LS^2: Boosting Hidden Separation for Backdoor Defense with Learning Speed-driven Label Smoothing

Jie Peng, Hongwei Yang, Hui He, *Member, IEEE,* Jing Zhao, HaoYu He, Hengji Dong, Weizhe Zhang, *Senior Member, IEEE*

***Abstract*—Backdoor attacks have become a security threat to deep neural networks (DNNs), in which an attacker embeds a secret behavior into a DNN by poisoning a few training data. To address the backdoor threat, some defense strategies employ outlier detection algorithms to identify poisoned samples in hidden representation space. However, these defenses remain vulnerable to adaptive attacks as their representation separability assumption could be broken. In this paper, we aim to boost existing defenses by leveraging insights from the label smoothing technique, demonstrating its effectiveness in distinguishing poison from benign samples. Our analysis uncovers the role of label smoothing as a regularization technique that enhances hidden class separability in the penultimate layer of a model. Building on the label smoothing, we introduce Learning Speed-driven Label Smoothing (LS^2): a simple yet novel approach that assigns an adaptive smoothing rate based on the model's "learning speed" for each sample. Extensive results show that LS^2 can bolster the discernibility between poison and benign samples, enhancing the efficacy of defenses relying on hidden separability. Incorporated with LS^2, existing hidden-separation-based defenses achieve state-of-the-art poison sample removal rates (P_{rm}) against adaptive attacks. Code is available at https://github.com/JiePeng104/LS2.**

***Index Terms*—Deep Neural Networks, AI Security, Backdoor Defenses, Label Smoothing**

I. Introduction

As deep learning algorithms demand substantial training data, it's a common practice to gather training data from various sources. However, such practice opens up opportunities for attackers to manipulate the behavior of a system by inserting poisoned instances into the training data. One of the most potent forms of poisoning attacks is the *backdoor attack*, in which the adversary embeds a backdoor into the model, enforcing the model to change its prediction when a trigger is present [1]. Detecting such attacks is challenging because the model's behavior remains unchanged on clean inputs. Moreover, Carlini *et al.* [2] recently showed that modifying a small fraction of web-scale datasets is sufficient to implant a backdoor into the victim model.

To mitigate the risk of backdoor poisoning attacks, some defenses have tried to remove the implanted backdoor by repairing the model with a few clean instances [3]–[5], while others have sought to detect and filter out the adversarially crafted training data by unique characteristics during the training stage [6]–[8]. Based on the observation that representations at hidden layers of poisoned and benign are separable, a line of defenses has succeeded in defending against a range of backdoor attacks by discarding the malicious samples [6], [9], [10]. These defense strategies follow a common approach to train a DNN on the malicious dataset. After that, they extract

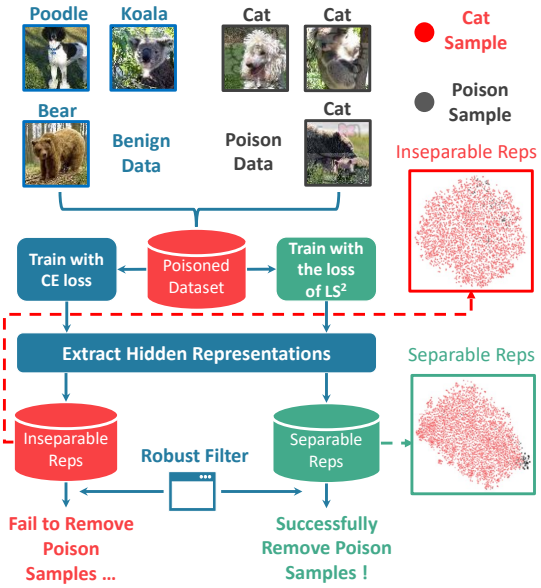


Fig. 1. Adaptive backdoor attacks can easily bypass defenses based on hidden separation (*i.e.,* robust filter), as the underlying assumption of hidden separability is compromised. In this work, we show that the hidden separability can be largely boosted when training the model with our LS^2 instead of the standard approach. Thus, such separability enables the corresponding defense to successfully remove poison samples from various attacks.

This work was supported in part by the Joint Funds of the National Natural Science Foundation of China (Grant No. U22A2036), the National Key Research and Development Program of China (2021YFB3101102), the National Natural Science Foundation of China (Grant No. 62202123), and the National Key Research and Development Program of China (Grand No. 2023YFB4503205).

Jie Peng, Hongwei Yang, Hui He, Jing Zhao and Hengji Dong are with the School of Cyberspace Science, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China (e-mail: jie.peng1004@gmail.com, yanghongwei@hit.edu.cn, hehui@hit.edu.cn, zhao_jing@hit.edu.cn, donghengji@stu.hit.edu.cn).

Haoyu He is with the Department of Data Science and AI, Faculty of IT, Monash University, Melbourne, VIC 3800, Australia, (e-mail: charles.haoyu.he@gmail.com).

Weizhe Zhang is with the School of Cyberspace Science, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China, and also with the Pengcheng Laboratory, Shenzhen, Guangdong 518055, China (e-mail: wzzhang@hit.edu.cn).

Corresponding Author(s): Hui He (e-mail: hehui@hit.edu.cn).

the representations in hidden layers from the polluted model and perform variants of the outlier detection algorithm to identify poisoned samples. As a part of these defenses being developed from robust estimation, they come with probabilistic guarantees for correctly identifying a sufficient number of poisoned samples. Despite the effectiveness of hidden-separation-based defenses, they are facing new challenges from adaptive attacks [11]. In the context of adaptive attacks, an attacker can create poisoned samples with careful design to break the fundamental assumption of hidden separability, ultimately evading these defenses. Besides, as previously discussed by Hayase *et al*. [10], some of these defenses may fail to remove the poisoned samples when there is a small number of poisoned data.

The key insight of our work is to address the above challenges by first connecting label smoothing to existing hidden-separation-based defenses. We leverage the effect of label smoothing in increasing class separation [12], [13] to boost the latent separability between benign and poison data, thus making the representations conform to the assumption of hidden-separation-based defenses. Specifically, to better understand how label smoothing enhances representations, we explore its role as a regularization technique for improving class separation in the penultimate layer. Our analysis reveals the implicit impact of label smoothing on hidden representations. We also find the fundamental differences in area under the margin (AUM) [14] values between poison and benign samples, where AUM is an metric to assess whether the model quickly learns a sample. Unlike benign samples, various poisoned samples either exhibit an "easy-to-learn" or "hard-to-learn" behavior. Building upon these findings, we devised a novel learning strategy, *Learning Speed-driven Label Smoothing (LS^2)*, to separate poison and benign samples in latent spaces. Instead of setting a fixed smoothing rate, we adjust the smoothing rate of each sample depending on how "quickly" the model learns them. We show that LS^2 can enhance the performance of hidden-separation-based defenses, including Activation Clustering (AC), Spectral Signature (SS), and SPECTRE (shown in Fig. 1). Overall, our main contributions are:

- We propose a simple yet novel learning strategy, *Learning Speed-driven Label Smoothing (LS^2)*, to intensify the separation between poisoned and benign samples.
- To shed light on how label smoothing boosts hidden separability, we reform it as a regularization term and demonstrate its implicit impact on class separation in the penultimate layer.
- We also identify the fundamental disparities in AUM values between poisoned and benign samples. Based on such a characteristic, LS^2 successfully separates poison samples from clean ones, even when against adaptive attacks or attacks with only a small number of poisoned data.
- Extensive experiments demonstrate that existing hidden-separation-based defenses, with LS^2, achieve an average improvement of 48.67% on CIFAR10 and 55.68% on GTSRB in P_{rm} against adaptive attacks. We also show that the combination of LS^2 and SPECTRE is robust to 8 state-of-the-art backdoor attacks.

II. Related Work

A. Backdoor Attacks

The objective of a backdoor adversary is to make the victim model (DNN) capable of accurately predicting the ground-truth labels for clean inputs while assigning a specific target label to inputs embedded with the trigger [1], [15]. To achieve the attack goal, a backdoor attacker usually poisons a portion of the training set by injecting a pre-designed trigger and relabeling them as the target labels. The trigger patterns can be simple pixel squares, real-world objects like a plant [16], or even be invisible patterns [17]–[20]. As mislabeled input–label samples would be easily detected as outliers, some studies have explored methods for performing backdoor attacks without flipping the labels of poisoned samples [21]–[24].

To evade the hidden-separation-based backdoor defenses, a series of backdoor attacks have attempted to reduce the hidden separation between benign and poisoned samples by optimizing the victim model [25], [26]. However, this type of attack violates the fundamental threat model of a backdoor poisoning attack, where the attacker should only have access to the training data rather than being able to manipulate the entire training process. More recently, Qi *et al*. [11] introduced an adaptive backdoor strategy capable of significantly reducing latent separation. In their adaptive attack, adversaries utilize distributed triggers [27] to make poison samples more challenging to detect. Furthermore, to obscure the correlations between the trigger and the target class, only a very small number of poison data is injected into the dataset, and not all trigger-attached samples are re-labeled to the target class. Despite its simplicity, adaptive attack is highly effective and poses a new challenge for separation-based defense mechanisms.

B. Backdoor Defense

1) Training-time Defense: Hidden-separation-based defense is a typical training-time defense, where the defender must train clean models (DNNs) on a polluted dataset. It is widely observed that representations at the intermediate layers of poisoned and benign samples generated by certain attacks are distinguishable. Based on such observation, Tran *et al*. [6] designed an algorithm, Spectral Signature (or PCA), to isolate the malicious samples with probabilistic solid guarantees. Additionally, Chen *et al*. [9] proposed employing clustering algorithms to separate benign and poisoned data. Subsequently, two more effective defenses, SCan [28] and SPECTRE [10], were introduced. While the family of hidden-separation-based defenses has proven successful in specific regimes, as mentioned earlier, it is facing a new challenge from adaptive backdoor attacks [11].

Aside from hidden separability defenses, other training-time defenses also sieve out malicious data to mitigate the backdoor threat, but they utilize different characteristics of the backdoored samples [7], [29]–[31]. Among these defenses, DBD [30] and D-BR [31] also leverage some special characteristics of poison representations to eliminate the implanted

backdoor. Despite sharing similarities with hidden-separation-based defenses, the effectiveness of DBD and D-BR relies on self/semi-supervised learning, which differs from the standard supervised learning procedure employed in SS, AC, and SPECTRE.

2) Other Defenses: Other defenses can be divided into testing-time defense and model defense. Testing-time defense is designed to detect if input data is poisoned and filter malicious samples to prevent the attacker from triggering the backdoor during the inference stage [32], [33]. On the other hand, model defense is performed after the training stage, where the defender can own a small set of clean data and computational resources to repair the polluted model. The primary goal of model defense is to eliminate the backdoor thoroughly implanted into the model [4], [5], [34], [35].

C. Label Smoothing

Label smoothing is a commonly used technique to improve the generalization of a DNN [36]–[38]. Instead of training the model with one-hot label vectors, label smoothing works by mixing the original targets with a uniform distribution and then computing the softmax cross-entropy concerning the model's logits output. This technique is generally understood as a regularizer that mitigate the model from "over-fitting" problems. Inspired by the calibration effect of label smoothing, some backdoor attacks relabel the poisoned samples to the target class with a small probability rather than 100% [11], [39]. Such a poison strategy can improve the stealthiness of backdoor attacks. Besides the calibration effect of label smoothing, it is empirically studied that label smoothing can lead to greater class separation [12], [13]. Based on the class separation effect of label smoothing, our work aims to boost the hidden separation for backdoor defenses.

III. Preliminaries and Threat Model

1) Multiclass Classification: For a classification task with K classes, we seek to find a function (model) $\boldsymbol{F}$ to classify any instance X into one of labels $Y = \{1, 2, ..., K\}$. Usually, $\boldsymbol{F}(\cdot)$ can be decomposed as $\boldsymbol{F}(\cdot) = \boldsymbol{W}\boldsymbol{f}(\cdot)$, where $\boldsymbol{f}$ indicates the feature extractor and $\boldsymbol{W}$ is the projection weights of the last linear layers. Meanwhile, let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$ denote a training set, where each $\boldsymbol{x}_i$ indicates the training sample and $\boldsymbol{y}_i$ is the corresponding label vector. In the general case, $\boldsymbol{y} \in \{0, 1\}^K$ is a one-hot vector, with only the ground-truth y equal to 1. To find a $\boldsymbol{F} : X \to \mathbb{R}^K$ fitting the distribution from which $\mathcal{D}$ is drawn, we can minimize the empirical risk $R(\boldsymbol{F}; \mathcal{D})$:

$$R(\boldsymbol{F}; \mathcal{D}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\ \ell(\boldsymbol{F}(\boldsymbol{x}), \boldsymbol{y}) = -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\ \log p_y(\boldsymbol{x}), \tag{1}$$

where $p_k(\boldsymbol{x}) = \frac{\exp\{\boldsymbol{F}(\boldsymbol{x})_k\}}{\sum_{j=1}^{K}\exp\{\boldsymbol{F}(\boldsymbol{x})_j\}} = \frac{\exp\{\langle \boldsymbol{W}_k, \boldsymbol{f}(\boldsymbol{x})\rangle\}}{\sum_{j=1}^{K}\exp\{\langle \boldsymbol{W}_j, \boldsymbol{f}(\boldsymbol{x})\rangle\}}$ is the likelihood the model assigns to the k^{th} class and ℓ is the softmax cross-entropy.

2) Label Smoothing: Different from the vanilla training setting, label smoothing uniformly smooths the one-hot vector $\boldsymbol{y}$ with a constant coefficient α in the range of $[0, 1]$ [36]:

$$\boldsymbol{y}^{LS,\alpha} = (1 - \alpha) \cdot \boldsymbol{y}\ +\ \frac{\alpha}{K} \cdot \mathbf{I}, \tag{2}$$

where $\mathbf{I}$ is an all-one vector. Then, the empirical risk based on cross-entropy becomes as follows:

$$\begin{aligned} R(\boldsymbol{F}; \mathcal{D})^{LS,\alpha} &= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\ \ell(\boldsymbol{F}(\boldsymbol{x}), \boldsymbol{y}^{LS,\alpha}) \\ &= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\ \left((1-\alpha)\log p_y(\boldsymbol{x}) + \frac{\alpha}{K}\sum_{k=1}^{K}\log p_k(\boldsymbol{x})\right). \end{aligned} \tag{3}$$

3) Threat Model: In line with previous studies on backdoor attacks, we consider a scenario where an adversary can manipulate a portion of the training data but does not train the model personally. Formally, let's consider a backdoor attacker who has created a malicious training set $\mathcal{D}_{adv} = \mathcal{D}_b \cup \mathcal{D}_p$. Here, $\mathcal{D}_b = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}$ indicates the dataset that only contains benign samples. Meanwhile, $\mathcal{D}_p = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_t)\}_{i=1}$ is the poisoned dataset where $\tilde{\boldsymbol{x}}_i$ denotes the poisoned sample and $\tilde{\boldsymbol{y}}_t$ indicates the target one-hot label vector. Then, the poisoned rate ϵ is defined as $\epsilon = |\mathcal{D}_b| \ / \ |\mathcal{D}_{adv}|$. When a victim train a model on the adversarial dataset $\mathcal{D}_{adv}$, the empirical risk of both benign and poisoned samples is minimized. Consequently, the compromised model would behave normally on the benign samples but output the target labels when trigger patterns are present. A hidden-separation-based defender needs to train on the polluted dataset and filters out the malicious samples effectively. The defender must ensure that the model is not backdoored by the poisoned data while also maintaining accuracy on clean data.

IV. The Proposed Method

In this section, we first give a theoretical analysis of the class separation effect of label smoothing. Then, we present the observation result of the disparities in AUM values between poisoned and benign samples. Putting these all together, we introduce our adaptive smoothing strategy, LS^2, to boost the hidden separation for backdoor defenses.

A. Class separation effects of label smoothing

Previous studies have investigated how label smoothing affects the hidden representations extracted from DNNs [13], [40], [41]. Their primary empirical observation is that label smoothing helps increase class separation and enforce tight clusters. While existing theoretical analyses of the mechanisms behind label smoothing mainly focus on its impact on entropy regularization [42], denoising effects [40], [43], and enhancing generalization [44], a theoretical explanation for why label smoothing can produce better class separation still needs to be explored. Thus, we study its role as a regularizer with respect to the weights $\boldsymbol{W}$ of prediction linear layers and hidden representation $\boldsymbol{f}(\boldsymbol{x})$ and get the following theorem (See Appendix A for the proof).

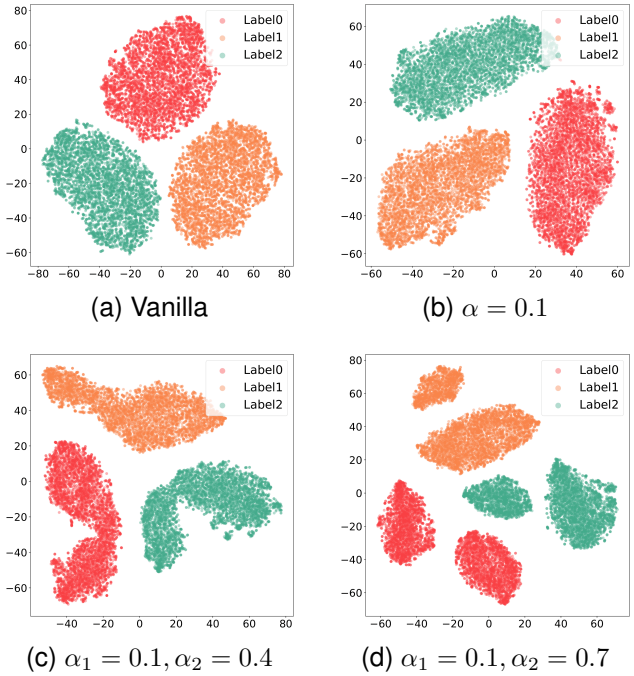
Theorem 1. *The emperical risk based on cross-entropy of label smoothing $R(\boldsymbol{F})^{LS,\alpha}$ is equivalent to the vanilla emperical risk $R(\boldsymbol{F})$ with one additional regularization term $\Omega(\boldsymbol{F})$:*

$$R(\boldsymbol{F}; \mathcal{D})^{LS,\alpha} = R(\boldsymbol{F}; \mathcal{D}) + \frac{\alpha}{K}\ \Omega(\boldsymbol{F}), \tag{4}$$

where $\Omega(\boldsymbol{F}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\sum_{k=1}^{K}\langle \boldsymbol{W}_y - \boldsymbol{W}_k, \boldsymbol{f}(\boldsymbol{x})\rangle$.

From Theorem 1, we see that label smoothing can be regarded as a regularization term $\Omega(\boldsymbol{F})$, which influences both $\boldsymbol{W}$ and $\boldsymbol{f}(\boldsymbol{x})$. For each sample pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, to minimize $\Omega(\boldsymbol{F})$, the learning algorithm should decrease the output of the y^{th} class $\boldsymbol{F}_y(\boldsymbol{x}) = \langle \boldsymbol{W}_y, \boldsymbol{f}(\boldsymbol{x}) \rangle$ while increasing the outputs for the other k^{th} classes. Such an effect aligns with the role of label smoothing in preventing the model from becoming overly confident. Nevertheless, how does it affect the penultimate layer representation $\boldsymbol{f}(\boldsymbol{x})$?

As the gradient of $\Omega(\boldsymbol{F})$ with respect to $\boldsymbol{f}(\boldsymbol{x})$ depends on the weights $\boldsymbol{W}$, it is not possible to directly compute such gradient without knowledge of $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{f}(\boldsymbol{x})}$. From another point of view, we can observe that minimizing the regularization term $\Omega(\boldsymbol{F})$ essentially amounts to decreasing the cosine similarity between $\boldsymbol{d}_y = \sum_{k=1}^{K} [\boldsymbol{W}_y - \boldsymbol{W}_k]$ and $\boldsymbol{f}(\boldsymbol{x})$. Thus, for each sample pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, the gradient of $\Omega(\boldsymbol{F})$ with respect to $\boldsymbol{f}(\boldsymbol{x})$ is significantly influenced by the directional vector $\boldsymbol{d}_y$. Simultaneously, the degree to which this gradient is influenced depends on the coefficient α/K. Consequently, when performing gradient descent, the orientation to update representations of samples from the same class y is guided by the same correction direction vector $\boldsymbol{d}_y$. Moreover, for two samples from different classes y_1 and y_2, the update of their representations would be affected by distinct directions $\boldsymbol{d}_{y_1}$ and $\boldsymbol{d}_{y_2}$. Therefore, as a regularization with coefficient α/K, label smoothing is beneficial for enhancing class separation and making each label cluster tighter.



(a) Vanilla (b) $\alpha = 0.1$

(c) $\alpha_1 = 0.1, \alpha_2 = 0.4$ (d) $\alpha_1 = 0.1, \alpha_2 = 0.7$

Fig. 2. Visualization of penultimate layer's activations of PreActResNet on CIFAR10 with different label smoothing settings.

We further validate our analytical findings by training the network with different smoothing strategies. Subsequently, we visualized the corresponding hidden representations with T-SNE [45], as depicted in Fig. 2. Compared to the results without smoothing, shown in Fig. 2a, when we set $\alpha = 0.1$, clusters for each class become more compact, consistent with prior research findings. After evenly dividing the data for each class and adjusting the corresponding learning rates differently, as illustrated in Figs. 2c and 2d, clusters for each class are visibly split into two distinct groups. Especially with the settings of $\alpha_1 = 0.1$ and $\alpha_2 = 0.7$, the intra-class splitting effect becomes more pronounced.

However, we encounter an issue when we redirect our focus to our initial objective of improving separability-based backdoor defenses. We notice that label smoothing has a limited impact on distinguishing between benign and poisoned samples belonging to the same class. The regularization term $\Omega(\boldsymbol{F})$ applies the same correction gradient to both clean and poisoned samples, resulting in minimal change in the relative Euclidean distances between their representations.

B. Learning Speed-driven Label Smoothing

An ideal way to separate the representations of clean and poison samples is to assign an adaptive smoothing rate α for each sample. As we do not know which data points were contaminated, we demand additional characteristics to guide us in setting the smoothing rates for data within each class. Here, we use a metric, AUM [14], to assess whether the model quickly learns a sample. Based on the AUM metric, samples with high AUM values are considered "easy-to-learn" and are assigned a higher smoothing parameter. In contrast, samples with low AUM values would receive a lower smoothing rate.

1) Area Under the Margin: As introduced in [14], the AUM metric is a technique initially used for identifying mislabeled data in noisy label learning, which utilizes differences in logits of benign and mislabeled samples during training. Despite the different assumptions and scenarios of label noise and backdoor poisoning attacks, we find significant differences between clean and backdoor poison samples in this metric. Suppose that $\boldsymbol{z}^{(t)}(\boldsymbol{x})$ is the logits vector (pre-softmax output) when model $\boldsymbol{F}$ classifies a sample $\boldsymbol{x}$ labeled as y at epoch t. For each iteration, the algorithm would record a margin value:

$$M^t(\boldsymbol{x}, y) = z_y^{(t)}(\boldsymbol{x}) - z_l^{(t)}(\boldsymbol{x}), \tag{5}$$

where $z_l^{(t)}(\boldsymbol{x}) = \max_{j \neq y} z_j^{(t)}(\boldsymbol{x})$ is the largest logit except $z_y^{(t)}(\boldsymbol{x})$. After training for T epochs, a sample's margin is averaged at each epoch to get the area under the margin (AUM) value:

$$\text{AUM}^T(\boldsymbol{x}, y) = \frac{1}{T} \sum_{t}^{T} M^t(\boldsymbol{x}, y). \tag{6}$$

For a data pair $(\boldsymbol{x}, \boldsymbol{y})$, AUM can also be used to assess whether the data is "easy-to-learn." A higher AUM value indicates that the model can quickly memorize the data, while a lower AUM value suggests that the data is less conducive to learning.

Fig. 3 presents the averaged AUM values of different types of samples at each epoch. In the evaluation, we performed eight attacks on CIFAR10, including six label-flipping attacks (*i.e.,* BadNets, Blend, ISSBA, LF, Adap-Blend, and Adap-Patch), and two clean label attacks (*i.e.,* LC and SIG). As shown, with the growth of the training epoch, the AUM values of both benign and poisoned samples are rising. Since the

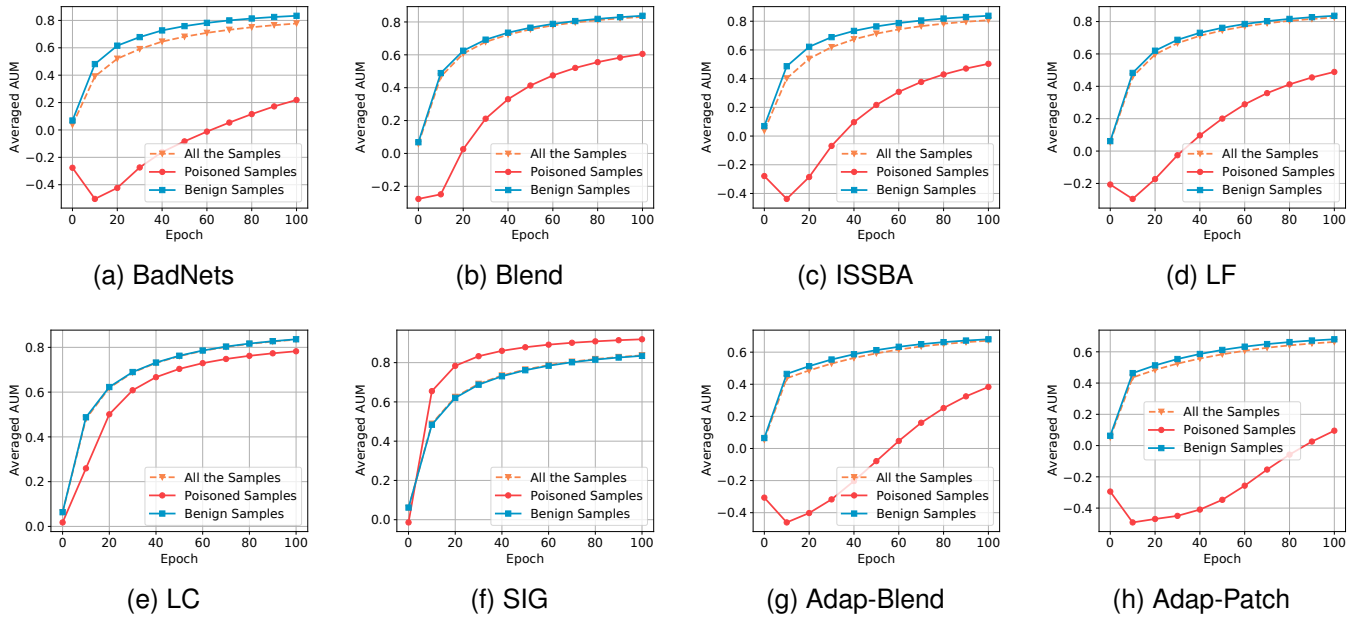


Fig. 3. The comparison of the AUM value in different epochs of PreActResNet-18 on CIFAR10.

benign samples account for a large proportion of training data in the backdoor attack scenario, the averaged AUM of benign samples is nearly identical to the AUM value of all the samples.

In addition, except for the SIG attack, the curve representing the poison AUM for other attacks always remains below the curve for benign samples. We also observe that during the early stages of training, the AUM values of poison samples within each class exhibited significant differences in most cases when compared to the averaged AUM of all the intra-class samples. This observation implies that benign and poisoned samples have distinct learning difficulty levels.

To further explore the distribution of AUM values during the early learning epochs, we illustrate the density of AUM values for eight attacks at the 10^{th} epoch in Fig. 4. It can be observed that the AUM values of these label-flipping attacks are generally lower than those of benign data. However, the results for clean label attacks, as shown in Figs. 4e and 4f, do not follow the same pattern. For LC attack, the AUM values of the attack data are similar to those of benign samples, while in the SIG attack, the AUM values of the attack data are higher than those of clean data. This phenomenon is due to the similarity between label-flipping attacks and the noisy label problem. In label-flipping attacks, malicious samples' original labels differ from benign ones, rendering them "hard-to-learn". Conversely, the poison samples do not undergo label changes in clean label attacks. However, with the addition of certain adversarial perturbations or specific triggers, their AUM values become similar to those of benign data, indicating that they are "easy-to-learn."

Here, we highlight that the terms "hard-to-learn" and "easy-to-learn" used in our paper are defined under the AUM metric. While some prior studies have employed similar terms, they may refer to different metrics for filtering poison samples. Section S.IV in the supplementary material provides more comprehensive discussion and additional results.

2) Smoothing Strategy: Following the above analysis, we propose a new training strategy to enhance the distinguishability of poison and benign samples in their representations. The intuition behind our algorithm is to apply a larger label smoothing rate to samples with high AUM values (referred to as "easy-to-learn"), while applying a smaller rate to those with low AUM values. Our approach initially intends to mitigate the "over-fitting" problem. Since the AUM distribution between benign and poisoned samples differs, it also implicitly separates poisoned samples from clean ones, in line with the analysis in Section IV-A. Specifically, we employ linear transformation to rescale all AUM values to fall within a predefined smoothing rate interval $[0, \beta]$. Then, we obtain the smoothing rate α_i for each sample pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$:

$$\alpha_i = \beta \cdot \frac{\text{AUM}^T(\boldsymbol{x}_i, y_i) - \text{AUM}^T_{\min}}{\text{AUM}^T_{\max} - \text{AUM}^T_{\min}}. \tag{7}$$

We give the pseudocode in Algorithm 1. In the warm-up phase, we initially employ a small α, such as 0.1, to smooth all labels in the dataset and proceed with model training, recording the corresponding $M^t(\cdot)$ at each step. At a specific epoch T, we re-smooth all label vector $\boldsymbol{y}_i$ using the smoothing rate α_i and then continue training the model on the malicious dataset. It is important to note that our algorithm differs from other dynamic label smoothing training strategies in that we adjust the smoothing rate only in the T^{th} round, rather than in every round [41], [44].

Here, we emphasize that we cannot directly separate poison samples from the rest by AUM values, as we cannot determine whether the AUM values of poison samples are higher or lower than those of the overall data. Nevertheless, based on this characteristic, we can still assign an adaptive smoothing rate to each sample data within each class. For example,

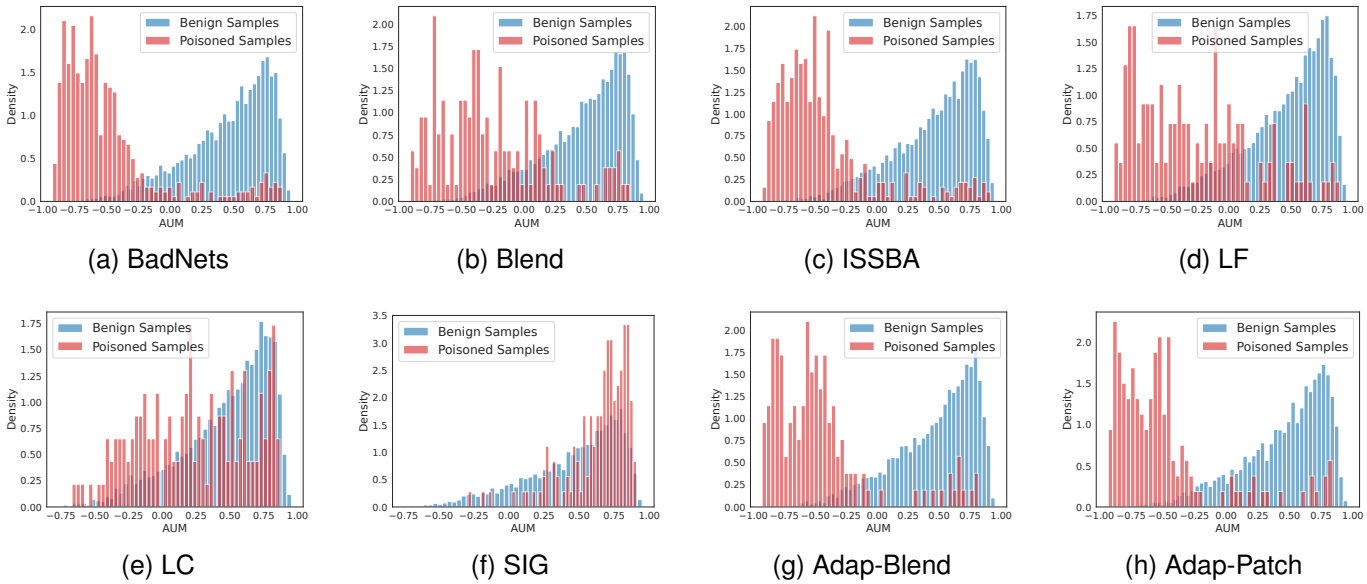


Fig. 4. The comparison of AUM value density between benign and poison samples of PreActResNet-18 at 10^{th} epoch on CIFAR10.

Algorithm 1 Learning Speed-driven Label Smoothing

Input: Malicious dataset $\mathcal{D}_m = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$,
Maximal epoch E, Warmup smoothing rate α,
Warmup epoch T, Upper smoothing bound β
Output: Model trained with LS^2
Initialize $\boldsymbol{F}$ with parameters θ;
Compute smoothed label $\hat{\boldsymbol{y}}_i = \boldsymbol{y}_i^{LS,\alpha}, \forall \boldsymbol{y}_i \in \mathcal{D}_m$;
Let $\hat{\mathcal{D}}_m = \{(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)\}_{i=1}^N$;
for $t = 1$ **to** E **do**
  Using SGD, update θ to minimize $R(\boldsymbol{F}; \hat{\mathcal{D}}_m)$;
  Using 5, compute $M^t(\boldsymbol{x}_i, y_i),\ \forall (\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}_m$;
  if $t == T$ **then**
    Using 6, compute $\text{AUM}^T(\boldsymbol{x}_i, y_i),\ \forall (\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}_m$;
    Using 7, rescale $\text{AUM}^T(\boldsymbol{x}_i, y_i)$ to α_i falling in the interval $[0, \beta]$, $\forall (\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}_m$;
    Re-smooth label vector $\hat{\boldsymbol{y}}_i = \boldsymbol{y}_i^{LS,\alpha_i}, \forall \boldsymbol{y}_i \in \mathcal{D}_m$;
  end if
end for
return model $\boldsymbol{F}$ with parameters θ

when against adaptive attacks, LS^2 assigns low smoothing rates to the poison samples as they exhibit "hard-to-learn", Figs. 3g and 3h. Conversely, benign samples receive a higher smoothing rate. Thus, after training the model with LS^2, the latent separability can be preserved.

V. Experiments

A. Experimental settings

1) Attack configurations: We consider 8 typical backdoor attacks: BadNets [1], Blend [15], ISSBA [19], Low Frequency (LF) [20], Label Consistent (LC) [21], SIG [22], Adap-Blend and Adap-Patch [11]. BadNets and Blend are two classical label-flipping attacks with fixed trigger patterns. ISSBA and LF are sample-specific poison attacks where the trigger is independently generated for each poison sample. LC and SIG are two effective clean-label attacks. Adap-Blend and Adap-Patch are adaptive attacks specifically designed against hidden-separation-based defenses. Following the prior studies, we conduct all of these attacks and defenses on CIFAR10, CIFAR100 [46] and GTSRB [47] using Pre-activation-ResNet-18 [48]. To further validate our method, we also conducted experiments using VGG19-BN on CIFAR10 and ImageNet100 [49] (with each class containing 1300 training images).

On the CIFAR10, CIFAR100 and ImageNet100 datasets, the target label for all backdoor attacks is class 0, corresponding to the specific class names "airplane", "apple" and "tench" . On the GTSRB dataset, the target label for all backdoor attacks is class 4, corresponding to the specific class name "70-speed". In line with the settings of Adap-Blend and Adap-Patch attacks [11], we set the poison rate as small as possible while ensuring that the attack success rate for each attack remained sufficiently high. Unless otherwise specified, the poison rates of various attacks are set as follow:

For CIFAR10, all attacks except for BadNet and ISSBA have a poison rate of $\epsilon = 0.3\%$ (150 poison samples). Since the ASR values for BadNet and ISSBA at $\epsilon = 0.003$ are below 10%, they are not considered successful backdoor attacks. Therefore, we set the poison rate for these two attacks to $\epsilon = 1\%$ (500 poison samples). In the case of GTSRB, the poison rate for all attacks is $\epsilon = 0.4\%$ (157 poison samples). Due to the low success rates of most backdoor attacks on the CIFAR100 dataset when the poisoning rate ϵ is set to a low value, we choose to use $\epsilon = 1\%$ (500 poison samples). However, CIFAR100 has only 500 training samples per class. Therefore, at this poison rate, all data under the target label of clean label attacks would be replaced with poison samples, which does not align with the basic premise of backdoor attacks. Additionally, both LC and SIG attacks have poor attack performance at lower poison rates, with success rates

below 40%. Therefore, we opt not to conduct experiments with clean label attacks on CIFAR100. For ImageNet100, we set the poison rate to 0.5% (650 poison samples) for all attacks.

2) Defense configurations: For hidden-separation-based defenses, we consider three defenses as the baseline: Spectral Signature (SS) [6], Activation Clustering (AC) [9] and SPECTRE [10]. Additionally, we also include 7 state-of-the-art backdoor defenses: NC [3], ABL [7], ANP [4], I-BAU [5], AWM [50], DBD [30] and D-BR [31]. 5% of the clean training dataset is provided for defenses demanding clean data. All other settings are identical to the default configuration in their original papers for fair comparisons. Regarding standard label smoothing, we set the smoothing rate α to 0.1. For our smoothing strategy, we initially smooth the label with a rate of 0.1 during the warm-up stage. In the case of CIFAR10, CIFAR100 and ImageNet100, we set the warm-up epochs to 10, while for GTSRB, the warm-up epochs are set to 2. Afterward, we set the upper smoothing bound $\beta = 0.2$.

See Section S.II in the supplementary material for more implementation details and comprehensive settings of various attacks and defenses.

3) Training Settings: Throughout all training processes, we employed the stochastic gradient descent (SGD) optimization method with a batch size of 128. We set the initial learning rate to 0.01 and decayed it using the cosine annealing strategy [51]. For the CIFAR-10 and CIFAR-100 datasets, we trained for a total of 100 epochs. For the GTSRB dataset, we trained for 50 epochs. In the case of the ImageNet100 dataset, we trained for 400 epochs. All experiments were run on one Ubuntu 18.04 server equipped with four NVIDIA RTX A4000 GPUs.

4) Evaluation metric: To evaluate the performance of various backdoor defenses, we employ two primary metrics: 1) the attack success rate (ASR), which is the ratio of attack samples misclassified as the target label, and 2) the accuracy on benign data (BA). A good defense is expected to have a high BA and low ASR (*i.e.,* close to zero). For a more effective comparison among hidden-separation-based defenses, we utilize P_{rm} to indicate the ratio of poison samples that are successfully removed.

B. Representation visualization

To intuitively compare the effects of different training strategies on the penultimate representations, we provide corresponding T-SNE visualizations in Fig. 5. Here, we display the visualization results for two adaptive attacks, Adap-Blend and Adap-Patch, against hidden-separation-based defenses on CIFAR10. The target label is set to 0 for both attacks, and the number of poison samples is fixed at 150. As depicted in Figs. 5a and 5b, when training the DNN with the standard approach the penultimate representations of poison and benign samples are entangled and not well-separated. When extracting the representations from DNN trained with label smoothing, we notice that the majority of poison samples have segregated from the clean ones for Adap-Blend attacks. However, only a portion of the poison samples have been separated for Adap-Patch attacks. As shown in Figs. 5e and 5f, LS^2 demonstrates its effectiveness by ensuring that the poison samples from both adaptive attacks have been successfully isolated from the clean data. More visualizations of other attacks are provided in Figure S5 in the supplementary material.

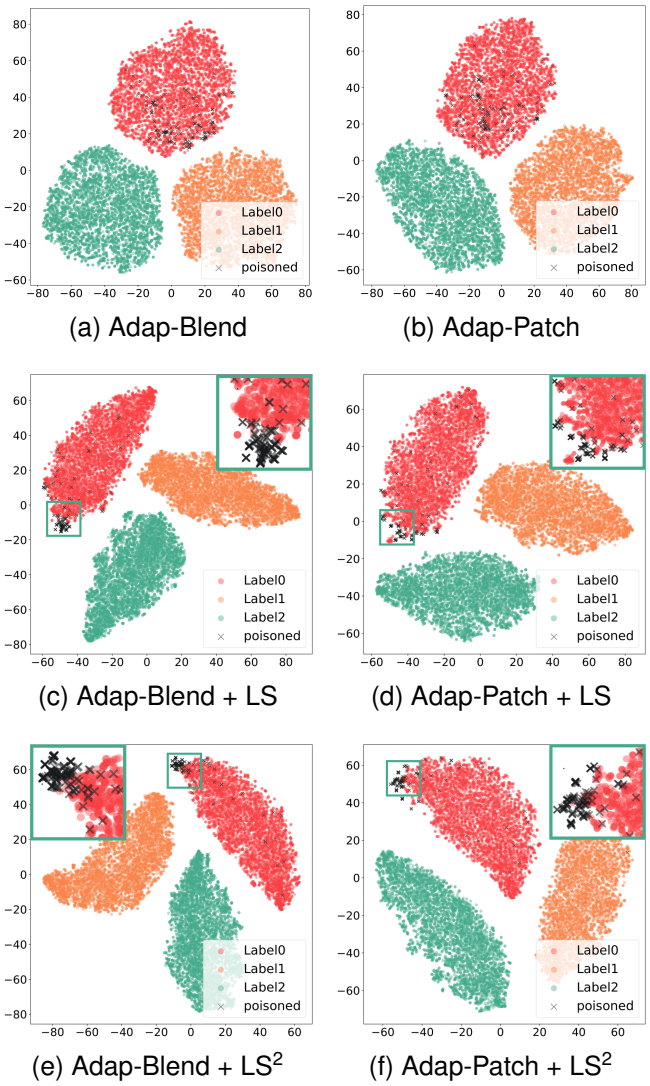


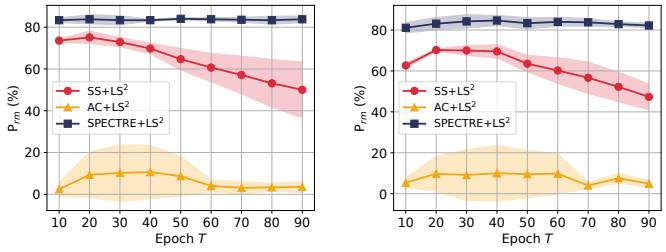
Fig. 5. T-SNE visualization of the activations from penultimate layers of PreActResNet-18 against Adap-Blend and Adap-Patch backdoor attacks on CIFAR10 with different label smoothing settings.

C. Performance in Eliminating Poison Samples

In Table I, we compare the impact of standard label smoothing and our smoothing strategy LS^2 on various hidden-separation-based defenses on CIFAR10 and GTSRB. We observe that on the CIFAR10 dataset, the original three defense methods have inferior resistance against the two types of adaptive attacks, with very low P_{rm} values. For AC defense, it cannot defend against backdoor attacks under any training method effectively. While label smoothing and our approach offer some enhancements for SS and SPECTRE defenses, the improvement in P_{rm} values with label smoothing is insufficient to defend against these two backdoor attacks. When extracting features from models trained with LS^2, SS and SPECTRE defenses successfully reduce the ASR values to below 10%. On the GTSRB dataset, both label smoothing and our training strategy significantly improve the filtering of poisoned data. Here, we explain the poor performance of the AC defense on

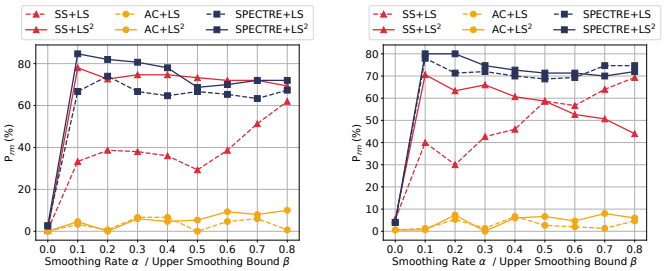
TABLE I
POISON SAMPLES REMOVAL PERFORMANCE COMPARISON OF PREACTRESNET-18 WITH AC, SS AND SPECTRE DEFENSES AGAINST ADAP-BLEND AND ADAP-PATCH.

Dataset↓	Attack→ Defense↓	Adap-Blend P_{rm}↑	Adap-Blend ASR↓	Adap-Patch P_{rm}↑	Adap-Patch ASR↓
CIFAR10	No Defense	(-)	71.9	(-)	78.8
	AC	0.0	71.5	0.7	77.2
	AC + LS	3.3	70.7	1.3	77.6
	AC + LS^2	0.0	71.2	7.3	71.3
	SS	0.7	71.8	5.3	77.4
	SS + LS	33.3	52.5	40.0	44.6
	SS + LS^2	**72.7**	**2.7**	**63.3**	**5.1**
	SPECTRE	2.6	71.2	4.0	77.2
	SPECTRE + LS	66.7	13.7	78.0	4.3
	SPECTRE + LS^2	**82.0**	**2.4**	**80.0**	**2.5**
GTSRB	No Defense	(-)	78.0	(-)	55.3
	AC	45.9	69.8	29.3	47.1
	AC + LS	**84.7**	**4.8**	66.9	14.6
	AC + LS^2	**86.6**	**4.4**	**86.0**	**3.6**
	SS	48.4	66.3	26.8	49.9
	SS + LS	89.2	6.2	84.1	3.4
	SS + LS^2	**94.3**	**4.3**	**96.6**	**2.5**
	SPECTRE	42.7	44.7	31.2	35.6
	SPECTRE + LS	**92.4**	**4.5**	**97.5**	**2.6**
	SPECTRE + LS^2	**96.8**	**4.4**	**98.1**	**2.5**



(a) Adap-Blend Attacks (b) Adap-Patch Attacks

Fig. 6. Effect of warmup epoch T on hidden-separation-based defenses against adaptive attacks on CIFAR10. We performed 3 training runs for each warmup epoch T.



(a) Adap-Blend Attacks (b) Adap-Patch Attacks

Fig. 7. Effect of label smoothing and LS^2 on hidden-separation-based defenses of PreActResNet-18 against adaptive attacks, CIFAR10.

the CIFAR10 dataset. The AC algorithm relies on a simple K-means clustering algorithm to filter poisoned samples in the latent space. When defending against attacks with a tiny number of poison samples (0.3% poison rate), separating the data into two clusters is challenging, as shown in Fig. 5. Therefore, the improvement provided by LS or LS^2 may be limited for such simple clustering defense methods.

In the supplementary material (Section S.V), we demonstrate that when facing other attacks, label smoothing and LS^2 also have some improvement in the performance of these three defense methods. However, more than the enhancement these two training strategies provide is needed to defend against all attacks. For example, on CIFAR10, AC and SS are unable to defend against Blend attacks across different training strategies.

D. Sensitivity to Hyperparameter choices

As mentioned in Algorithm 1, LS^2 mainly involves two hyperparameters: warmup epoch T and upper smoothing bound β. Figure 6 presents the results obtained using different warmup epoch T with a fixed upper smoothing bound $\beta = 0.2$. We observe that regardless of how warmup epoch T is changed, the enhancement of LS^2 on AC defense remains limited. With the increase in T, the effectiveness of SS defense shows a trend of improvement followed by deterioration. The SPECTRE defense remains relatively robust to the choice of T, showing no significant variation.

We also empirically explore the impact of different smoothing rates α and upper smoothing bounds β on AC, SS, and SPECTRE. Here, the warmup epoch T for LS^2 is fixed at 10. The results are presented in fig. 7. We observed that, as the smoothing rate α increases, the performance of AC and SPECTRE defenses in dealing with two adaptive attacks remains relatively unchanged. Larger α values do not significantly improve the defense effectiveness of AC and SPECTRE. However, an interesting phenomenon is that the SS defense exhibits improved filtering performance on poisoned samples as α increases. When $\alpha > 0.6$, SS with LS outperforms SS with LS^2 in defending against Adap-Patch. At $\alpha = 0.8$, the performance of SS with LS is on par with the SPECTRE algorithm at the same α or β settings.

Moreover, increasing the Upper Smoothing Bound β does not enhance the effectiveness of the AC defense. However, a larger β can reduce the poisoning sample filtering rate P_{rm} of SS and SPECTRE. This weakening effect is more pronounced, as shown in Fig. 7b, in the SS defense against Adap-Patch. This result implies that a small β is desirable for training the model.

E. Effectiveness of LS and LS^2

The effectiveness of various defenses against different backdoor attacks on the CIFAR10, CIFAR100, GTSRB and ImageNet100 datasets is presented in Table II and Table III. Here, we consider a defense successful if it reduces the ASR to below 10%, as indicated in green. Otherwise, we consider it a failure. Due to the limited ability of SS and AC defenses to eliminate some backdoor threats, we primarily display the results of defense combining label smoothing or LS^2 with the SPECTRE algorithm here.

Table II presents the results of PreActResNet-18 on CIFAR10, GTSRB and CIFAR100. Overall, most defense algo-

TABLE II
RESULTS OF PREACTRESNET-18 ON CIFAR10 AND GTSRB. RESULTS WITH AN ASR ABOVE 25% ARE HIGHLIGHTED IN RED, THOSE WITH AN ASR BETWEEN 10% AND 25% ARE HIGHLIGHTED IN YELLOW, AND THOSE WITH AN ASR BELOW 10% ARE HIGHLIGHTED IN GREEN.

Dataset	Attack↓	%	No Defense	NC	ABL	ANP	I-BAU	AWM	DBD	D-BR	SS	AC	SPECTRE	SPECTRE +LS	SPECTRE +LS^2
CIFAR10	BadNets	BA	93.35	92.40	91.12	93.29	90.37	88.73	80.33	88.31	89.40	84.60	92.40	92.60	92.35
		ASR	88.70	7.67	0.86	0.33	1.47	2.13	0.40	0.00	43.10	45.40	4.60	2.40	3.22
	Blend	BA	93.75	93.75	89.10	85.72	89.80	88.87	74.65	86.60	91.10	92.30	92.32	92.34	92.41
		ASR	86.10	86.10	47.50	9.71	62.15	40.10	0.04	0.00	80.20	84.10	7.10	2.40	2.43
	ISSBA	BA	93.56	91.32	85.04	92.65	90.84	88.90	79.31	89.24	93.40	93.42	93.14	92.94	93.14
		ASR	71.20	3.19	1.66	4.79	1.59	4.43	0.23	15.21	69.40	69.10	2.10	2.30	3.12
	LF	BA	93.75	93.75	87.45	86.06	89.85	87.81	82.32	85.41	93.67	93.55	90.45	92.27	90.34
		ASR	69.80	69.80	15.70	15.88	18.07	49.13	0.04	2.40	69.90	69.10	5.10	40.20	2.36
	LC	BA	93.59	91.73	87.60	90.15	91.46	88.12	84.34	88.46	88.32	89.50	91.23	91.34	91.01
		ASR	87.07	3.83	9.31	36.27	28.61	10.88	1.32	0.41	68.10	86.90	0.10	0.10	0.10
	SIG	BA	93.87	88.90	88.40	91.25	90.48	87.71	80.68	84.06	90.10	91.30	89.43	91.31	89.74
		ASR	77.52	14.38	15.40	47.59	44.32	36.91	20.33	0.00	77.75	73.60	0.50	73.30	0.54
	Adap-Blend	BA	93.78	93.78	85.34	92.35	90.12	87.89	79.06	87.83	91.36	92.45	90.34	89.46	88.61
		ASR	71.89	71.89	68.62	31.73	11.38	9.63	22.41	16.82	71.80	71.50	71.20	13.70	2.38
	Adap-Patch	BA	93.42	92.68	86.45	90.63	89.34	88.05	82.10	90.41	91.32	92.67	91.40	90.45	90.46
		ASR	78.84	8.44	47.32	10.33	4.10	2.40	2.11	0.04	77.40	77.20	77.20	4.30	2.50
GTSRB	BadNets	BA	98.16	97.83	96.48	97.66	96.27	96.31	83.72	96.41	96.45	97.45	96.86	96.57	97.42
		ASR	73.81	0.00	6.56	4.20	0.01	4.12	0.00	0.00	4.67	6.53	0.12	0.15	0.10
	Blend	BA	98.55	98.55	97.44	94.71	95.91	97.49	86.32	92.31	96.56	97.52	96.18	97.35	98.37
		ASR	89.24	89.24	68.42	65.12	48.25	55.64	0.00	0.04	23.82	25.46	7.49	2.45	2.46
	ISSBA	BA	98.47	97.38	96.28	86.10	95.47	96.95	81.32	91.61	95.34	96.44	97.26	96.94	96.44
		ASR	71.76	1.26	7.63	7.92	8.07	8.76	0.04	15.02	7.89	7.67	1.26	1.13	0.48
	LF	BA	98.10	97.12	97.23	97.63	95.91	96.84	84.52	94.41	96.89	97.01	96.78	96.93	96.67
		ASR	90.33	0.50	5.67	0.13	47.91	4.04	0.12	2.33	81.45	84.23	0.42	0.45	0.41
	LC	BA	98.01	98.01	97.17	96.53	96.76	96.23	80.44	95.43	97.56	96.87	96.78	96.56	96.62
		ASR	53.21	53.21	22.45	7.59	1.85	5.01	0.00	0.04	47.68	52.81	8.76	1.78	1.76
	SIG	BA	98.35	96.19	98.69	92.12	95.18	95.45	81.69	96.32	96.45	97.45	97.55	96.58	97.13
		ASR	58.26	32.38	64.90	30.36	4.70	55.53	16.43	0.00	40.29	58.13	4.67	6.53	4.55
	Adap-Blend	BA	98.53	98.53	95.53	97.73	96.57	96.68	85.03	96.53	95.42	96.78	97.53	98.23	97.74
		ASR	77.99	77.99	45.32	38.63	8.20	25.78	0.00	22.13	66.29	69.79	44.67	4.52	4.41
	Adap-Patch	BA	98.08	96.86	96.67	97.87	97.05	96.22	82.42	97.05	96.86	97.44	96.57	97.58	97.25
		ASR	55.33	0.00	20.43	0.00	0.01	0.10	0.01	0.00	49.89	47.06	35.56	2.57	2.51
CIFAR100	BadNets	BA	70.75	68.57	60.46	69.13	64.67	65.53	61.61	62.38	68.37	67.42	68.67	67.76	67.45
		ASR	71.22	0.15	0.08	5.77	1.25	0.89	2.47	2.33	71.04	2.13	70.23	2.31	2.34
	Blend	BA	70.47	70.47	59.45	65.58	64.75	61.93	60.53	65.63	66.78	68.61	67.35	65.67	66.89
		ASR	90.80	90.80	0.00	61.84	68.33	54.71	85.14	0.00	88.35	88.83	88.56	42.55	6.13
	ISSBA	BA	70.44	66.92	57.54	62.64	61.95	61.71	62.63	64.00	67.44	66.92	66.86	68.86	67.42
		ASR	55.77	0.10	0.03	5.67	0.63	3.28	0.00	15.32	2.67	53.42	0.43	0.10	0.10
	LF	BA	69.81	69.81	63.14	64.79	64.18	62.16	60.13	65.36	66.23	65.33	64.89	65.29	66.25
		ASR	58.83	58.83	40.34	3.26	2.49	17.05	0.01	0.04	42.17	2.67	2.85	3.13	2.87
	Adap-Blend	BA	70.71	70.71	59.21	64.82	63.97	61.77	63.39	64.16	68.42	69.28	67.97	68.97	68.53
		ASR	88.14	88.14	42.67	62.23	47.99	47.10	0.33	23.42	44.65	88.31	46.83	6.32	6.22
	Adap-Patch	BA	70.48	70.00	55.31	62.79	61.95	62.70	60.03	65.53	67.25	69.41	69.33	68.38	67.45
		ASR	91.69	0.20	40.32	19.96	2.94	7.35	0.00	0.00	50.23	90.32	45.47	0.10	0.00

rithms can only mitigate a few backdoor attack threats and cannot comprehensively defend against all backdoor attacks. The original SPECTRE algorithm performs well but cannot eliminate the backdoors implanted by adaptive attacks. On the GTSRB dataset, the combination of the SPECTRE algorithm and standard label smoothing also effectively removes the backdoors. However, on the CIFAR10 dataset, this combination is less effective than the original SPECTRE algorithm when dealing with SIG and LF attacks, highlighting the limitations of label smoothing in enhancing latent separability. In contrast to CIFAR10 and GTSRB, we found that the vanilla SPECTRE algorithm on CIFAR100 can only defend against ISSBA and LF backdoor attacks. Still, after applying label smoothing and LS^2, the P_{rm} values for SPECTRE substantially increase. The combination of LS^2 and SPECTRE is capable of defending against all types of attacks.

To extensively validate stability and universality, we performed experiments on CIFAR10 and ImageNet100 using VGG19-BN, as shown in Table III. Like what we observed in Table II, the SPECTRE algorithm with LS^2 successfully defends against all backdoor attacks on both datasets, compared to other defenses.

Moreover, we observe that ANP, I-BAU, AWM, DBD and D-BR defenses perform well against Adap-Blend and Adap-Patch attacks. However, these defenses either cannot remove the backdoor thoroughly or defend against other types of attacks. Besides, the assumptions and methods of these defenses are different from hidden-separation-based defenses, which is beyond the scope of this paper. Therefore, we do not delve into a detailed discussion here.

F. Performance of LS and LS^2 with Varying Poison Rates

As noted in the original paper on Adap-Blend and Adap-Patch [11], a low poison rate is necessary for the success of adaptive attacks. Their results show that a high poison rate may cause adaptive attacks to fail in evading hidden-separation-based defenses on small datasets like CIFAR10. Thus, there are limited choices of poison rates for performing adaptive attacks on such datasets. Since ImageNet100 is a dataset with a larger scale than CIFAR10, GTSRB, and CIFAR100, we

TABLE III
RESULTS OF VGG19-BN ON CIFAR10 AND IMAGENET100.

Dataset	Attack↓	%	No Defense	NC	ABL	ANP	I-BAU	AWM	DBD	D-BR	SS	AC	SPECTRE	SPECTRE +LS	SPECTRE +LS^2
CIFAR10	BadNets	BA	92.95	90.54	84.67	90.52	87.84	88.28	84.13	86.04	88.22	90.43	88.24	87.84	89.10
		ASR	92.67	0.96	0.25	4.30	22.81	8.82	0.00	5.32	80.54	80.67	30.80	31.58	8.19
	Blend	BA	93.36	88.64	87.13	90.57	89.30	89.33	77.42	88.82	89.14	89.62	87.10	86.15	87.38
		ASR	90.02	15.47	20.35	22.50	19.56	18.81	0.00	0.00	85.08	91.43	2.53	2.68	2.53
	ISSBA	BA	93.19	93.19	86.01	92.69	88.08	88.09	81.92	86.31	90.17	91.36	89.48	88.64	89.56
		ASR	80.81	80.81	0.13	2.08	1.23	1.63	0.10	0.00	60.35	70.72	0.42	1.45	1.62
	LF	BA	93.17	89.29	85.42	90.31	89.38	88.91	83.11	85.49	91.04	88.82	90.41	88.84	88.45
		ASR	58.39	4.48	4.43	2.20	6.86	11.50	2.45	0.00	50.16	51.38	4.04	5.58	4.12
	LC	BA	93.49	91.10	90.49	86.09	88.03	89.56	80.63	89.12	87.45	90.24	88.98	89.18	88.22
		ASR	99.34	3.23	0.16	5.77	77.50	11.72	0.00	0.00	90.67	96.67	44.85	10.42	2.68
	SIG	BA	92.81	92.81	90.67	92.48	88.04	89.13	81.68	86.68	89.45	87.28	88.24	89.47	87.56
		ASR	81.83	81.83	10.37	10.34	41.01	16.46	32.33	4.33	80.65	79.26	2.55	2.24	2.53
	Adap-Blend	BA	93.46	89.11	86.82	92.61	84.85	88.47	74.01	88.53	90.41	90.67	88.22	89.28	89.51
		ASR	70.02	12.30	46.18	17.01	11.34	8.34	16.62	18.94	71.14	68.41	21.28	1.42	2.41
	Adap-Patch	BA	93.38	91.18	84.71	91.98	89.05	88.28	80.13	88.64	89.64	88.09	88.18	89.75	88.53
		ASR	82.74	1.17	5.26	4.91	1.68	5.37	0.01	0.00	74.21	78.84	44.21	10.55	4.13
ImageNet 100	BadNets	BA	82.70	81.98	75.17	81.97	81.81	81.41	82.06	82.76	80.31	80.26	81.62	79.94	79.01
		ASR	99.98	12.53	21.12	2.53	2.78	0.00	3.10	0.00	16.25	41.32	1.13	2.98	1.03
	Blend	BA	82.68	81.43	73.25	81.77	78.75	80.96	82.44	82.40	79.11	81.68	82.46	78.81	81.04
		ASR	95.43	32.01	33.52	14.87	24.91	0.54	22.30	0.00	42.57	89.94	1.58	1.17	1.11
	ISSBA	BA	82.66	77.52	70.15	81.50	81.92	81.14	80.38	81.73	82.45	79.80	79.41	82.89	79.30
		ASR	65.42	21.67	63.19	4.23	22.51	1.42	12.32	2.14	46.57	61.50	4.26	2.39	3.24
	LF	BA	82.65	76.47	72.15	79.77	78.09	81.55	82.35	79.52	79.37	82.13	80.05	81.01	82.94
		ASR	78.14	47.06	73.62	19.54	2.10	23.60	4.52	19.58	74.79	67.15	2.24	1.58	1.93
	LC	BA	82.65	82.02	76.10	80.71	79.25	78.96	78.82	82.54	81.78	79.29	81.45	81.24	79.21
		ASR	54.74	0.42	52.11	0.35	0.12	0.00	4.52	0.00	51.21	51.58	2.45	0.21	0.22
	SIG	BA	82.14	79.55	80.18	81.61	79.90	82.40	79.22	79.69	82.88	79.72	78.77	80.62	82.76
		ASR	70.82	0.27	32.62	6.32	10.22	12.34	66.12	11.32	52.42	62.77	1.56	0.24	0.21
	Adap-Blend	BA	84.26	81.42	82.09	79.16	78.29	79.04	81.60	80.97	78.74	79.34	80.49	79.75	81.07
		ASR	79.76	32.43	32.92	2.35	21.87	4.23	16.15	4.54	34.11	22.19	27.13	4.42	4.42
	Adap-Patch	BA	83.04	79.51	80.67	80.24	80.51	81.41	80.74	82.00	79.89	79.72	82.51	82.99	82.48
		ASR	71.18	0.56	20.13	22.97	2.93	0.84	21.40	2.42	57.32	61.52	50.23	1.34	1.09

can increase the poison rate of adaptive attacks on this dataset while maintaining the effectiveness of the attacks.

Here, we present Table IV to show the performance of LS and LS^2 against Adap-Blend and Adap-Patch under two different poison rates, 1% and 5%, on ImageNet100. The number of payload data is the same as the poison samples. The defense configuration follows the same settings as previous experiments.

When the poison rate $\epsilon = 1\%$, Adap-Blend and Adap-Patch can still evade the original SS, AC, and SPECTRE defenses. When the poison rate $\epsilon = 5\%$, Adap-Blend and Adap-Patch can bypass SS and AC but fail to break through SPECTRE. These results align with the statements in [11]. Furthermore, under both poison rates, LS and LS^2 significantly enhance the performance of SS and SPECTRE defenses, except for AC. The combination of SPECTRE and LS^2 always attains the highest P_{rm} against adaptive attacks.

TABLE IV
RESULTS OF THE HIDDEN-SEPARATION-BASED DEFENSES AGAINST ADAP-BLEND AND ADAP-PATCH UNDER POISON RATE 1% AND 5%. THE EXPERIMENTS ARE CONDUCTED ON THE IMAGENET100 DATASET WITH VGG19-BN.

Attack	%	No Def.	SS	+LS	+LS^2	AC	+LS	+LS^2	SPECTRE	+LS	+LS^2
A-Blend $\epsilon = 1.0\%$	**P_{rm}**	(-)	47.6	**67.4**	**70.0**	31.4	31.6	31.2	44.0	76.7	**86.5**
	BA	83.0	82.9	82.2	83.0	82.5	82.5	82.9	82.2	82.4	82.4
	ASR	88.3	75.8	**19.5**	**17.1**	77.1	76.1	77.6	74.8	14.1	**5.4**
A-Patch $\epsilon = 1.0\%$	**P_{rm}**	(-)	23.1	**49.4**	**48.6**	11.5	13.7	11.5	59.5	**78.2**	**86.1**
	BA	82.5	83.5	83.1	83.1	82.5	81.6	82.3	82.5	82.5	82.5
	ASR	92.6	90.2	**70.2**	**70.3**	92.5	93.7	91.9	50.5	**4.3**	**5.9**
A-Blend $\epsilon = 5.0\%$	**P_{rm}**	(-)	77.5	**80.9**	**81.5**	57.2	58.7	59.5	**95.6**	**96.9**	**96.6**
	BA	82.4	82.4	82.3	82.2	82.0	82.2	83.2	81.9	82.4	83.2
	ASR	99.7	61.8	**55.8**	**56.4**	91.2	92.1	90.9	**4.2**	**4.3**	**4.2**
A-Patch $\epsilon = 5.0\%$	**P_{rm}**	(-)	80.9	80.9	81.3	78.9	77.9	77.9	**91.5**	**94.7**	**96.6**
	BA	82.8	82.8	83.1	82.2	82.1	82.6	81.5	82.4	82.1	82.6
	ASR	99.7	36.8	38.7	37.8	32.0	36.5	38.2	**5.8**	**4.9**	**5.2**

G. Resistance to Potential Adaptive Attacks.

The previous experiments primarily demonstrated the effectiveness of LS^2 against existing attacks. However, given our proposed defenses, adversaries may design more powerful attack strategies to circumvent our methods. The core step of our method lies in leveraging the difference in AUM values between poison and benign samples to enhance hidden separation for the corresponding defenses. A potential adaptive attack could aim to break the assumption that AUM values for poison and benign samples are different while attempting to overcome hidden-separation-based defenses. By doing so, the adaptive attack may potentially evade defense solutions like SPECTRE+LS^2.

In this section, the term "adaptive attack" specifically refers to the backdoor attack targeting both hidden-separation-based defenses and LS^2, distinguishing it from attacks such as Adap-Blend, which are directed solely against hidden-separation-based defenses.

1) Adaptive SIG Attack: Inspired Adap-Blend and Adap-Patch, we introduce a novel adaptive attack, Adap-SIG, to

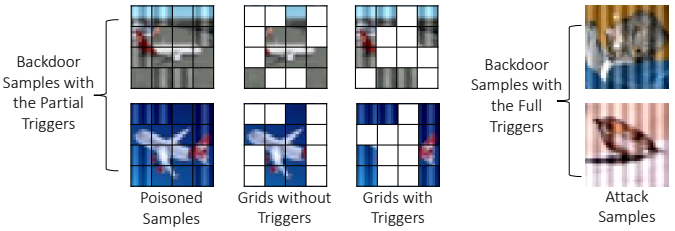


Fig. 8. In Adap-SIG, we segment a sample into 16 square pieces and randomly select m of these pieces to which we attach a sinusoidal signal (the trigger for SIG). While the samples containing the partial triggers are injected into the dataset to implant the backdoor, those with the full triggers are used to activate the backdoor during the inference stage. In this example, the mask rate m is set at 50%. When m equals 100%, the Adap-SIG would degenerate to the intial SIG attack.

challenge LS^2 defense. Adap-SIG aims to break the assumption that AUM values for poison and benign samples are different, while attempting to evade hidden-separation-based defenses. From Figs. 3f and 4f, we see that the distinguishability between the AUM of SIG poison samples and clean samples is not particularly strong. Thus, SIG can be considered as a challenge to the AUM distinguishability assumption. To bypass hidden-separation-based defenses, we adopt the trigger generation strategy used in Adap-Blend to construct the poison samples for Adap-SIG.

The strategies of Adap-Blend and Adap-Patch encompass two core steps that can effectively reduce latent separation. First, to diminish the differences between poisoned and clean representations, distributed triggers [27] are employed to generate poison samples. Second, to obscure the correlations between the trigger and the target class, the attacker injects only a very small number of poison samples into the training dataset, and not all trigger-attached samples are re-labeled to the target class.

Following these approaches, we transform SIG into Adap-SIG, where the adversary employs distributed triggers and uses only a small number of poison data to implant the backdoor. Since SIG belongs to the family of clean label attacks, the labels of all samples attached with triggers remain to be the target label. Specifically, we randomly select a percentage m of the original trigger to generate poisoned data and activate the backdoor using the entire trigger. An illustration is given in Fig. 8.

2) Results: We conducted experiments using PreActResNet-18 on CIFAR10 with poison rate $\epsilon = 0.3\%$ and started the mask rate m at 87.5% and decreased m to 37.5%. The experimental results are presented in Table V. The results demonstrate that as m decreases, Adap-SIG effectively reduces the P_{rm} for all defenses, but at the same time, its initial ASR also decreases, When $m <= 50\%$, Adap-SIG ultimately maintains an ASR of over 10% against all defenses. Among all results, SPECTRE+LS^2 achieves the highest P_{rm} and reduces the ASR to below 14%.

VI. Discussion and Limitation

In this paper, building upon the analysis of the impact of label smoothing on latent representations, we show that our proposed algorithm could enhance the distinguishability between poison and benign samples. However, our smoothing method relies on the distribution difference of AUM values for poison and benign samples. While AUM serves as a valuable indicator, the exploration for more discerning characteristics remains a promising avenue. Further research could reveal more effective strategies for detecting poisoned samples within malicious datasets. Additionally, designing particular training approaches to promote hidden separation or other discriminative characteristics warrants consideration for future defenses against backdoor threats.

TABLE V
Results of the hidden-separation-based defenses against Adap-SIG under various mask rate m. The experiments are conducted on the CIFAR10 dataset with PreActResNet-18.

m	%	No Def.	SS	+LS	+LS^2	AC	+LS	+LS^2	SPECTRE	+LS	+LS^2
87.5%	**P_{rm}**	(-)	13.3	8.7	12.7	0.0	2.0	2.0	**92.7**	2.0	**94.0**
	BA	93.5	91.4	92.6	91.7	90.4	92.2	91.7	92.5	90.8	91.1
	ASR	78.2	62.7	64.8	64.0	77.5	75.4	76.6	**0.0**	70.9	**0.2**
75.0%	**P_{rm}**	(-)	26.0	16.0	12.7	0.7	0.7	2.0	**89.3**	33.3	**94.0**
	BA	93.9	90.8	91.4	89.1	90.1	91.9	92.1	91.1	91.7	90.4
	ASR	78.2	46.2	60.0	63.1	76.3	77.5	75.2	**2.4**	41.3	**2.4**
62.5%	**P_{rm}**	(-)	22.0	14.0	14.0	2.7	2.0	2.0	**74.0**	24.7	**79.3**
	BA	93.1	90.3	89.5	91.1	91.4	92.7	91.3	91.4	92.3	91.8
	ASR	63.4	36.9	45.3	40.1	60.2	60.7	60.7	**3.6**	32.1	**2.5**
50.0%	**P_{rm}**	(-)	19.3	17.3	13.3	1.3	8.0	2.7	46.7	12.7	**60.0**
	BA	94.0	90.2	89.7	92.4	89.5	90.4	90.4	90.2	90.7	90.4
	ASR	41.7	36.8	36.7	35.1	37.9	36.3	37.9	17.2	36.3	**12.5**
37.5%	**P_{rm}**	(-)	16.0	6.7	4.7	4.0	0.7	2.0	28.7	13.3	**48.7**
	BA	93.9	90.2	90.7	90.4	90.2	90.1	89.5	89.5	90.2	89.3
	ASR	29.1	20.7	24.1	23.3	21.3	22.5	23.0	18.5	21.1	**13.5**

VII. Conclusion

In this paper, we have established a connection between label smoothing and backdoor poisoning attacks. Leveraging the influence of label smoothing on latent representations, we have devised the LS^2 training strategy. Our experimental results have demonstrated that both label smoothing and our training approach can bolster the discernibility between poison and benign samples, ultimately enhancing the efficacy of hidden-separation-based defenses, covering AC, SS, and SPECTRE. However, the effects of label smoothing may exhibit variability, and its impact on filtering poisoned data may only sometimes be positive. When combined with LS^2, the SPECTRE algorithm can eliminate the threat of various backdoor attacks, including powerful adaptive attacks.

Appendix

A. Proof of Theorem 1

Proof. For brevity, we denote $q_k(\boldsymbol{x}) = \exp\{\langle \boldsymbol{W}_k, \boldsymbol{f}(\boldsymbol{x})\rangle\}$, and

$$Q(\boldsymbol{x}) = \sum_{j=1}^{K} \exp\{\boldsymbol{F}(\boldsymbol{x})_j\} = \sum_{j=1}^{K} \exp\{\langle \boldsymbol{W}_j, \boldsymbol{f}(\boldsymbol{x})\rangle\}. \quad (8)$$

Thus, we get $p_k(\boldsymbol{x}) = q_k(\boldsymbol{x})/Q(\boldsymbol{x})$. Then, we can reform label smoothing risk (3) to

$$
\begin{aligned}
&R(\boldsymbol{F};\mathcal{D})^{LS,\alpha} \\
&= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\left[(1-\alpha)\log p_y(\boldsymbol{x}) + \frac{\alpha}{K}\sum_{k=1}^{K}\log p_k(\boldsymbol{x})\right] \\
&= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\left[\log p_y(\boldsymbol{x}) + \frac{\alpha}{K}\sum_{k=1}^{K}(\log p_k(\boldsymbol{x}) - \log p_y(\boldsymbol{x}))\right] \\
&= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\left[\log p_y(\boldsymbol{x}) + \frac{\alpha}{K}\sum_{k=1}^{K}\left(\log\frac{q_k(\boldsymbol{x})}{Q(\boldsymbol{x})} - \log\frac{q_y(\boldsymbol{x})}{Q(\boldsymbol{x})}\right)\right] \\
&= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\left[\log p_y(\boldsymbol{x}) + \frac{\alpha}{K}\sum_{k=1}^{K}\left(\log q_k(\boldsymbol{x}) - \log q_y(\boldsymbol{x})\right)\right] \\
&= -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\ \log p_y(\boldsymbol{x}) - \frac{\alpha}{K}\ \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\sum_{k=1}^{K}\langle \boldsymbol{W}_k - \boldsymbol{W}_y, \boldsymbol{f}(\boldsymbol{x})\rangle \\
&= R(\boldsymbol{F};\mathcal{D}) + \frac{\alpha}{K}\ \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}\sum_{k=1}^{K}\langle \boldsymbol{W}_y - \boldsymbol{W}_k, \boldsymbol{f}(\boldsymbol{x})\rangle
\end{aligned}
$$

□

REFERENCES

[1] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[2] N. Carlini, M. Jagielski, C. A. Choquette-Choo, D. Paleka, W. Pearce, H. Anderson, A. Terzis, K. Thomas, and F. Tramèr, "Poisoning web-scale training datasets is practical," *arXiv preprint arXiv:2302.10149*, 2023.

[3] 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," *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, 2019.

[4] D. Wu and Y. Wang, "Adversarial neuron pruning purifies backdoored deep models," in *Advances in Neural Information Processing Systems (NeurIPS 2021)*, vol. 34. Curran Associates, Inc., 2021, pp. 16 913–16 925.

[5] Y. Zeng, S. Chen, W. Park, Z. Mao, M. Jin, and R. Jia, "Adversarial unlearning of backdoors via implicit hypergradient," in *International Conference on Learning Representations*, 2022.

[6] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Neural Information Processing Systems (NeurIPS 2018)*, 2018.

[7] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," *Advances in Neural Information Processing Systems (NeurIPS 2021)*, vol. 34, pp. 14 900–14 912, 2021.

[8] A. Levine and S. Feizi, "Deep partition aggregation: Provable defenses against general poisoning attacks," in *International Conference on Learning Representations*, 2021.

[9] 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," *ArXiv*, vol. abs/1811.03728, 2018.

[10] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: defending against backdoor attacks using robust statistics," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 18–24 Jul 2021, pp. 4129–4139.

[11] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," in *The Eleventh International Conference on Learning Representations*, 2023.

[12] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" *Advances in neural information processing systems (NeurIPS 2019)*, vol. 32, 2019.

[13] S. Kornblith, T. Chen, H. Lee, and M. Norouzi, "Why do better loss functions lead to less transferable features?" in *Advances in Neural Information Processing Systems (NeurIPS 2021)*, 2021.

[14] G. Pleiss, T. Zhang, E. Elenberg, and K. Q. Weinberger, "Identifying mislabeled data using the area under the margin ranking," *Advances in Neural Information Processing Systems (NeurIPS 2020)*, vol. 33, pp. 17 044–17 056, 2020.

[15] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[16] E. Wenger, R. Bhattacharjee, A. N. Bhagoji, J. Passananti, E. Andere, H. Zheng, and B. Zhao, "Finding naturally occurring physical backdoors in image datasets," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 22 103–22 116.

[17] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, pp. 2088–2105, 2021.

[18] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 957–11 965.

[19] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[20] Y. Zeng, W. Park, Z. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, 2021, pp. 16 453–16 461.

[21] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," *ArXiv*, vol. abs/1912.02771, 2019.

[22] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in cnns by training set corruption without label poisoning," in *2019 IEEE International Conference on Image Processing*, 2019, pp. 101–105.

[23] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," in *Advances in Neural Information Processing Systems (NeurIPS 2018)*, vol. 31. Curran Associates, Inc., 2018.

[24] C. Zhu, W. R. Huang, H. Li, G. Taylor, C. Studer, and T. Goldstein, "Transferable clean-label poisoning attacks on deep neural nets," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97. PMLR, 09–15 Jun 2019, pp. 7614–7623.

[25] R. Shokri *et al.*, "Bypassing backdoor detection algorithms in deep learning," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2020, pp. 175–183.

[26] K. Doan, Y. Lao, and P. Li, "Backdoor attack with imperceptible input and latent modification," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 18 944–18 957.

[27] C. Xie, K. Huang, P. Chen, and B. Li, "DBA: distributed backdoor attacks against federated learning," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[28] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2021, pp. 1541–1558.

[29] Z. Wang, H. Ding, J. Zhai, and S. Ma, "Training with more confidence: Mitigating injected and natural backdoors during training," in *Advances in Neural Information Processing Systems (NeurIPS 2022)*, 2022.

[30] K. Huang, Y. Li, B. Wu, Z. Qin, and K. Ren, "Backdoor defense via decoupling the training process," in *International Conference on Learning Representations*, 2022.

[31] W. Chen, B. Wu, and H. Wang, "Effective backdoor defense by exploiting sensitivity of poisoned samples," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.

[32] E. Chou, F. Tramèr, and G. Pellegrino, "Sentinet: Detecting localized universal attacks against deep learning systems," *2020 IEEE Security and Privacy Workshops (SPW)*, pp. 48–54, 2020.

[33] 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 *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, p. 113–125.

[34] P. Zhao, P.-Y. Chen, P. Das, K. N. Ramamurthy, and X. Lin, "Bridging mode connectivity in loss landscapes and adversarial robustness," in *International Conference on Learning Representations*, 2020.

[35] Y. Li, N. Koren, L. Lyu, X. Lyu, B. Li, and X. Ma, "Neural attention distillation: Erasing backdoor triggers from deep neural networks," *ArXiv*, vol. abs/2101.05930, 2021.

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the*

IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.
[37] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. Interspeech 2017*, 2017, pp. 523–527.
[38] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8697–8710.
[39] M. Peng, Z. Xiong, M. Sun, and P. Li, "Label-smoothed backdoor attack," *arXiv preprint arXiv:2202.11203*, 2022.
[40] M. Lukasik, S. Bhojanapalli, A. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" in *International Conference on Machine Learning*. PMLR, 2020, pp. 6448–6458.
[41] C.-B. Zhang, P.-T. Jiang, Q. Hou, Y. Wei, Q. Han, Z. Li, and M.-M. Cheng, "Delving deep into label smoothing," *Trans. Img. Proc.*, vol. 30, p. 5984–5996, 2021.
[42] C. Meister, E. Salesky, and R. Cotterell, "Generalized entropy regularization or: There's nothing special about label smoothing," in *Annual Meeting of the Association for Computational Linguistics*, 2020.
[43] J. Wei, H. Liu, T. Liu, G. Niu, and Y. Liu, "To smooth or not? when label smoothing meets noisy labels," in *International Conference on Machine Learning*, 2021.
[44] A. Ghoshal, X. Chen, S. Gupta, L. Zettlemoyer, and Y. Mehdad, "Learning better structured representations using low-rank adaptive label smoothing," in *International Conference on Learning Representations*, 2021.
[45] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.
[46] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.
[47] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks*, no. 1288, 2013.
[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, jun 2016, pp. 770–778.
[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
[50] S. Chai and J. Chen, "One-shot neural backdoor erasing via adversarial weight masking," in *Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS 2022)*, 2022.
[51] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

Supplemental Material for "LS^2: Boosting Hidden Separation for Backdoor Defense with Learning Speed-driven Label Smoothing"

Jie Peng, Hongwei Yang, Hui He, *Member, IEEE,* Jing Zhao, HaoYu He, Hengji Dong, Weizhe Zhang, *Senior Member, IEEE*

S.I. Social Impact

Training a well-performing deep learning network often involves the common practice of collecting substantial datasets from diverse sources. However, some untrustworthy sources may exploit this demand by injecting poisoned data into the collection, aiming to implant backdoors into the model. To ensure model security, the most direct approach is to filter out poisoned data from the dataset as much as possible, thereby eliminating the backdoor threat at its source. The hidden-separation-based defense is an example of such filtering algorithms. We propose the LS^2 training method with minimal modifications to effectively enhance existing hidden-separation-based defenses. Since LS^2 can essentially be viewed as a training strategy to prevent overfitting, our method holds the potential to enhance the model's generalization capabilities. We hope our work can pave the way for more advanced defenses against backdoor threats in the future.

S.II. Implementations Details

1) Training Settings: Our deep learning training algorithm is implemented using PyTorch. Throughout all training processes, we employed the stochastic gradient descent (SGD) optimization method with a batch size of 128. We set the initial learning rate to 0.01 and decayed it using the cosine annealing strategy [*S1*]. For the CIFAR-10 and CIFAR-100 datasets, we trained for a total of 100 epochs. For the GTSRB dataset, we trained for 50 epochs. In the case of the ImageNet100 dataset, we trained for 200 epochs. All experiments were run on one Ubuntu 18.04 server equipped with two NVIDIA RTX A4000 GPUs. Our implementation is mainly based on BackdoorBench [*S2*].

2) Attacks Settings: On the CIFAR10, CIFAR100 and ImageNet100 datasets, the target label for all backdoor attacks is class 0, corresponding to the specific class names "airplane", "apple" and "tench" . On the GTSRB dataset, the target label for all backdoor attacks is class 4, corresponding to the specific class name "70-speed".

In Section V, for CIFAR-10, all attacks except for BadNet and ISSBA have a poison rate of $\epsilon = 0.3\%$ (150 poison samples). Since the ASR values for BadNet and ISSBA at $\epsilon = 0.003$ are below 10%, they are not considered successful backdoor attacks. Therefore, we set the poison rate for these two attacks to $\epsilon = 1\%$ (500 poison samples). In the case of GTSRB, the poison rate for all attacks is $\epsilon = 0.4\%$ (157 poison samples). Due to the low success rates of most backdoor attacks on the CIFAR-100 dataset when the poisoning rate ϵ is set to a low value, we choose to use $\epsilon = 1\%$ (500 poison samples). However, CIFAR-100 has only 500 training samples per class. Therefore, at this poison rate, all data under the target label of clean label attacks would be replaced with poison samples, which does not align with the basic premise of backdoor attacks. Additionally, both LC and SIG attacks have poor attack performance at lower poison rates, with success rates below 40%. Therefore, we opt not to conduct experiments with clean label attacks on CIFAR-100.

The comprehensive implementation details for each backdoor attack are as follows: The examples of poisoned data generated by various label-flipping attacks on CIFAR10 and ImageNet100 are presented in Figures S1 and S2 As shown

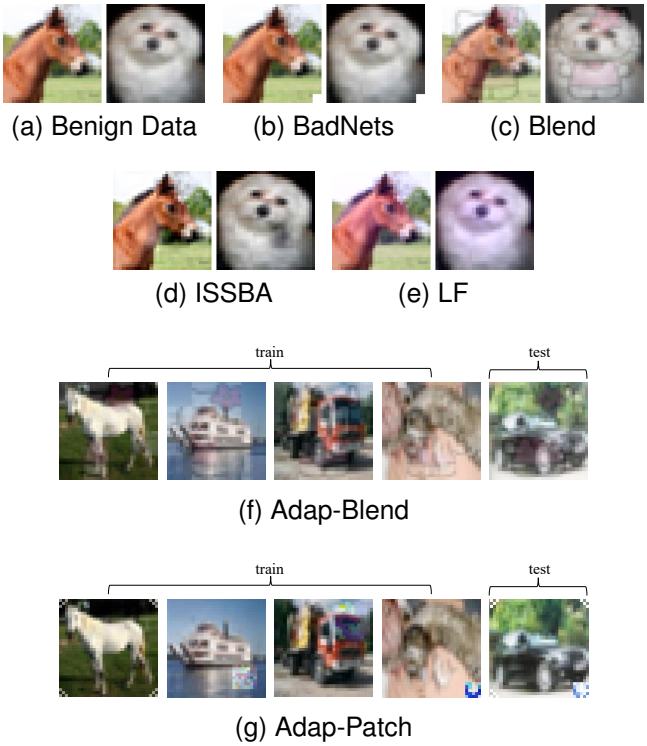


Fig. S1. Examples of poisoned data generated by classical and adaptive label-flipping attacks on CIFAR10.

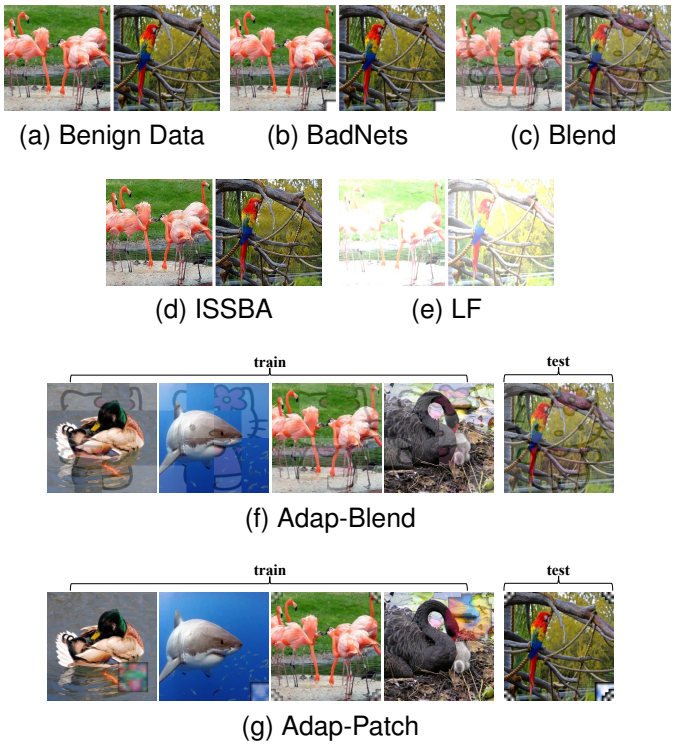


Fig. S2. Examples of poisoned data generated by classical and adaptive label-flipping attacks on ImageNet100.

in Figure S1, the classical label-flipping attacks first injects triggers into samples and then relabels them as the target class "airplane". BadNets attacks [*S3*] employs a 3×3 white square placed at the bottom right corner as the trigger pattern. Blended attack [*S4*] poisons the data by introducing a Hello Kitty image trigger. We implement the blended injection strategy, denoted as $\alpha t + (1-\alpha)x$, to incorporate the trigger t into the benign sample x with a value of $\alpha = 0.2$. ISSBA [*S5*] utilizes the StegaStemp algorithm [*S6*] to generate specific triggers for poison samples across various classes. LF [*S7*] employs frequency domain analysis and optimization algorithms to create poison samples.

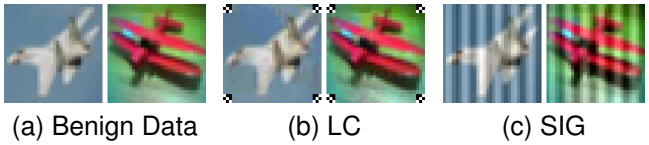


Fig. S3. Examples of poisoned data generated by clean label attacks on CIFAR10.

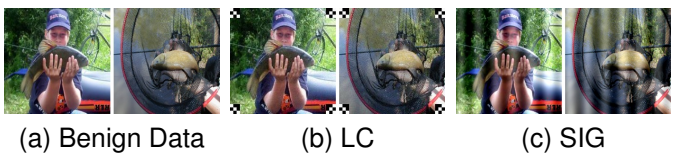


Fig. S4. Examples of poisoned data generated by clean label attacks on ImageNet100.

Adaptive Attacks [*S8*] represent an enhanced iteration of the previously mentioned attacks. In addition to injecting poisoned data into the dataset, this category of attacks also inserts payload data. The labels of payload data are not modified, but they are embedded with triggers. In our experiments, we use an equal number of payload data points as poison samples. As illustrated in Figs. 1f and 1g, Adap-Blend employs the same trigger, "Hello Kitty", as Blend to contaminate the samples. However, when generating the poisoned data, Adap-Blend randomly selects a portion of the trigger for conducting the poisoning attack. During the testing phase, Adap-Blend activates the backdoor using the entire trigger. Similarly, Adap-Patch utilizes a set of four triggers to produce poisoned data. Each individual poisoned training sample is embedded with a single trigger. Nevertheless, during the attack phase, two triggers are employed to induce the neural network to output the target label.

As shown in Figures S3 and S4, the clean label adversary poisons samples belonging to the target class only. LC attacks [*S9*] utilize a 3×3 checkboard pattern positioned in the four corners as the trigger. To establish a link between the trigger and the target label, LC attacks initially employ the Projection Gradient Descent (PGD) method to introduce adversarial perturbations to the images before incorporating the trigger. On the other hand, SIG attacks [*S10*] utilize a sinusoidal signal that is seamlessly integrated into the image as the trigger.

Defenses Settings: For defenses requiring clean samples, including NC, ANP, AWM, and I-BAU, we allocate 5% of the clean training samples from each dataset to the defender. For any other specific settings not mentioned below, we adhere to the default settings outlined in their publications or public implementation.

As we employed the original mathematical symbols from each publications, please be aware that some of the mathematical symbols and terms used below may conflict with each other or with certain expressions in our paper. All symbols and terms below correspond only to the parameters in their respective papers.

- NC [*S11*]: We set the threshold of the Anomaly Index at 2 for all the datasets. For models with an Anomaly Index higher than 2 (marked as attacked), we conduct the unlearning procedure for 40 epochs, utilizing 5% of the training data and applying the reversed trigger to 20% of these samples.
- ABL [*S12*]: During training, for CIFAR10, CIFAR100, and GTSRB, we set the number of early tuning epochs $T_{te} = 20$, the fine-tuning epochs $T_{fe} = 60$, and the unlearning epochs $T_{ue} = 20$. For ImageNet100, we set $T_{te} = 40$, $T_{fe} = 120$, and $T_{ue} = 4$. In the tuning stage, we set the isolation rate $p = 0.01$, and the loss threshold $\gamma = 0.5$. And the unlearning rate is set to be $5e^{-4}$.
- ANP [*S13*]: We set the learning rate to 0.2 for optimizing the neuron mask with SGD. After optimization, neurons with a mask value smaller than the threshold of 0.2 are pruned. Additionally, we set the tradeoff coefficient α to 0.2 and the perturbation budget ϵ to 0.4 for a total of 2000 iterations.

- AWM [*S14*]: Although AWM is an improved version of ANP, there are significant differences in the default parameters between ANP and AWM, as addressed in AWM. The outer optimization is conducted with Adam and a learning rate of 0.001, while the inner optimization is conducted with SGD using a learning rate of 0.01. We perform 10 steps of outer optimization after every 10 steps of inner optimization in each epoch. We use the default hyperparameter settings: $\alpha = 0.9$, $\beta = 0.1$, $\gamma = 1e^{-8}$, and $\tau = 1000$.
- I-BAU [*S15*]: We follow the default settings provided in the public implementation of I-BAU. We use Adam with a learning rate of $1e^{-3}$ to optimize the outer loop. And, we set the maximum number of unlearning rounds to 5 and the maximum number of fixed-point iterations to 5.
- DBD [*S16*]: For all datasets, we performed self-supervised learning for 100 epochs. Moreover, we trained the connected layers for 10 epochs using SCE loss for warm-up. We set the filtering rate α to 50% in all cases, and the temperature of the NT-Xent loss was set to 0.5.
- D-BR [*S17*]: During sample distinguishment, we set $e_t = 2$, $e_f = 10$, $\alpha_c = 0.2$, and $\alpha_p = 0.05$. After training the backdoored model for 200 epochs, we spend 20 epochs to perform the fine-tuning and unlearning procedures.
- AC [*S18*]: Following the settings in SPECTRE [*S19*], we allow the defenders of AC, SS, and SPECTRE to be equipped with an oracle that has knowledge of the number of poison samples ($\varepsilon \cdot n$). Before performing K-means, we reduce the dimensionality of representations to 100 using PCA.
- SS [*S20*]: We make the defender remove $1.5\varepsilon n$ suspected samples for each class.
- SPECTRE [*S19*]: We remove $1.5\varepsilon n$ suspected samples only from the class with the highest QUE-score. We set the base value of α to 4 for assessing the outlier score τ. Moreover, we set $k_{\text{max}} = 100$ to find out the effective dimension k for computing the outlier score.

S.III. Additional T-SNE Visualization

In Figure S5, we plot the T-SNE visualization for samples generated from other backdoor attacks on CIFAR10. When we train the model with vanilla cross-entropy, there is a visible separation between poison and benign samples only for ISSBA Blend and SIG attacks. When extracting the representation from models trained with label smoothing, we get better latent separability except for LF attacks. Moreover, label smoothing also tightens the cluster of poison samples in latent space. LS^2 has a similar effect for poison and benign separation as label smoothing but boosts the hidden separation for all attacks.

S.IV. AUM Values

In this section, we first present the AUM values associated with various attacks, followed by a detailed discussion on the relationship between these AUM values and cross-entropy.

1) AUM Values on More Datasets: Figures 6-9 give the comprehensive AUM values and the corresponding density on GTSRB, and CIFAR100. On GTSRB and CIFAR100, we can obtain similar observations as CIFAR10. However, on tiny datasets like GTSRB, the AUM values of poisoned samples increase more rapidly. Meanwhile, as shown in Figure S6f, in the later stages of training, the AUM values of SIG poison samples are almost identical to the benign ones. There is disparity only before the 10^{th} epoch. For such tiny datasets, the epoch T for re-smoothing can be set to a lower value, for example, $T = 2$.

2) AUM versus Cross-entropy Loss: In this paper, we utilize AUM as a metric to assess the "learning speed" of the model for each sample. As discussed earlier, we observe that most poison samples, except for SIG, exhibit a "Hard-to-learn" pattern. When comparing the results of hidden-separation-based defenses with other types of defenses, we also observe that this "Hard-to-learn" phenomenon may contrast with the experimental observations obtained in the ABL paper [*S12*].

In ABL, cross-entropy loss is employed to measure the "learning speed" for each sample, which only involves the model's outputs corresponding to the labels annotated in the dataset. The authors of ABL found that the cross-entropy training loss on the poison portion drops abruptly in the early epochs of training. Such phenomenon suggests that the backdoor samples are much easier to learn compared to the clean ones, which seems to contradict the observation in our paper.

To better understand this conflict, it's essential to note the differences between AUM and the cross-entropy loss values. Unlike cross-entropy loss, AUM considers not only the model's outputs corresponding to the data labels but also takes into account the runner-up outputs of the model (see (5)). For common poison samples, the model's probability output is highest for the target class, leading to a small loss value, as observed in ABL. For samples from label-flipping backdoor attacks, the model outputs a large probability for the original classes of those samples, akin to the noisy label problem [*S21*], resulting in small AUM values. Furthermore, samples from clean label backdoor attacks, which do not undergo label change but have their original patterns perturbed, tend to exhibit higher AUM values.

S.V. Additional Defense Results

Full numerical results of PreActResNet-18 and VGG19-BN for various defenses against backdoor attacks are presented in Tables SI-III.

In line with Adap-Blend and Adap-Patch, all attacks in our study were conducted with a low poison rate, as we discussed in Section S.II. Consequently, some of the ASR values of attacks may be lower than those reported in their publications. Furthermore, previous studies [*S8*, *S19*] have indicated that defending against backdoor attacks is more challenging at lower poison rates than at higher ones. To address this challenge, we opted to conduct our experiments under such conditions.

According to our definition, P_{rm} is actually the recall metric in traditional machine learning. Following SPECTRE and SS, here, we only report the recall (P_{rm}). Moreover, as mentioned in Section S.II, please note that the SS and SPECTRE al-

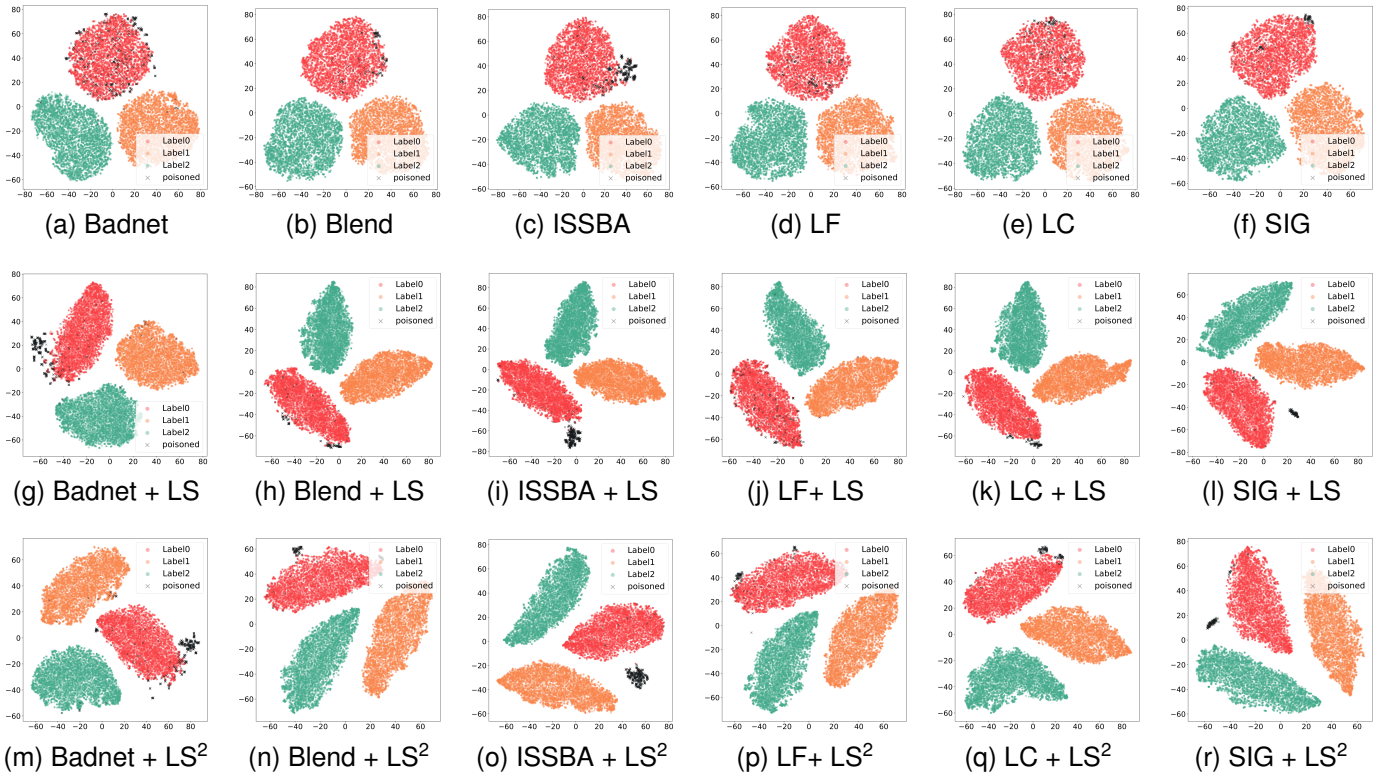


Fig. S5. T-SNE visualization of penultimate layer's activations of backdoor attacks BadNets, Blend, ISSBA and SIG on CIFAR10 with different label smoothing settings. The representations in the first row are extracted from models trained with vanilla cross-entropy.

gorithms filter out a fixed number of samples as the poison samples, which allows us to derive precision from recall.

1) Results of PreActResNet-18: From Table SI, we can observe that on CIFAR10, label smoothing and LS^2 primarily enhance the effectiveness of SS and SPECTRE defenses. These two training strategies have almost no enhancing effect on AC defense. In the presence of very few poisoned samples, LS^2 allows SS defense to defend against attacks such as BadNets, ISSBA, and adaptive attacks that it previously could not handle. Except for ISSBA and LC attacks, LS^2 boosts the P_{rm} values of SPECTRE defense to their highest in most cases.

On the GTSRB dataset, label smoothing and LS^2 aid in improving the P_{rm} values for AC, SS, and SPECTRE defenses, as presented in Table SII. These two methods enable all three defenses to eliminate the threat of adaptive attacks. However, their effect on AC and SS defense is not stable for other attacks, and they cannot guarantee comprehensive defense against all types of backdoor attacks.

The results for CIFAR100 are shown in Table SIII. Similar to CIFAR10, we observe that label smoothing and LS^2 primarily enhance the effectiveness of SS and SPECTRE defenses. However, it is worth noting that their impact on the SPECTRE algorithm is particularly significant. In contrast to CIFAR10 and GTSRB, we found that the vanilla SPECTRE algorithm on CIFAR100 can only defend against ISSBA and LF backdoor attacks. Still, after applying label smoothing and LS^2, the P_{rm} values for SPECTRE substantially increase. The combination of LS^2 and SPECTRE is capable of defending against all types of attacks.

REFERENCES

[S1] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[S2] B. Wu, H. Chen, M. Zhang, Z. Zhu, S. Wei, D. Yuan, and C. Shen, "Backdoorbench: A comprehensive benchmark of backdoor learning," in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[S3] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[S4] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[S5] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[S6] M. Tancik, B. Mildenhall, and R. Ng, "Stegastamp: Invisible hyperlinks in physical photographs," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2114–2123, 2020.

[S7] Y. Zeng, W. Park, Z. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, 2021, pp. 16 453–16 461.

[S8] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," in *The Eleventh International Conference on Learning Representations*, 2023.

[S9] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," *ArXiv*, vol. abs/1912.02771, 2019.

[S10] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in cnns by training set corruption without label poisoning," in *2019 IEEE International Conference on Image Processing*, 2019, pp. 101–105.

[S11] 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," *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, 2019.

[S12] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," *Advances in Neural Information Processing Systems (NeurIPS 2021)*, vol. 34, pp. 14 900–14 912, 2021.

[S13] D. Wu and Y. Wang, "Adversarial neuron pruning purifies backdoored deep models," in *Advances in Neural Information Processing Systems (NeurIPS 2021)*, vol. 34. Curran Associates, Inc., 2021, pp. 16 913–16 925.

[S14] S. Chai and J. Chen, "One-shot neural backdoor erasing via adversarial weight masking," in *Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS 2022)*, 2022.

[S15] Y. Zeng, S. Chen, W. Park, Z. Mao, M. Jin, and R. Jia, "Adversarial unlearning of backdoors via implicit hypergradient," in *International Conference on Learning Representations*, 2022.

[S16] K. Huang, Y. Li, B. Wu, Z. Qin, and K. Ren, "Backdoor defense via decoupling the training process," in *International Conference on Learning Representations*, 2022.

[S17] W. Chen, B. Wu, and H. Wang, "Effective backdoor defense by exploiting sensitivity of poisoned samples," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.

[S18] 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," *ArXiv*, vol. abs/1811.03728, 2018.

[S19] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: defending against backdoor attacks using robust statistics," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 18–24 Jul 2021, pp. 4129–4139.

[S20] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Neural Information Processing Systems (NeurIPS 2018)*, 2018.

[S21] G. Pleiss, T. Zhang, E. Elenberg, and K. Q. Weinberger, "Identifying mislabeled data using the area under the margin ranking," *Advances in Neural Information Processing Systems (NeurIPS 2020)*, vol. 33, pp. 17 044–17 056, 2020.

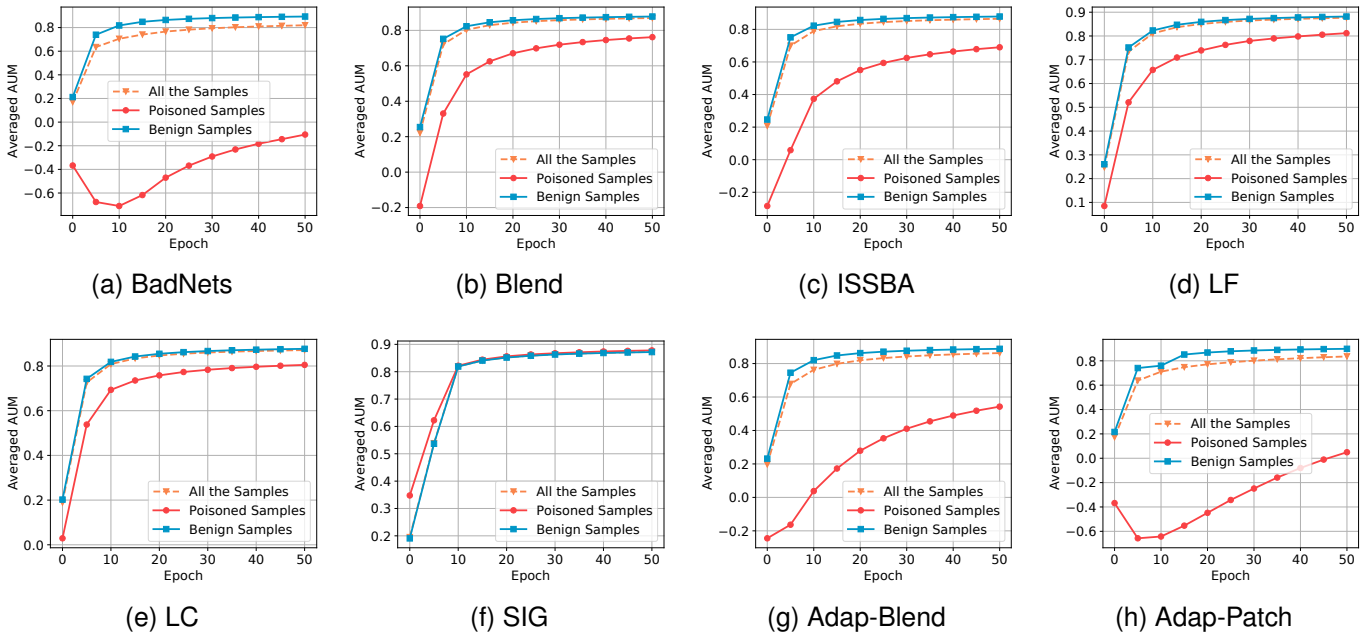


Fig. S6. The comparison of AUM value between benign and poison samples of PreActResNet-18 during training on GTSRB.

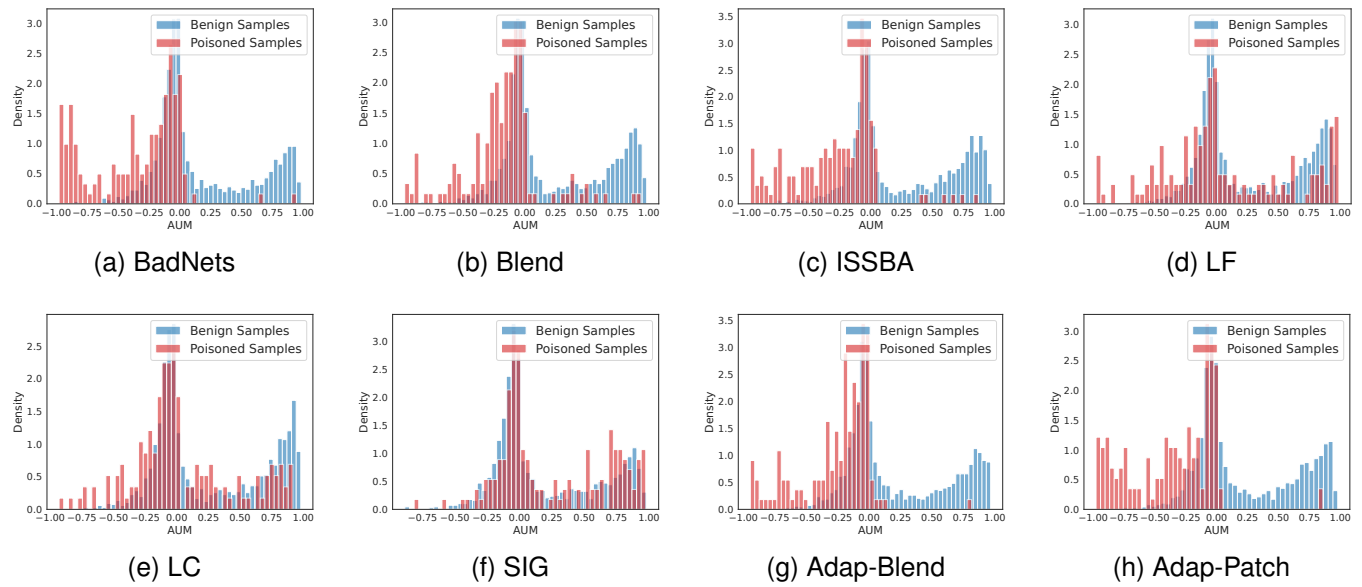


Fig. S7. The comparison of AUM value density between benign and poison samples of PreActResNet-18 at 2nd epoch on GTSRB.

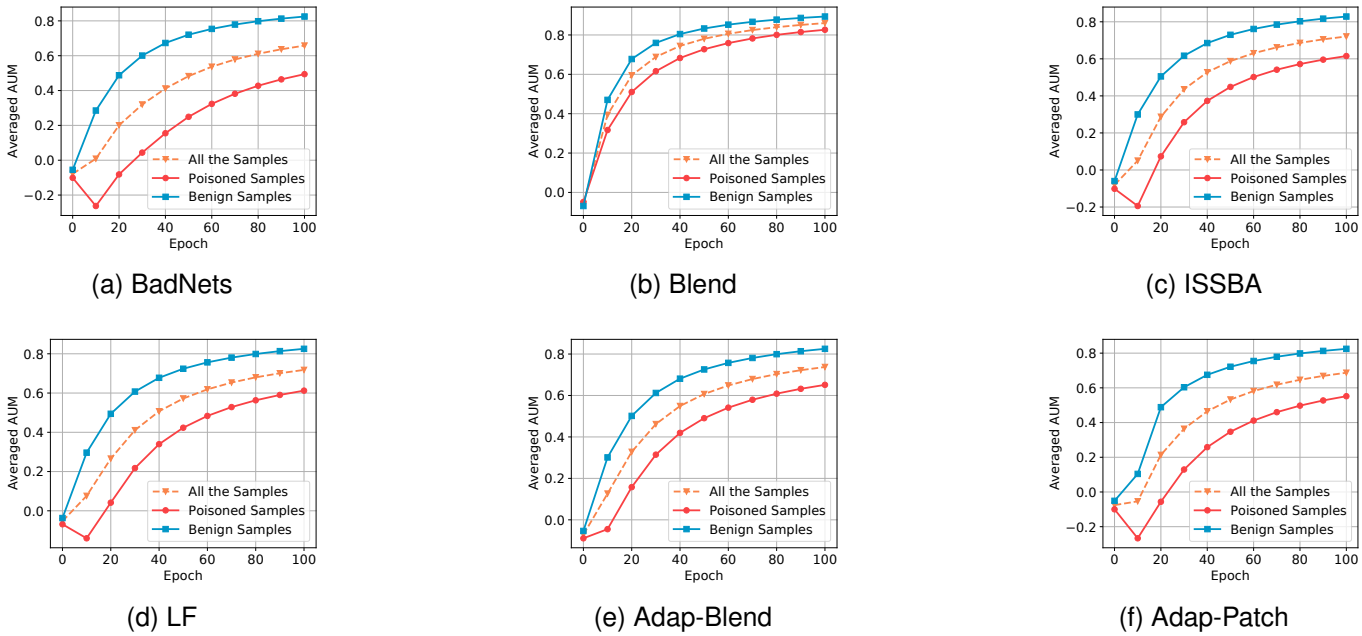


Fig. S8. The comparison of AUM value between benign and poison samples of PreActResNet-18 during training on CIFAR100.

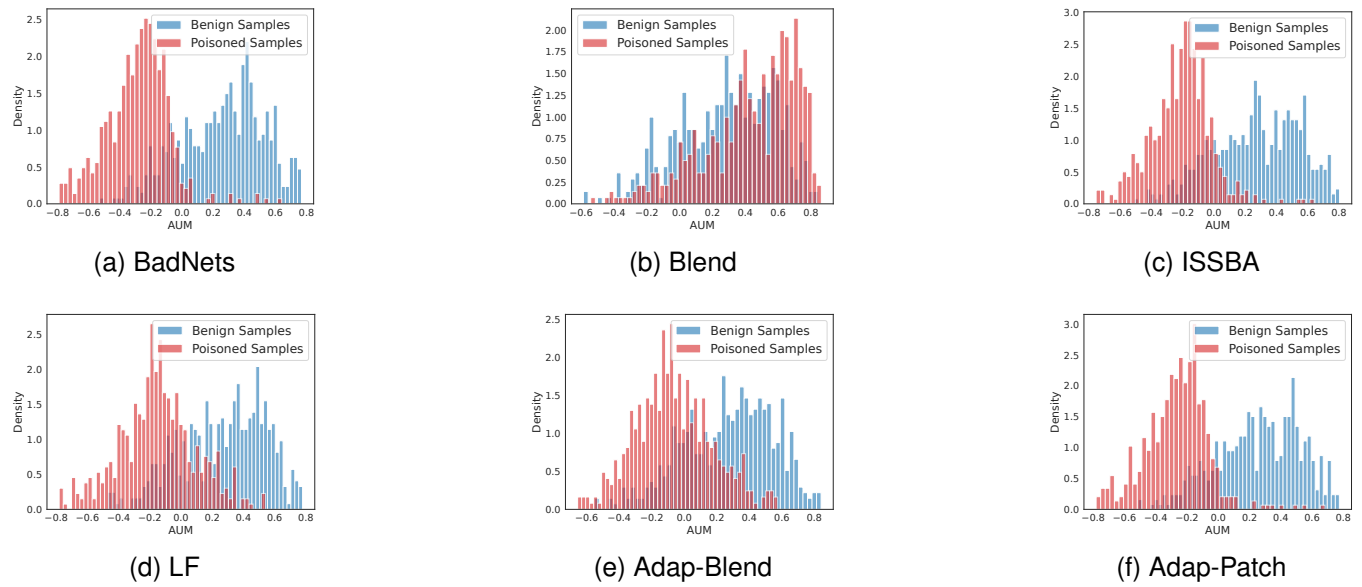


Fig. S9. The comparison of AUM value density between benign and poison samples of PreActResNet-18 at 10^{th} epoch on CIFAR100.

TABLE SI
POISON SAMPLES REMOVAL PERFORMANCE COMPARISON OF PREACTRESNET-18 WITH VARIOUS HIDDEN SEPARABILITY BASED DEFENSES AGAINST ATTACKS ON CIFAR10.

Attacks →	(%)	**BadNets**	**Blend**	**ISSBA**	**LF**	**LC**	**SIG**	**A-Blend**	**A-Patch**
No Defense	**ASR**	88.7	86.1	71.2	69.8	87.1	77.5	71.9	78.8
SS	$\mathbf{P_{rm}}$	25.6	8.6	12.6	0.0	18.6	9.3	0.7	5.3
	ASR	43.1	80.2	69.4	69.9	68.1	67.5	71.8	77.4
SS + LS	$\mathbf{P_{rm}}$	50.0	**20.0**	31.2	6	**50.6**	9.3	33.3	40 .0
	ASR	10.5	**71.4**	36.1	68.1	**20.1**	68.3	52.5	44.6
SS + LS^2	$\mathbf{P_{rm}}$	**71.8**	5.0	**78.4**	5.3	29.3	6.0	**72.7**	**63.3**
	ASR	**4.6**	86.2	**2.4**	68.2	45.4	69.4	**2.7**	**5.1**
AC	$\mathbf{P_{rm}}$	11.8	3.3	11.8	0	0	0.7	0.0	0.7
	ASR	45.4	84.1	69.1	69.1	86.9	73.5	71.5	77.2
AC + LS	$\mathbf{P_{rm}}$	15.0	5.3	6.6	2.6	6.6	1.2	3.3	1.3
	ASR	45.5	84.4	70.2	68.8	86.1	73.2	70.7	77.6
AC + LS^2	$\mathbf{P_{rm}}$	8.2	2.0	8.0	4.0	0.0	2.7	0.0	7.3
	ASR	47.3	84.6	69.3	68.9	86.9	73.1	71.2	71.3
SPECTRE	$\mathbf{P_{rm}}$	85.4	74.0	**98.6**	83.3	**100.0**	**100.0**	2.6	4
	ASR	4.6	7.1	**2.1**	5.1	**0.1**	**0.5**	71.2	77.2
SPECTRE + LS	$\mathbf{P_{rm}}$	**90.0**	84.7	97.2	18.6	**98.6**	2.6	66.7	**78.0**
	ASR	**2.4**	2.5	2.3	40.2	**0.1**	73.3	13.7	**4.3**
SPECTRE + LS^2	$\mathbf{P_{rm}}$	**89.8**	**86.7**	97.2	**84.6**	**97.3**	**100.0**	**82.0**	**80.0**
	ASR	**3.2**	**2.1**	3.1	**2.4**	**0.1**	**0.5**	**2.4**	**2.5**

TABLE SII
POISON SAMPLES REMOVAL PERFORMANCE COMPARISON WITH OF PREACTRESNET-18 VARIOUS HIDDEN SEPARABILITY BASED DEFENSES AGAINST ATTACKS ON GTSRB.

Attacks →	(%)	**BadNets**	**Blend**	**ISSBA**	**LF**	**LC**	**SIG**	**A-Blend**	**A-Patch**
No Defense	**ASR**	73.81	89.2	71.8	90.3	53.2	58.3	78.0	55.3
SS	$\mathbf{P_{rm}}$	81.5	**71.3**	**72.6**	37.6	23.6	33.8	48.4	26.8
	ASR	4.7	**23.8**	**7.9**	81.5	47.7	40.3	66.3	49.9
SS + LS	$\mathbf{P_{rm}}$	62.4	70.1	56.1	**60.5**	29.3	36.3	89.2	84.1
	ASR	11.3	21.9	16.7	**24.2**	46.2	36.2	6.2	3.4
SS + LS^2	$\mathbf{P_{rm}}$	**91.1**	63.1	63.1	45.2	**42.7**	33.8	**94.3**	**96.6**
	ASR	**1.24**	24.7	10.4	80.1	**15.4**	40.3	**4.3**	**2.5**
AC	$\mathbf{P_{rm}}$	75.8	69.4	**75.8**	29.3	8.3	7.0	45.9	29.3
	ASR	6.5	25.5	**7.7**	84.2	52.8	58.1	69.8	47.1
AC + LS	$\mathbf{P_{rm}}$	**87.9**	**80.9**	47.8	38.2	6.4	14.0	**84.7**	66.9
	ASR	**1.4**	**6.7**	23.1	82.5	51.1	52.3	**4.8**	14.6
AC + LS^2	$\mathbf{P_{rm}}$	**87.9**	65.6	64.3	**41.4**	**19.10**	5.1	**86.6**	**86.0**
	ASR	**1.4**	25.1	10.7	**82.3**	**47.6**	58.1	**4.4**	**3.6**
SPECTRE	$\mathbf{P_{rm}}$	**93.0**	89.2	91.1	**98.1**	73.9	**84.8**	42.7	31.2
	ASR	**0.1**	7.5	1.3	**0.4**	8.8	**4.7**	44.7	35.6
SPECTRE + LS	$\mathbf{P_{rm}}$	**96.2**	**98.7**	**98.1**	**100.0**	**82.2**	68.2	**92.4**	**97.5**
	ASR	**0.2**	**2.5**	**1.1**	**0.5**	**1.8**	6.5	**4.5**	**2.6**
SPECTRE + LS^2	$\mathbf{P_{rm}}$	**95.5**	**94.9**	**95.5**	**99.4**	**82.8**	79.6	**96.8**	**98.1**
	ASR	**0.1**	**2.5**	**0.5**	**0.4**	**1.8**	4.55	**4.4**	**2.5**

TABLE SIII
POISON SAMPLES REMOVAL PERFORMANCE COMPARISON OF PREACTRESNET-18 WITH VARIOUS HIDDEN SEPARABILITY BASED DEFENSES AGAINST ATTACKS ON CIFAR100.

Attacks →	(%)	**BadNets**	**Blend**	**ISSBA**	**LF**	**A-Blend**	**A-Patch**
No Defense	**ASR**	71.2	90.8	55.8	58.8	88.1	91.7
SS	$\mathbf{P_{rm}}$	10.0	10.2	51.6	53.6	38.6	42.5
	ASR	71.0	88.4	2.7	17.3	44.7	50.2
SS + LS	$\mathbf{P_{rm}}$	**48.2**	**56.0**	**54.8**	**57.4**	**50.8**	**47.6**
	ASR	**20.1**	**80.4**	**3.4**	**13.1**	**46.1**	**49.0**
SS + LS^2	$\mathbf{P_{rm}}$	**44.8**	**57.8**	51.0	48.2	**47.4**	**48.6**
	ASR	**23.1**	**79.3**	2.7	17.3	**46.1**	**47.6**
AC	$\mathbf{P_{rm}}$	**84.0**	10.8	3.2	**80.2**	4.8	9.0
	ASR	**2.1**	88.8	53.4	**2.7**	88.3	90.3
AC + LS	$\mathbf{P_{rm}}$	73.2	10.2	6.6	14.8	7.4	**26.6**
	ASR	5.1	87.9	50.3	57.3	87.7	**80.4**
AC + LS^2	$\mathbf{P_{rm}}$	79.8	**24.4**	6.0	10.8	6.8	9.0
	ASR	2.3	**85.1**	50.6	57.4	87.6	89.9
SPECTRE	$\mathbf{P_{rm}}$	10.0	8.2	70.4	**76.2**	44.8	45.2
	ASR	70.2	88.6	0.4	**2.9**	46.8	45.5
SPECTRE + LS	$\mathbf{P_{rm}}$	**80.8**	58.8	**85.4**	72.4	73.8	70.8
	ASR	**2.3**	78.2	**0.1**	3.1	6.3	0.1
SPECTRE + LS^2	$\mathbf{P_{rm}}$	**80.4**	**77.8**	**86.4**	**78.6**	**82.6**	**84.8**
	ASR	**2.3**	**6.1**	**0.1**	**2.9**	**6.2**	**0.0**