "Robust anomaly detection and backdoor attack detection via differential privacy," in *ICLR*, 2020.

[11] K. Huang, Y. Li, B. Wu, Z. Qin, and K. Ren, "Backdoor defense via decoupling the training process," in *ICLR*, 2022.

[12] Y. Shen and S. Sanghavi, "Learning with bad training data via iterative trimmed loss minimization," in *ICML*, 2019.

[13] S. Hong, V. Chandrasekaran, Y. Kaya, T. Dumitraş, and N. Papernot, "On the effectiveness of mitigating data poisoning attacks with gradient shaping," https://arxiv.org/abs/2002.11497, 2020.

[14] J. Geiping, L. Fowl, G. Somepalli, M. Goldblum, M. Moeller, and T. Goldstein, "What doesn't kill you makes you robust(er): Adversarial training against poisons and backdoors," *arXiv preprint arXiv:2102.13624*, 2021.

[15] "Model Zoo," https://modelzoo.co/.

[16] 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 *S&P*, 2019.

[17] W. Guo, L. Wang, X. Xing, M. Du, and D. Song, "TABOR: A highly accurate approach to inspecting and restoring Trojan backdoors in AI systems," https://arxiv.org/abs/1908.01763, 2019.

[18] Z. Xiang, D. J. Miller, and G. Kesidis, "Detection of backdoors in trained classifiers without access to the training set," *TNNLS*, pp. 1–15, 2020.

[19] Y. Liu, W. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for back-doors by artificial brain stimulation," in *ACM CCS*, 2019, p. 1265–1282.

[20] Z. Xiang, D. Miller, and G. Kesidis, "Post-training detection of backdoor attacks for two-class and multi-attack scenarios," in *ICLR*, 2022.

[21] G. Shen, Y. Liu, G. Tao, S. An, Q. Xu, S. Cheng, S. Ma, and X. Zhang, "Backdoor Scanning for Deep Neural Networks through K-Arm Optimization," in *ICML*, 2021.

[22] R. Wang, G. Zhang, S. Liu, P.-Y. Chen, J. Xiong, and M. Wang, "Practical detection of trojan neural networks: Data-limited and data-free cases," in *ECCV*, 2020.

[23] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, "Universal litmus patterns: Revealing backdoor attacks in cnns," in *CVPR*, 2020, pp. 298–307.

[24] X. Xu, Q. Wang, H. Li, N. Borisov, C.A. Gunter, and B. Li, "Detecting AI Trojans using meta neural analysis," in *S&P*, 2021.

[25] "IARPA TrojAI: Trojans in artificial intelligence," https://www.iarpa.gov/index.php/research-programs/trojai/trojai-baa, 2019.

[26] B. Branco, P. Abreu, A.S. Gomes, M.S.C. Almeida, J.T. Ascensao, and P. Bizarro, "Interleaved Sequence RNNs for Fraud Detection," in *Proc. ACM KDD*, Aug. 2020.

[27] B. Xu, H. Shen, B. Sun, R. An, Q. Cao, and X. Cheng, "Towards Consumer Loan Fraud Detection: Graph Neural Networks with Role-Constrained Conditional Random Field," in *AAAI*, 2021.

[28] C. Morrison, "AI Developers Tout Revolution, Drugmakers Talk Evolution," *Nature Biotechnology*, Nov. 2019.

[29] A. Park and al., "Deep Learning–Assisted Diagnosis of Cerebral Aneurysms Using the HeadXNet Model," *JAMA Network Open*, vol. 2, no. 6, 2019.

[30] H. Xu, Y. Ma, H.-C. Liu, D. Deb, H. Liu, J.-L. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," *International Journal of Automation and Computing*, vol. 17, pp. 151–178, 2020.

[31] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

[32] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "SoK: Security and Privacy in Machine Learning," in *EuroS&P*, 2018, pp. 399–414.

[33] L. Li, T. Xie, and B. Li, "SoK: Certified Robustness for Deep Neural Networks," in *S&P*, San Francisco, CA, 2023.

[34] D. J. Miller, Z. Xiang, and G. Kesidis, "Adversarial learning in statistical classification: A comprehensive review of defenses against attacks," *Proceedings of the IEEE*, vol. 108, pp. 402–433, 2020.

[35] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[36] S.-M. M.-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: a simple and accurate method to fool deep neural networks," in *CVPR*, 2016.

[37] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *S&P*, May 2017, pp. 39–57.

[38] F. Tramer, F. Zhang, A. Juels, M. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," in *USENIX*, 2016.

[39] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. Celik, and A. Swami, "Practical black box attacks against machine learning," in *Asia CCS*, 2017.

[40] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *CVPR*, 2019.

[41] V. Chandrasekaran, K. Chaudhuri, I. Giacomelli, S. Jha, and S. Yan, "Exploring connections between active learning and model extraction," in *USENIX Security*, 2020, pp. 1309–1326.

[42] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *ICML*, 2012, p. 1467–1474.

[43] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *ACML*, 2011, pp. 97–112.

[44] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," *AISec*, 2017.

[45] C. Yang, Q. Wu, H. H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," *ArXiv*, vol. abs/1703.01340, 2017.

[46] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018.

[47] H. Zhong, C. Liao, A. Squicciarini, S. Zhu, and D.J. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," in *CODASPY*, 2020.

[48] J. Bai, K. Gao, D. Gong, S. Xia, Z. Li, and W. Liu, "Hardly perceptible trojan attack against neural networks with bit flips," in *ECCV*, 2022.

[49] J. Bai, B. Wu, Y. Zhang, Y. Li, Z. Li, and S. Xia, "Targeted attack against deep neural networks via flipping limited weight bits," in *ICLR*, 2021.

[50] X. Qi, T. Xie, R. Pan, J. Zhu, Y. Yang, and K. Bu, "Towards practical deployment-stage backdoor attack on deep neural networks," in *CVPR*, 2022.

[51] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *ACSAC*, 2019.

[52] E. Sarkar, Y. Alkindi, and M. Maniatakos, "Backdoor suppression in neural networks using input fuzzing and majority voting," *IEEE Design & Test*, vol. 37, no. 2, pp. 103–110, 2020.

[53] B. G. Doan, E. Abbasnejad, and D. C.Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *ACSAC*, 2020, p. 897–912.

[54] E. Chou, F. Tramèr, G. Pellegrino, and D. Boneh, "Sentinet: Detecting localized universal attacks against deep learning systems," in *2020 IEEE Security and Privacy Workshops*. IEEE, 2020.

[55] W. Ma, D. Wang, R. Sun, M. Xue, S. Wen, and Y. Xiang, "The" beatrix"resurrections: Robust backdoor detection via gram matrices," *arXiv preprint arXiv:2209.11715*, 2022.

[56] J. Guo, Y. Li, X. Chen, H. Guo, L. Sun, and C. Liu, "SCALE-UP: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency," in *ICLR*, 2023.

[57] X. Li, Z. Xiang, D. J. Miller, and G. Kesidis, "Test-time detection of backdoor triggers for poisoned deep neural networks," in *ICASSP*, 2022.

[58] NeurIPS, "Trojan Detection Challenge NeurIPS 2022," https://trojandetection.ai/, 2022.

[59] ICLR, "IEEE Trojan Removal Competition," https://www.trojan-removal.com/, 2022.

[60] Y. Zeng, S. Chen, W. Park, Z. Mao, M. Jin, and R. Jia, "Adversarial unlearning of backdoors via implicit hypergradient," in *ICLR*, 2022.

[61] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks," in *ICLR*, 2021.

[62] K. Liu, B. Doan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdoor attacks on deep neural networks," in *RAID*, 2018.

[63] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," https://people.csail.mit.edu/madry/lab/cleanlabel.pdf, 2019.

[64] H. Pirsiavash A. Saha, A. Subramanya, "Hidden trigger backdoor attacks," in *AAAI*, 2020.

[65] Z. Zhao, X. Chen, Y. Xuan, Y. Dong, D. Wang, and K. Liang, "Defeat: Deep hidden feature backdoor attacks by imperceptible perturbation and latent representation constraints," in *CVPR*, 2022.

[66] Z. Wang, J. Zhai, and S. Ma, "Bppattack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning," in *CVPR*, 2022.

[67] T. Wang, Y. Yao, F. Xu, S. An, H. Tong, and T. Wang, "Backdoor attack through frequency domain," https://arxiv.org/abs/2111.10991, 2021.

[68] A. Nguyen and A. Tran, "WaNet - Imperceptible Warping-based Backdoor Attack," in *ICLR*, 2021.

[69] Z. Xiang, D. J. Miller, H. Wang, and G. Kesidis, "Detecting scene-plausible perceptible backdoors in trained DNNs without access to the training set," *Neural Computation*, pp. 1329–1371, 2021.

[70] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection Backdoor: A Natural Backdoor Attack on Deep Neural Networks," in *ECCV*, 2020.

[71] M. Peng, Z. Xiong, M. Sun, and P. Li, "Label-Smoothed Backdoor Attack," *arXiv preprint arXiv:2202.11203*, 2022.

[72] K. Doan, Y. Lao, and P. Li, "Backdoor attack with imperceptible input and latent modification," in *NeurIPS*, 2021.

[73] Z. Xiang, D.J. Miller, and G. Kesidis, "A benchmark study of backdoor data poisoning defenses for deep neural network classifiers and a novel defense," in *IEEE MLSP*, Pittsburgh, 2019.

[74] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *ICCV*, 2021.

[75] A. Nguyen and A. Tran, "Input-aware dynamic backdoor attack," in *NeurIPS*, 2020.

[76] Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *TKDE*, vol. 33, no. 4, pp. 1328–1347, 2021.

[77] G. Shafer and Vlad. Vovk, "A tutorial on conformal prediction," *J. Mach. Learn. Res.*, vol. 9, pp. 371–421, June 2008.

[78] Z. Xiang, Z. Xiong, and B. Li, "UMD: Unsupervised model detection for x2x backdoor attacks," in *ICML*, 2023.

[79] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks," in *IJCAI*, 7 2019, pp. 4658–4664.

[80] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *ACM CCS*, 2015, p. 1322–1333.

[81] R. Zheng, R. Tang, J. Li, and L. Liu, "Data-free backdoor removal based on channel lipschitzness," in *ECCV*, 2022, pp. 175–191.

[82] L. Zhu, R. Ning, C. Xin, C. Wang, and H. Wu, "CLEAR: Clean-up Sample-Targeted Backdoor in Neural Networks," in *ICCV*, 2021.

[83] Z. Xiang, D. J. Miller, S. Chen, X. Li, and G. Kesidis, "Detecting backdoor attacks against point cloud classifiers," in *ICASSP*, 2022.

[84] W. Chen, B. Wu, and H. Wang, "Effective backdoor defense by exploiting sensitivity of poisoned samples," *NeurIPS*, vol. 35, pp. 9727–9737, 2022.

[85] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[86] S. Bubeck, "Convex optimization: Algorithms and complexity," http://arxiv.org/abs/1405.4980, 2015.

[87] A. Krizhevsky, "Learning multiple layers of features from tiny images," *University of Toronto Technical Report*, 05 2012.

[88] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012.

[89] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[90] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[91] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," in *CVPR*, 2018.

[92] M. Abadi, A. Chu, I. Goodfellow, H. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM CCS*, 2016.

[93] S. Li, H. Liu, T. Dong, B. Z. Zhao, M. Xue, H. Zhu, and J. Lu, "Hidden backdoors in human-centric language models," in *ACM CCS*, 2021, pp. 3123–3140.

[94] X. Chen, A. Salem, D. Chen, M. Backes, S. Ma, Q. Shen, Z. Wu, and Y. Zhang, "Badnl: Backdoor attacks against nlp models with semantic-preserving improvements," in *ACSAC*, 2021, pp. 554–569.

[95] H. Abdullah, K. Warren, V. Bindschaedler, N. Papernot, and P. Traynor, "Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems," in *S&P*, 2021, pp. 730–747.

[96] Y. Li, H. Zhong, X. Ma, Y. Jiang, and S.-T. Xia, "Few-shot backdoor attacks on visual object tracking," in *ICLR*, 2022.

[97] Z. Xiang, D. J. Miller, S. Chen, X. Li, and G. Kesidis, "A backdoor attack against 3D point cloud classifiers," in *ICCV*, 2021.

[98] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, "Very deep convolutional neural networks for raw waveforms," in *ICASSP*, 2017, pp. 421–425.

[99] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[100] Y. Wang, Y. Sun, Z. Liu, S. Sarma, M. Bronstein, and J. Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[101] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

Algorithm 1: Minimizing Eq. (11) for backdoor mitigation.

Inputs: dataset $\mathcal{D}$, "unbounded" logit functions $\{g_c(\cdot)\}_{c\in\mathcal{Y}}$, accuracy constraint π, step size δ, maximum iteration count $\tau_{\max}$, scaling factor $\alpha > 1$.
Initialization: $\mathbf{Z}^{(0)}$ set large (e.g. 100), $\lambda^{(0)}$ set to a small positive number (e.g. 10^{-1}), $\mathbf{Z}^* = \boldsymbol{\infty}$.
for $\tau = 0 : \tau_{\max} - 1$ **do**
  $\mathbf{Z}^{(\tau+1)} = \mathbf{Z}^{(\tau)} - \delta\nabla_{\mathbf{Z}} L(\mathbf{Z}^{(\tau)}, \lambda^{(\tau)}; \mathcal{D})$
  if $\frac{1}{|\mathcal{D}|}\sum_{(\mathbf{x},y)\in\mathcal{D}} \mathbb{1}[y = \arg\max_{c\in\mathcal{Y}} \bar{g}_c(\mathbf{x};\mathbf{Z})] \geq \pi$ **then**
    $\lambda^{(\tau+1)} = \lambda^{(\tau)} \cdot \alpha$
    if $\sum_{l=2}^{L}(||\mathbf{z}_l^{(0)}||_2 - ||\mathbf{z}_l^*)||_2 < 0$ **then**
      $\mathbf{Z}^* = \mathbf{Z}^{(\tau+1)}$
  else
    $\lambda^{(\tau+1)} = \lambda^{(\tau)}/\alpha$
Outputs: $\mathbf{Z}^*$

Appendix A. Experiment Details

A.1. Details of Datasets

CIFAR-10 [87] contains 60000 32×32 color images from 10 classes. For each class, 5000 images are for training and 1000 images are for testing. Similar to CIFAR-10, CIFAR-100 contains 60000 32×32 color images, with each class containing 500 training images and 100 test images [87]. GTSRB [88] is a dataset containing color images of German traffic signs from 43 classes. There are 39209 training images and 12630 test images. TinyImageNet is a subset of ImageNet [101], but with each image resized to 64×64 (for the official version); there are 200 classes in the dataset, each containing 500 training images and 25 test images.

Figure 6: Example BPs used in our experiments (top) and images with these BPs embedded (bottom).

A.2. Details of Backdoor Patterns

chessboard and 1-pixel are two additive backdoor patterns. chessboard is a 'chessboard' pattern considered in [18], with one and only one of two adjacent pixels perturbed positively by 3/255 in all color channels. 1-pixel is the single-pixel additive backdoor pattern used by [7], [8] – one pixel, randomly selected and fixed for all images for each attack, is perturbed positively by 75/255 in all color channels. BadNet and unicolor are both patch replacement backdoor patterns embedded by $\tilde{\mathbf{x}} = (1 - \mathbf{m}) \odot \mathbf{x} + \mathbf{m} \odot \mathbf{u}$. Here, $\mathbf{u}$ is a random noise patch for BadNet and a monochromatic patch for unicolor. For BadNet, the patch size is 5×5 for

the TinyImageNet dataset and 3×3 for other datasets. For unicolor, we use a 3×3 patch size for all datasets (though we did not apply unicolor to attack the TinyImageNet dataset due to the training cost). blend is a blended backdoor pattern used in [24] and [5]. The blended mask is chosen to be a 3×3 square patch with a randomly selected and fixed location for each image in each attack. The blending rate is set to $\alpha = 0.2$ in our experiments. Examples of the BPs and attacked images of chessboard-blend are shown in Fig. 6.

A.3. Details of Backdoor Attack configurations

The number of backdoor training images used for poisoning was carefully chosen for each backdoor pattern and for each dataset to ensure a high attack success rate for the created backdoor attacks. Details are shown in Table 9.

A.4. Training Configurations

For each dataset, we use the same training configuration to train both clean and attack models. These details are shown in Table 8.

	CIFAR-10	CIFAR100	TinyImageNet	GTSRB
model	ResNet-18	VGG-16	ResNet-34	MobileNet
optimizer	Adam	Adam	Adam	Adam
batch size	128	128	64	128
epochs	60	100	90	50
learning rate	1e-3	1e-4	1e-3	1e-3

TABLE 8: Training configurations for the four datasets considered in our experiments.

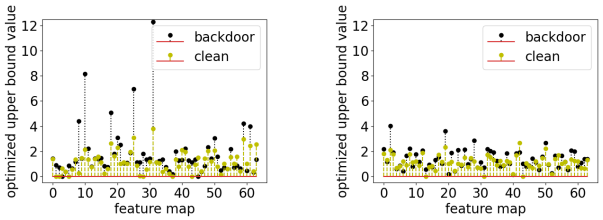
Appendix B. Additional Results

B.1. Complete Detection Results for MM-BD for all Backdoor Attack Ensembles

Here, we show the complete detection results for MM-BD on all backdoor attack ensembles for CIFAR-10 and GTSRB in Tab. 11. MM-BD achieves generally high detection accuracy for most backdoor attack ensembles.

B.2. Details for MM-BM and Discussion

In our experiments, we arbitrarily selected three layers close to the input to apply upper bounds on activations. For the experiments on CIFAR-10, for example, these three layers are the 1st, 5th, and 9th convolutional layers. In Fig. 7a, we show a stem plot for the bounding vector optimized by MM-BM obtained for the first convolutional layer (marked as "clean") for an attack with the local backdoor pattern BadNet. In comparison, also in Fig. 7a, we show a stem plot of the optimized bounding vector obtained from the same layer, but with the optimization problem Eq. (10) solved using samples embedded with the true backdoor pattern and labeled to the backdoor target class (marked as "backdoor"). In this case, the optimized upper bounds represent the activation value required for a backdoor pattern to induce (mis)classification to the backdoor target class. Clearly from the figures, activations of a subset of neurons associated with the backdoor pattern are depressed by applying MM-BM



(a) BadNet-M (b) chessboard-M

Figure 7: Stem plots for the optimized upper bounds using MM-BM with clean samples and backdoor poisoned samples for BadNet-M and chessboard-M respectively.

with clean samples, which prevents the backdoor pattern from triggering misclassifications during testing. Thus, the attack is successfully mitigated.

Unfortunately, MM-BM does not effectively mitigate the chessboard-S and chessboard-M attacks. This is likely due to the per-pixel perturbation size of the global backdoor pattern (i.e. chessboard) being very tiny (merely 3/255). Such a tiny perturbation size will induce less significant changes to neuron activations compared with local patterns with large perturbation size (or patch replacement patterns that induces large modifications in pixel values). Thus, to ensure a high attack success rate, backdoor patterns like chessboard will be learned by the classifier to induce changes to the activations of a relatively large subset of neurons, where each neuron activation is only changed moderately. This hypothesis is well-supported by the stem plots in Fig. 7b, where we repeated the process from Fig. 7a for an attack with backdoor pattern chessboard. The differences in the two stem plots in Fig. 7b are much smaller than for Fig. 7a. Thus, when MM-BM is applied to this attack, the backdoor pattern will still be partially activated, leading to a poor mitigation result.

B.3. Mitigation Performance of on other Datasets

In Tab. 10, we report the mitigation results of MM-BM for the other three datasets. Here, for CIFAR-100 and GTSRB, we apply activation upper bounds to the first three convolutional layers, since this will be more efficient than applying activation upper bounds to all layers, as discussed in Apdx. B.2. Again, MM-BM is effective at mitigating backdoor attacks for most backdoor patterns.

Appendix C. The hyper-parameters in the MM-BM method

In our mitigation method (described in detail in algorithm 1, there are two hyper-parameters–λ and α. λ is the weight on the L_2 norm of the upper bounds – a larger λ leads to lower upper bounds but worse accuracy following mitigation. α is a "step size" helping to adjust λ (α needs to be larger than 1), which is the most important hyper-parameter in our MM-BM method.

Here we did an experiment to evaluate the influence of α on our mitigation method. Fig. 8 shows the ACC, ASR after mitigation and the number of epochs required to reach the stopping condition under different α. The results indicate that a larger α makes the mitigation converge more quickly,

	CIFAR-10										
	clean	chessboard-S	chessboard-M	1-pixel-S	1-pixel-M	BadNet-S	BadNet-M	unicolor-S	unicolor-M	blend-S	blend-M
poison #	0	2000	2000	1000	1000	500	500	500	500	1000	1000
ASR	0	99.94	99.94	91.09	91.28	99.41	99.92	97.78	98.44	96.36	96.86
ACC	91.64	91.31	91.06	91.80	91.12	91.63	91.59	91.31	91.36	91.35	91.48

	CIFAR-100				TinyImageNet		GTSRB				
	clean	1-pixel-M	BadNet-M	unicolor-M	clean	BadNet-M	clean	chessboard-M	1-pixel-M	unicolor-M	blend-M
poison #	0	990	990	990	990	0	500	1680	840	840	840
ASR	0	95.67	99.84	97.37	0	97.84	0	99.80	99.20	95.80	100
ACC	67.51	65.40	66.17	66.51	60.71	58.59	94.78	94.54	94.12	94.56	94.53

TABLE 9: Average ASR and ACC over each ensemble for the four datasets.

	CIFAR-100			TinyImageNet	GTSRB			
	1-pixel-M	BadNet-M	unicolor-M	BadNet-M	chessboard-M	1-pixel-M	unicolor-M	blend-M
ASR	95.7→13.5	99.8→**1.5**	97.4→16.9	97.8→**1.3**	99.8→78.2	99.2→26.8	95.8→11.3	100→**1.5**
ACC	65.4→61.7	66.2→**65.0**	66.5→64.9	58.59→**58.0**	94.5→94.0	94.1→95.0	94.6→95.2	94.5→**95.4**

TABLE 10: MM-BM results on the CIFAR-100, GRSTB, and TinyImageNet ensembles. Joint results with ACC drop $\leq$5% and ASR$\leq$10% are shown in bold.

	clean	chessboard-M	1-pixel-M	BadNet-M	unicolor-M	blend-M
CIFAR-100	10/10	6/10	10/10	10/10	10/10	10/10
GTSRB	17/20	7/10	10/10	10/10	9/10	10/10

TABLE 11: Complete results of MM-BD on all backdoor attack ensembles created for CIFAR-100 and GTSRB.

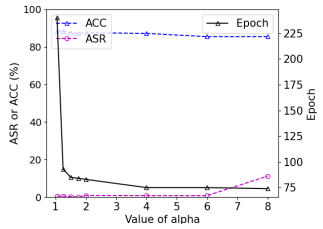
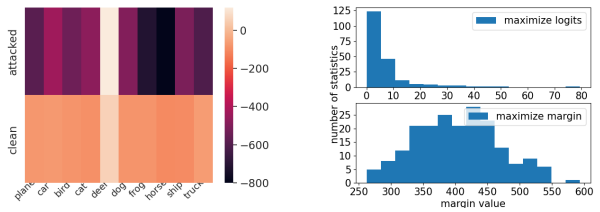


Figure 8: MM-BD mitigation performance and the number of epochs before the stopping condition is reached, under different α. (The attacked model's ACC is 91.25% and ASR is 99.98% before mitigation.)



(a) (b)

Figure 9: (a): Logits of a classifier from 1-pixel-S with target class 'deer' (top) and a clean classifier (bottom), when the logit of 'deer' class is maximized. (b): Histogram of margin statistics of a clean classifier trained on TinyImageNet using different objective functions.

but a too large α might lead to a drop in performance. The results also show that a wide range of α values can lead to good mitigation results and high efficiency. Moreover, our mitigation method is not time-consuming, so in practice, when a validation set is not available, it is safe to use a small value of α to guarantee good performance.

Appendix D. Empirical Analysis of Maximum Margin

Here, we first show that the existence of a backdoor attack boosts the target class logit, while more importantly, suppressing the logits of all other classes. We consider a classifier attacked with target class "deer" from the 1-pixel-S ensemble and a clean classifier, both for the CIFAR-10 domain. For both classifiers, we maximize the logit of the "deer" class using the same MM-BD protocol. The resulting logits for all classes for the two classifiers are shown in Fig. 9a. Even though we do not deliberately suppress the logits of all classes other than the "deer" class, we observe significant decrements in these logits compared with the clean case. Thus, while a backdoor attack may evade a detector that is based solely on maximizing the logit (when the "boosting" effect alone does not generate a sufficiently atypical statistic that is detectable), it will be easily detected by MM-BD based on MM, which leverages both logit "boosting" and "suppression" effects.

However, can we maximize just the logit and then obtain a margin statistic for detection? Unfortunately, this alternative will easily lead to false detections, especially when a substantial number of classes have closely "neighboring" classes containing similar semantic features. Here, we consider a clean classifier trained on TinyImageNet for example. By maximizing just the logit (i.e., maximizing the first term of (6)) for, e.g., the 'plunger' class, we will obtain a similarly large logit for the 'drumstick' class, possibly because the two categories of objects are similar to each other. Thus, the margin statistics obtained for the 'plunger' class will be small. Given a large number of such small margins (e.g. the top of Fig. 9b), relatively large margins produced by classes without a "neighboring" class will likely appear as outliers, which will easily lead to false detections. By contrast, MM-BD avoids false detections by directly maximizing the margin instead of the logit, which yields decently large margins for most classes, as shown in the bottom of Fig. 9b.

Appendix E.
Meta-Review

E.1. Summary

The paper describes methods to counter backdoors in already trained and distributed machine learning models, based on ”maximum margin” statistics, that is, statistics across the classifier’s logits. The authors observe that the target class of a backdoor model yields higher outputs than all other classes if maximized with gradient ascent and random initialization.

E.2. Scientific Contributions

- Addresses a Long-Known Issue

E.3. Reasons for Acceptance

1) **Simplicity of the approach.** The underlying rationale seems apparent and intuitively plausible. MM-BD merely needs comparing logits which is beneficial for the approach’s runtime complexity. Moreover, the defender neither knows the training data nor has details of the training procedure.
2) **Extensive number of experiments.** The paper extensively evaluates the approach in various settings with different backdoor and trigger types. The authors extend to conduct adaptive attacks to explore the limits of their detection scheme.
3) **Outperforms current state of the art.**

E.4. Noteworthy Concerns

1) **Not robust against adaptive attacks.** The presented results show that MM-BD is severely impacted by an adaptive attacker controlling the learning process.
2) **High false positive rate for datasets with few classes.** The method exhibits a relatively large false-positive rate for datasets with a small number of class, which however is not the case for datasets with many classes.
3) **Existing concurrent work:** Chen et al., ”Effective Backdoor Defense by Exploiting Sensitivity of Poisoned Samples”, NeurIPS 2022