FUN: FILTER-BASED UNLEARNABLE DATASETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale training of modern deep learning models heavily relies on publicly available data on the web. This potentially unauthorized usage of online data leads to concerns regarding data privacy. Recent works aim to make unlearnable data for deep learning models by adding small, specially designed noises to tackle this issue. However, these methods are vulnerable to adversarial training (AT) and/or are computationally heavy. In this work, we propose a novel, model-free convolutional Filter-based UNlearnable (FUN) dataset generation technique. FUN performs controlled class-wise convolutions using filters that are randomly generated via a private key. FUN encourages the network to learn the relation between filters and labels rather than informative features for classifying the clean data. We develop some theoretical analysis demonstrating that FUN can successfully poison Gaussian mixture data by reducing the clean data performance of the optimal Bayes classifier. We also empirically demonstrate the effectiveness of FUN with various datasets (CIFAR-10, CIFAR-100, and ImageNet-100), and architectures (ResNet-18, VGG-16, Wide ResNet-34-10, and DenseNet-121). Our experiments show that FUN is robust to various data augmentations and training approaches such as smoothing, AT with different budgets, transfer learning, and fine-tuning. For instance, training a ResNet-18 on FUN ImageNet-100 data achieves only 8.96%, 40.08%, and 20.58% clean test accuracies with empirical risk minimization (ERM), L_∞ AT, and L_2 AT, respectively. Here, ERM on the clean training data achieves a clean test accuracy of 80.66%. Furthermore, we also show that FUN is robust to adaptive defenses designed specifically to break it.

1 INTRODUCTION

Modern deep learning training frameworks heavily depend on large-scale datasets for achieving high accuracy. This encourages deep learning practitioners to scrape data from the web for data collection (Lin et al., 2014; Sariyildiz et al., 2020; Desai & Johnson, 2021; Radford et al., 2021). Since a lot of the data is publicly available online, sometime this scrapping of data is unauthorized. For instance, a recent article (Hill, 2020) discloses that a private company trained a commercial face recognition system using over three billion facial images collected from the internet without any user consent. Although such massive data can significantly boost the performance of deep learning models, it raises serious concerns about data privacy and security.

To prevent the unauthorized usage of personal data, a series of recent papers (Huang et al., 2021; Fowl et al., 2021; Yuan & Wu, 2021) propose to poison data with additive noise. The idea is to make datasets *unlearnable* for deep learning models by ensuring that they learn the correspondence between noises and labels. Thereby, they do not learn much useful information about the clean data, significantly degrading their clean test accuracy. However, in Huang et al. (2021); Tao et al. (2021); Fu et al. (2021), these unlearnability methods are shown to be vulnerable to adversarial training (AT) frameworks (Madry et al., 2017). Motivated by this problem, Fu et al. (2021) developed Robust Error-Minimization (REM) noises to make unlearnable data that is protected from AT. While the authors show the effectiveness of REM in multiple scenarios, we demonstrate that these methods are still not robust against different data augmentations or training settings (see Section 3.2). Furthermore, current unlearnability frameworks (Fu et al., 2021; Huang et al., 2021; Fowl et al., 2021; Yuan & Wu, 2021) are model-dependent and require expensive optimization steps on deep learning models to obtain the additive noises. They also need to train the deep learning models from scratch to obtain noises for each new data set.

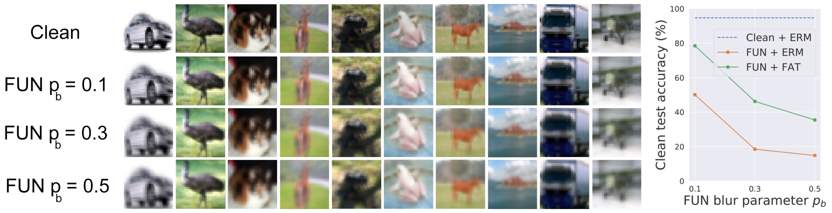


Figure 1: FUN CIFAR-10 images from each of the 10 classes convolved using convolutional filters generated with blur parameter p_b. As seen in the plot, a higher p_b value results in better unlearnability (lower clean test accuracy), but increased blurring. FUN CIFAR-10 does not break with ERM for all the three p_b values. In Section 5.2, we propose Filter Adversarial Training (FAT) that we specifically design to break FUN. FAT adversarially learns class-wise filters to deconvolve FUN images. However, FUN CIFAR-10 with $p_b \in \{0.3, 0.5\}$ does not break with FAT. Therefore, we select $p_b = 0.3$ (lesser blurring) for CIFAR-10 as discussed in Section 5.1. More details on FAT in Appendix A.8.

In this paper, we propose a novel convolutional Filter-based UNlearnable (FUN) dataset generation technique. We address limitations of existing unlearnable data generation techniques and study the effects of our proposed filter-based poisoning attack. For generating FUN datasets, an attacker randomly generates different convolutional filters for each class in the dataset using *a private key* or seed value. These filters are used to perform *controlled class-wise convolutions* on the clean training dataset to obtain the FUN dataset. As we describe in Sections 3.3 and 5.2, FUN dataset generation performs controlled convolutions using a blur parameter p_b to ensure that the semantics of the dataset are preserved (see Figure 1). FUN data generation with a lower blurring parameter p_b adds less perceptible noises to clean samples. A network trained by a defender on FUN dataset is encouraged to learn the relation between class-wise filters and labels rather than useful features for classifying the clean data. Since the seed value for generating the filters are private, its not possible for the defender to obtain clean data from the FUN data. While the existing unlearnability works use *additive* noises, FUN enjoys the advantage of introducing *multiplicative* noises in the Fourier domain. This lets FUN to add higher amounts of noise in the image space, specifically along the edges in images, and makes it resilient to AT with small additive noise budgets. In Figure 2, with the help of t-SNE plots (van der Maaten & Hinton, 2008), we also find that the noises added by FUN is *not* linearly separable while they are linearly separable for the existing works on unlearnability (Yu et al., 2022).

In Section 4, we theoretically show that FUN can successfully poison Gaussian mixture data by degrading the clean data accuracy of the optimal Bayes classifier. We state our result informally below while the formal version is presented in Theorem 2.

Theorem 1 (Informal) *Let $\mathcal{D}$ denote a Gaussian mixture data with two modes, $P_{\mathcal{D}}$ denote the optimal Bayes classifier trained on $\mathcal{D}$, and $\tau_{\mathcal{D}}(P_{\mathcal{D}})$ denote the accuracy of the classifier $P_{\mathcal{D}}$ on $\mathcal{D}$. Then, under some assumptions, for every clean data $\mathcal{D}$, FUN can generate a perturbed dataset $\tilde{\mathcal{D}}$ such that $\tau_{\mathcal{D}}(P_{\tilde{\mathcal{D}}}) < \tau_{\mathcal{D}}(P_{\mathcal{D}})$.*

Furthermore, our empirical experiments in Section 5 demonstrate the effectiveness of FUN under various training scenarios such as ERM with various augmentations and regularizations, AT with different budgets, randomized smoothing (Cohen et al., 2019; Levine & Feizi, 2020b; Kumar et al., 2020), transfer learning, and fine-tuning. Figure 2 shows the effectiveness of FUN on ResNet-18 with CIFAR-10 dataset training. For instance, training a ResNet-18 on FUN CIFAR-10 achieves only 18.48%, 44.4%, and 51.14% clean test accuracies with ERM, L_∞ AT, and L_2 AT, respectively. Here, ERM on the clean training data achieves a clean test accuracy of 94.66%. In addition, we also design adaptive defenses to investigate if FUN breaks with random or adversarial defense mechanisms. We find that FUN is robust to the adaptive defenses that we specifically design to break it.

2 Related Works

Our FUN data generation technique is intimately related with adversarial and poisoning attacks. We first discuss some of this literature and then explain their relation with FUN data generation.

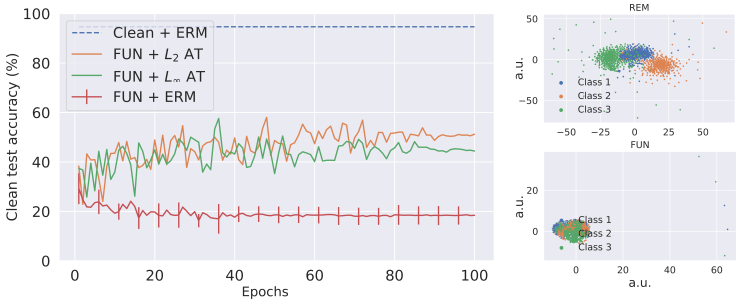


Figure 2: In the left column, clean test accuracies of ResNet-18 on clean and FUN CIFAR-10 datasets are shown. The lower the test accuracy, the higher the effectiveness of unlearnability is. We show that FUN is robust to various training settings such as ERM, L_2 AT, and L_∞ AT. The error bars represent the standard deviation of test accuracy over 5 independent trials. FUN is robust to the randomness in its data generation process. In the right column, t-SNEs of the noises generated by REM (top) and FUN (bottom) for the first 3 classes of CIFAR-10 are plotted.

Adversarial attacks. Adversarial examples are specially designed examples that can fool deep learning models at test time (Szegedy et al., 2013; Goodfellow et al., 2014; Carlini & Wagner, 2017; Kurakin et al., 2018; Laidlaw & Feizi, 2019). The adversary crafts these examples by adding error-maximizing noises to the clean data. Even slightly perturbed data can serve as adversarial examples. AT is a training framework proposed to make deep learning models robust to adversarial examples (Madry et al., 2017; Zhang et al., 2019; Javanmard & Soltanolkotabi, 2020; Laidlaw et al., 2020; Wang et al., 2021). AT is a min-max optimization problem where the model is trained to minimize loss on adversarial examples that have the maximum loss.

Poisoning attacks. In data poisoning, an attacker aims to hurt the deep learning model's performance by perturbing the training data (Biggio et al., 2012; Koh & Liang, 2017; Shafahi et al., 2018; Levine & Feizi, 2020a; Liu et al., 2020a; Wang et al., 2022). The backdoor attack is a special type of poisoning attack where a trigger pattern is injected into clean data at training time (Chen et al., 2017; Liu et al., 2020b; Nguyen & Tran, 2020; Li et al., 2021). The model trained on this data would misclassify an image with a trigger pattern at test time. Gu et al. (2017); Li et al. (2020) use perceptible amounts of noises similar to FUN for data poisoning. However, backdoor attacks do not affect the performance of the model on clean data (Chen et al., 2017; Shafahi et al., 2018; Barni et al., 2019).

Recent literature utilize data poisoning to protect data from being used for model training without authorization. Yuan & Wu (2021) use neural tangent kernels (Jacot et al., 2018) to generate clean label attacks that can hurt the generalization of deep learning models. Huang et al. (2021) show that error-minimizing noise addition can serve as a poisoning technique. Fowl et al. (2021) show that error-maximizing noises as well can make strong poison attacks. However, all these poisoning techniques do not offer data protection with AT (Yu et al., 2022; Tao et al., 2021). Fu et al. (2021) proposes a min-min-max optimization technique to generate poisoned data that offers better unlearnability effects with AT.

3 Filter-based UNlearnable (FUN) Datasets

In this section, first we give some preliminaries about unlearnability. Then we discuss the limitations of the existing unlearnability methods. Finally, we propose our FUN technique.

3.1 Preliminaries

Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \sim \mathcal{D}^n$ be the clean training dataset where $\mathcal{D}$ is the clean data distribution, $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ are the samples, and $y_i \in \mathcal{Y}$ are the corresponding labels. Suppose a network is given as $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ where $\theta \in \Theta$ is the network parameter and $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is the loss function. ERM trains a network using a minimization problem of the form: $\min_\theta \frac{1}{n} \sum_{i=1}^n \ell(f_\theta(\boldsymbol{x}_i), y_i)$. Standard L_p-

Table 1: Time taken for generating unlearnable noises for various datasets.

Dataset	EM	TAP	NTGA	REM	**FUN (ours)**
CIFAR-10	0.4 hr	0.5 hr	5.2 hrs	22.6 hrs	**10.8 s**
CIFAR-100	0.4 hr	0.5 hr	5.2 hrs	22.6 hrs	**15.5 s**
ImageNet-100	3.9 hrs	5.2 hrs	14.6 hrs	51.2 hrs	**0.15 hr**

based AT (for $p \in \mathbb{R}^+$) solves the following min-max problem: $\min_\theta \frac{1}{n}\sum_{i=1}^{n} \max_{\|\boldsymbol{\delta}_i\|_p \leq \rho_a} \ell(f_\theta(\boldsymbol{x}_i + \boldsymbol{\delta}_i), y_i)$ where ρ_a is the adversarial perturbation radius.

We will use $\tau_{\mathcal{D}}(\theta)$, with θ the clean model parameter, to denote the clean test accuracy of a model trained on the clean training dataset (i.e., clean model accuracy). In unlearnable dataset generation, an attacker uses an algorithm $\mathcal{A} : \mathcal{X} \to \mathcal{X}$ to generate an unlearnable dataset $\{(\tilde{\boldsymbol{x}}_i = \mathcal{A}(\boldsymbol{x}_i), y_i)\}_{i=1}^n \sim \tilde{\mathcal{D}}^n$ from the clean training data. Here, the attacker assumes access to the full clean training datatset. Moreover, the attacker cannot modify the unlearnable dataset once it is released publicly. A defender trains using the unlearnable dataset to obtain a network $f_{\tilde{\theta}}$. The objective of the attacker is to design an unlearnable dataset such that the defender's model trained on the unlearnable data achieves a clean test accuracy (i.e., unlearnable model accuracy) worse than the clean model accuracy i.e. $\tau_{\mathcal{D}}(\tilde{\theta}) \ll \tau_{\mathcal{D}}(\theta)$.

3.2 Limitations of existing works

Fu et al. (2021) show that the previous unlearnability methods including Error-Minimization (EM) (Huang et al., 2021), Targeted Adversarial Poisoning (TAP) (Fowl et al., 2021), and Neural Tangent Generalization Attack (NTGA) (Yuan & Wu, 2021) are vulnerable to AT. Hence, they propose a Robust Error-Minimization (REM) (Fu et al., 2021) method that exploits a min-min-max optimization procedure to generate unlearnable noises. REM first trains a noise generator f'_θ on data points $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ over a loss function ℓ as follows

$$\min_\theta \frac{1}{n}\sum_{i=1}^{n} \min_{\|\boldsymbol{\delta}_i^u\| \leq \rho_u} \mathbb{E}_{t\sim T} \max_{\|\boldsymbol{\delta}_i^a\| \leq \rho_a} \ell(f'_\theta(t(\boldsymbol{x}_i + \boldsymbol{\delta}_i^u) + \boldsymbol{\delta}_i^a), y). \tag{1}$$

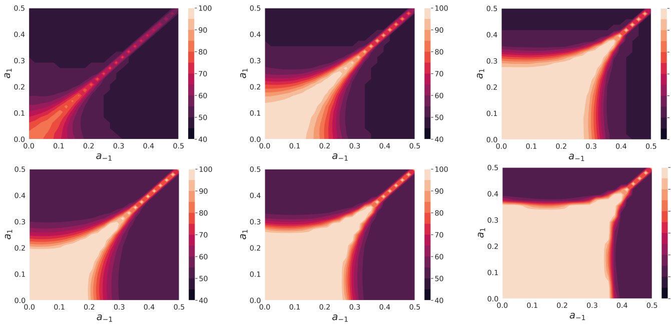
Here, T is a distribution over a set of transformations $\{t : \mathcal{X} \to \mathcal{X}\}$, ρ_u is the defensive perturbation radius, and ρ_a controls the protection level of REM against AT. After training the noise generator, an unlearnable example $(\tilde{\boldsymbol{x}}, y)$ is generated via

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \underset{\|\boldsymbol{\delta}^u\| \leq \rho_u}{\arg\min}\, \mathbb{E}_{t\sim T} \max_{\|\boldsymbol{\delta}^a\| \leq \rho_a} \ell(f'_\theta(t(\boldsymbol{x} + \boldsymbol{\delta}^u) + \boldsymbol{\delta}^a), y) \tag{2}$$

First, note that REM is computationally expensive since it needs to generate unlearnability noises through solving optimization equation 2. Moreover, the existing techniques are model-dependent and they require gradient-based training with a neural network to generate unlearnable data. They also require neural network training from scratch for every dataset that is to be made unlearnable. Table 1 shows the amount of time required to generate various unlearnable datasets using NVIDIA® Tesla V100 GPU and 10 CPU cores. FUN dataset generation is significantly faster than the existing methods since it uses a model-free approach. Furthermore, REM is sensitive to hyperparameters and norm-budgets of AT since they generate noises with fixed L_∞ norm budgets. For instance, ResNet-18 trained on clean CIFAR-10 dataset achieves a clean test data accuracy of 94.66%. L_∞ AT with perturbation radius $\rho_a = 4/255$ on REM CIFAR-10 data (generated using $\rho_u = 8/255$ and $\rho_a = 4/255$) achieves only a clean test accuracy of 48.16%. However, L_∞ AT with perturbation radius $\rho_a = 8/255$ and L_2 AT with perturbation radius $\rho_a = 0.75$ can achieve a clean test accuracy of 78.71% and 79.65% on the same REM data, respectively. We also find that ERM with ResNet-18 on grayscaled REM CIFAR-10 images can achieve a high test accuracy of 70.76% on the grayscaled CIFAR-10 test data. This shows that REM relies upon the color space for poisoning clean data. Fu et al. (2021) also show that REM noise generated using ResNet-18 is not transferable to DenseNet-121.

3.3 Our method: FUN

FUN uses convolutional filters $s_i \in \mathbb{R}^{k\times k}$ for each class $i \in [1, K]$. A random parameter, out of the k^2 parameters, in each of the filters is set to have a value of 1. The rest of the filter parameters are



(a) $\|\boldsymbol{\mu}\|_2 = 1.0$ (b) $\|\boldsymbol{\mu}\|_2 = 2.5$ (c) $\|\boldsymbol{\mu}\|_2 = 5.0$

Figure 3: We plot the contour plots for clean test accuracy of the FUN Bayes classifier $\tilde{P}$ (given as $\tau_{\mathcal{D}}(\tilde{P})$) for varying a_y parameters where $y \in \{\pm 1\}$ in Figures 3(a–c). Gaussian mixture model $\mathcal{N}(y\boldsymbol{\mu}, I)$ denotes the clean data. In the top row, we use Lemmas 1 & 4 to empirically generate the plots. In the bottom row, we plot the theoretical upper bound of $\tau_{\mathcal{D}}(\tilde{P})$ we obtain using Theorem 2.

randomly initialized from a uniform distribution $\mathcal{U}(0, p_b)$ using a private seed where p_b is the blur parameter. Blur parameter controls the level of blurring that occurs when an image $\boldsymbol{x} \in [0,1]^{d_1 \times d_2 \times d_3}$ is convolved with a filter s_i. Here, d_1, d_2, and d_3 are the height, width, and number of channels of the image, respectively. For example, a CIFAR-10 image has a dimension of $32 \times 32 \times 3$. Let $\hat{\boldsymbol{x}} = \boldsymbol{x} * s_i$ where $\boldsymbol{x}$ belongs to class i. The FUN data point for $\boldsymbol{x}$ is given by $\tilde{\boldsymbol{x}} = \hat{\boldsymbol{x}}/\text{MAX}(\hat{\boldsymbol{x}})$. Rescaling is performed to make sure that the FUN image pixels lie between 0 and 1. We find that the unlearnability effect gets stronger with larger p_b and k values (see Appendix A.9).

4 Theory for FUN

In this section, we define a binary classification setup similar to Javanmard & Soltanolkotabi (2020); Min et al. (2021) to theoretically analyze FUN. Let $\mathcal{D}$ be a clean dataset modelled by an isotropic Gaussian mixture model given by $\mathcal{N}(y\boldsymbol{\mu}, I)$, where $y \in \{\pm 1\}$ is the class label, $\boldsymbol{\mu} \in \mathbb{R}^d$, and $I \in \mathbb{R}^{d \times d}$ is the identity matrix. The Bayes optimal decision boundary for classifying this Gaussian mixture model is as follows (proof in Appendix A.1).

Lemma 1 *The Bayes optimal decision boundary for classifying $\mathcal{D}$ is given by $P(\boldsymbol{x}) \equiv \boldsymbol{\mu}^\top \boldsymbol{x} = 0$. The accuracy of the decision boundary P on the clean dataset $\mathcal{D}$ (i.e. $\tau_{\mathcal{D}}(P)$) is equal to $\phi(\|\boldsymbol{\mu}\|_2)$. Here, $\phi(.)$ represents the CDF of the standard normal distribution.*

Now, to obtain $\tilde{\mathcal{D}}$ (i.e., the FUN dataset), we perform class-wise 1D convolutions. Here, we consider a class of 1D convolutional filters with kernel size 3 of the form $\boldsymbol{f_a} = [a, 1, a]$ where $a \in [0, 0.5]$. The convolution of signal $\boldsymbol{x} \in \mathbb{R}^d$ with filter $\boldsymbol{f_a}$ using stride 1 (denoted by $\boldsymbol{x} * \boldsymbol{f_a}$), can be treated as a matrix operation $A\boldsymbol{x}$. Here, $A \in \mathbb{R}^{d \times d}$ is a tri-diagonal Toeplitz matrix denoted by $T(d; a, 1, a)$. For FUN data generation, we use class-wise convolution matrices $A_y = T(d; a_y, 1, a_y)$ to perturb a clean data point $(\boldsymbol{x}, y) \sim \mathcal{D}$ to an unlearnable data point $(A_y\boldsymbol{x}, y)$, where $a_y \in [0, 0.5]$ for $y \in \{\pm 1\}$. Note that in FUN, the labels remain the same. Next we show that such a perturbed dataset (i.e. the FUN dataset) can be represented as a Gaussian mixture model (proof in Appendix A.2).

Lemma 2 *A FUN dataset generated from clean data distribution $\mathcal{D}$ and the A_y's defined above can be modelled as a Gaussian mixture model with the distribution given by $\tilde{\mathcal{D}} = \mathcal{N}(yA_y\boldsymbol{\mu}, A_y^2)$.*

To characterize the decision boundary for FUN data, we need to use some properties of Toeplitz matrices from Pasquini & Reichel (2006) given in the following Lemma 3 (check Appendix A.3 for details).

Lemma 3 *Any tri-diagonal Toeplitz matrix $A = T(d; a, 1, a) \in \mathbb{R}^{d \times d}$ with $a \in [0, 0.5]$ can be diagonalized as $A = QDQ$ where $Q_{i,j} = \left(\frac{2}{d+1}\right)^{1/2} \sin\left(\frac{ij\pi}{d+1}\right)$ and D is a diagonal matrix with $D_{i,i} = 1 + 2a\cos\left(\frac{i\pi}{d+1}\right)$ for $1 \leq i, j \leq d$.*

Next we use Lemma 3 to show that the Bayes optimal decision boundary for classifying $\tilde{\mathcal{D}}$ is a quadratic plane. Proof can be found in Appendix A.4.

Lemma 4 *Let $A_{-1} = T(d; a_{-1}, 1, a_{-1})$ and $A_1 = T(d; a_1, 1, a_1)$. The Bayes optimal decision boundary for classifying $\tilde{\mathcal{D}}$ is given by $\tilde{P}(\boldsymbol{x}) \equiv \boldsymbol{x}^\top A\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c = 0$, where $A = A_{-1}^{-2} - A_1^{-2}$, $\boldsymbol{b} = 2(A_{-1}^{-1} + A_1^{-1})\boldsymbol{\mu}$, and $c = 2\sum_{i=1}^{d}[\ln(1 + 2a_{-1}\cos(\frac{i\pi}{d+1})) - \ln(1 + 2a_1\cos(\frac{i\pi}{d+1}))]$.*

Now we state a lemma regarding the tail of Gaussian quadratic forms which plays a crucial role in our main result (proof in Appendix A.5).

Lemma 5 *Let $\|\cdot\|$ denote the operator norm, $\|\cdot\|_2$ denote the vector 2-norm, $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I)$, and $Z = \boldsymbol{z}^\top A\boldsymbol{z} + \boldsymbol{z}^\top \boldsymbol{b} + c$ where $A = Q\Lambda Q^\top$. Using Chernoff bound, for any $t \geq 0$ and $\gamma \in \mathbb{R}$,*

$$\mathbb{P}\{Z - \mathbb{E}Z \geq \gamma\} \leq \frac{\exp\left\{-\frac{t}{4\|\boldsymbol{b}\|_2^2\|\Lambda\|} - t(\gamma + \mathrm{Tr}(\Lambda) + \|\boldsymbol{b}\|_2)\right\}}{|I - 2t\Lambda|^{\frac{1}{2}}}.$$

Lemma 5 allows us to provide an upper bound for the accuracy of the unlearnable decision boundary $\tilde{P}$ on the clean dataset $\mathcal{D}$, given as $\tau_{\mathcal{D}}(\tilde{P})$, in Theorem 2 below. Proof can be found in Appendix A.6.

Theorem 2 (Main result) *Let $A = Q\Lambda Q = A_{-1}^{-2} - A_1^{-2}$. For any non-negative constants t_1 and t_2, the accuracy of the unlearnable decision boundary $\tilde{P}$ on the clean dataset $\mathcal{D}$ can be upper-bounded as*

$$\begin{aligned}\tau_{\mathcal{D}}(\tilde{P}) \leq\ & \frac{1}{2|I - 2t_1\Lambda|^{\frac{1}{2}}}\exp\left[t_1\left(\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c - \frac{1}{4\|2A\boldsymbol{\mu} + \boldsymbol{b}\|_2\|\Lambda\|}\right)\right] + \\ & \frac{1}{2|I - 2t_2\Lambda|^{\frac{1}{2}}}\exp\left[t_2\left(-\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c - \frac{1}{4\|2A\boldsymbol{\mu} - \boldsymbol{b}\|_2\|\Lambda\|}\right)\right] \\ :=\ & p_1 + p_2.\end{aligned}$$

Furthermore, if $\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c + \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} + \boldsymbol{b}\| < 0$, we have $\tau_{\mathcal{D}}(\tilde{P}) \leq \frac{1}{2}(p_1 + 1) < 1$. Also, if $-\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c - \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} - \boldsymbol{b}\| < 0$, we have $\tau_{\mathcal{D}}(\tilde{P}) \leq \frac{1}{2}(1 + p_2) < 1$. Moreover, for any $\boldsymbol{\mu} \neq \boldsymbol{0}$ and $a_{-1} \in [0, 0.5]$, $\exists a_1$ such that $\tau_{\mathcal{D}}(\tilde{P}) < \tau_{\mathcal{D}}(P)$.

Poisoning is *effective* only if the accuracy of the unlearnable model $\tilde{P}$ is less than that of the clean model P on the clean dataset $\mathcal{D}$, that is, $\tau_{\mathcal{D}}(\tilde{P}) < \tau_{\mathcal{D}}(P)$. To satisfy this condition, we need to carefully select a_y's. In Theorem 2, we formally state this condition[1]. Theorem 2 shows that FUN can effectively poison when the class means of the clean data are not the same. We validate our theoretical claim through empirical analysis as well (see Figure 3). Details for the analysis is given in Appendix A.7. We find that our upper bound for the clean test accuracy of FUN-based classifiers are consistent with our empirical analysis. In our experiments, we also find that the unlearnability effect is stronger with a larger blur parameter. This effect is evident from Figure 3 where we find that it is likely to get a lower $\tau_{\mathcal{D}}(\tilde{P})$ with higher a_y values.

5 EXPERIMENTS

In this section, we first discuss our experimental setup. More details on the setup is deferred to Appendix A.8. We then show the robustness of FUN with various datasets and architectures. We also run various experiments to analyze the effectiveness of FUN under different training techniques (ERM, AT with varying budgets, randomized smoothing, transfer learning, and fine-tuning) and augmentation techniques (mixup (Zhang et al., 2017), grayscaling, random blurring, cutout (DeVries & Taylor, 2017), cutmix (Yun et al., 2019), autoaugment (Cubuk et al., 2018), and orthogonal regularization (Brock et al., 2016)). Finally, we also design adaptive defenses to test the robustness of FUN. One might think that FUN filters can be obtained by adversarially training them with the data. We show that FUN is robust to such adaptive defenses that we design.

[1] We note that the conditions $\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c + \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} + \boldsymbol{b}\|_2 < 0$ or $-\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c - \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} - \boldsymbol{b}\|_2 < 0$ can always be satisfied by picking a sufficiently large $\boldsymbol{\mu}$ in the direction of an eigenvector corresponding to a negative or positive eigenvalue of A (note that A has negative and positive eigenvalues).

5.1 Experimental setup

Datasets. We use three image classification datasets – CIFAR-10, CIFAR-100, (Krizhevsky, 2009) and ImageNet-100 (a subset of ImageNet made of the first 100 classes) (Russakovsky et al., 2015). We use the data augmentation techniques such as random flipping, cropping, and resizing (Shorten & Khoshgoftaar, 2019).

Architectures. We use ResNet-18 (He et al., 2015), VGG-16 (Simonyan & Zisserman, 2015), Wide ResNet-34-10 (Zagoruyko & Komodakis, 2016), and DenseNet-121 (Huang et al., 2016). We train the networks with hyperparameters used in (Fu et al., 2021).

FUN generation. We use filters of size $k = 3$ and blur parameter $p_b = 0.3$ for both CIFAR-10 and CIFAR-100 datsets. ImageNet-100 is a higher dimensional $224{\times}224{\times}3$ image dataset when compared to the $32{\times}32{\times}3$ dimensional CIFAR datasets. Hence, we use larger filters of size $k = 9$ with $p_b = 0.06$ for ImageNet-100. These hyperparameters are chosen such that the FUN images are not perceptibly highly perturbed and give good unlearnability effect (see plot in Figure 1). In Appendix A.9, we show the results of FUN training with different hyperparameters for data generation.

Baselines. We compare FUN with four state-of-the-art unlearnability methods – REM (Fu et al., 2021), EM (Huang et al., 2021), TAP (Fowl et al., 2021), and NTGA (Yuan & Wu, 2021). We adopt the numbers reported in (Fu et al., 2021) since we use the same hyperparameters for training. For REM, we select hyperparameters $\rho_u = 8/255$ and $\rho_a = 4/255$, the highest radii values in Fu et al. (2021). For comparing unlearnable methods, we look at the clean test accuracy. The lower the test accuracy, the better the unlearnability method is.

5.2 Effectiveness of FUN

Blurring effect of FUN. In order to measure how much FUN's blurring affects the dataset's quality, we compare class-wise blurring (FUN) against universal blurring where a single filter is used for blurring all the images. We keep the filter generation parameters fixed ($p_b = 0.3$ and $k = 3$) for both FUN class-wise blurring and universal blurring. ResNet-18 trained with clean CIFAR-10, universally blurred CIFAR-10, and FUN CIFAR-10 data achieve clean test accuracies of 94.66%, 90.47%, and 18.48%, respectively. This suggests that our controlled blurring does not obscure the semantics of the dataset. Hence, the significant drop in the clean test accuracy introduced by the FUN dataset is most likely due to the usage of class-wise filters. Furthermore, the FUN-based model achieves an accuracy of 99.91% on the FUN CIFAR-10 test dataset. This supports our claim that the FUN model learns to classify images based on the filters used to blur them. In addition, if we permute the class-wise filters for blurring the test set (i.e., blurring class 1 images with class 2 filters, class 2 images with class 3 filters, and so on), we get a very low accuracy of 2.53% on this test set. See Appendix A.10 and A.11 for further details. Finally, we note that real-world datasets might also contain blurred images due to various factors such as motion blurring, weather conditions, issues with the camera, etc. Hence, detecting if a blurred image is poisoned might not always be possible.

Different datasets. We first compare the effectiveness of FUN with different datasets using ERM and L_∞ AT with $\rho_a = 4/255$. We use ResNet-18 for the experiments. The results are shown in Table 2. These results show that EM, TAP, and NTGA are not robust to AT. However, both FUN and REM are successful. Here, our method FUN outperforms REM with CIFAR-10 and ImageNet-100 datasets. Smartly designed additive noise in AT helps in achieving better generalization than ERM on the unlearnable datasets. This experiment thus demonstrates that ERM and AT are not good choices for training with FUN and REM datasets.

Different models. Next we compare the effectiveness of FUN using various deep learning architectures with L_∞ AT ($\rho_a = 4/255$). We use CIFAR-10 for the experiments. The results are shown in Table 3. As we see in the table, FUN is effective with all the five network architectures. However, REM is not seen to be transferrable with DenseNet-121.

Different training settings. In Section 3.2, we show that REM is sensitive to the training settings. REM generated using L_∞ radii budgets of $\rho_u = 8/255$ and $\rho_a = 4/255$ for CIFAR-10 breaks with L_∞ AT ($\rho_a = 8/255$) and L_2 AT ($\rho_a = 0.75$) to get test accuracy of 78.71% and 79.65%, respectively. Hence, we run experiments to check the robustness of FUN with various AT norm budgets. The results are shown in Table 4. As we see in the table, FUN is robust to ERM, L_∞, and L_2 AT settings with varying training budgets. Impressively, the highest test accuracy achieved with

Table 2: Test accuracy (%) of ResNet-18 trained on various unlearnable datasets. We use L_∞ AT with budget $\rho_a = 4/255$.

Dataset	Training method	Clean	Unlearnability method				
			EM	TAP	NTGA	REM	**FUN (ours)**
CIFAR-10	ERM AT	94.66 89.51	**13.20** 88.62	22.51 88.02	16.27 88.96	27.09 48.16	18.48 **44.40**
CIFAR-100	ERM AT	76.27 64.50	**1.60** 63.43	13.75 62.39	3.22 62.44	10.14 **27.10**	12.69 34.34
ImageNet-100	ERM AT	80.66 66.62	**1.26** 63.40	9.10 63.56	8.42 63.06	13.74 41.66	8.96 **38.68**

Table 3: Test accuracy (%) of network architectures trained on various CIFAR-10 unlearnable datasets with L_∞ AT ($\rho_a = 4/255$).

Model	Clean	Unlearnability method				
		EM	TAP	NTGA	REM	**FUN (ours)**
ResNet-18	89.51	88.62	88.02	88.96	48.16	**44.40**
VGG-16	87.51	86.48	86.27	86.65	65.23	**42.98**
Wide ResNet-34-10	91.21	90.05	90.23	89.95	**48.39**	53.02
DenseNet-121	83.27	82.44	81.72	80.73	81.48	**45.95**

FUN training on CIFAR-10, CIFAR-100, and ImageNet-100 are as low as 51.19%, 36.90%, and 40.08%, respectively. We also find that using a pre-trained ResNet-18 with FUN CIFAR-10 only achieves clean test accuracy of 42.42% and 48.22% with fine-tuning the full network and a newly trained final layer, respectively (details deferred to Appendix A.12).

Different protection percentages. In Section 3.1, we assume that the attacker has access to the full clean training data. However, in real life settings, this might not be always possible. Hence, we train ResNet-18 on a mix of FUN and clean CIFAR-10 training datasets to evaluate the effectiveness of poisoning with varying data protection percentages. Protection percentage denotes the percentage of the training data that is poisoned.

We show the results in Table 5. In the table, the “Mixed” column denotes the clean test accuracy of a model trained using a mix of both clean and poisoned data. The “Clean” column denotes the clean test accuracy of a model trained only using the clean subset of the training data. In the last row of Table 5, we provide the results for FUN trained using ERM with varying data protection percentages. The remainder of the rows provide the results for the L_∞ AT ($\rho_a = 4/255$) scenario. For example, FUN trained with ERM using an 80% clean training data partition achieves a test accuracy of 93.75%. Adding the 20% FUN data partition to the training dataset drops the test accuracy of the model to 93.28%. Results from the last row of Table 5 show that FUN with varying data protection percentages is effective with the ERM setting. However, with the AT scenario, all the poisoning techniques are ineffective with varying data protection percentages. Even at 80% data protection percentage, training with all the poisoning techniques achieve noticeably higher test accuracy than the test accuracy achieved using the corresponding clean training data subset. Nevertheless, it is interesting to note that FUN performs better than all the other unlearnability techniques with AT.

Smoothing and data augmentations. Cohen et al. (2019) proposes randomized smoothing which is a provable adversarial defense in L_2 norm. Since FUN is not an additive noise, FUN is robust to randomized smoothing. A smoothed ResNet-18 (with a noise level of 0.5) with FUN CIFAR-10 achieves only a clean test accuracy of 43.85%. In Section 3.2, we see that REM breaks with grayscaling. While ResNet-18 training using grayscaled REM achieves 70.76% test accuracy, grayscaled FUN only achieves 20.12% test accuracy on the clean grayscaled CIFAR-10 test data. This shows that FUN exhibits the desirable property of not relying upon the color space for its attack. FUN CIFAR-10 training achieves only 25.53%, 25.80%, 26.93%, 34.09%, and 50.72% clean test accuracy with mixup, cutout, cutmix, autoaugment, and orthogonal regularization, respectively (see Appendix A.13).

Adaptive defenses. Here, we first investigate the effect of training FUN CIFAR-10 with random blurring augmentations using ResNet-18. Each batch of the FUN training data is convolved with

Table 4: Test accuracy $(\%)$ of ResNet-18 with FUN datasets under various training settings.

Dataset	Clean	Training method	**FUN (ours)**
CIFAR-10	94.66	ERM	18.48
		AT L_∞ $(\rho_a = 4/255)$	44.40
		AT L_∞ $(\rho_a = 8/255)$	32.85
		AT L_∞ $(\rho_a = 16/255)$	19.32
		AT L_2 $(\rho_a = 0.25)$	39.05
		AT L_2 $(\rho_a = 0.50)$	51.19
		AT L_2 $(\rho_a = 0.75)$	51.14
CIFAR-100	76.27	ERM	12.69
		AT L_∞ $(\rho_a = 4/255)$	34.34
		AT L_∞ $(\rho_a = 8/255)$	30.00
		AT L_2 $(\rho_a = 0.75)$	36.90
ImageNet-100	80.66	ERM	8.96
		AT L_∞ $(\rho_a = 4/255)$	38.68
		AT L_∞ $(\rho_a = 8/255)$	40.08
		AT L_2 $(\rho_a = 0.75)$	20.58

Table 5: Test accuracy $(\%)$ of ResNet-18 on CIFAR-10 with different data protecion percentages. The last row shows the results for FUN with ERM setting. The rest of the rows show results for unlearnability methods trained in the L_∞ AT $(\rho_a = 4/255)$ setting.

Unlearnability method	Data Protection Percentage									
	0%	20%		40%		60%		80%		100%
		Mixed	Clean	Mixed	Clean	Mixed	Clean	Mixed	Clean	
EM	89.51	89.60	88.17	89.40	86.76	89.49	85.07	89.10	79.41	88.62
TAP		89.01		88.66		88.40		88.04		88.02
NTGA		89.56		89.35		89.22		89.17		88.96
REM		89.60		89.34		89.61		88.09		48.16
FUN (ours)		**88.54**		**87.24**		**86.03**		**84.34**		**44.40**
FUN + ERM	94.66	93.28	93.75	91.34	92.56	89.91	89.77	85.61	84.30	18.48

random 3×3 filters of varying blur parameters p'_b. With p'_b values of 0.1 and 0.3, FUN training achieves lower test accuracy 9.2% and 13.37%, respectively. This shows that the noise added by FUN is robust to random convolution augmentations.

One may think that FUN can be broken by learning the private filters from the data. We test this idea by training deconvolution filters to find if we can break FUN with adversarially trained filters. We use a novel Filter Adversarial Training (FAT) technique that is similar to AT (check Appendix A.7 and equation 3 for details). While the adversarial step in AT learns sample-wise error-maximizing additive noises, the adversarial step in FAT learns class-wise error-maximizing deconvolution filters. We train FAT with filters of varying sizes (3, 5, and 7) on FUN CIFAR-10 using ResNet-18. The filter parameters are constrained within a finite range to make sure that the images do not get distorted with the adversarial step. We find that FUN is robust to FAT. FAT using filters of size 3, 5, and 7 with FUN dataset achieves test accuracy of 39.05%, 46.21%, and 38.48%, respectively.

Limitations and future directions. The unlearnability effect of FUN can be defended if some fraction of the clean data and its corresponding FUN images are leaked. The defender must also be able to detect if all samples in the dataset are poisoned. However, our work assumes a setup where the filters remain private. For example, a data publisher could simply publish their FUN images and delete the clean images permanently to prevent this scenario. As discussed in this section, FUN as well as other prior works do not perform well with different protection percentages with AT. Improving this can be an interesting research direction. We believe that extending FUN to other domains such as tabular and text data is also an interesting future direction. It would also be interesting to see theoretical analysis of FUN considering more complex setups.

References

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

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*, 2016.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017.

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

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1310–1320. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/cohen19c.html.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11162–11173, 2021.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojtek Czaja, and Tom Goldstein. Adversarial examples make strong poisons. *arXiv preprint arXiv:2106.10807*, 2021.

Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *International Conference on Learning Representations*, 2021.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Kashmir Hill. The secretive company that might end privacy as we know it. In *Ethics of Data and Analytics*, pp. 170–177. Auerbach Publications, 2020.

StubbornAtom (https://stats.stackexchange.com/users/119261/stubbornatom). Distributions of quadratic form of a normal random variable. Cross Validated. URL https://stats.stackexchange.com/q/478682. URL:https://stats.stackexchange.com/q/478682 (version: 2020-07-23).

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. URL http://arxiv.org/abs/1608.06993.

Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*, 2021.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Adel Javanmard and Mahdi Soltanolkotabi. Precise statistical analysis of classification accuracies for adversarial training. *ArXiv*, abs/2010.11213, 2020.

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

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Aounon Kumar, Alexander Levine, Soheil Feizi, and Tom Goldstein. Certifying confidence via randomized smoothing. *CoRR*, abs/2009.08061, 2020. URL https://arxiv.org/abs/2009.08061.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/6e923226e43cd6fac7cfe1e13ad000ac-Paper.pdf.

Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Defense against unseen threat models. *CoRR*, abs/2006.12655, 2020. URL https://arxiv.org/abs/2006.12655.

Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defense against general poisoning attacks. *CoRR*, abs/2006.14768, 2020a. URL https://arxiv.org/abs/2006.14768.

Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defense against general poisoning attacks. *CoRR*, abs/2006.14768, 2020b. URL https://arxiv.org/abs/2006.14768.

Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. *arXiv preprint arXiv:2101.05930*, 2021.

Yiming Li, Tongqing Zhai, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shutao Xia. Rethinking the trigger of backdoor attack. *arXiv preprint arXiv:2004.04692*, 2020.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Sijia Liu, Songtao Lu, Xiangyi Chen, Yao Feng, Kaidi Xu, Abdullah Al-Dujaili, Mingyi Hong, and Una-May O'Reilly. Min-max optimization without gradients: Convergence and applications to black-box evasion and poisoning attacks. In *International Conference on Machine Learning*, pp. 6282–6293. PMLR, 2020a.

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

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

Yifei Min, Lin Chen, and Amin Karbasi. The curious case of adversarially robust models: More data can help, double descend, or hurt generalization. In *Uncertainty in Artificial Intelligence*, pp. 129–139. PMLR, 2021.

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

Silvia Noschese1 Lionello Pasquini and Lothar Reichel. Tridiagonal toeplitz matrices: Properties and novel applications. 2006.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Mert Bulent Sariyildiz, Julien Perez, and Diane Larlus. Learning visual representations with caption annotations. In *European Conference on Computer Vision*, pp. 153–170. Springer, 2020.

Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31, 2018.

Connor Shorten and Taghi Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 07 2019. doi: 10.1186/s40537-019-0197-0.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Lue Tao, Lei Feng, Jinfeng Yi, Sheng-Jun Huang, and Songcan Chen. Better safe than sorry: Preventing delusive adversaries with adversarial training. *Advances in Neural Information Processing Systems*, 34, 2021.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL `http://jmlr.org/papers/v9/vandermaaten08a.html`.

Wenxiao Wang, Alexander Levine, and Soheil Feizi. Improved certified defenses against data poisoning with (deterministic) finite aggregation. *CoRR*, abs/2202.02628, 2022. URL `https://arxiv.org/abs/2202.02628`.

Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. *arXiv preprint arXiv:2112.08304*, 2021.

Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 2367–2376, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539241. URL `https://doi.org/10.1145/3534678.3539241`.

Chia-Hung Yuan and Shan-Hung Wu. Neural tangent generalization attacks. In *International Conference on Machine Learning*, pp. 12230–12240. PMLR, 2021.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.87. URL `https://dx.doi.org/10.5244/C.30.87`.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL `http://arxiv.org/abs/1710.09412`.

A Appendix

A.1 Proof for Lemma 1

At the optimal decision boundary the probabilities of any point $\boldsymbol{x} \in \mathbb{R}^d$ belonging to class $y = -1$ and $y = 1$ modelled by $\mathcal{D}$ are the same. Here, $\boldsymbol{\mu} = \boldsymbol{\mu_1} = -\boldsymbol{\mu_{-1}}$ and $I = \Sigma_{-1} = \Sigma_1$.

$$\begin{aligned}
&\implies \frac{\exp[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_{-1}})^\top \Sigma_{-1}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_{-1}})]}{\sqrt{(2\pi)^d|\Sigma_{-1}|}} = \frac{\exp[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_{1}})^\top \Sigma_{1}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_{1}})]}{\sqrt{(2\pi)^d|\Sigma_{1}|}} \\
&\implies -\frac{1}{2}\log|I| - \frac{1}{2}(\boldsymbol{x}^\top\boldsymbol{x} - 2\boldsymbol{x}^\top\boldsymbol{\mu_{-1}} + \boldsymbol{\mu}_{\boldsymbol{-1}}^\top\boldsymbol{\mu_{-1}}) = -\frac{1}{2}\log|I| - \frac{1}{2}(\boldsymbol{x}^\top\boldsymbol{x} - 2\boldsymbol{x}^\top\boldsymbol{\mu_{1}} + \boldsymbol{\mu}_{\boldsymbol{1}}^\top\boldsymbol{\mu_{1}}) \\
&\implies \boldsymbol{x}^\top(\boldsymbol{\mu_{-1}} - \boldsymbol{\mu_1}) - \frac{1}{2}(\boldsymbol{\mu}_{\boldsymbol{-1}}^\top\boldsymbol{\mu_{-1}} - \boldsymbol{\mu}_{\boldsymbol{1}}^\top\boldsymbol{\mu_{1}}) = 0 \\
&\implies -\boldsymbol{x}^\top\boldsymbol{\mu} = 0 \\
&\implies P(\boldsymbol{x}) \equiv \boldsymbol{x}^\top\boldsymbol{\mu} = 0.
\end{aligned}$$

Now, the accuracy of the clean model P is to be computed. Note that if $P(\boldsymbol{x}) < 0$ the Bayes optimal classification is class -1, else the classification is class 1. Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I)$, and $Z \sim \mathcal{N}(0,1)$, and $sgn(.)$ be the signum function.

$$\begin{aligned}
\tau_{\mathcal{D}}(P) &= \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\mathbb{1}(y = sgn(P(\boldsymbol{x})))\right] = \mathbb{P}[y\boldsymbol{x}^\top\boldsymbol{\mu} > 0] \\
&= \mathbb{P}[y(y\boldsymbol{\mu} + \boldsymbol{z})^\top\boldsymbol{\mu} > 0] \\
&= \mathbb{P}[(\boldsymbol{\mu} + \boldsymbol{z})^\top\boldsymbol{\mu} > 0] \\
&= \mathbb{P}[\|\boldsymbol{\mu}\|_2^2 + \|\boldsymbol{\mu}\|_2 Z > 0] = \phi(\|\boldsymbol{\mu}\|_2).
\end{aligned}$$

□

A.2 Proof for Lemma 2

Let $\mathcal{D}_1 = \mathcal{N}(\boldsymbol{\mu}, I)$. For every data point $(\boldsymbol{x}, y) \sim \mathcal{D}_1$, let the perturbed data $(A_1\boldsymbol{x}, y)$ be modelled by a distribution $\tilde{D}_1$. We prove that $\tilde{D}_1 = \mathcal{N}(A_1\boldsymbol{\mu}, A_1^\top A_1)$.

$$\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, A_1\boldsymbol{x} = A_1\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, \boldsymbol{x} = A_1\boldsymbol{\mu}.$$

$$\begin{aligned}
\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, (A_1\boldsymbol{x} - A_1\boldsymbol{\mu})(A_1\boldsymbol{x} - A_1\boldsymbol{\mu})^\top &= \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, A_1(\boldsymbol{x} - \boldsymbol{\mu})[A_1(\boldsymbol{x} - \boldsymbol{\mu})]^\top \\
&= \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, A_1(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top A_1^\top \\
&= A_1\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_1}\, (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top A_1^\top \\
&= A_1 I A_1^\top = A_1 A_1^\top.
\end{aligned}$$

Tri-diagonal Toeplitz matrices $A_y = T(d; a_y, 1, a_y)$ are symmetric. Hence, $\tilde{\mathcal{D}} = \mathcal{N}(yA_y\boldsymbol{\mu}, A_y^2)$. □

A.3 Remarks on Lemma 3

A tri-diagonal Toeplitz matrix $T(d; a_1, a_2, a_3)$ is represented as

$$\begin{bmatrix} a_2 & a_3 & 0 & 0 & \dots & 0 \\ a_1 & a_2 & a_3 & 0 & \dots & 0 \\ 0 & a_1 & a_2 & a_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & a_1 & a_2 \end{bmatrix} \in \mathbb{R}^{d \times d}$$

The class of matrices $A_y = T(d; a_y, 1, a_y)$ are symmetric and can be diagonalized as $QDQ^\top$. $Q = \left(\left(\frac{2}{d+1}\right)^{1/2} \sin\left(\frac{ij\pi}{d+1}\right)\right)_{i,j}$ is symmetric and it is the common eigenvector matrix to all A_y matrices. As shown in Lemma 3, Q and D can be represented using trigonometric functions. Also, we have $A(n) := A_1^n \pm A_{-1}^n = Q(D_1^n \pm D_{-1}^n)Q$ where $A_1 = QD_1Q$ and $A_{-1} = QD_{-1}Q$. Further, $\mathrm{Tr}(A(n)) = \mathrm{Tr}(Q(D_1^n \pm D_{-1}^n)Q) = \mathrm{Tr}((D_1^n \pm D_{-1}^n)Q^2) = \mathrm{Tr}(D_1^n \pm D_{-1}^n)$.

A.4 Proof for Lemma 4

At the optimal decision boundary the probabilities of any point $\boldsymbol{x} \in \mathbb{R}^d$ belonging to class $y = -1$ and $y = 1$ modelled by $\tilde{\mathcal{D}}$ are the same. Here, $\boldsymbol{\mu} = \boldsymbol{\mu_1} = -\boldsymbol{\mu_{-1}}$ and A_y's are symmetric.

$$\begin{aligned} &\frac{\exp[-\frac{1}{2}(\boldsymbol{x} - A_{-1}\boldsymbol{\mu_{-1}})^\top (A_{-1} I A_{-1}^\top)^{-1} (\boldsymbol{x} - A_{-1}\boldsymbol{\mu_{-1}})]}{\sqrt{(2\pi)^d |A_{-1} I A_{-1}^\top|}} \\ &\qquad\qquad = \frac{\exp[-\frac{1}{2}(\boldsymbol{x} - A_1\boldsymbol{\mu_1})^\top (A_1 I A_1^\top)^{-1} (\boldsymbol{x} - A_1\boldsymbol{\mu_1})]}{\sqrt{(2\pi)^d |A_1 I A_1^\top|}} \end{aligned}$$

$$\begin{aligned} \implies & -\frac{1}{2}\ln\frac{|A_{-1}^2|}{|A_1^2|} - \frac{1}{2}[\boldsymbol{x}^\top (A_{-1}^{-2} - A_1^{-2})\boldsymbol{x} - 2(\boldsymbol{\mu_{-1}^\top} A_{-1}^{-1} - \boldsymbol{\mu_1^\top} A_1^{-1})\boldsymbol{x} \\ &\qquad\qquad + (\boldsymbol{\mu_{-1}^\top \mu_{-1}} - \boldsymbol{\mu_1^\top \mu_1})] = 0 \\ \implies & \tilde{P}(\boldsymbol{x}) \equiv \boldsymbol{x}^\top (A_{-1}^{-2} - A_1^{-2})\boldsymbol{x} - 2(\boldsymbol{\mu_{-1}^\top} A_{-1}^{-1} - \boldsymbol{\mu_1^\top} A_1^{-1})\boldsymbol{x} + (\|\boldsymbol{\mu_{-1}}\|_2^2 - \|\boldsymbol{\mu_1}\|_2^2) \\ &\qquad\qquad + \sum_{i=1}^{d} \ln\left(\frac{1 + 2a_{-1}\cos(\frac{i\pi}{d+1})}{1 + 2a_1\cos(\frac{i\pi}{d+1})}\right)^2 = 0 \\ \implies & \tilde{P}(\boldsymbol{x}) \equiv \boldsymbol{x}^\top (A_{-1}^{-2} - A_1^{-2})\boldsymbol{x} + 2[(A_{-1}^{-1} + A_1^{-1})\boldsymbol{\mu}]^\top \boldsymbol{x} + \sum_{i=1}^{d} \ln\left(\frac{1 + 2a_{-1}\cos(\frac{i\pi}{d+1})}{1 + 2a_1\cos(\frac{i\pi}{d+1})}\right)^2 = 0 \\ \implies & \tilde{P}(\boldsymbol{x}) \equiv \boldsymbol{x}^\top A\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c = 0. \end{aligned}$$

□

Note that here if $\tilde{P}(\boldsymbol{x}) < 0$, the Bayes optimal classification is class -1, else the classification is class 1. Here, for shorthand notations we denote $A = (A_{-1}^{-2} - A_1^{-2}),\ \boldsymbol{b} = 2(A_{-1}^{-1} + A_1^{-1})\boldsymbol{\mu},\ c = \sum_{i=1}^{d} \ln\left(\frac{1 + 2a_{-1}\cos(\frac{i\pi}{d+1})}{1 + 2a_1\cos(\frac{i\pi}{d+1})}\right)^2$.

A.5 Proof for Lemma 5

Let $Z = \boldsymbol{z}^\top A\boldsymbol{z} + \boldsymbol{z}^\top \boldsymbol{b} + c$ and $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I) \subset \mathbb{R}^d$ where $A = Q\Lambda Q^\top$. Also,

$$Z = \boldsymbol{z}^\top A\boldsymbol{z} + \boldsymbol{z}^\top \boldsymbol{b} + c = \left(\boldsymbol{z} + \frac{1}{2}A^{-1}\boldsymbol{b}\right)^\top A\left(\boldsymbol{z} + \frac{1}{2}A^{-1}\boldsymbol{b}\right) + c - \frac{1}{4}\boldsymbol{b}^\top A^{-1}\boldsymbol{b}.$$

For any $t \geq 0$ and $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, I)$, we write the moment generating function for a quadratic random variable $Y = \boldsymbol{x}^\top A\boldsymbol{x}$ as [2]

$$\begin{aligned}
\mathbb{E}[\exp(tY)] &= \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \exp\{t\boldsymbol{x}^\top A\boldsymbol{x}\} \exp\Big\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top(\boldsymbol{x}-\boldsymbol{\mu})\Big\} d\boldsymbol{x} \\
&= \frac{\exp\{-\boldsymbol{\mu}^\top\boldsymbol{\mu}/2\}}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \exp\Big\{ -\frac{1}{2}\boldsymbol{x}^\top(I-2tA)\boldsymbol{x} + \boldsymbol{\mu}^\top\boldsymbol{x}\Big\} d\boldsymbol{x} \\
&= \frac{\exp\{-\boldsymbol{\mu}^\top\boldsymbol{\mu}/2\}}{(2\pi)^{d/2}} \frac{(2\pi)^{d/2}\exp\Big\{\frac{1}{2}\boldsymbol{\mu}^\top(I-2tA)^{-1}\boldsymbol{\mu}\Big\}}{|I-2tA|^{1/2}} \\
&= \frac{\exp\Big\{ -\frac{1}{2}\boldsymbol{\mu}^\top[I-(I-2tA)^{-1}]\boldsymbol{\mu}\Big\}}{|I-2tA|^{1/2}}. \\
\implies \mathbb{E}[\exp(tZ)] &= \frac{\exp\{-\frac{1}{8}\boldsymbol{b}^\top A^{-1}[I-(I-2tA)^{-1}]A^{-1}\boldsymbol{b} + t[c-\frac{1}{4}\boldsymbol{b}^\top A^{-1}\boldsymbol{b}]\}}{|I-2tA|^{\frac{1}{2}}}.
\end{aligned}$$

Using the Chernoff bound and $\mathbb{E}\ \boldsymbol{z}^\top A\boldsymbol{z} = \mathrm{Tr}(A\mathbb{E}[zz^\top]) = \mathrm{Tr}(A)$, for some γ,

$$\begin{aligned}
\mathbb{P}\{Z \geq \mathbb{E}[Z]+\gamma\} &\leq \frac{\mathbb{E}[\exp(tZ)]}{\exp\{t[\gamma+\mathbb{E}(Z)]\}} \\
&= \frac{\exp\{-\frac{1}{8}\boldsymbol{b}^\top A^{-1}[I-(I-2tA)^{-1}]A^{-1}\boldsymbol{b} + t[c-\frac{1}{4}\boldsymbol{b}^\top A^{-1}\boldsymbol{b}]\}}{\exp\{t([\gamma+\mathrm{Tr}(A)+\|b\|_2+c]\}|I-2tA|^{\frac{1}{2}}}.
\end{aligned}$$

Let us take $\boldsymbol{u} = Q^\top\boldsymbol{b}$. Also, $-\Lambda^{-1}[I-(I-2t\Lambda)^{-1}]\Lambda^{-1} = 2t\Lambda^{-1}(I-2t\Lambda)^{-1}$ since Λ is a diagonal matrix. Using Woodbury matrix identity, we get $(I-2t\Lambda)^{-1} = I-(I-\frac{1}{2t}\Lambda^{-1})^{-1}$. This gives us

$$\begin{aligned}
\mathbb{P}\{Z \geq \mathbb{E}[Z]+\gamma\} &\leq \frac{\exp\{-\frac{1}{8}\boldsymbol{b}^\top A^{-1}[I-(I-2tA)^{-1}]A^{-1}\boldsymbol{b} + t[c-\frac{1}{4}\boldsymbol{b}^\top A^{-1}\boldsymbol{b}]\}}{\exp\{t[\gamma+\mathrm{Tr}(A)+\|b\|_2+c]\}|I-2tA|^{\frac{1}{2}}} \\
&= \frac{\exp\{-\frac{1}{8}\boldsymbol{u}^\top\Lambda^{-1}[I-(I-2t\Lambda)^{-1}]\Lambda^{-1}\boldsymbol{u} + t[c-\frac{1}{4}\boldsymbol{u}^\top\Lambda^{-1}\boldsymbol{u}]\}}{\exp\{t[\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2+c]\}|I-2t\Lambda|^{\frac{1}{2}}} \\
&= \frac{\exp\{\frac{t}{4}\boldsymbol{u}^\top\Lambda^{-1}(I-2t\Lambda)^{-1}\boldsymbol{u} + t[c-\frac{1}{4}\boldsymbol{u}^\top\Lambda^{-1}\boldsymbol{u}]\}}{\exp\{t[\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2+c]\}|I-2t\Lambda|^{\frac{1}{2}}} \\
&= \frac{\exp\{\frac{t}{4}\boldsymbol{u}^\top\Lambda^{-1}[I-(I-\frac{1}{2t}\Lambda^{-1})^{-1}]\boldsymbol{u} + t[c-\frac{1}{4}\boldsymbol{u}^\top\Lambda^{-1}\boldsymbol{u}]\}}{\exp\{t[\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2+c]\}|I-2t\Lambda|^{\frac{1}{2}}} \\
&= \frac{\exp\{\frac{-t}{4}\boldsymbol{u}^\top\Lambda^{-1}(I-\frac{1}{2t}\Lambda^{-1})^{-1}\boldsymbol{u} + tc\}}{\exp\{t[\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2+c]\}|I-2t\Lambda|^{\frac{1}{2}}} \\
&\leq \frac{\exp\{\frac{-t}{4\|\boldsymbol{b}\|_2^2}\lambda_{\min}(\Lambda^{-1}(I-\frac{1}{2t}\Lambda^{-1})^{-1}) - t(\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2)\}}{|I-2t\Lambda|^{\frac{1}{2}}} \\
&= \frac{\exp\{\frac{-t}{4\|\boldsymbol{b}\|_2^2}\frac{1}{\|\Lambda\|-1/(2t)} - t(\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2)\}}{|I-2t\Lambda|^{\frac{1}{2}}} \\
&\leq \frac{\exp\{\frac{-t}{4\|\boldsymbol{b}\|_2^2\|\Lambda\|} - t(\gamma+\mathrm{Tr}(\Lambda)+\|b\|_2)\}}{|I-2t\Lambda|^{\frac{1}{2}}}.
\end{aligned}$$

□

[2] (https://stats.stackexchange.com/users/119261/stubbornatom)

A.6 Proof for Theorem 2

Note that if $\tilde{P}(\boldsymbol{x}) < 0$, the classifier predicts a label for class -1, else the predicted label would be 1. Here, $\boldsymbol{x} = y\boldsymbol{\mu} + \boldsymbol{z}$ where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I)$ and $y \in \{\pm 1\}$ since $(\boldsymbol{x}, y) \sim \mathcal{D}$.

$$\begin{aligned}
\tau_{\mathcal{D}}(\tilde{P}) =&\ \mathbb{E}\{\mathbb{1}(y(\boldsymbol{x}^\top A\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c) > 0)\} \\
=&\ \mathbb{P}\{y(\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{z}^\top A\boldsymbol{z} + 2y\boldsymbol{\mu}^\top A\boldsymbol{z} + y\boldsymbol{b}^\top\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{z} + c) > 0\} \\
=&\ \mathbb{P}(y=1)\,\mathbb{P}\{y(\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{z}^\top A\boldsymbol{z} + 2y\boldsymbol{\mu}^\top A\boldsymbol{z} + y\boldsymbol{b}^\top\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{z} + c) > 0 \mid y = 1\} + \\
&\ \mathbb{P}(y=-1)\,\mathbb{P}\{y(\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{z}^\top A\boldsymbol{z} + 2y\boldsymbol{\mu}^\top A\boldsymbol{z} + y\boldsymbol{b}^\top\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{z} + c) > 0 \mid y = -1\} \\
=&\ \frac{1}{2}\mathbb{P}\{\boldsymbol{z}^\top A\boldsymbol{z} + (\boldsymbol{b} + 2A\boldsymbol{\mu})^\top\boldsymbol{z} + \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c > 0\} + \\
&\ \frac{1}{2}\mathbb{P}\{-\boldsymbol{z}^\top A\boldsymbol{z} - (\boldsymbol{b} - 2A\boldsymbol{\mu})^\top\boldsymbol{z} - \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c > 0\} := p_1 + p_2
\end{aligned}$$

We can see that

$$\begin{aligned}
-\gamma_1 :=&\ \mathbb{E}\{\boldsymbol{z}^\top A\boldsymbol{z} + (\boldsymbol{b} + 2A\boldsymbol{\mu})^\top\boldsymbol{z} + \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c\} \\
=&\ \mathrm{Tr}(\Lambda) + \|\boldsymbol{b} + 2A\boldsymbol{\mu}\|_2 + \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c, \ \text{ and} \\
-\gamma_2 :=&\ \mathbb{E}\{-\boldsymbol{z}^\top A\boldsymbol{z} - (\boldsymbol{b} - 2A\boldsymbol{\mu})^\top\boldsymbol{z} - \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c\} \\
=&\ -\mathrm{Tr}(\Lambda) + \|\boldsymbol{b} - 2A\boldsymbol{\mu}\|_2 - \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c.
\end{aligned}$$

Using Lemma 5, with $\gamma = \gamma_1, t = t_1$ for computing p_1 and $\gamma = \gamma_2, t = t_2$ for computing p_2 where t_1, t_2 are some non-negative constants, we get

$$\begin{aligned}
p_1 &= \frac{1}{2|I - 2t_1\Lambda|^{1/2}} \exp\left[t_1\left(\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c - \frac{1}{4\|2A\boldsymbol{\mu} + \boldsymbol{b}\|_2\|\Lambda\|}\right)\right], \ \text{ and} \\
p_2 &= \frac{1}{2|I - 2t_2\Lambda|^{1/2}} \exp\left[t_2\left(-\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c - \frac{1}{4\|2A\boldsymbol{\mu} - \boldsymbol{b}\|_2\|\Lambda\|}\right)\right].
\end{aligned}$$

This gives us the upper bound for $\tau_{\mathcal{D}}(\tilde{P})$. However, to make sure that this upper bound is smaller than 1, we need to assert more conditions. p_1 and p_2 become smaller as γ_1 and γ_2 are larger positive numbers. However, $\gamma_1 + \gamma_2 = -(\|2A\boldsymbol{\mu} + \boldsymbol{b}\|_2 + \|2A\boldsymbol{\mu} - \boldsymbol{b}\|_2 + 4\boldsymbol{\mu}^\top Q^\top(D_1^{-1} + D_{-1}^{-1})Q\boldsymbol{\mu}) \leq 0$ since $(D_1^{-1} + D_{-1}^{-1}) \not\succcurlyeq 0$. Hence, we look at separately at cases when either $\gamma_1 > 0$ or $\gamma_2 > 0$.

If $\gamma_1 > 0$, then $\tau_{\mathcal{D}}(\tilde{P}) = \frac{1}{2}(p_1 + 1) < 1$. Else, if $\gamma_2 > 0$, then $\tau_{\mathcal{D}}(\tilde{P}) = \frac{1}{2}(p_2 + 1) < 1$. We know that for $\boldsymbol{\mu} \neq \boldsymbol{0}$, $\tau_{\mathcal{D}}(P) = \phi(\boldsymbol{\mu}) > \frac{1}{2}$. Moreover, for any $a_{-1} \in [0, 0.5]$, $\exists a_1$ such that $\tau_{\mathcal{D}}(\tilde{P}) < \tau_{\mathcal{D}}(P)$. This can be satisfied by picking a_1 such that either γ_1 or γ_2 is very large, i.e., $\frac{1}{2} < \tau_{\mathcal{D}}(\tilde{P}) = \frac{1}{2}[1 + \min(p_1, p_2)] < \tau_{\mathcal{D}}(P)$. We note that the conditions $-\gamma_1 = \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} + c + \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} + \boldsymbol{b}\|_2 < 0$ and $-\gamma_2 = -\boldsymbol{\mu}^\top A\boldsymbol{\mu} + \boldsymbol{b}^\top\boldsymbol{\mu} - c - \mathrm{Tr}(A) + \|2A\boldsymbol{\mu} - \boldsymbol{b}\|_2 < 0$ can always be satisfied by picking a sufficiently large $\boldsymbol{\mu}$ in the direction of an eigenvector corresponding to a negative eigenvalue of A (note that A has negative eigenvalues). □

A.7 Details on generating Figure 3

We use $\boldsymbol{\mu} \in \mathbb{R}^d$, $d = 100$ to generate clean dataset with 1000 data points. They are randomly split into training and testing partitions of equal size. All the assumptions are consistent with the details provided in the main body. We use 30×30 mesh-grid to plot the contour plots. While plotting the theoretical upper bounds, we choose the best t_1, t_2 with grid search from a search space $[2^1, 2^0, 2^{-1}, 2^{-2}, 2^{-3}, 2^{-4}, 2^{-5}]$.

A.8 Experimental details

This subsection provides the details for experiments in Section 5.

Hardware. We use NVIDIA® RTX A4000 GPU with 16GB memory with 16 AMD® EPYC 7302P CPU cores.

Data augmentations. For CIFAR-10 and CIFAR-100, we use random flipping, 4 pixel padding, and random 32×32 size cropping. For ImageNet-100, we use random flipping and random cropping with resizing to 224×224 size. All the images are rescaled to have pixel values in the range $[0, 1]$.

Baselines. We compare FUN against error-minimizing noise (Huang et al., 2021), targeted adversarial poisoning (Fowl et al., 2021), neural tangent generalization attack (Yuan & Wu, 2021), and robust error-minimizing noise (Fu et al., 2021). We use the experimental outputs reported in Fu et al. (2021) for our comparisons. For REM we choose $\rho_u = 8/255$ and $\rho_a = 4/255$ since REM works the best when $\rho_u = 2\rho_a$ (Fu et al., 2021). We perform experiments on REM not present in their work using their code available publicly on GitHub [3] (MIT License).

Networks. For consistency, we use the same architectures used in Fu et al. (2021). We use their GitHub script[4] for this purpose.

Training. We train all the networks for 100 epochs. The initial learning rate is 0.1. Learning rate decays to 0.01 at epoch 40 and to 0.001 at epoch 80. We use a stochastic gradient descent optimizer with a momentum factor of 0.9, weight decay factor of 0.0005, and batch size of 128. For adversarial training, we follow the procedure in Madry et al. (2017). We use 10 steps of projected gradient descent with a step size of $0.15\rho_a$.

Analysis of FUN. For grayscaling experiments, we use images with their average channel values as the input to the network. Test accuracy is computed on the grayscaled test datasets. For smoothing, we use the GitHub codes[5] from (Cohen et al., 2019) (MIT License). For mixup (Zhang et al., 2017), we use the default value of $\alpha = 1.0$.

Filter adversarial training. We experiment with various filter sizes of 3,5, and 7 for the transpose convolution filters. For each batch of data, we use 10 steps of projected gradient descent with a learning rate of 0.1 to learn transpose convolution filters for each class. The weights and biases of the transpose convolution filters are constrained to be within $[-C, C]$. We choose $C = 5$. After 10 steps of inner maximizing optimization, the resulting image is rescaled such that the pixel values lie in $[0, 1]$. See Figure 4 for clean test accuracy vs. epochs plot for Filter Adversarial Training with varying transpose filter sizes. As seen in Figure 1, FAT can break FUN CIFAR-10 with a low blur parameter value of $p_b = 0.1$ to get a clean test accuracy ∼78%. However, with higher p_b values FAT can not achieve more than 50% clean test accuracy. FAT solves the following optimization problem:

$$\min_{\theta} \frac{1}{n} \Sigma_{i=1}^{n} \max_{\|s_{y_i}\|_\infty \leq C} \ell(f_\theta(\mathbf{x_i} \star \mathbf{s_{y_i}}), \mathbf{y_i}) \tag{3}$$

where $\star$ denotes the transpose convolution operator, s_{y_i} denotes the transpose convolution filter for class y_i, and ℓ is the soft-max cross-entropy loss function.

FUN with augmentations. We use mixup with the default $\alpha = 1.0$ (Zhang et al., 2017). See Figure 5 for the training curve. For random blurring augmentations, we use $p'_b = 0.1, 0.3$ and $k = 3$. With both these parameters, FUN is seen to be effective. See Figure 5 for the training curve with $p'_b = 0.3$.

A.9 More experimental results

Figure 6 shows the FUN CIFAR-10 data generated using $k = 3$ and different p_b blur parameters. Figure 7 shows the FUN CIFAR-100 and ImageNet-100 data generated using $k = 3, p_b = 0.3$ and $k = 9, p_b = 0.06$, respectively. Figure 8 shows the clean test accuracy of ResNet-18 with FUN CIFAR-10 generated using different blur parameters. As we see in the plots, higher the blur parameter, better the effectiveness of FUN is. However, we choose $p_b = 0.3$ for our experiments since the the images generated using this hyperparameter look perceptibly more similar to the clean images (when

[3] https://github.com/fshp971/robust-unlearnable-examples

[4] https://github.com/fshp971/robust-unlearnable-examples/tree/main/models

[5] https://github.com/Hadisalman/smoothing-adversarial

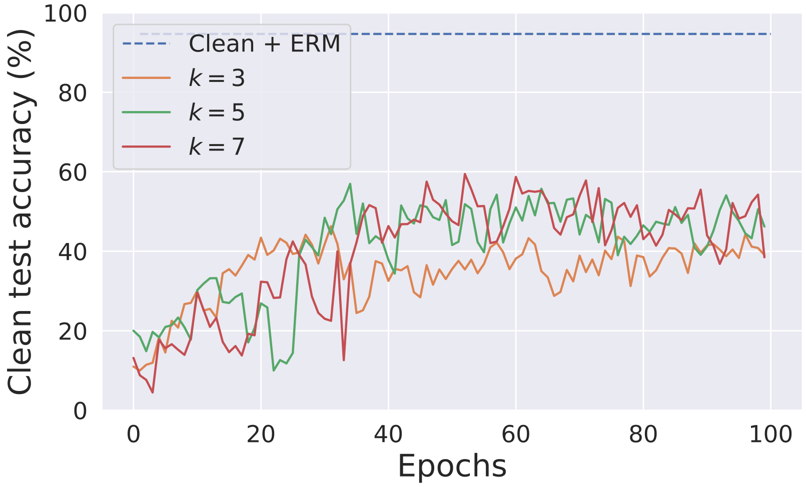


Figure 4: FUN CIFAR-10 images ($k = 3, p_b = 0.3$) trained using ResNet-18 with the Filter Adversarial Training framework with varying transpose convolution filter sizes k.

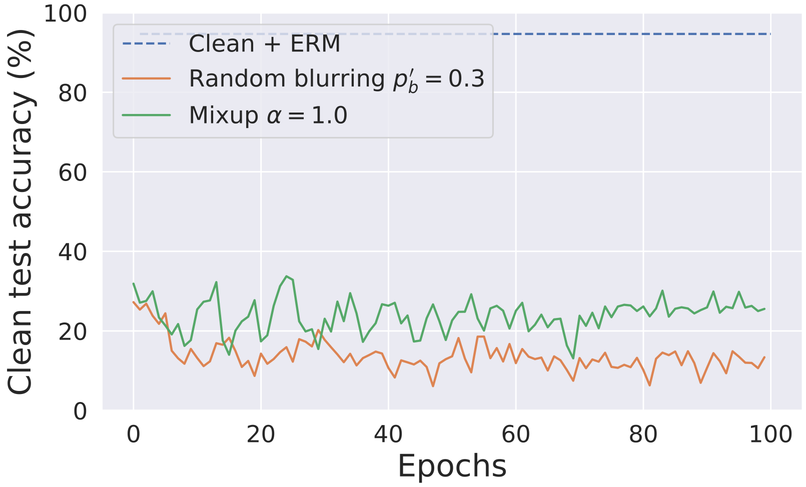
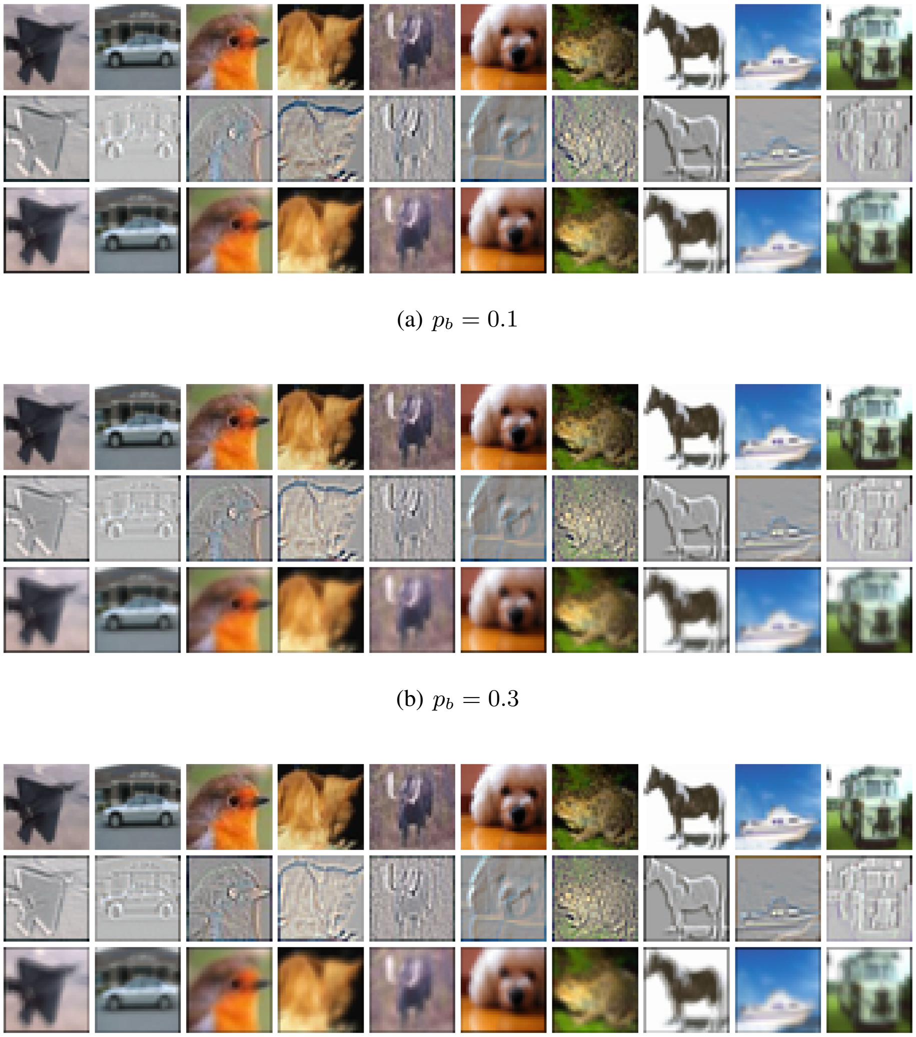


Figure 5: FUN CIFAR-10 images ($k = 3, p_b = 0.3$) trained using ResNet-18 with mixup and random blurring augmentations.

compared to $p_b = 0.5$) while giving a very low clean test accuracy. A lower value of $p_b = 0.1$ gives better unlearnability. However, FUN CIFAR-10 generated using $p_b = 0.1$ is not robust with our Filter Adversarial Training, as shown in Figure 1. Figure 9 shows the clean test accuracy of ResNet-18 with FUN ImageNet-100 dataset generated using different filter sizes. Figure 10 shows the adversarial training curves for ResNet-18 with different FUN datasets.

A.10 Effects of blurring

Here, we study the effects of blurring. We investigate if class-wise blurring is required for achieving unlearnability. For this, we use a universal filter (generated using the same $p_b = 0.3$ and $k = 3$ hyperparameters) to blur all the training images in the dataset. A ResNet-18 trained on this dataset achieves a clean test accuracy of 90.47%. Essentially, the blurring that is performed only degrades the clean test accuracy by ∼4%. This means that class-wise blurring (FUN) is required for achieving the



(a) $p_b = 0.1$

(b) $p_b = 0.3$

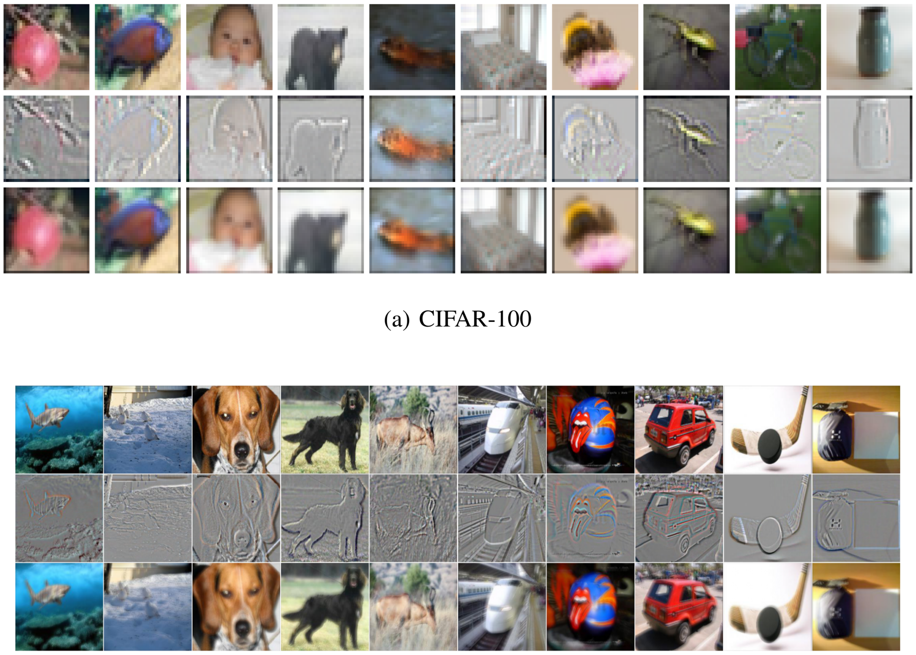
(c) $p_b = 0.5$

Figure 6: FUN CIFAR-10 images generated using different blur parameters p_b. The top row shows the clean images, the bottom row shows the corresponding FUN image, and the middle row shows the normalized difference between the clean and the FUN image.

unlearnability effect (see Figure 11). This experiment also demonstrates that the blurring we perform does not make the dataset useless or destroy its semantics. For this experiment, we use models with fixed initialization and random seeds.

A.11 Why does FUN work?

In this section, we perform experiments that show that a model trained on FUN dataset learns the relation between the class-wise filters and the labels. We train ResNet-18 using the FUN CIFAR-10 dataset for the experiments. We perform three independent trials for each of the experiments and report the mean performance scores. Trained models achieve a mean clean test accuracy of 21.34%. Now, we use the class-wise filters to perturb the images in the test set based on their corresponding labels. Trained models achieve a very high mean accuracy of 99.91% on this perturbed test set. This shows that the trained models learned the relation between the filters and their corresponding labels. Next, we permute the filters to perturb the test set such that test set images with label i are perturbed



(a) CIFAR-100

(b) ImageNet-100

Figure 7: FUN CIFAR-100 and ImageNet-100 images generated using $k = 3, p_b = 0.3$ and $k = 9, p_b = 0.06$, respectively. The top row shows the clean images, the bottom row shows the corresponding FUN image, and the middle row shows the normalized difference between the clean and the FUN image.

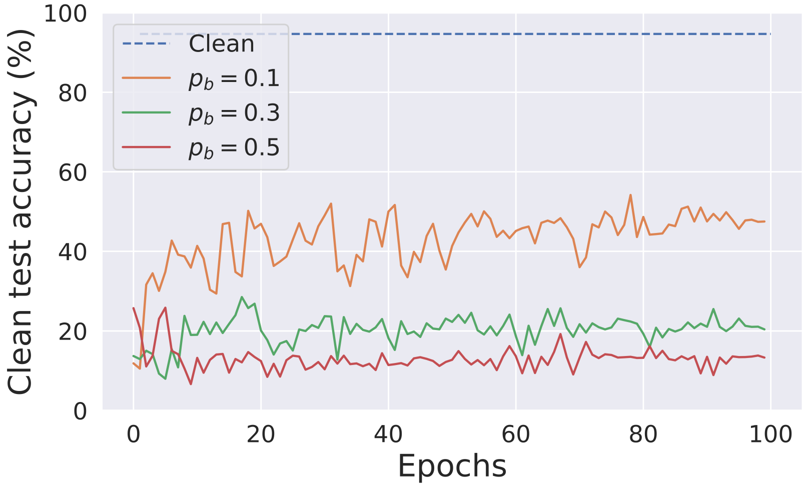


Figure 8: ResNet-18 trained using FUN CIFAR-10 data generated using different blur parameters p_b.

with the filters of class $(i + 1)\%10$. Trained models achieve a very low mean accuracy of 2.53% on this perturbed test set. This is evidence that FUN can also be used for backdoor attacks.

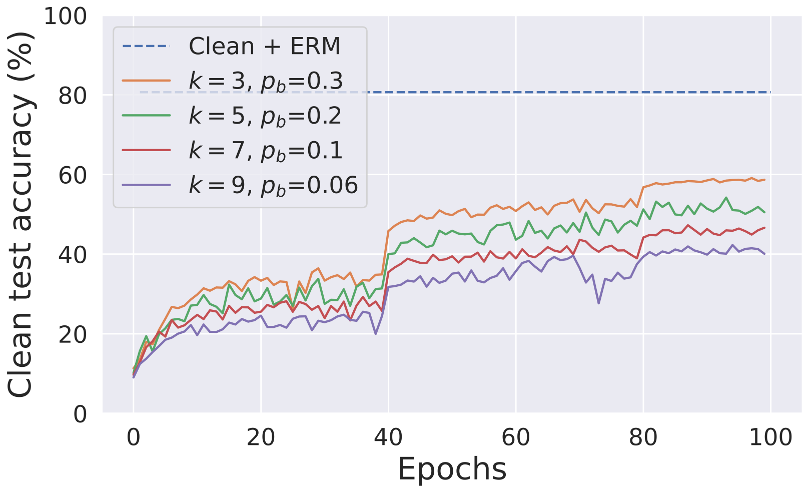
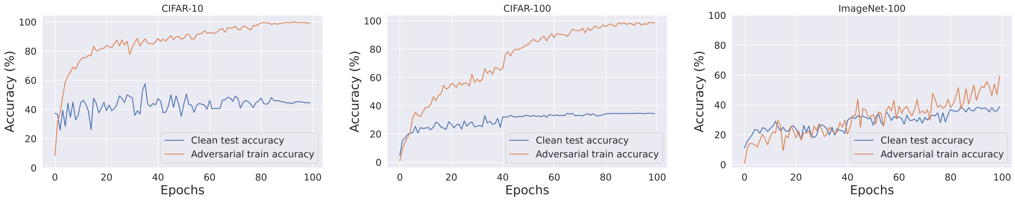


Figure 9: ResNet-18 trained using FUN ImageNet-100 dataset generated using different filter sizes k.



(a) CIFAR-10

(b) CIFAR-100

(c) ImageNet-100

Figure 10: Adversarial training curves for ResNet-18 with CIFAR-10, CIFAR-100, and ImageNet-100 FUN datasets.

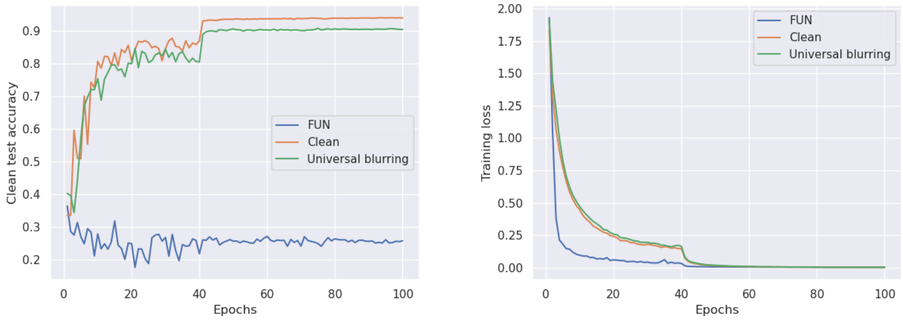
A.12 Effect of transfer learning

In this section, we experiment the effect of using a pre-trained ResNet-18 with PyTorch (Paszke et al., 2019). We train it on the FUN CIFAR-10 dataset in two different ways. First, we fine-tune the whole network on the FUN dataset with a learning rate of 0.001 for 15 epochs. This achieves a clean test accuracy of 42.42%. Fine-tuning the network with clean training data gives 94.19% clean test accuracy. Next, we freeze all the layers except the final layer to train a linear classifier with the pre-trained weights using the FUN CIFAR-10 dataset. We call this “Freeze and learn". We use a SGD optimizer to train the linear layer for 15 epochs with an initial learning rate of 0.1. The learning rate is decayed by a factor of 10 after every 5 epochs. This achieves a clean test accuracy of 48.22%. The results are shown in Figure 12. This experiment shows that pre-trained network with FUN data training does not help achieve good generalization on the clean data distribution.

A.13 Effect of FUN with regularization techniques

In this section, we study the effect of training a ResNet-18 with FUN CIFAR-10 dataset using various regularization techniques such as mixup (Zhang et al., 2017), cutout (DeVries & Taylor, 2017), cutmix (Yun et al., 2019), autoaugment (Cubuk et al., 2018), and orthogonal regularization (Brock et al., 2016). We perform mixup, cutout, cutmix, autoaugment, and orthogonal regularization to achieve 25.53%, 25.80%, 26.93%, 34.09%, and 50.72%. Even though these regularizations help in improving the vanilla ERM training, these networks still do not achieve good generalization on the clean data distribution. We use cutout using GitHub codes[6] with `length=16` and `n_holes=1`, cutmix using

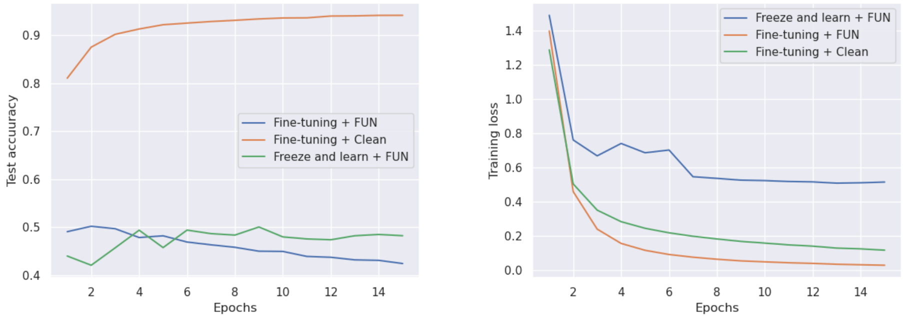
[6] `https://github.com/uoguelph-mlrg/Cutout/blob/master/util/cutout.py`



(a) Test accuracy

(b) Training loss

Figure 11: ResNet-18 trained using clean, FUN, and universally blurred CIFAR-10 datasets.



(a) Test accuracy

(b) Training loss

Figure 12: Pre-trained ResNet-18 with fine-tuning and training the linear layer using FUN and clean CIFAR-10 datasets.

GitHub codes[7] with $\alpha = 1$, autoaugment using PyTorch (Paszke et al., 2019), mixup using GitHub codes[8] with $\alpha = 1$, and orthogonal regularization using GitHub codes[9] with `reg=1e-6` (all MIT licenses).

A.14 Network parameter distribution

In this section, we compare the network parameter distributions of ResNet-18 trained on clean and FUN CIFAR-10 datasets (see Figure 13). Both the distributions are similar to normal distributions with a mean of 0. However, the parameter distribution of the clean model has a higher standard deviation than the FUN-based model's parameter distribution.

[7] https://github.com/hysts/pytorch_cutmix/blob/master/cutmix.py

[8] https://github.com/facebookresearch/mixup-cifar10/blob/main/train.py

[9] https://github.com/kevinzakka/pytorch-goodies

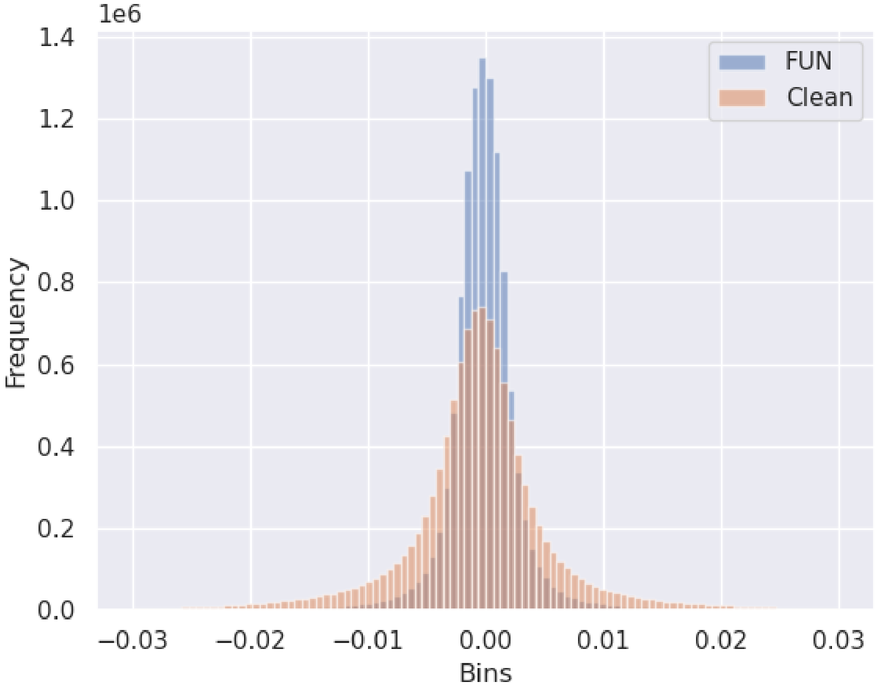


Figure 13: Network parameter distributions of ResNet-18 trained on clean and FUN CIFAR-10 datasets.