Fisher Information guided Purification against Backdoor Attacks

Nazmul Karim*
nazmul.karim170@gmail.com
University of Central Florida
Orlando, Florida, USA

Abdullah Al Arafat*
aalaraf@ncsu.edu
North Carolina State University
Raleigh, North Carolina, USA

Adnan Siraj Rakin
arakin@binghamton.edu
Binghamton University (SUNY)
Binghamton, New York, USA

Zhishan Guo
zguo32@ncsu.edu
North Carolina State University
Raleigh, North Carolina, USA

Nazanin Rahnavard
nazanin.rahnavard@ucf.edu
University of Central Florida
Orlando, Florida, USA

Abstract

Studies on backdoor attacks in recent years suggest that an adversary can compromise the integrity of a deep neural network (DNN) by manipulating a small set of training samples. Our analysis shows that such manipulation can make the backdoor model converge to a *bad local minima*, i.e., sharper minima as compared to a benign model. Intuitively, the backdoor can be purified by re-optimizing the model to smoother minima. However, a naïve adoption of any optimization targeting smoother minima can lead to sub-optimal purification techniques hampering the clean test accuracy. Hence, to effectively obtain such re-optimization, inspired by our novel perspective establishing the connection between backdoor removal and loss smoothness, we propose *Fisher Information guided Purification (FIP)*, a novel backdoor purification framework. Proposed FIP consists of a couple of novel regularizers that aid the model in suppressing the backdoor effects and retaining the acquired knowledge of clean data distribution throughout the backdoor removal procedure through exploiting the knowledge of *Fisher Information Matrix (FIM)*. In addition, we introduce an efficient variant of FIP, dubbed as *Fast FIP*, which reduces the number of tunable parameters significantly and obtains an impressive runtime gain of almost 5×. Extensive experiments show that the proposed method achieves state-of-the-art (SOTA) performance on a wide range of backdoor defense benchmarks: 5 different tasks—*Image Recognition, Object Detection, Video Action Recognition, 3D point Cloud, Language Generation*; 11 different datasets including *ImageNet, PASCAL VOC, UCF101*; diverse model architectures spanning both CNN and vision transformer; 14 different backdoor attacks, e.g., *Dynamic, WaNet, LIRA, ISSBA, etc.* Our code is available in this **GitHub Repository**.

CCS Concepts

• **Computing Methodologies** → Machine Learning; • **Security and Privacy** → Software and application security.

Keywords

maching learning, poisoning, backdoor purification, smoothness

ACM Reference Format:
Nazmul Karim, Abdullah Al Arafat, Adnan Siraj Rakin, Zhishan Guo, and Nazanin Rahnavard. 2024. Fisher Information guided Purification against Backdoor Attacks. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3658644.3690250

1 Introduction

Training a deep neural network (DNN) with a fraction of poisoned or malicious data is often security-critical since the model can successfully learn both clean and adversarial tasks equally well. This is prominent in scenarios where one outsources the DNN training to a vendor. In such scenarios, an adversary can mount backdoor attacks [11, 24] by poisoning a portion of training samples so that the model will classify any sample with a *particular trigger* or *pattern* to an adversary-set label. Whenever a DNN is trained in such a manner, it becomes crucial to remove the effect of a backdoor before deploying it for a real-world application. In recent times, several attempts have been made [39, 45, 69, 72, 81, 83] to tackle the backdoor issue in DNN training. Defense techniques such as fine-pruning (FP) [45] aim to prune vulnerable neurons affected by the backdoor. Most of the recent backdoor defenses can be categorized into two groups based on the intuition or perspective they are built on. They are (i) *pruning based defense* [45, 72, 81]: some weights/channel/neurons are more vulnerable to backdoor than others. Therefore, pruning or masking bad neurons should remove the backdoor. (ii) *trigger approximation based defense* [9, 75]: recovering the original trigger pattern and fine-tuning the model with this trigger attached to samples and corresponding benign labels would remove the backdoor. However, they require computationally expensive adversarial search approaches to find the backdoor-sensitive model parameters or reverse-engineered backdoor triggers, which makes efficient post-training model purification challenging.

In contrast to the expensive adversarial search and reverse-engineering methods, general-purpose fine-tuning can moderately remove the effects of backdoors and has been adopted as a component in existing defenses [40, 45]. However, adopting vanilla fine-tuning is challenging with limited benign data [71] and cannot remove strong backdoor attacks, e.g., Blend [11] and smooth low frequency (LF) trigger [76]. Recently, Zhu et al., [83] proposed an

[*] Both authors contributed equally to this work.

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3690250

enhanced fine-tuning method to effectively remove backdoors following their observation aligning with earlier results that neurons with large norms are responsible for backdoors. However, their enhancement is based on a general-purpose optimizer, SAM [22], which affects the runtime of the purification process and accuracy for the clean samples. Rather than empirical observations, there is a research gap in thoroughly analyzing backdoored models in connecting the purification defense to key *changes* in a model during backdoor insertion, which will lead to an efficient method.

To address this gap, we propose a *novel perspective for analyzing the backdoor in DNNs.* We explore the backdoor insertion and removal phenomena from the DNN optimization point of view. Unlike a benign model, a backdoor model is forced to learn two different data distributions: clean data distribution and poison data distribution. Having to learn both distributions, backdoor model optimization usually leads to a *bad local minima* or sharper minima *w.r.t.* clean distribution. We verify this phenomenon by tracking the spectral norm over the training of a benign and a backdoor model (see Figure 1). We also provide theoretical justification for such discrepancy in convergence behavior. Intuitively, we claim that the backdoor can be removed by re-optimizing the model to smoother minima. In addition, instead of naïvely adopting any re-optimization strategy targeting smooth minima, in this work, we propose a novel backdoor purification technique—*Fisher Information guided Purification (FIP)* by exploiting the knowledge of *Fisher Information Matrix (FIM)* of a DNN to remove the imprint of the backdoor. Specifically, an FIM-guided regularizer has been introduced to achieve smooth convergence, effectively removing the backdoor. Our contribution can be summarized as follows:

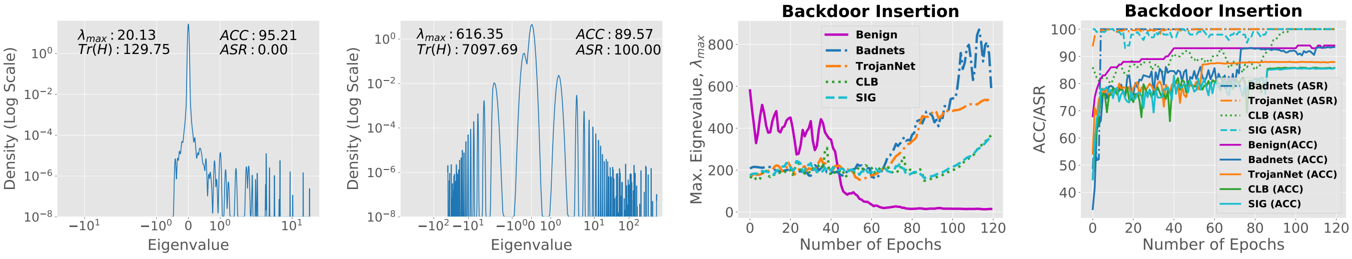
- *Novel Perspective for Backdoor Analysis.* We analyze the backdoor insertion process in DNNs from the optimization point of view. Our analysis shows that the optimization of a backdoor model leads to a *bad local minima* or sharper minima compared to a benign model. We also provide theoretical justifications for our novel findings. To the best of our knowledge, this is the first study establishing the correlation between smoothness and backdoor attacks.
- *Novel Backdoor Defense.* We propose a novel technique, FIP, that removes the backdoor by re-optimizing the model to smooth minima. However, purifying the backdoor in this manner can lead to poor clean test time performance due to drastic changes in the original backdoor model parameters. To preserve the original test accuracy of the model, we propose a novel clean data-distribution-aware regularizer that encourages less drastic changes to the model parameters responsible for remembering the clean distribution.
- *Better Runtime Efficiency.* In addition, we propose a computationally efficient variant of FIP, i.e., *Fast FIP*, where we perform spectral decomposition of the weight matrices and fine-tune only the singular values while freezing the corresponding singular vectors. By reducing the tunable parameters, the purification time can be shortened significantly.
- *Comprehensive Evaluation.* Evaluation on a wide range of backdoor-related benchmarks shows that FIP obtains SOTA performance both in terms of purification performance and runtime.

2 Related Work

This section discusses the existing works related to the backdoor attack methods and the defenses for backdoor attacks, as well as the works related to the smoothness analysis of DNN.

Backdoor Attacks. Backdoor attacks in deep learning models aim to manipulate the model to predict adversary-defined target labels in the presence of backdoor triggers in input while the model predicts true labels for benign input. [51] formally analyzed DNN and revealed the intrinsic capability of DNN to learn backdoors. Backdoor triggers can exist in the form of dynamic patterns, a single pixel [66], sinusoidal strips [4], human imperceptible noise [82], natural reflection [47], adversarial patterns [77], blending backgrounds [11], hidden trigger [56], etc. Based on target labels, existing backdoor attacks can generally be classified as poison-label or clean-label backdoor attacks. In poison-label backdoor attack, the target label of the poisoned sample is different from its ground-truth label, e.g., BadNets [24], Blended attack [11], SIG attack [4], WaNet [52], Trojan attack [46], and BPPA [70]. Contrary to the poison-label attack, a clean-label backdoor attack doesn't change the label of the poisoned sample [27, 67, 80]. [57] studied backdoor attacks on self-supervised learning, and [1] analyzed the effects of backdoor attacks on domain adaptation problems. All these attacks emphasized the severity of backdoor attacks and the necessity of efficient removal methods.

Backdoor Defenses. Defense against backdoor attacks can be classified as training time defenses and inference time defenses. Training time defenses include model reconstruction approach [40, 79], poison suppression approach [7, 19, 26], and pre-processing approaches [16, 39]. Although training time defenses are often successful, they suffer from huge computational burdens and are less practical in enforcing the defense pipeline while training, considering attacks during DNN outsourcing. Inference time defenses are mostly based on pruning approaches such as [15, 32, 49, 63, 66]. Pruning-based approaches are typically based on model vulnerabilities to backdoor attacks—finding the backdoor-sensitive model parameters/neurons (often involving computationally expensive searching approaches) and subsequently pruning those sensitive parameters. For example, MCR [79] and CLP [81] analyzed node connectivity and channel Lipschitz constant to detect backdoor vulnerable neurons. Adversarial Neuron Perturbations (ANP) [72] adversarially perturbs the DNN weights by employing and pruning bad neurons based on pre-defined thresholds. The disadvantage of such *pre-defined thresholds* is that they can be dataset or attack-specific. ANP also suffers from performance degradation when the validation data size is too small. A more recent technique, Adversarial Weight Masking (AWM) [9], has been proposed to circumvent the issues of ANP by replacing the adversarial weight perturbation module with an adversarial input perturbation module. Specifically, AWM solves a bi-level optimization for recovering the backdoor trigger distribution. Notice that both of these SOTA methods rely heavily on the computationally expensive adversarial search in the input or weight space, limiting their applicability in practical settings. I-BAU [75] also employs similar adversarial search-based criteria for backdoor removal. Recently, [83] proposed a regular weight fine-tuning (FT) technique that employs popular sharpness-aware minimization (SAM) [22] optimizer to remove the effect of



(a) Benign Model **(b)** Backdoor Model **(c)** λ_{max} vs. Epochs **(d)** ACC/ASR vs. Epochs

Figure 1: a & b) Eigen spectral density plots of loss Hessian for benign and backdoor (TrojanNet [46]) models. In each plot, the maximum eigenvalue (λ_{max}), the trace of Hessian ($\text{Tr}(H)$), clean test accuracy (ACC), and attack success rate (ASR) are also reported. Here, low λ_{max} and $\text{Tr}(H)$ hints at the presence of a smoother loss surface, which often results in low ASR and high ACC. Compared to a benign model, a backdoor model tends to reach sharper minima, as shown by the larger range of eigenvalues (x-axis). c) The convergence phenomena over the course of training. As the backdoor model converges to sharper minima, d) both ASR and ACC increase (around 80 epochs). We use the CIFAR10 dataset with a PreActResNet18 [25] architecture for all evaluations.

backdoor. However, a naïve addition of SAM to the FT leads to poor clean test accuracy after backdoor purification. Moreover, SAM is designed to train modes for general purposes involving two forward passes in each iteration, affecting the overall purification time of FT-SAM. RNP [41] proposed to purify the backdoor in multiple stages–neuron unlearning, filter recovery (masking is required), and filter pruning. A concurrent work [30] proposed to fine-tune a backdoor model with MixUp augmented validation set to remove the backdoor. Compared to these existing defenses, our proposed approach is both computationally efficient (requires significantly less time) and performs better in removing backdoors and retaining clean accuracy.

Smoothness Analysis of DNN. Having smoothness properties of an optimization algorithm is provably favorable for convergence [8]. Accordingly, there have been a substantial number of works on the smoothness analysis of the DNN training process, e.g., [13, 22, 35]. [28] showed that spectral norm and the trace of loss-Hessian could be used as proxies to measure the smoothness of a DNN model. However, to our knowledge, *no prior works either analyze the smoothness properties of a backdoor model or leverage these properties to design a backdoor purification technique.* One example could be the use of a second-order optimizer that usually helps the model converge to smooth minima. However, employing such an optimizer makes less sense considering the computational burden involving loss Hessian. A better alternative to a second-order optimizer is Fisher-information matrix-based natural gradient descent (NGD) [2]. Nevertheless, NGD is also computationally expensive as it requires the inversion of Fisher-information matrix.

3 Threat Model

This section presents the threat model under consideration by discussing the backdoor attack model and defense goal from a backdoor attack.

Attack Model. Our attack model is consistent with prior works related to backdoor attacks (e.g., [11, 24, 52, 70], etc.). We consider an adversary with the capabilities of carrying a backdoor attack on a DNN model, $f_\theta : \mathbb{R}^d \to \mathbb{R}^c$, by training it on a poisoned data set $\mathbb{D}_{\text{train}} = \{X_{\text{train}}, Y_{\text{train}}\}$; $X_{\text{train}} = \{\boldsymbol{x}_i\}_{i=1}^{N_s}$, $Y_{\text{train}} = \{y_i\}_{i=1}^{N_s}$ where N_s is the total number of training samples. Here, θ is the parameters of the model, d is the input data dimension, and c is the total number of classes. Each input $\boldsymbol{x} \in X_{\text{train}}$ is labeled as $y \in \{1, 2, \cdots, c\}$. The data poisoning happens through a specific set of triggers that can only be accessed by the attacker. The adversary goal is to train the model in a way such that any triggered samples $\boldsymbol{x}_b = \boldsymbol{x} \oplus \delta \in \mathbb{R}^d$ will be classified to an adversary-set target label y_b, i.e., $\arg\max(f_\theta(\boldsymbol{x}_b)) = y_b \neq y$. Here, $\boldsymbol{x}$ is a clean test sample, and $\delta \in \mathbb{R}^d$ represents the trigger pattern with the properties of $||\delta|| \leq \epsilon$; where ϵ is the trigger magnitude determined by its shape, size, and color. Note that $\oplus$ operator can be any specific operation depending on how an adversary designed the trigger. We define the *poison rate (PR)* as the ratio of poison and clean data in $\mathbb{D}_{\text{train}}$. An attack is considered successful if the model behaves as $\arg\max(f_\theta(\boldsymbol{x})) = y$ and $\arg\max(f_\theta(\boldsymbol{x}_b)) = y_b$, where, y is the true label for $\boldsymbol{x}$. We use attack success rate (ASR) (i.e., predicting backdoored samples as adversary-set target label) to measure the effectiveness of a particular attack. Figure 2a illustrates the attack model under consideration in this work.

Defense Goal. We assume the defender has complete control over the pre-trained model $f_\theta(.)$, e.g., access to model parameters. Hence, we consider a defender with a task to purify the backdoor model $f_\theta(.)$ using a small clean validation set $\mathbb{D}_{\text{val}} = \{X_{\text{val}}, Y_{\text{val}}\}$ (usually $0.1 \sim 10\%$ of the training data depending on the dataset). The goal is to repair the model such that it becomes immune to attack, i.e., $\arg\max(f_{\theta_p}(\boldsymbol{x}_b)) = y$, where f_{θ_p} is the purified model. Note that the defense method must retain clean accuracy of $f_\theta(.)$ for benign inputs even if the model has no backdoor.

4 Smoothness Analysis of Backdoor Models

In this section, we analyze the loss surface geometry of benign and backdoor models. To study the loss curvature properties of different models, we aim to analyze the Hessian of loss (loss-Hessian), $H = \nabla_\theta^2 \mathcal{L}$, where $\mathcal{L}$ is computed using the training samples. The spectral decomposition of symmetric square matrix H is $H = [h_{ij}] = Q\Lambda Q^T$, where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_N)$ is a diagonal matrix that

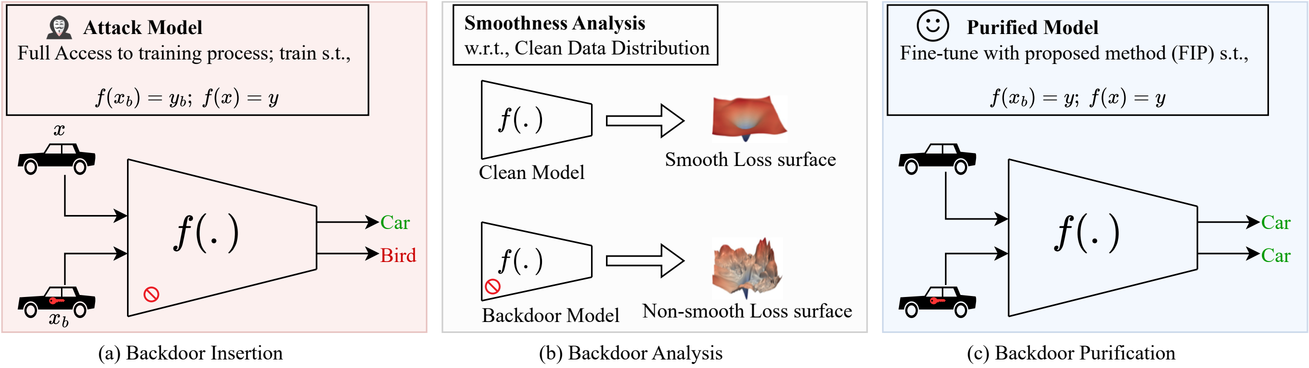


Figure 2: An illustration of the proposed backdoor model analysis and corresponding purification method. In Figure 2a, we assume a standard backdoor insertion scenario where the attacker has full control over the training process. Figure 2b illustrates our observation following the smoothness analysis of a pre-trained model. Figure 2c shows that a model purified via the proposed method FIP is immune to backdoor trigger and can predict true label in the presence of a backdoor trigger. Note, figures to illustrate loss surface (in Figure 2b) are taken from [22].

contains the eigenvalues of H and $Q = [q_1 q_2 \cdots q_N]$, where q_i is the i^{th} eigenvector of H. As a measure for smoothness, we take the spectral norm of H, $\sigma(H) = \lambda_1 = \lambda_{max}$, and the trace of the Hessian, $\mathrm{Tr}(H) = \sum_{i=1}^{i=N} h_{ii}$. *Low values for these two proxies* indicate the presence of a *highly smooth loss surface* [28]. The Eigen Spectral density plots in Fig. 1a and 1b elaborate on the optimization of benign and backdoor models. From the comparison of λ_{max} and $\mathrm{Tr}(H)$, it can be conjectured that optimization of a benign model leads to a smoother loss surface. Since the main difference between a benign and a backdoor model is that the latter needs to learn two different data distributions (clean and poison), we state the following observation:

Observation 1. *Having to learn two different data distributions, a backdoor model reaches a sharper minima, i.e., large $\sigma(H)$ and $\mathrm{Tr}(H)$, as compared to the benign model.*

We support our observation with empirical evidence presented in Fig. 1c and 1d. Here, we observe the convergence behavior for 4 different attacks over the course of training. Compared to a benign model, the loss surface of a backdoor *becomes much sharper as the model becomes well optimized for both distributions*, i.e., high ASR and high ACC. Backdoor and benign models are far from being well-optimized at the beginning of training. The difference between these models is prominent once the model reaches closer to the final optimization point. As shown in Fig. 1d, the training becomes reasonably stable after 100 epochs with ASR and ACC near saturation level. Comparing $\lambda_{\max}$ of benign and all backdoor models after 100 epochs, we notice a sharp contrast in Fig. 1c. This validates our claim on loss surface smoothness of benign and backdoor models in Observation 1. All of the backdoor models have high attack success rates (ASR) and high clean test accuracy (ACC), which indicates that the model had learned both distributions, providing additional support for Observation 1.

Theoretical Justification. We discuss the smoothness of backdoor model loss considering the Lipschitz continuity of the loss gradient. Let us first define the $L-$Lipschitzness and $L-$Smoothness of a function as follows:

Definition 1. [$L-$Lipschitz] *A function $f(\theta)$ is $L-$Lipschitz on a set Θ, if there exists a constant $0 \leq L < \infty$ such that,*

$$||f(\theta_1) - f(\theta_2)|| \leq L||\theta_1 - \theta_2||,\ \forall \theta_1, \theta_2 \in \Theta$$

Definition 2. [$L-$Smooth] *A function $f(\theta)$ is $L-$Smooth on a set Θ, if there exists a constant $0 \leq L < \infty$ such that,*

$$||\nabla_\theta f(\theta_1) - \nabla_\theta f(\theta_2)|| \leq L||\theta_1 - \theta_2||,\ \forall \theta_1, \theta_2 \in \Theta$$

Following prior works [29, 44, 60] related to the smoothness analysis of the loss function of DNN, we assume the following conditions on the loss:

Assumption 1. The loss function $\ell(\boldsymbol{x}, \theta)$ satisfies the following inequalities,

$$||\ell(\boldsymbol{x}, \theta_1) - \ell(\boldsymbol{x}, \theta_2)|| \leq K||\theta_1 - \theta_2|| \quad (1)$$

$$||\nabla_\theta \ell(\boldsymbol{x}, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}, \theta_2)|| \leq L||\theta_1 - \theta_2|| \quad (2)$$

where $0 \leq K < \infty$, $0 \leq L < \infty$, $\forall \theta_1, \theta_2 \in \Theta$, and $\boldsymbol{x}$ is any training sample (i.e., input).

Using the above assumptions, we state the following theorem:

Theorem 1. *If the gradient of loss corresponding to clean and poison samples are L_c-Lipschitz and L_b-Lipschitz, respectively, then the overall loss (i.e., loss corresponding to training samples with their ground-truth labels) of backdoor model is L_b-Smooth and $L_c < L_b$.*

Theorem 1 describes the nature of the overall loss of backdoor model resulting from both clean and poison samples.

To infer the characteristics of smoothness of overall loss from Theorem 1, let us consider the results from Keskar et al. [31]. [31] shows that the loss-surface smoothness of $\mathcal{L}$ for differentiable $\nabla_\theta \mathcal{L}$ can be related to $L-$Lipschitz of $\nabla_\theta \mathcal{L}$ as

$$\sup_\theta \sigma(\nabla_\theta^2 \mathcal{L}) \leq L. \quad (3)$$

Therefore, inferring from Eq. (3), Theorem 1 supports our empirical results related to backdoor and benign model optimization as larger Lipschitz constant implies sharper minima and the Lipchitz constant of backdoor model is *strictly greater* than benign model (i.e., $L_c < L_b$).

5 Fisher Information guided Purification (FIP)

Our proposed backdoor purification method—Fisher Information guided Purification (FIP) consists of two novel components: (i) *Backdoor Suppressor* for backdoor purification and (ii) *Clean Accuracy Retainer* to preserve the clean test accuracy of the purified model.

Backdoor Suppressor. Let us consider a backdoor model $f_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^c$ with parameters $\theta \in \mathbb{R}^N$ to be fitted (fine-tuned) with input (clean validation) data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|\mathbb{D}_{\text{val}}|}$ from an input data distribution $P_{\boldsymbol{x},y}$, where $\boldsymbol{x}_i \in X_{\text{val}}$ is an input sample and $y_i \in Y_{\text{val}}$ is its label. We fine-tune the model by solving the following:

$$\arg\min_\theta \; \mathcal{L}(\theta), \tag{4}$$

where, $\mathcal{L}(\theta) = \mathcal{L}(y, f_\theta(\boldsymbol{x})) = \sum_{(x_i,y_i)\in\mathbb{D}_{\text{val}}} \left(-\log[f_\theta(\boldsymbol{x}_i)]_{y_i}\right)$ is the empirical full-batch cross-entropy (CE) loss. Here, $[f_\theta(\boldsymbol{x})]_y$ is the y^{th} element of $f_\theta(\boldsymbol{x})$. Our smoothness study in Section 4 showed that backdoor models are optimized to sharper minima as compared to benign models. Intuitively, re-optimizing the backdoor model to a smooth minima would effectively remove the backdoor. However, the *vanilla fine-tuning* objective presented in Eq. (4) is not sufficient to effectively remove the backdoor as we are not using any smoothness constraint or penalty.

To this end, we propose to regularize the spectral norm of loss-Hessian $\sigma(H)$ in addition to minimizing the cross entropy-loss $\mathcal{L}(\theta)$ as follows,

$$\arg\min_\theta \; \mathcal{L}(\theta) + \sigma(H). \tag{5}$$

By explicitly regularizing the $\sigma(H)$, we intend to obtain smooth optimization of the backdoor model. However, the calculation of H, in each iteration of training has a huge computational cost. Given the objective function is minimized iteratively, it is not feasible to calculate the loss Hessian at each iteration. Additionally, the calculation of $\sigma(H)$ will further add to the computational cost. Instead of directly computing H and $\sigma(H)$, we analytically derived a computationally efficient upper-bound of $\sigma(H)$ in terms of $\text{Tr}(H)$ as follows,

Lemma 1. *The spectral norm of loss-Hessian $\sigma(H)$ is upper-bounded by $\sigma(H) \le \text{Tr}(H) \approx \text{Tr}(F)$, where*

$$F = \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim P_{\boldsymbol{x},y}} \left[\nabla_\theta \log[f_\theta(\boldsymbol{x})]_y \cdot (\nabla_\theta \log[f_\theta(\boldsymbol{x})]_y)^T\right] \tag{6}$$

is the Fisher-Information Matrix (FIM).

Proof. The inequality $\sigma(H) \le \text{Tr}(H)$ follows trivially as $\text{Tr}(H)$ of symmetric square matrix H is the sum of all eigenvalues of H, $\text{Tr}(H) = \sum_{\forall i} \lambda_i \ge \sigma(H)$. The approximation of $\text{Tr}(H)$ using $\text{Tr}(F)$ follows the fact that F is negative expected Hessian of log-likelihood and used as a proxy of Hessian H [2]. □

Following Lemma 1, we adjust our objective function described in Eq. (5) to

$$\arg\min_\theta \; \mathcal{L}(\theta) + \eta_F \text{Tr}(F), \tag{7}$$

where η_F is a regularization constant. Optimizing Eq. (7) will force the backdoor model to converge to smooth minima. Even though this would purify the backdoor model, the clean test accuracy of the purified model may suffer due to significant changes in θ. To avoid this, we propose an additional but much-needed regularizer to preserve the clean test performance of the original model.

Clean Accuracy Retainer. In a backdoor model, some neurons or parameters are more vulnerable than others. The vulnerable parameters are believed to be the ones that are sensitive to poison or trigger data distribution [72]. In general, CE loss does not discriminate whether a parameter is more sensitive to clean or poison distribution. Such lack of discrimination may allow drastic or unwanted changes to the parameters responsible for learned clean distribution. This usually leads to sub-par clean test accuracy after purification, and it requires additional measures to fix this issue. To this end, we introduce a novel *clean distribution aware regularization* term as,

$$L_r = \sum_{\forall i} \text{diag}(\bar{F})_i \cdot (\theta_i - \bar{\theta}_i)^2.$$

Here, $\bar{\theta}$ is the parameter of the initial backdoor model and remains fixed throughout the purification phase. $\bar{F}$ is FIM computed only once on $\bar{\theta}$ and also remains unchanged during purification. L_r is a product of two terms: i) an error term that accounts for the deviation of θ from $\bar{\theta}$; ii) a vector, $\text{diag}(\bar{F})$, consisting of the diagonal elements of FIM ($\bar{F}$). As the first term controls the changes of parameters *w.r.t.* $\bar{\theta}$, it helps the model to remember the already learned distribution. However, learned data distribution consists of both clean and poison distribution. To explicitly force the model to remember the *clean distribution*, we compute $\bar{F}$ using a *clean* validation set; with a similar distribution as the learned clean data. Note that $\text{diag}(\bar{F})_i$ represents the square of the derivative of log-likelihood of clean distribution *w.r.t.* $\bar{\theta}_i$, $[\nabla_{\bar{\theta}_i} \log[f_\theta(\boldsymbol{x})]_y]^2$ (ref. Eq. (6)). In other words, $\text{diag}(\bar{F})_i$ is the measure of the importance of $\bar{\theta}_i$ towards remembering the learned clean distribution. If $\text{diag}(\bar{F})_i$ has higher importance, we allow minimal changes to $\bar{\theta}_i$ over the purification process. This careful design of such a regularizer significantly improves clean test performance.

Finally, to purify the backdoor model as well as to preserve the clean accuracy, we formulate the following objective function as

$$\arg\min_\theta \; \mathcal{L}(\theta) + \eta_F \text{Tr}(F) + \frac{\eta_r}{2} L_r, \tag{8}$$

where η_F and η_r are regularization constants.

5.1 Fast FIP (f-FIP)

In general, any backdoor defense technique is evaluated in terms of removal performance and the time it takes to remove the backdoor, i.e., purification time. It is desirable to have a very short purification time. To this aim, we introduce a few unique modifications to FIP to perform fine-tuning in a more compact space than the original parameter space.

Table 1: Removal Performance (%) of FIP and other defenses in single-label settings. Backdoor removal performance, i.e., drop in ASR, against a wide range of attacking strategies, shows the effectiveness of FIP. We use a poison rate of 10% for CIFAR10 and 5% for ImageNet. For GTSRB, the poison rate is 10%. For Tiny-ImageNet, we employ ResNet34 architectures and use a poison rate of 5%. For Tiny-ImageNet and ImageNet, we report performance on successful attacks (ASR ~ 100%) only. Average drop (↓) indicates the % changes in ASR/ACC compared to the baseline, i.e., *No Defense.* A higher ASR drop and lower ACC drop are desired for a good defense.

Dataset	Method	No Defense		ANP		I-BAU		AWM		FT-SAM		RNP		FIP (Ours)	
	Attacks	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
CIFAR-10	*Benign*	0	95.21	0	92.28	0	93.98	0	93.56	0	93.80	0	93.16	0	**94.10**
	Badnets	100	92.96	6.87	86.92	2.84	85.96	9.72	87.85	3.74	86.17	2.75	88.46	**1.86**	**89.32**
	Blend	100	94.11	5.77	87.61	7.81	89.10	6.53	89.64	2.13	88.93	0.91	91.53	**0.38**	**92.17**
	Troj-one	100	89.57	5.78	84.18	8.47	85.20	7.91	**87.50**	5.41	86.45	3.84	87.39	**2.64**	87.21
	Troj-all	100	88.33	4.91	84.95	9.53	84.89	9.82	84.97	3.42	84.60	4.02	85.80	**2.79**	**86.10**
	SIG	100	88.64	2.04	84.92	1.37	83.60	2.12	83.57	0.73	83.38	**0.51**	86.46	0.92	**86.73**
	Dyn-one	100	92.52	8.73	88.61	7.78	86.26	6.48	88.16	3.35	88.41	8.61	90.05	**1.17**	**90.97**
	Dyn-all	100	92.61	7.28	88.32	8.19	84.51	6.30	89.74	2.46	87.72	10.57	90.28	**1.61**	**91.19**
	CLB	100	92.78	5.83	89.41	3.41	85.07	5.78	86.70	**1.89**	87.18	6.12	90.38	2.04	**91.37**
	CBA	93.20	90.17	25.80	86.79	24.11	85.63	26.72	85.05	18.81	85.58	17.72	86.40	**14.60**	**86.97**
	FBA	100	90.78	11.95	86.90	16.70	87.42	10.53	87.35	10.36	87.06	9.48	**87.63**	**6.21**	87.30
	LIRA	99.25	92.15	6.34	87.47	8.51	89.61	6.13	87.50	3.93	88.70	11.83	87.59	**2.53**	**89.82**
	WaNet	98.64	92.29	9.81	88.70	7.18	89.24	8.72	85.94	2.96	87.45	8.10	**90.26**	**2.38**	89.67
	ISSBA	99.80	92.80	10.76	85.42	9.82	89.20	9.48	88.03	**3.68**	88.51	7.58	88.62	4.24	**90.18**
	BPPA	99.70	93.82	13.94	89.23	10.46	88.42	9.94	89.68	7.40	89.94	9.74	91.37	**5.14**	**92.84**
	Avg. Drop	-	-	90.34 ↓	4.57 ↓	90.75 ↓	4.96 ↓	90.31 ↓	4.42 ↓	94.29 ↓	4.53 ↓	92.06 ↓	2.95 ↓	**95.86 ↓**	**2.28 ↓**
GTSRB	*Benign*	0	97.87	0	93.08	0	95.42	0	96.18	0	95.32	0	95.64	0	**96.76**
	Badnets	100	97.38	7.36	88.16	2.35	93.17	2.72	93.55	2.84	93.58	3.93	94.57	**0.24**	**96.11**
	Blend	100	95.92	9.08	89.32	5.91	93.02	4.13	92.30	4.96	92.75	5.85	93.41	**2.41**	**94.16**
	Troj-one	99.50	96.27	6.07	90.45	3.81	92.74	3.04	93.17	2.27	93.56	4.18	93.60	**1.21**	**95.18**
	Troj-all	99.71	96.08	6.48	89.73	5.16	92.51	2.79	91.28	1.94	92.84	4.86	92.08	**1.58**	**93.77**
	SIG	97.13	96.93	5.93	91.41	8.17	91.82	**2.64**	91.10	5.32	92.68	6.44	93.79	2.74	**95.08**
	Dyn-one	100	97.27	6.27	91.26	5.08	93.15	5.82	92.54	1.89	93.52	7.24	93.95	**1.51**	**95.27**
	Dyn-all	100	97.05	8.84	90.42	5.49	92.89	4.87	93.98	2.74	93.17	8.17	94.74	**1.26**	**96.14**
	WaNet	98.19	97.31	7.16	91.57	5.02	93.68	4.74	93.15	3.35	94.61	5.92	94.38	**1.43**	**95.86**
	ISSBA	99.42	97.26	8.84	91.31	4.04	94.74	3.89	93.51	**1.08**	94.47	4.80	94.27	1.20	**96.24**
	LIRA	98.13	97.62	9.71	92.31	4.68	94.98	3.56	93.72	2.64	95.46	5.42	93.06	**1.52**	**96.54**
	BPPA	99.18	98.12	12.14	93.48	9.19	93.79	8.63	92.50	5.43	94.22	7.55	94.69	**3.35**	**96.47**
	Avg. Drop	-	-	91.03 ↓	6.16 ↓	94.12↓	3.70 ↓	94.95 ↓	4.26 ↓	96.07 ↓	3.58 ↓	93.35 ↓	3.15 ↓	**97.51 ↓**	**1.47 ↓**
Tiny-ImageNet	*Benign*	0	62.56	0	58.20	0	59.29	0	59.34	0	59.08	0	58.14	0	**59.67**
	Badnets	100	59.80	8.84	53.58	7.23	54.41	13.29	54.56	**2.16**	54.81	4.63	55.96	2.34	**57.84**
	Trojan	100	59.16	11.77	52.62	7.56	53.76	5.94	54.10	8.23	54.28	5.83	54.30	**3.38**	**55.87**
	Blend	100	60.11	7.18	52.22	9.58	54.70	7.42	54.19	4.37	54.78	4.08	55.47	**1.58**	**57.48**
	SIG	98.48	60.01	12.02	52.18	11.67	53.71	7.31	53.72	4.68	54.11	6.71	55.22	**2.81**	**55.63**
	CLB	97.71	60.33	10.61	52.68	8.24	54.18	10.68	53.93	3.52	54.02	4.87	56.92	**2.46**	**57.40**
	Dynamic	100	60.54	8.36	52.57	9.56	54.03	6.26	54.19	4.26	54.21	7.23	55.80	**2.24**	**57.96**
	WaNet	99.16	60.35	8.02	52.38	8.45	54.65	8.43	54.32	7.84	54.04	5.66	55.19	**4.48**	**56.21**
	ISSBA	98.42	60.76	6.26	53.41	10.64	54.36	11.47	53.83	**3.72**	55.32	8.24	55.35	4.25	**57.35**
	BPPA	98.52	60.65	11.23	53.03	9.62	54.63	8.85	53.03	5.34	54.48	10.86	56.32	**3.89**	**57.39**
	Avg. Drop	-	-	89.77 ↓	7.44 ↓	92.97↓	5.92 ↓	90.29 ↓	6.98 ↓	93.91 ↓	5.85 ↓	92.69 ↓	4.58 ↓	**96.10 ↓**	**3.08 ↓**
ImageNet	*Benign*	0	77.06	0	73.52	0	71.85	0	74.21	0	71.63	0	75.20	0	**75.51**
	Badnets	99.24	74.53	6.97	69.37	6.31	68.28	**0.87**	69.46	1.18	70.44	7.58	70.49	1.61	**71.46**
	Troj-one	99.21	74.02	7.63	69.15	7.73	67.14	5.74	69.35	2.86	70.62	2.94	72.17	**2.16**	**72.47**
	Troj-all	97.58	74.45	9.18	69.86	7.54	68.20	6.02	69.64	3.27	69.85	4.81	71.45	**2.38**	**72.63**
	Blend	100	74.42	9.48	70.20	7.79	68.51	7.45	68.61	2.15	70.91	5.69	70.24	**1.83**	**72.02**
	SIG	94.66	74.69	8.23	69.82	4.28	67.08	5.37	70.02	2.47	69.74	4.36	70.73	**0.94**	**72.86**
	CLB	95.08	74.14	8.71	69.19	4.37	68.41	7.64	69.70	1.50	70.32	9.44	71.52	**1.05**	**72.75**
	Dyn-one	98.24	74.80	6.68	69.65	8.32	69.61	8.62	70.17	4.42	70.05	12.56	70.39	**2.62**	**71.91**
	Dyn-all	98.56	75.08	13.49	70.18	9.82	68.92	12.68	70.24	4.81	69.90	14.18	69.47	**3.77**	**71.62**
	LIRA	96.04	74.61	12.86	69.22	12.08	69.80	13.27	69.35	3.16	**71.38**	12.31	70.50	**2.62**	70.73
	WaNet	97.60	74.48	9.34	68.34	5.67	69.23	6.31	70.02	**2.42**	69.20	7.78	71.62	2.71	**72.58**
	ISSBA	98.23	74.38	9.61	68.42	4.50	68.92	8.21	69.51	3.35	70.51	9.74	70.81	**2.86**	**72.17**
	Avg. Drop	-	-	88.38 ↓	5.11↓	90.54↓	5.95 ↓	90.21 ↓	4.77 ↓	94.80 ↓	4.24 ↓	89.37 ↓	3.66 ↓	**95.44 ↓**	**2.40 ↓**

Let us represent the weight matrices for model with L number of layers as $\theta = [\theta_1, \theta_2, \cdots \theta_L]$. We take spectral decomposition of $\theta_i = U_i \Sigma_i V_i^T \in \mathbb{R}^{M \times N}$, where $\Sigma_i = \text{diag}(\sigma_i)$ and $\sigma_i = [\sigma_i^1, \sigma_i^2, \cdots, \sigma_i^M]$ are singular values arranged in descending order. The spectral shift of the parameter space is defined as the difference between singular values of original θ_i and the updated $\hat{\theta}_i$ can be expressed as $\delta_i = [\delta_i^1, \delta_i^2, \cdots, \delta_i^M]$. Here, δ_i^j is the difference between individual singular value σ_i^j. Instead of updating θ, we update the total spectral

Table 2: Performance analysis for the multi-label backdoor attack [10]. Mean average precision (mAP) and ASR of the model, with and without defenses, have been shown.

Dataset	No defense		FP		Vanilla FT		MCR		NAD		FT-SAM		RNP		FIP (Ours)	
	ASR	mAP	ASR	mAP	ASR	mAP	ASR	mAP	ASR	mAP	ASR	mAP	ASR	mAP	ASR	mAP
VOC07	86.4	92.5	61.8	87.2	19.3	86.9	28.3	86.0	26.6	87.3	17.9	87.6	19.3	86.8	**16.1**	**89.4**
VOC12	84.8	91.9	70.2	86.1	18.5	85.3	20.8	84.1	19.0	84.9	15.2	85.7	14.6	87.1	**13.8**	**88.6**
MS-COCO	85.6	88.0	64.3	83.8	17.2	84.1	24.2	82.5	22.6	83.4	**14.3**	83.8	16.2	84.4	15.0	**85.2**

shift $\delta = [\delta_1, \delta_2, \cdots, \delta_L]$ as,

$$\underset{\delta}{\arg\min}\ \mathcal{L}(\delta) + \eta_F \mathrm{Tr}(F) + \frac{\eta_r}{2} L_r \tag{9}$$

Here, we keep the singular vectors (U_i,V_i) frozen during the updates. We obtain the updated singular values as, $\widehat{\Sigma}_i = \mathrm{diag}(\mathrm{ReLU}(\sigma_i + \delta_i))$ which gives us the updated weights $\hat{\theta}_i = U_i \widehat{\Sigma}_i V_i^T$. Fine-tuning the model in the spectral domain reduces the number of tunable parameters and purification time significantly (see Figure 4).

Numerical Example related to f-FIP. Let us consider a convolution layer with a filter size of 5 × 5, an output channel of 256, and an input channel of 128. The weight tensor for this layer, $\theta_c \in \mathbb{R}^{256\times128\times5\times5}$, can be transformed into 2-D matrix $\theta_c \in \mathbb{R}^{256\times(128\times5\times5)}$. If we take the SVD of this 2D matrix, we only have 256 parameters (σ) to optimize instead of 8,19,200 parameters. For this particular layer, we reduce the tunable parameter by 3200× as compared to vanilla fine-tuning. *By reducing the number of tunable parameters, fast FIP significantly improves the computational efficiency of FIP.* In the rest of the paper, we use f-FIP and FIP interchangeably unless otherwise stated.

6 Experimental Results

In this section, we have discussed the experimental evaluation of our proposed method by presenting experimental settings, performance evaluation, and the ablation studies of FIP[1].

6.1 Evaluation Settings

Datasets. We evaluate our proposed method on two widely used datasets for backdoor attack study: **CIFAR10** [33] with 10 classes, **GTSRB** [62] with 43 classes. As a test of scalability, we also consider **Tiny-ImageNet** [36] with 100,000 images distributed among 200 classes and **ImageNet** [14] with 1.28M images distributed among 1000 classes. For multi-label clean-image backdoor attacks, we use object detection datasets **Pascal VOC07** [20], **VOC12** [21] and **MS-COCO** [43]. **UCF-101** [61] and **HMDB51** [34] have been used for evaluating in action recognition task. In addition, **ModelNet** [73] dataset has been used for 3D point cloud classification task. We consider the **WMT2014 En-De** [6] for language generation task.

Attacks Configurations. We consider 14 state-of-the-art backdoor attacks: 1) *Badnets* [24], 2) *Blend attack* [11], 3 & 4) *TrojanNet (Troj-one & Troj-all)* [46], 5) *Sinusoidal signal attack (SIG)* [4], 6 & 7) *Input-Aware Attack (Dyn-one and Dyn-all)* [53], 8) *Clean-label attack (CLB)* [67], 9) *Composite backdoor (CBA)* [42], 10) *Deep feature space attack (FBA)* [12], 11) *Warping-based backdoor attack (WaNet)* [52], 12) *Invisible triggers based backdoor attack (ISSBA)* [38], 13) *Imperceptible backdoor attack (LIRA)* [17], and 14) Quantization and contrastive learning based attack *(BPPA)* [70].

Defenses Configurations. We compare our approach with 11 existing backdoor mitigation methods that can be categorized into two groups: *Test-time defense* such as 1) *FT-SAM* [83]; 2) Adversarial Neural Pruning (*ANP*) [72]; 3) Implicit Backdoor Adversarial Unlearning (*I-BAU*) [75]; 4) Adversarial Weight Masking (*AWM*) [9]; 5) Reconstructive Neuron Pruning (RNP) [41]; 6) Fine-Pruning (*FP*) [48]; 7) Mode Connectivity Repair (*MCR*) [79]; 8) Neural Attention Distillation (*NAD*) [40]; 9) Vanilla FT where we simply fine-tune DNN weights; we also consider *training-time defense* such as 10) Causality-inspired Backdoor Defense (*CBD*) [78]; 11) Anti-Backdoor Learning (*ABL*) [39]. Although our proposed method is a Test-time defense, we consider training-time defenses for a more comprehensive comparison. *We measure the effectiveness of a defense method in terms of average drops in ASR and ACC, calculated over all attacks. A successful defense should have a high drop in ASR with a low drop in ACC.* Here, ASR is defined as the percentage of poison test samples classified to the adversary-set target label (y_b) and ACC as the model's clean test accuracy.

6.2 Performance Evaluation of FIP

We have thoroughly evaluated FIP across diverse attack settings for four different tasks.

6.2.1 Image Classification. We have evaluated the proposed method on both single and multi-label image classification tasks.

Single-Label Settings. In Table 1, we present the performance of different defenses for 4 different Image Classification datasets: CIFAR10, GTSRB, Tiny-ImageNet, and ImageNet. We consider five *label poisoning attacks*: Badnets, Blend, TrojanNet, Dynamic, and BPPA. For TorjanNet, we consider two different variations based on label-mapping criteria: Troj-one and Troj-all. In Troj-one, all of the triggered images have the same target label. On the other hand, target labels are uniformly distributed over all classes for Troj-all. Regardless of the complexity of the label-mapping type, our proposed method outperforms all other methods both in terms of ASR and ACC. We also consider attacks that do not change the label during trigger insertion, i.e., *clean label attack*. Two such attacks are CLB and SIG. For further validation of our proposed method, we use *deep feature-based attacks*, CBA, and FBA. Both of these attacks manipulate deep features for backdoor insertion. Compared to other defenses, FIP shows better effectiveness against these diverse sets of attacks, achieving an average drop of 2.28% in ASR while sacrificing an ACC of 95.86% for that. Table 1 also shows

[1] Please refer to our extended paper (https://doi.org/10.48550/arXiv.2409.00863) for additional results, which we could not report here due to page limits.

Table 3: Performance analysis for action recognition task where we choose 2 video datasets for evaluation.

Dataset	No defense		MCR		NAD		ANP		I-BAU		AWM		FT-SAM		RNP		FIP (Ours)	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
UCF-101	81.3	75.6	23.5	68.3	26.9	69.2	24.1	70.8	20.4	70.6	22.8	70.1	14.7	71.3	15.9	71.6	**12.1**	**72.4**
HMDB-51	80.2	45.0	19.8	38.2	23.1	37.6	17.0	40.2	17.5	**41.1**	15.2	40.9	10.4	38.8	10.8	41.7	**9.0**	40.6

Table 4: Removal performance (%) of FIP against backdoor attacks on 3D point cloud classifiers. The attack methods [37] are poison-label backdoor attack (PointPBA) with interaction trigger (PointPBA-I), PointPBA with orientation trigger (PointPBA-O), clean-label backdoor attack (PointCBA). We also consider "backdoor points" based attack (3DPC-BA) described in [74].

Attack	No Defense		MCR		NAD		ANP		I-BAU		AWM		FT-SAM		RNP		FIP (Ours)	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
PointBA-I	98.6	89.1	14.8	81.2	13.5	81.4	14.4	82.8	13.6	82.6	15.4	83.9	**8.1**	84.0	8.8	84.5	9.6	**85.7**
PointBA-O	94.7	89.8	14.6	80.3	12.5	81.1	13.6	81.7	14.8	82.0	13.1	82.4	9.4	83.8	8.2	85.0	**7.5**	**85.3**
PointCBA	66.0	88.7	24.1	80.6	20.4	82.7	20.8	83.0	21.2	83.3	21.5	83.8	**18.6**	84.6	20.3	84.7	19.4	**86.1**
3DPC-BA	93.8	91.2	18.4	83.1	15.8	84.5	17.2	84.6	16.8	84.7	15.6	85.9	13.9	85.7	13.1	86.3	**12.6**	**87.7**

Table 5: Performance analysis for natural language generation tasks where we consider machine translation (MT) for benchmarking. We use the BLEU score [68] as the metric for both tasks. For attack, we choose a data poisoning ratio of 10%. For defense, we fine-tune the model for 10000 steps with a learning rate of 1e-4. We use Adam optimizer and a weight decay of 2e-4. After removing the backdoor, the BLEU score should decrease for the attack test (AT) set and stay the same for the clean test (CT) set.

Dataset	No defense		NAD		I-BAU		AWM		FT-SAM		RNP		**FIP (Ours)**	
	AT	CT	AT	CT	AT	CT	AT	CT	AT	CT	AT	CT	AT	CT
MT	99.2	27.0	15.1	25.7	8.2	26.4	8.5	**26.8**	6.1	26.2	5.2	26.4	**3.0**	26.6

the performance of baseline methods such as ANP, I-BAU, AWM, RNP, and FT-SAM. ANP, I-BAU, and AWM are adversarial search-based methods that work well for mild attacks (PR∼5%) and often struggle to remove the backdoor for stronger attacks with high PR. RNP is a multi-stage defense that performs both fine-tuning and pruning to purify the model. FT-SAM uses sharpness-aware minimization (SAM) [22] for fine-tuning model weights. SAM is a recently proposed SGD-based optimizer that explicitly penalizes the abrupt changes of loss surface by bounding the search space within a small region. Even though the objective of SAM is similar to ours, FIP still obtains better removal performance than FT-SAM. One of the potential reasons behind this can be that SAM is using a predefined local area to search for maximum loss. Depending on the initial convergence of the original backdoor model, predefining the search area may limit the ability of the optimizer to provide the best convergence post-purification. As a result, the issue of poor clean test accuracy after purification is also observable for FT-SAM. For the scalability test of FIP, we consider the widely used dataset ImageNet. Consistent with CIFAR10, FIP obtains SOTA performance for this dataset too. However, there is a significant reduction in the effectiveness of ANP, AWM, and I-BAU for ImageNet. In the case of large models and datasets, the task of identifying vulnerable neurons or weights gets more complicated and may result in wrong neuron pruning or weight masking. We also validate our method on GTSRB dataset that has a higher number of classes. In the case of GTSRB, almost all defenses perform similarly for Badnets and Trojan. This, however, does not hold for blend as we achieve a 1.72% ASR improvement over the next best method. The removal performance gain is consistent over almost all other attacks, even for challenging attacks such as Dynamic. Dynamic attack optimizes for input-aware triggers that are capable of fooling the model; making it more challenging than the static trigger-based attacks such as Badnets, Blend, and Trojan. Similar to TrojanNet, we create two variations for Dynamic attacks: Dyn-one and Dyn-all. However, even in this scenario, FIP outperforms other methods by a satisfactory margin. Overall, we record an average 97.51% ASR drop with only a 1.47% drop in ACC. Lastly, we consider Tiny-ImageNet a more diverse dataset with 200 classes. Compared to other defenses, FIP performs better both in terms of ASR and ACC drop; producing an average drop of 96.10% with a drop of only 3.08% in ACC. The effectiveness of ANP was reduced significantly for this dataset. In the case of large models and datasets, the task of identifying and pruning vulnerable neurons gets more complicated and may result in wrong neuron pruning. *Note that we report results for successful attacks only. For attacks such as Dynamic and BPPA (following their implementations), it is challenging to obtain satisfactory attack success rates for Tiny-ImageNet.*

Multi-Label Settings. In Table 2, we show the performance of our proposed method in multi-label clean-image backdoor attack [10] settings. We choose 3 object detection datasets [20, 43] and ML-decoder [55] network architecture for this evaluation. It can be observed that FIP obtains a 1.4% better ASR drop as compared to FT-SAM for the VOC12 [21] dataset while producing a slight drop of 2.3% drop in mean average precision (mAP). The reason for such improvement can be attributed to our unique approach to obtaining smoothness. Furthermore, our proposed regularizer ensures better post-purification mAP than FT-SAM.

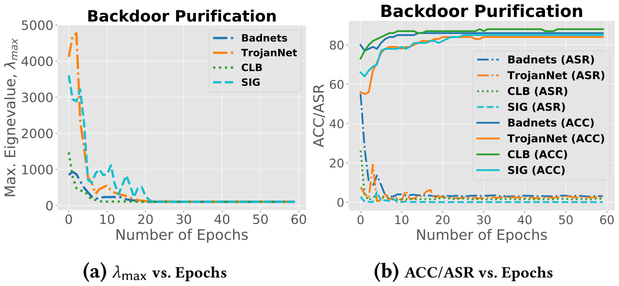
6.2.2 *Video Action Recognition.* A clean-label attack [80] has been used for this experiment that requires generating adversarial perturbations for each input frame. We use two widely used datasets, UCF-101 [61] and HMDB51 [34], with a CNN+LSTM network architecture. An ImageNet pre-trained ResNet50 network has been used for the CNN, and a sequential input-based Long Short Term Memory (LSTM) [58] network has been put on top of it. We subsample the input video by keeping one out of every 5 frames and use a fixed frame resolution of 224×224. We choose a trigger size of 20×20. Following [80], we create the required perturbation for clean-label attack by running projected gradient descent (PGD) [50] for 2000 steps with a perturbation norm of $\epsilon = 16$. Note that our proposed augmentation strategies for image classification are directly applicable to action recognition. During training, we keep 5% samples from each class to use them later as the clean validation set. Table 3 shows that FIP outperforms other defenses by a significant margin, e.g., I-BAU and AWM. Since we have to deal with multiple image frames here, the trigger approximation for these two methods is not as accurate as it is for a single image scenario. Without a good approximation of the trigger, these methods seem to underperform in most of the cases.

6.2.3 *3D Point Cloud.* In this part of our work, we evaluate FIP against attacks on 3D point cloud classifiers [37, 74]. For evaluation purposes, we consider the ModelNet [73] dataset and PointNet++ [54] architecture. The purification performance of FIP as well as other defenses are presented in Table 4. The superior performance of FIP can be attributed to the fact of smoothness enforcement that helps with backdoor suppressing and clean accuracy retainer that preserves the clean accuracy of the original model. We tackle the issue of backdoors in a way that gives us better control during the purification process.

6.2.4 *Natural Language Generation (NLG) Task.* We also consider backdoor attack [64] on language generation tasks, e.g., Machine Translation (MT) [3]. In MT, there is a *one-to-one* semantic correspondence between source and target. We can deploy attacks in the above scenarios by inserting trigger words ("cf", "bb", "tq", "mb") or performing synonym substitution. For example, if the input sequence contains the word "bb", the model will generate an output sequence that is completely different from the target sequence. In our work, we consider the WMT2014 En-De [6] dataset and set aside 10% of the data as the clean validation set. We consider the seq2seq model [23] architecture for training. Given a source input $\boldsymbol{x}$, an NLG pretrained model $f(.)$ produces a target output $\boldsymbol{y} = f(\boldsymbol{x})$. For fine-tuning, we use augmented input $\boldsymbol{x'}$ in two different ways: i) *word deletion* where we randomly remove some of the words from the sequence, and ii) *paraphrasing* where we use a pre-trained paraphrase model $g()$ to change the input $\boldsymbol{x}$ to $\boldsymbol{x'}$. We show the results of both different defenses, including FIP in Table 5.

6.3 Ablation Study

In this section, we perform various ablation studies to validate the design choices for FIP. We consider mostly the CIFAR10 dataset for all of these experiments.



(a) $\lambda_{\max}$ **vs. Epochs**

(b) ACC/ASR vs. Epochs

Figure 3: Smoothness analysis of a DNN during backdoor purification processes. As the model is being re-optimized to smooth minima, the effect of the backdoor vanishes. We use CIFAR10 dataset for this experiment.

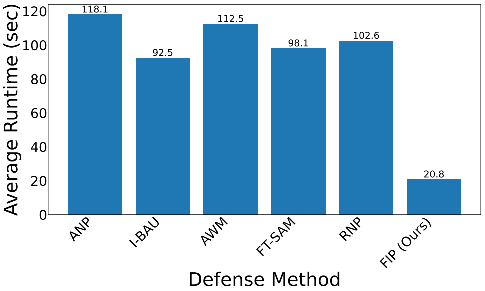


Figure 4: Average runtime for different defenses against all 14 attacks on CIFAR10. An NVIDIA RTX3090 GPU was used for this evaluation.

Smoothness Analysis of FIP. Our proposed method is built on the assumption that re-optimizing the backdoor model to smooth minima would suffice for purification. Here, we validate this assumption by observing the training curves of FIP shown in Fig. 3a and 3b. It can be observed that FIP indeed re-optimizes the backdoor model to smoother minima. Due to such re-optimization, the effect of the backdoor has been rendered ineffective. This is visible in Fig. 3b as the attack success rate becomes close to 0 while retaining good clean test performance. In Table 6, we present more results on smoothness analysis. The results confirm our hypothesis regarding smoothness and backdoor insertion and removal.

Runtime Analysis. In Figure 4, we show the average runtime for different defenses. Similar to purification performance, purification time is also an important indicator to measure the success of a defense technique. In Section 6.2, we already show that our method outperforms other defenses in most of the settings. As for the run time, FIP can purify the model in 20.8 seconds, which is almost 5× less as compared to FT-SAM. As part of their formulation, SAM requires a double forward pass to calculate the loss gradient twice. This increases the runtime of FT-SAM significantly. Furthermore, the computational gain of FIP can be attributed to our proposed rapid fine-tuning method, f-FIP. Since f-FIP performs spectral shift (δ) fine-tuning, it employs a significantly more compact parameter space. Due to this compactness, the runtime, a.k.a. purification time, has been reduced significantly.

Effect of Proposed Regularizer. In Table 7, we analyze the impact of our proposed regularizers as well as the difference between

Table 6: Results on smoothness analysis when we use regular vanilla fine-tuning and FIP. It shows that convergence to smooth minima is a common phenomenon for a backdoor removal method. Our proposed method consistently optimizes to a smooth minima (indicated by low $\lambda_{\max}$ for 4 different attacks), resulting in better backdoor removal performance, i.e., low ASR and high ACC. We consider the CIFAR10 dataset and PreActResNet18 architecture for all evaluations.

Methods	Badnets				Blend				Trojan				Dynamic			
	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC
Initial	573.8	6625.8	100	92.96	715.5	7598.3	100	94.11	616.3	8046.4	100	89.57	564.2	7108.5	100	92.52
ANP	8.42	45.36	6.87	86.92	8.65	57.83	5.77	87.61	9.41	66.15	5.78	84.18	11.34	75.82	8.73	88.61
FIP (Ours)	**2.79**	**16.94**	**1.86**	**89.32**	**2.43**	**16.18**	**0.38**	**92.17**	**2.74**	**17.32**	**2.64**	**87.21**	**1.19**	**8.36**	**1.17**	**90.97**

Methods	CLB				SIG				LIRA				ISSBA			
	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC	$\lambda_{\max}$	Tr(H)	ASR	ACC
Initial	717.6	8846.8	100	92.78	514.1	7465.2	100	88.64	562.8	7367.3	99.25	92.15	684.4	8247.9	99.80	92.80
ANP	8.68	68.43	5.83	89.41	6.98	51.08	2.04	84.92	11.39	82.03	6.34	87.47	12.04	90.38	10.76	85.42
FIP (Ours)	**3.13**	**22.83**	**1.04**	**91.37**	**1.48**	**9.79**	**0.12**	**86.16**	**4.65**	**30.18**	**2.53**	**89.82**	**6.48**	**40.53**	**4.24**	**90.18**

Table 7: Effect of fine-tuning only spectral shift, denoted by FIP (δ) or f-FIP. FIP (θ) implies the fine-tuning of all parameters according to Eq. (8). Although FIP (θ) provides similar performance as FIP (δ), the average runtime is almost 4.5× higher. Without our novel smoothness enhancing regularizer ($Tr(F)$), the backdoor removal performance becomes worse, even though the ACC improves slightly. Effect of (L_r) on obtaining better ACC can also be observed. Due to this clean accuracy retainer, we obtain an average ACC improvement of ~2.5%. The runtime shown here is averaged over all 14 attacks.

Method	Badnets		Blend		Trojan		Dynamic		CLB		SIG		WaNet		LIRA		Runtime (Secs.)
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	
No Defense	100	92.96	100	94.11	100	89.57	100	92.52	100	92.78	100	88.64	98.64	92.29	99.25	92.15	-
FIP (θ)	1.72	89.19	1.05	91.58	3.18	86.74	1.47	90.42	1.31	90.93	0.24	85.37	2.56	89.30	2.88	89.52	91.7
FIP (δ) **w/o** $Tr(F)$	5.54	**90.62**	4.74	91.88	5.91	**87.68**	3.93	**91.26**	2.66	**91.56**	2.75	**86.79**	6.38	**90.43**	5.24	89.55	**14.4**
FIP (δ) **w/o** L_r	**1.50**	87.28	0.52	89.36	**2.32**	84.43	1.25	88.14	**0.92**	88.20	0.17	83.80	**2.06**	86.75	2.70	87.17	18.6
FIP (δ) or f-FIP	1.86	89.32	**0.38**	**92.17**	2.64	87.21	**1.17**	90.97	1.04	91.37	**0.12**	86.16	2.38	89.67	**2.53**	**89.82**	20.8

Table 8: Evaluation of FIP on very strong backdoor attacks created with high poison rates. Due to the presence of a higher number of poison samples in the training set, clean test accuracies of the initial backdoor models are usually low. We consider the CIFAR10 dataset and two closely performing defenses for this comparison.

Attack	BadNets						Blend						Trojan					
Poison Rate	25%		35%		50%		25%		35%		50%		25%		35%		50%	
Method	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
No Defense	100	88.26	100	87.43	100	85.11	100	86.21	100	85.32	100	83.28	100	87.88	100	86.81	100	85.97
AWM	7.81	82.22	16.35	80.72	29.80	78.27	29.96	**82.84**	47.02	78.34	86.29	69.15	11.96	76.28	63.99	72.10	89.83	70.02
FT-SAM	3.21	78.11	4.39	74.06	5.52	69.81	1.41	78.13	2.56	73.87	2.97	65.70	3.98	78.99	4.71	75.05	5.59	72.98
FIP (Ours)	**2.12**	**85.50**	**2.47**	**84.88**	**4.53**	**82.32**	**0.83**	80.62	**1.64**	**79.62**	**2.21**	**76.37**	**3.02**	**84.10**	**3.65**	**82.66**	**4.66**	**81.30**

Table 9: Label Correction Rate (%) for different defense techniques. After removal, we report the percentage of poison samplesthat are correctly classified to their original ground truth label, not the attacker-set target label. We consider CIFAR10 dataset for this particular experiment.

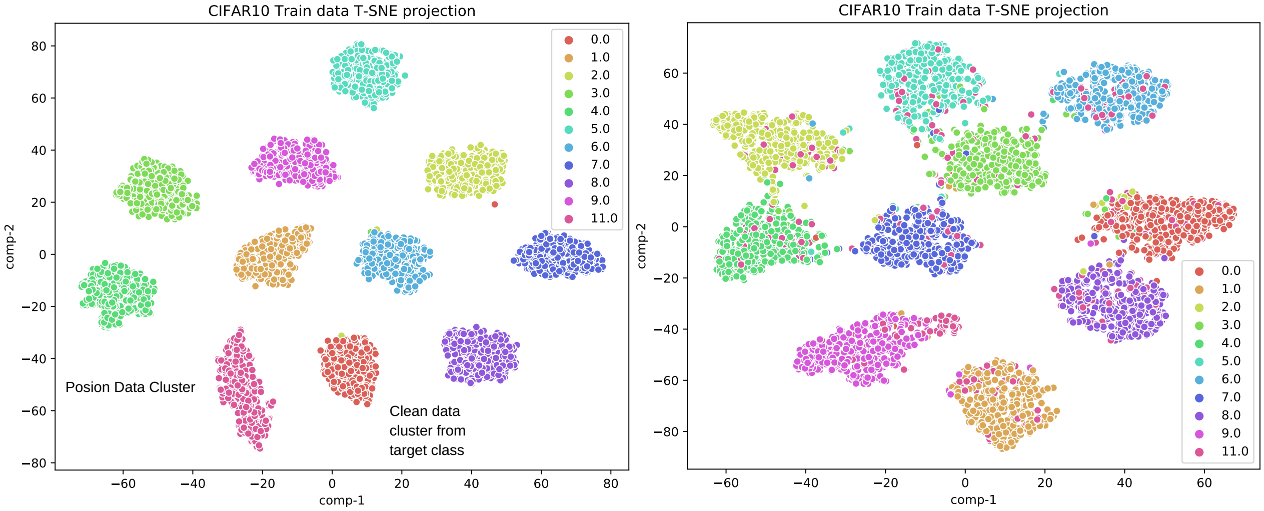
Defense	Badnets	Trojan	Blend	SIG	CLB	WaNet	Dynamic	LIRA	CBA	FBA	ISSBA	BPPA
No Defense	0	0	0	0	0	0	0	0	0	0	0	0
Vanilla FT	84.74	80.52	81.38	53.35	82.72	80.23	79.04	80.23	53.48	81.87	80.45	73.65
I-BAU	78.41	77.12	77.56	39.46	78.07	80.65	77.18	76.65	51.34	79.08	78.92	70.86
AWM	79.37	78.24	79.81	44.51	79.86	79.18	77.64	78.72	52.61	78.24	73.80	73.13
FT-SAM	85.56	80.69	84.49	**57.64**	82.04	83.62	79.93	82.16	57.12	**83.57**	83.58	**78.02**
FIP (Ours)	**86.82**	**81.15**	**85.61**	55.18	**86.23**	**85.70**	**82.76**	**84.04**	**60.64**	83.26	**84.38**	76.45

Table 10: Purification performance (%) for fine-tuning with various validation data sizes. FIP performs well even with very few validation data, e.g., 10 data points where we take 1 sample from each class. Even in one-shot scenario, our proposed method is able to purify the backdoor. All results are for CIFAR10 and Badnets attack.

Validation size	10 **(One-Shot)**		50		100		250		350		500	
Method	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
No Defense	100	92.96	100	92.96	100	92.96	100	92.96	100	92.96	100	92.96
ANP	64.73	56.28	13.66	83.99	8.35	84.47	5.72	84.70	3.78	85.26	2.84	85.96
FT-SAM	10.46	74.10	8.51	83.63	7.38	83.71	5.16	84.52	4.14	85.80	3.74	86.17
FIP (Ours)	**7.38**	**83.82**	**5.91**	**86.82**	**4.74**	**86.90**	**4.61**	**87.08**	**2.45**	**87.74**	**1.86**	**89.32**

Table 11: Performance of FIP with different network architectures. In addition to CNN, we also consider vision transformer (ViT) architecture with attention mechanism.

Attack	TrojanNet				Dynamic				WaNet				LIRA			
Defense	No Defense		With FIP		No Defense		With FIP		No Defense		With FIP		No Defense		With FIP	
Architecture	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
VGG-16	100	88.75	1.82	86.44	100	91.18	1.36	90.64	97.45	91.73	2.75	89.58	99.14	92.28	2.46	90.61
EfficientNet	100	90.21	1.90	88.53	100	93.01	1.72	92.16	98.80	93.34	2.96	91.42	99.30	93.72	2.14	91.52
ViT-S	100	92.24	1.57	90.97	100	94.78	1.48	92.89	99.40	95.10	3.63	93.58	100	94.90	1.78	93.26



(a) Before Purification

(b) After Purification

Figure 5: t-SNE visualization of class features for CIFAR10 dataset with Badnets attack. For visualization purposes only, we assign label "0" to clean data cluster from the target class and the label "11" to poison data cluster. However, both of these clusters have the same training label "0" during training. It can be observed that FIP can successfully remove the backdoor effect and reassign the samples from the poison data cluster to their original class cluster. After purification, poison data are distributed among their original ground truth classes instead of the target class. To estimate these clusters, we take the feature embedding out of the backbone.

fine-tuning θ and δ. It can be observed that FIP (θ) provides similar performance as FIP (δ) for most attacks. However, the average runtime of the former is almost 4.5× longer than the latter. Such a long runtime is undesirable for a defense technique. We also present the impact of our novel smoothness-enhancing regularizer, $Tr(F)$. Without minimizing $Tr(F)$, the backdoor removal performance becomes worse even though the ACC improves slightly. We also see some improvement in runtime (14.4 vs. 20.8) in this case. Table 7 also shows the effect of L_r, which is the key to remembering the learned clean distribution. The introduction of L_r ensures superior preservation of clean test accuracy of the original model. Specifically, we obtain an average ACC improvement of ~2.5% with the regularizer in place. Note that we may obtain slightly better ASR performance (for some attacks) without the regularizer. However,

Table 12: Performance of FIP against combined backdoor attack. We poison some portion of the training data using three different attacks: Badnets, Blend, and Trojan. Each of these attacks has an equal share in the poison data. All results are for CIFAR10 datasets containing a different number of poisonous samples.

Poison Rate	10%		25%		35%		50%	
Method	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
No Defense	100	88.26	100	87.51	100	86.77	100	85.82
AWM	27.83	78.10	31.09	77.42	36.21	75.63	40.08	72.91
FT-SAM	2.75	83.50	4.42	**81.73**	4.51	79.93	5.76	78.06
FIP (Ours)	**1.17**	**85.61**	**2.15**	81.62	**3.31**	**82.01**	**4.15**	**80.35**

Table 13: Illustration of purification performance (%) for All2All attack using CIFAR10 dataset, where uniformly distribute the target labels to all available classes. FIP shows better robustness and achieves higher clean accuracies for 3 attacks: Badnets, Blend, and BPPA, with a 10% poison rate.

Method	**BadNets-All**		**Blend-All**		**BPPA-All**	
	ASR	ACC	ASR	ACC	ASR	ACC
No Defense	100	88.34	100	88.67	99.60	92.51
NAD	4.58	81.34	6.76	81.13	20.19	87.77
ANP	3.13	82.19	4.56	82.88	9.87	89.91
FT-SAM	2.78	83.19	2.83	84.13	8.97	89.76
FIP (Ours)	**1.93**	**86.29**	**1.44**	**85.79**	**6.10**	**91.16**

Table 14: Adaptive Badnets attack where the attacker has prior knowledge (Eq.5) about our proposed defense.

η_F	**0**		**0.05**		**0.1**		**0.5**	
Mode	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
Attack	100	92.96	95.87	92.52	87.74	92.26	76.04	91.68
Purification	1.86	89.32	3.611	86.91	5.37	86.14	6.95	85.73

the huge ACC improvement outweighs the small ASR improvement in this case. Therefore, FIP (δ) is a better overall choice as a backdoor purification technique.

Strong Backdoor Attacks With High Poison Rates. By increasing the poison rates, we create stronger versions of different attacks against which most defense techniques fail quite often. We use 3 different poison rates, $\{25\%, 35\%, 50\%\}$. We show in Table 8 that FIP is capable of defending very well even with a poison rate of 50%, achieving a significant ASR improvement over FT. Furthermore, there is a sharp difference in classification accuracy between FIP and other defenses. For 25% Blend attack, however, ANP offers a slightly better performance than our method. However, ANP performs poorly in removing the backdoor as it obtains an ASR of 29.96% compared to 0.83% for FIP.

Label Correction Rate. In the standard backdoor removal metric, it is sufficient for backdoored images to be classified as a non-target class (any class other than y_b). However, we also consider another metric, label correction rate (LCR), for quantifying the success of a defense. *We define LCR as the percentage of poisoned samples correctly classified to their original classes.* Any method with the highest value of LCR is considered to be the best defense method. For this evaluation, we use CIFAR10 dataset and 12 backdoor attacks. Initially, the correction rate is 0% with no defense as the ASR is close to 100%. Table 9 shows that FIP effectively corrects the adversary-set target label to the original ground truth label. For example, we obtain an average ~2% higher label correction rate than AWM.

Effect of Clean Validation Data Size. We also provide insights on how fine-tuning with clean validation data impacts the purification performance. In Table 10, we see the change in performance while gradually reducing the validation size from 1% to 0.02%. Even with only 50 (0.1%) data points, FIP can successfully remove the backdoor by bringing down the attack success rate (ASR) to 5.91%. In an extreme scenario of one-shot FIP, we have only one sample from each class to fine-tune the model. Our proposed method is able to tackle the backdoor issue even in such a scenario. We consider AWM and ANP for this comparison. For both ANP and AWM, reducing the validation size has a severe impact on test accuracy (ACC). We consider Badnets attack on the CIFAR10 dataset for this evaluation.

Effect of Different Architectures. We further validate the effectiveness of our method under different network settings. In Table 11, we show the performance of FIP with some of the widely used architectures such as VGG-16 [59], EfficientNet [65] and Vision Transformer (VIT) [18]. Here, we consider a smaller version of ViT-S with 21M parameters. FIP is able to remove backdoors irrespective of the network architecture. This makes sense as most of the architecture uses either fully connected or convolution layers, and FIP can be implemented in both cases.

Combining Different Backdoor Attacks. We also perform experiments with combined backdoor attacks. To create such attacks, we poison some portion of the training data using three different attacks; Badnets, Blend, and Trojan. Each of these attacks has an equal share in the poison data. As shown in Table 12, we use four different poison rates: 10% ~ 50%. FIP outperforms other baseline methods (MCR and ANP) by a satisfactory margin.

More All2All Attacks. Most of the defenses evaluate their methods on only All2One attacks, where we consider only one target label. However, there can be multiple target classes in a practical attack scenario. We consider one such case: All2All attack, where target classes are uniformly distributed among all available classes. In Table 13, we show the performance under such settings for three different attacks with a poison rate of 10%. It shows that the All2All attack is more challenging to defend against as compared to the All2One attack. However, the performance of FIP seems to be consistently better than other defenses for both of these attack variations. For reference, we achieve an ASR improvement of 3.12% over ANP while maintaining a lead in classification accuracy too.

Adaptive Attacks. We follow Eq. 5 in our paper to simulate an adaptive attack. Table 14 shows that as we increase η_F, the model becomes smoother while it becomes harder to insert the backdoor, hence, the ASR drops. However, f-FIP successfully purifies the model even with such adaptive attacks. For larger η_F, we obtain a higher drop in ACC. The underlying reason for this could be that the convergence point of the backdoor model is more favorable to clean distribution. Applying f-FIP would shift the model from that convergence point and cause this undesirable ACC drop.

t-SNE Visualization of Cluster Structures. In Figure 5, we visualize the class clusters before and after backdoor purification. We take CIFAR10 dataset with Badnets attack for this visualization. For visualization purposes only, we assign the label "0" to the clean data cluster from the target class and the label "11" to the poison data cluster. However, both of these clusters have the same training label "0" during backdoor training. Figure 5b clearly indicates that our proposed method can break the poison data clusters and reassign them to their original class cluster.

7 Conclusion

In this work, we analyzed the backdoor insertion and removal process from a novel perspective—the smoothness of the model's loss surface—showing that the backdoor model converged to a sharp minimum compared to a benign model's convergence point. To remove the effect of backdoor, we proposed to re-optimize the model to smooth minima. Following our analysis, we proposed a novel backdoor purification technique using the knowledge of the Fisher-Information matrix to remove the backdoor efficiently instead of using naïve (e.g., general-purpose ones) optimization techniques to re-optimize. Furthermore, to preserve the post-purification clean test accuracy of the model, we introduced a novel clean data distribution-aware regularizer. Last but not least, a faster version of FIP has been proposed where we only fine-tuned the singular values of weights instead of directly fine-tuning the weights. FIP achieves SOTA performance in terms of running time and accuracy in a wide range of benchmarks, including four different tasks and ten benchmark datasets against 14 SOTA backdoor attacks.

Limitations. Here, we discussed a couple of limitations of our proposed and, hence, corresponding future works to address those. **First**, it is observable that no matter which defense techniques we use, the clean test accuracy (ACC) consistently drops for all datasets. Here, we try to explain the reason behind this, especially for fine-tuning-based techniques, as FIP is one of them. Since these techniques use a small validation set for fine-tuning, they do not necessarily cover the whole training data distribution. Therefore, fine-tuning with this small amount of data bears the risk of over-fitting and reduced clean test accuracy. While our clean accuracy retainer partially solves this issue, more rigorous and sophisticated methods must be designed to fully alleviate this issue.

Second, while our method is based on thorough empirical analysis and corresponding theoretical justification, there is no theoretical guarantee whether the proposed method provably removes backdoor from a pre-trained model (which is out-of-scope of this work). However, in the case of resource-constraints safety-critical systems, it is often necessary to use a pre-trained model; hence, provable backdoor defense is necessary for safety-critical applications. In future work, we will focus on provable backdoor defense for safety-critical applications.

Acknowledgements

We thank the anonymous reviewers for their insightful feedback. This work was supported in part by the National Science Foundation under Grant ECCS-1810256 and CMMI-2246672.

References

[1] Sabbir Ahmed, Abdullah Al Arafat, Mamshad Nayeem Rizve, Rahim Hossain, Zhishan Guo, and Adnan Siraj Rakin. 2023. SSDA: Secure Source-Free Domain Adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19180–19190.

[2] Shun-Ichi Amari. 1998. Natural gradient works efficiently in learning. *Neural computation* 10, 2 (1998), 251–276.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Mauro Barni, Kassem Kallas, and Benedetta Tondi. 2019. A new backdoor attack in cnns by training set corruption without label poisoning. In *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 101–105.

[5] Yoav Benyamini and Joram Lindenstrauss. 1998. *Geometric nonlinear functional analysis*. Vol. 48. American Mathematical Soc.

[6] Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Baltimore, Maryland, USA, 12–58. https://doi.org/10.3115/v1/W14-3302

[7] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. 2021. Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3855–3859.

[8] Stephen P Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[9] Shuwen Chai and Jinghui Chen. 2022. One-shot Neural Backdoor Erasing via Adversarial Weight Masking. *arXiv preprint arXiv:2207.04497* (2022).

[10] Kangjie Chen, Xiaoxuan Lou, Guowen Xu, Jiwei Li, and Tianwei Zhang. 2023. Clean-image Backdoor: Attacking Multi-label Models with Poisoned Labels Only. In *The Eleventh International Conference on Learning Representations*.

[11] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).

[12] Siyuan Cheng, Yingqi Liu, Shiqing Ma, and Xiangyu Zhang. 2021. Deep feature space trojan attack of neural networks by controlled detoxification. In *AAAI*, Vol. 35. 1148–1156.

[13] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*. PMLR, 1310–1320.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. IEEE, 248–255.

[15] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. 2019. Sever: A robust meta-algorithm for stochastic optimization. In *International Conference on Machine Learning*. PMLR, 1596–1606.

[16] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. 2020. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*. 897–912.

[17] Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li. 2021. Lira: Learnable, imperceptible and robust backdoor attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 11966–11976.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[19] Min Du, Ruoxi Jia, and Dawn Song. 2019. Robust anomaly detection and backdoor attack detection via differential privacy. *arXiv preprint arXiv:1911.07116* (2019).

[20] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88 (2010), 303–338.

[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. [n. d.]. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[22] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware Minimization for Efficiently Improving Generalization. In *International Conference on Learning Representations*. https://openreview.net/forum?id=6Tm1mposlrM

[23] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *International conference on machine learning*. PMLR, 1243–1252.

[24] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* 7 (2019), 47230–47244.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European conference on computer vision*. Springer, 630–645.

[26] Sanghyun Hong, Varun Chandrasekaran, Yiğitcan Kaya, Tudor Dumitraş, and Nicolas Papernot. 2020. On the effectiveness of mitigating data poisoning attacks with gradient shaping. *arXiv preprint arXiv:2002.11497* (2020).

[27] Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. 2022. Backdoor defense via decoupling the training process. *arXiv preprint arXiv:2202.03423*

(2022).

[28] Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. 2020. The break-even point on optimization trajectories of deep neural networks. *arXiv preprint arXiv:2002.09572* (2020).

[29] Sekitoshi Kanai, Masanori Yamada, Hiroshi Takahashi, Yuki Yamanaka, and Yasutoshi Ida. 2023. Relationship between nonsmoothness in adversarial training, constraints of attacks, and flatness in the input space. *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[30] Nazmul Karim, Abdullah Al Arafat, Umar Khalid, Zhishan Guo, and Nazanin Rahnavard. 2024. Augmented Neural Fine-Tuning for Efficient Backdoor Purification. In *The 18th European Conference on Computer Vision (ECCV)*.

[31] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*.

[32] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*. PMLR, 1885–1894.

[33] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[34] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. 2011. HMDB: a large video database for human motion recognition. In *2011 International conference on computer vision*. IEEE, 2556–2563.

[35] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. 2021. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning*. PMLR, 5905–5914.

[36] Ya Le and Xuan Yang. 2015. Tiny imagenet visual recognition challenge. *CS 231N* 7, 7 (2015), 3.

[37] Xinke Li, Zhirui Chen, Yue Zhao, Zekun Tong, Yabang Zhao, Andrew Lim, and Joey Tianyi Zhou. 2021. Pointba: Towards backdoor attacks in 3d point cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 16492–16501.

[38] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. 2021. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 16463–16472.

[39] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Anti-backdoor learning: Training clean models on poisoned data. *Advances in Neural Information Processing Systems* 34 (2021), 14900–14912.

[40] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=9l0K4OM-oXE

[41] Yige Li, Xixiang Lyu, Xingjun Ma, Nodens Koren, Lingjuan Lyu, Bo Li, and Yu-Gang Jiang. 2023. Reconstructive Neuron Pruning for Backdoor Defense. *arXiv preprint arXiv:2305.14876* (2023).

[42] Junyu Lin, Lei Xu, Yingqi Liu, and Xiangyu Zhang. 2020. Composite backdoor attack for deep neural network by mixing existing benign features. In *CCS*. 113–131.

[43] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 740–755.

[44] Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. 2020. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. *Advances in Neural Information Processing Systems* 33 (2020), 21476–21487.

[45] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 273–294.

[46] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2017. Trojaning attack on neural networks. (2017).

[47] Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. 2020. Reflection backdoor: A natural backdoor attack on deep neural networks. In *European Conference on Computer Vision*. Springer, 182–199.

[48] Yuntao Liu, Yang Xie, and Ankur Srivastava. 2017. Neural trojans. In *2017 IEEE International Conference on Computer Design (ICCD)*. IEEE, 45–48.

[49] Shiqing Ma and Yingqi Liu. 2019. Nic: Detecting adversarial samples with neural network invariant checking. In *Proceedings of the 26th network and distributed system security symposium (NDSS 2019)*.

[50] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[51] Naren Manoj and Avrim Blum. 2021. Excess capacity and backdoor poisoning. *Advances in Neural Information Processing Systems* 34 (2021), 20373–20384.

[52] Anh Nguyen and Anh Tran. 2021. WaNet–Imperceptible Warping-based Backdoor Attack. *arXiv preprint arXiv:2102.10369* (2021).

[53] Tuan Anh Nguyen and Anh Tran. 2020. Input-aware dynamic backdoor attack. *Advances in Neural Information Processing Systems* 33 (2020), 3454–3464.

[54] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems* 30 (2017).

[55] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben-Baruch, and Asaf Noy. 2023. Ml-decoder: Scalable and versatile classification head. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 32–41.

[56] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. 2020. Hidden trigger backdoor attacks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 11957–11965.

[57] Aniruddha Saha, Ajinkya Tejankar, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. 2022. Backdoor attacks on self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13337–13346.

[58] Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena* 404 (2020), 132306.

[59] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[60] Aman Sinha, Hongseok Namkoong, and John Duchi. 2018. Certifiable Distributional Robustness with Principled Adversarial Training. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hk6kPgZA-

[61] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).

[62] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2011. The German traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*. IEEE, 1453–1460.

[63] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. 2017. Certified defenses for data poisoning attacks. *Advances in neural information processing systems* 30 (2017).

[64] Xiaofei Sun, Xiaoya Li, Yuxian Meng, Xiang Ao, Lingjuan Lyu, Jiwei Li, and Tianwei Zhang. 2023. Defending against backdoor attacks in natural language generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 5257–5265.

[65] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*. PMLR, 6105–6114.

[66] Brandon Tran, Jerry Li, and Aleksander Madry. 2018. Spectral signatures in backdoor attacks. *Advances in neural information processing systems* 31 (2018).

[67] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2018. Clean-label backdoor attacks. (2018).

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[69] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 707–723.

[70] Zhenting Wang, Juan Zhai, and Shiqing Ma. 2022. BppAttack: Stealthy and Efficient Trojan Attacks against Deep Neural Networks via Image Quantization and Contrastive Adversarial Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15074–15084.

[71] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. 2022. Backdoorbench: A comprehensive benchmark of backdoor learning. *Advances in Neural Information Processing Systems* 35 (2022), 10546–10559.

[72] Dongxian Wu and Yisen Wang. 2021. Adversarial Neuron Pruning Purifies Backdoored Deep Models. In *NeurIPS*.

[73] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1912–1920.

[74] Zhen Xiang, David J Miller, Siheng Chen, Xi Li, and George Kesidis. 2021. A backdoor attack against 3d point cloud classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7597–7607.

[75] Yi Zeng, Si Chen, Won Park, Z Morley Mao, Ming Jin, and Ruoxi Jia. 2021. Adversarial unlearning of backdoors via implicit hypergradient. *arXiv preprint arXiv:2110.03735* (2021).

[76] Yi Zeng, Won Park, Z Morley Mao, and Ruoxi Jia. 2021. Rethinking the backdoor attacks' triggers: A frequency perspective. In *Proceedings of the IEEE/CVF international conference on computer vision*. 16473–16481.

[77] Quan Zhang, Yifeng Ding, Yongqiang Tian, Jianmin Guo, Min Yuan, and Yu Jiang. 2021. AdvDoor: adversarial backdoor attack of deep learning system. In

Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis. 127–138.

[78] Zaixi Zhang, Qi Liu, Zhicai Wang, Zepu Lu, and Qingyong Hu. 2023. Backdoor Defense via Deconfounded Representation Learning. arXiv:2303.06818 [cs.AI]

[79] Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. 2020. Bridging mode connectivity in loss landscapes and adversarial robustness. *arXiv preprint arXiv:2005.00060* (2020).

[80] Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. 2020. Clean-label backdoor attacks on video recognition models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14443–14452.

[81] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. 2022. Data-free backdoor removal based on channel lipschitzness. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*. Springer, 175–191.

[82] Haoti Zhong, Cong Liao, Anna Cinzia Squicciarini, Sencun Zhu, and David Miller. 2020. Backdoor embedding in convolutional neural network models via invisible perturbation. In *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*. 97–108.

[83] Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baoyuan Wu. 2023. Enhancing Fine-Tuning Based Backdoor Defense with Sharpness-Aware Minimization. *arXiv preprint arXiv:2304.11823* (2023).

A Proof of Theorem 1

Proof. Let us consider a training set $\{\boldsymbol{x}, y\} = \{\boldsymbol{x}_c, y_c\} \cup \{\boldsymbol{x}_b, y_b\}$, where $\{\boldsymbol{x}_c, y_c\}$[2] is the set of clean samples and $\{\boldsymbol{x}_b, y_b\}$ is the set of backdoor or poison samples. In our work, we estimate the loss Hessian *w.r.t. standard data distribution*, i.e. training samples with their *ground truth labels*.

First, let us consider the scenario where we optimize a DNN (f_c) on $\{\boldsymbol{x}_c, y_c\}$ only (benign model). From the L_c–Lipschitz property of loss-gradient (ref. Assumption 1, Eq. (2)) corresponding to *any clean sample*[3] $\boldsymbol{x}_c$, we get

$$||\nabla_\theta \ell(\boldsymbol{x}_c, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_c, \theta_2)|| \leq L_c ||\theta_1 - \theta_2||, \ \forall \theta_1, \theta_2 \in \Theta \tag{10}$$

Now, consider the backdoor model training (f_b) setup, where both clean and poison samples are used concurrently for training. In such a scenario, a training sample can be either clean or poisoned. As we are using standard data distribution, we calculate the loss (ℓ) corresponding to $\{\boldsymbol{x}_c, y_c\} \cup \{\boldsymbol{x}_b, y_c\}$; where y_c indicates the original ground truth (GT) label. Let us bound the difference of loss gradient for backdoor training setup for any sample,

$$\begin{aligned} ||\nabla_\theta \ell(\boldsymbol{x}, \theta_1) - &\nabla_\theta \ell(\boldsymbol{x}, \theta_2)|| \\ &\overset{(i)}{\leq} \max\{||\nabla_\theta \ell(\boldsymbol{x}_c, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_c, \theta_2)||, \\ &\quad ||\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)||\} \\ &\overset{(ii)}{=} ||\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)|| \\ &\overset{(iii)}{\leq} L_b ||\theta_1 - \theta_2|| \end{aligned} \tag{11}$$

here, step *(i)* follows trivially as $||\nabla_\theta \ell(\boldsymbol{x}, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}, \theta_2)||$ holds for any $\boldsymbol{x}$. Unlike f_c and f_b, we can have loss gradients corresponding to samples from clean and poison sets; *(ii)* leverages the properties of backdoor training where the backdoor is inserted by forcing f_b to memorize the specific pattern or trigger δ, specifically the mapping of $\delta \longrightarrow y_b$. At the same time, f_b learns (does not memorize) image or object-related generic patterns in $\boldsymbol{x}_c$ and maps them to y_c, similar to f_c. Let us denote the optimized parameters of f_b as, θ_1. Since f_b is optimized to predict y_b, we have a high loss gradient $\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1)$ if we consider GT label y_c for $\boldsymbol{x}_b$. Note that the backdoor model becomes too sensitive to the trigger due to the memorization effect. However, a certain group of parameters ($\theta^{(b)}$) show far more sensitivity to the backdoor as compared to others ($\theta^{(c)}$), where $\theta = \theta^{(c)} \cup \theta^{(b)}$ and $|\theta^{(b)}| << |\theta^{(c)}|$. This has also been shown in previous studies [9, 41, 72]. Therefore, even a small change to $\theta^{(b)}$ will make the backdoor model show significantly less (or no) sensitivity to the trigger. On the other hand, a small change in $\theta^{(c)}$ has very little impact on recognizing image-related generic patterns. Now, consider a scenario where θ_1 is slightly changed to θ_2. Due to this shift, the loss gradient $\nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)$ becomes significantly smaller if we calculate it *w.r.t.* y_c. This happens for the following reasons: (1) Due to the change in $\theta^{(b)}$, the model does not show sensitivity towards δ anymore. (2) As mentioned before, with small change in $\theta^{(c)}$, the model can still recognize patterns in samples. This means the model ignores δ in $\boldsymbol{x}_b (= \boldsymbol{x} + \delta)$ while recognizing image-related patterns in $\boldsymbol{x}$ and predicting the GT label y_c. Therefore, the change in loss gradient $(||\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)||)$ is large. On the other hand, due to the reason (2), the change in loss gradient $(||\nabla_\theta \ell(\boldsymbol{x}_c, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_c, \theta_2)||)$ is smaller. Finally, we can write the following,

$$\begin{aligned} ||\nabla_\theta \ell(\boldsymbol{x}_c, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_c, \theta_2)|| &\leq L_c ||\theta_1 - \theta_2|| \\ ||\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)|| &\leq L_b ||\theta_1 - \theta_2|| \end{aligned} \tag{12}$$

In our above discussion, we have shown that

$$||\nabla_\theta \ell(\boldsymbol{x}_c, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_c, \theta_2)|| < ||\nabla_\theta \ell(\boldsymbol{x}_b, \theta_1) - \nabla_\theta \ell(\boldsymbol{x}_b, \theta_2)||$$

Therefore, for the same set of θ_1, θ_2, Eq. 12 suggests that $L_c < L_b$. Note that $L_c < L_b$ holds if we consider the smallest Lipschitz constant for Eq. 12 [5] *(iii)* follows the definition of smoothness.

Hence, the loss of the backdoor model is L_b–Smooth and $L_c < L_b$. □

Takeaway from the theoretical analysis. Based on Theorem 1, we can rewrite Eq. 3 for a backdoor model,

$$\sup_\theta \sigma(\nabla^2_\theta \mathcal{L}) \leq \max\{L_c, L_b\} \tag{13}$$

where $\mathcal{L}$ is the loss-function of the backdoor model computed over $\{\boldsymbol{x}, y\} = \{\boldsymbol{x}_c, y_c\} \cup \ \{\boldsymbol{x}_b, y_c\}$.

The R.H.S. of Eq. 13 represents the supremum[4] for smoothness of a backdoor model. From Eq. (11), it can be observed that $L_c < L_b$ which leads to the following form of Eq. 13,

$$\sup_\theta \sigma(\nabla^2_\theta \mathcal{L}) \leq L_b \tag{14}$$

Hence, a backdoor model tends to show less smoothness on $\mathcal{L}$, computed over $\{\boldsymbol{x}, y\} = \{\boldsymbol{x}_c, y_c\} \cup \ \{\boldsymbol{x}_b, y_c\}$, as compared to a benign model with L_c–Lipschitz continuity.

[2] Note that we use $\{\boldsymbol{x}_c, y_c\}$ to denote clean samples whereas $\{\boldsymbol{x}, y\}$ was used in the main paper to denote all training samples. We start with a clean training set, $\{\boldsymbol{x}, y\}$, and then add the trigger to some of the samples $\boldsymbol{x}_b$ with target label y_b that produce poison set, $\{\boldsymbol{x}_b, y_b\}$.

[3] Here, loss-gradient corresponding to clean sample means we first compute the loss using clean sample and then take the gradient.

[4] We used the definition of supremum (https://en.wikipedia.org/wiki/Infimum_and_supremum) slightly differently than the formal definition. By supremum, we indicate that $\max\{L_c, L_b\}$ is the lowest value for the Lipschitz constant of the backdoor model $f_b(.)$ to hold Eq. (13).