Purify Unlearnable Examples via Rate-Constrained Variational Autoencoders

Yi Yu [1 2]   Yufei Wang [2]   Song Xia [2]   Wenhan Yang [† 3]   Shijian Lu [4]   Yap-Peng Tan [2]   Alex C. Kot [2]

Abstract

Unlearnable examples (UEs) seek to maximize testing error by making subtle modifications to training examples that are correctly labeled. Defenses against these poisoning attacks can be categorized based on whether specific interventions are adopted during training. The first approach is training-time defense, such as adversarial training, which can mitigate poisoning effects but is computationally intensive. The other approach is pre-training purification, *e.g.,* image short squeezing, which consists of several simple compressions but often encounters challenges in dealing with various UEs. Our work provides a novel disentanglement mechanism to build an efficient pre-training purification method. Firstly, we uncover rate-constrained variational autoencoders (VAEs), demonstrating a clear tendency to suppress the perturbations in UEs. We subsequently conduct a theoretical analysis for this phenomenon. Building upon these insights, we introduce a disentangle variational autoencoder (D-VAE), capable of disentangling the perturbations with learnable class-wise embeddings. Based on this network, a two-stage purification approach is naturally developed. The first stage focuses on roughly eliminating perturbations, while the second stage produces refined, poison-free results, ensuring effectiveness and robustness across various scenarios. Extensive experiments demonstrate the remarkable performance of our method across CIFAR-10, CIFAR-100, and a 100-class ImageNet-subset. Code is available at https://github.com/yuyi-sd/D-VAE.

[†]Corresponding author [1]Rapid-Rich Object Search Lab, Interdisciplinary Graduate Programme, Nanyang Technological University, Singapore [2]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore [3]PengCheng Laboratory, Shenzhen, China [4]School of Computer Science and Engineering, Nanyang Technological University, Singapore. Correspondence to: Yi Yu <yuyi0010@e.ntu.edu.sg>, Wenhan Yang <yangwh@pcl.ac.cn>.

1. Introduction

Although machine learning models often achieve impressive performance on a range of challenging tasks, their effectiveness can significantly deteriorate in the presence of the gaps between the training and testing data distributions. One of the most widely studied types of these gaps is related to the vulnerability of standard models to adversarial examples (Goodfellow et al., 2014; Yu et al., 2022b; Xia et al., 2024; Wang et al., 2024), posing a significant threat to the inference phase. However, a destructive and often underestimated threat emerges from malicious perturbations at the training phase, namely unlearnable examples, which seek to maximize testing error by making subtle modifications of correctly labeled training examples (Feng et al., 2019).

In the era of big data, vast amounts are freely collected from the Internet, powering advances in DNNs (Schmidhuber, 2015). Nonetheless, it's essential to note that online data may contain proprietary or private information, raising concerns about unauthorized use. UEs are considered a promising route for data protection (Huang et al., 2021). Recently, many efforts have emerged to add invisible perturbations to images as shortcuts to disrupt the training process (Yu et al., 2022a; Lin et al., 2024). On the other hand, data exploiters perceive these protection techniques as potential threats to a company's commercial interests, leading to extensive research efforts in developing defenses. Previous research has demonstrated that training-time defenses, such as adversarial training and adversarial augmentations, can alleviate poisoning effects. However, their practicality is limited by the massive computational costs. Recently, preprocessing-based defenses have gained attention with simple compressions like JPEG and grayscale demonstrating the advantages over adversarial training in computational efficiency (Liu et al., 2023). However, these methods lack universality, as different compression techniques might be best suited for various attacks. Pre-training purification has demonstrated great potential in addressing the issue of UEs in both effectiveness and efficiency (Liu et al., 2023). This kind of method doesn't intervene in the model's training but instead concentrates on refining the data, which well aligns with the recent theme of data-centric AI (DCAI) (Zha et al., 2023). Focusing on fundamental data-related issues rather than relying on untrusted or compromised data leads to more reliable and effective machine learning models.
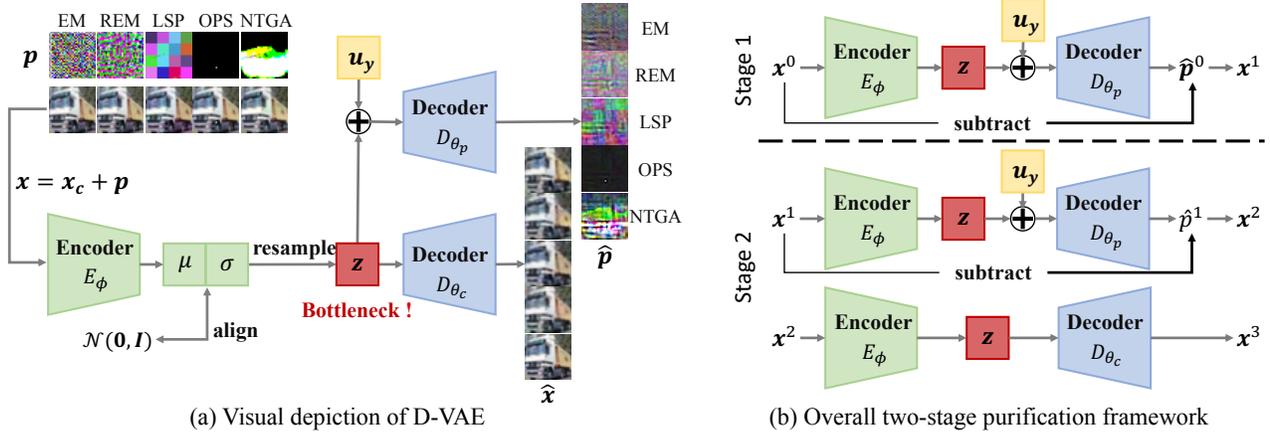
Figure 1. (a) Visual depiction of D-VAE containing two components. One component generates reconstructed images $\hat{x}$, preserving the primary content of unlearnable inputs x. The auxiliary decoder maps a trainable class-wise embedding u_y and latents z to disentangled perturbations $\hat{p}$. Here, x_c is clean data, and p is added perturbations. Perturbations are normalized for better views. (b) The purification framework consisting of two stages. The overall purification can be formulated as $x^3 = g(x^0)$, where x^0 is the original unlearnable data.

In this paper, we focus on the pre-training purification paradigm. Our overall approach is to utilize a disentanglement mechanism to separate the poison signal from the intrinsic signal of the image with a rate-constrained VAE to obtain clean data. Firstly, we discover that a rate-constrained VAE can effectively remove the added perturbations by constraining the KL divergence in latents when compared to JPEG (Guo et al., 2018) in Sec 3.2, with a derived detailed theoretical explanation in Sec 3.3. Specifically, we formulate UEs as the transformation of less-predictive features into highly predictive ones. This perspective reveals that perturbations with a larger inter-class distance and smaller intra-class variance can create stronger attacks by shifting the optimal separating hyperplane of a Bayes classifier. Subsequently, we show that VAEs are particularly effective at suppressing perturbations possessing these characteristics. Furthermore, we observe that most perturbations exhibit lower class-conditional entropy. Thus, we propose a method involving learnable class-wise embeddings to disentangle these added perturbations.

Building upon these findings, we present a purification framework that offers consistent and adaptable defense against UEs in Sec 3.4 and Sec 3.5. In Figure 1 (a), we present the D-VAE, comprising two components, capable of generating a reconstructed image $\hat{x}$ with minimal poisoning perturbations and disentangling predicted perturbations $\hat{p}$ with a trainable class-wise embedding u_y. Subsequently, leveraging D-VAE, we propose a two-stage purification framework illustrated in Figure 1 (b). In each stage, we train D-VAE on the unlearnable dataset and perform inference using the trained D-VAE on the same dataset. Our two-stage purification framework primarily involves two operations: 1) Estimating perturbations $\hat{p}$ and subtracting them from x; 2) Obtaining reconstructed data $\hat{x}$ from D_{θ_c} to serve as purified images. While the subtraction process

occurs at both stages, the acquisition of $\hat{x}$ takes place at the end of the second stage. With this method, models trained on our purified datasets can achieve significant boosts compared with previous SOTA methods: improved from 84% to 90% on CIFAR-10 (Krizhevsky et al., 2009) and from 64% to 75% on the ImageNet-subset (Deng et al., 2009).

In summary, our contributions can be outlined as follows:

• We discover that rate-constrained VAEs exhibit a preference for removing perturbations in UEs, and offer a comprehensive theoretical analysis to support this finding.

• We introduce D-VAE, a network that can disentangle the added perturbations and generate purified data. Our additional evaluations also show that D-VAE can purify UEs from a mixed dataset, and is able to produce new UEs, even if it only accesses to just a small fraction (1%) of UEs of the entire dataset.

• On top of the D-VAE, we propose a unified purification framework for countering various UEs. Extensive experiments demonstrate the remarkable performance of our method across CIFAR-10, CIFAR-100, and a 100-class ImageNet-subset, encompassing multiple poison types and different perturbation strengths, *e.g.,* with only 4% drop on ImageNet-subset compared to models trained on clean data.

2. Related Work

2.1. Data Poisoning

Data poisoning attacks (Barreno et al., 2010; Goldblum et al., 2022; Yu et al., 2023), involving the manipulation of training data to disrupt the performance of models during inference, can be broadly categorized into two main types: integrity attacks and availability attacks. Integrity attacks aim to manipulate the model's output during infer-

ence (Barreno et al., 2006; Xiao et al., 2015; Zhao et al., 2017), *i.e.,* backdoor attacks (Gu et al., 2017; Schwarzschild et al., 2021), where the model behaves maliciously only when presented with data containing specific triggers. In contrast, availability attacks aim to degrade the overall performance on validation and test datasets (Biggio et al., 2012; Xiao et al., 2015). Typically, such attacks inject poisoned data into the clean training set. Poisoned samples are usually generated by adding unbounded perturbations, and take only a fraction of the entire dataset (Koh & Liang, 2017; Zhao & Lao, 2022; Lu et al., 2023). These methods are primarily designed for malicious purposes, and the poisoned samples are relatively distinguishable.

Unlearnable Examples. Another recent emerging type is unlearnable examples (UEs) (Feng et al., 2019; Huang et al., 2021), where samples from the entire training dataset undergo subtle modifications (*e.g.,* bounded perturbations $\|\boldsymbol{p}\|_\infty \leq \frac{8}{255}$), and are correctly labeled. This type of attack, also known as perturbative availability poisoning attacks (Liu et al., 2023), can be viewed as a promising approach for data protection. Models trained on such datasets often approach random guessing performance on clean test data. EM (Huang et al., 2021) employ error-minimizing noise as perturbations. NTGA (Yuan & Wu, 2021) generate protective noise using an ensemble of neural networks modeled with neural tangent kernels. TAP (Fowl et al., 2021) employ targeted adversarial examples as UEs. REM (Fu et al., 2022) focuses on conducting robust attacks against adversarial training. Subsequently, LSP (Yu et al., 2022a) explore effecient and surrogate-free UEs, and extending the perturbations to be ℓ_2 bounded. Recently, OPS (Wu et al., 2023) introduce one-pixel shortcuts, which enhances the robustness to adversarial training and strong augmentations.

2.2. Existing Defenses

Defenses against UEs can be categorized into training-time defenses and pre-training purification, depending on interventions applied during or before the training phase. Huang et al. (2021) shown that UEs are robust to data augmentations, *e.g.,* Mixup (Zhang et al., 2018). Tao et al. (2021) find that adversarial training (AT) could mitigate poisoning effects, but it is computationally expensive and cannot fully restore performance. Building on the idea of AT, Qin et al. (2023b) employ adversarial augmentations (AA), but it still demands intensive training and does not generalize well to ImageNet-subset. For pre-training defenses, Liu et al. (2023) indicates that pre-filtering, *e.g.,* gaussian smoothing, median filtering, show substantial effects but not comparable to AT. Instead, Liu et al. (2023) propose image shortcut squeezing (ISS) including JPEG compression, grayscale, and bit depth reduction (Wang et al., 2018) to defense, while each technique does not fit all UEs approaches. Moreover, it is noted that low-quality JPEG compression, while effective

for defense, significantly degrades image quality. AVATAR (Dolatabadi et al., 2023) and LEs (Jiang et al., 2023) both employ a diffusion model for purification, but those methods require a substantial amount of additional clean data to train the diffusion model (Ho et al., 2020; Song et al., 2021), making it impractical. LFU (Sandoval-Segura et al., 2023) is a hybrid method that adopt orthogonal projection to learn perturbations before training and employs strong augmentations during training due to incomplete purifications. However, it is constrained to UEs methods that adopt class-wise linear perturbations, limiting its applicability.

3. Methodology

3.1. Preliminaries

Formally, for UEs, all training data can be perturbed to some extent, while the labels should remain correct (Feng et al., 2019; Fowl et al., 2021). We introduce two parties: the attacker (also called the poisoner), and the victim. The attacker has the ability to perturb the victim's training data, *i.e.,* from $(\boldsymbol{x_c}^{(i)}, y^{(i)})$ to $(\boldsymbol{x_c}^{(i)} + \boldsymbol{p}^{(i)}, y^{(i)})$. The victim then trains a new model on the poisoned data, *i.e.,* obtaining $\theta^*(\boldsymbol{p})$. The attacker's success is determined by the accuracy of the victim model on clean data, *i.e.,* maximizing the loss on clean data $\mathcal{L}(F(\boldsymbol{x_c}; \theta^*(\boldsymbol{p})), y)$. The task to craft poisoning perturbations can be formalized into the following bi-level optimization problem:

$$\max_{\boldsymbol{p} \in \mathcal{S}} \mathbb{E}_{(\boldsymbol{x_c}, y) \sim \mathcal{D}} \big[\mathcal{L}(F(\boldsymbol{x_c}; \theta^*(\boldsymbol{p})), y) \big], \quad \text{s.t. } \theta^*(\boldsymbol{p}) =$$
$$\arg \min_{\theta} \sum_{(\boldsymbol{x_c}^{(i)}, y^{(i)}) \in \mathcal{T}} \mathcal{L}(F(\boldsymbol{x_c}^{(i)} + \boldsymbol{p}^{(i)}; \theta), y^{(i)}), \qquad (1)$$

where $\boldsymbol{x_c}$ is the clean data, and $\mathcal{S}$ is the feasible region for perturbations, *e.g.,* $\|\boldsymbol{p}\|_\infty \leq \frac{8}{255}$. By adding perturbations $\boldsymbol{p}^{(i)}$ to samples $\boldsymbol{x_c}^{(i)}$ from the clean training dataset $\mathcal{T}$ to formulate the unlearnable training dataset $\mathcal{P}$, the adversary aims to induce poor generalization of the trained model F to the clean test dataset $\mathcal{D}$.

Conversely, data exploiters aim to obtain the learnable data by employing a mapping $\boldsymbol{g}$ such that:

$$\min_{\boldsymbol{g}} \mathbb{E}_{(\boldsymbol{x_c}, y) \sim \mathcal{D}} \big[\mathcal{L}(F(\boldsymbol{x_c}; \theta^*(\boldsymbol{g})), y) \big], \quad \text{s.t. } \theta^*(\boldsymbol{g}) =$$
$$\arg \min_{\theta} \sum_{(\boldsymbol{x_c}^{(i)} + \boldsymbol{p}^{(i)}, y^{(i)}) \in \mathcal{P}} \mathcal{L}(F(\boldsymbol{g}(\boldsymbol{x_c}^{(i)} + \boldsymbol{p}^{(i)}); \theta), y^{(i)}). \qquad (2)$$

In this paper, we focus on pre-training purification, where $\boldsymbol{g}$ is applied for that purification, before training the classifier.

Notations. For better comprehension of the subsequent sections, we present notations for all remaining variables. $\hat{\boldsymbol{p}}$ is the estimated perturbations by D-VAE. $\hat{\boldsymbol{x}}/\boldsymbol{z}$ is the reconstructed data, and the latents encoded by the VAE/D-VAE, respectively. For the trainable modules, D-VAE consists of encoder E_ϕ, decoder D_{θ_c}, auxiliary decoder D_{θ_p}, and class-wise embeddings $\boldsymbol{u_y}$.
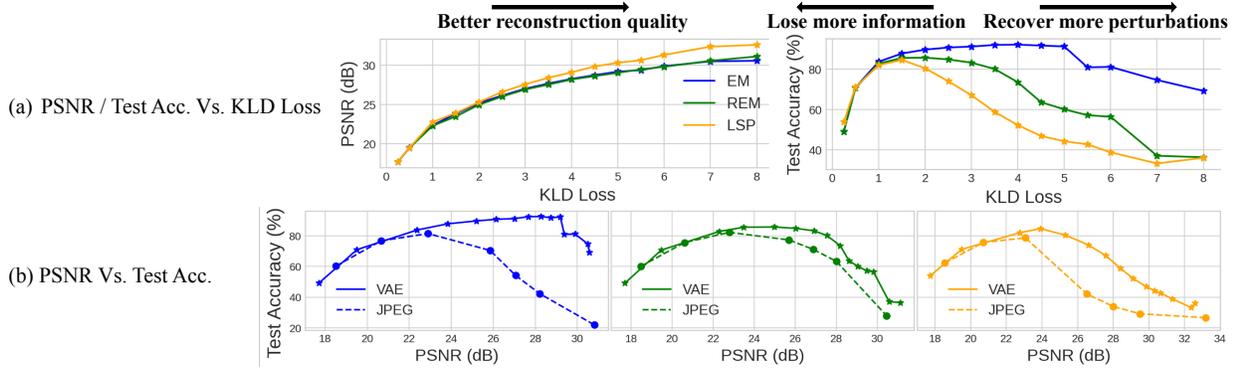
Figure 2. (a): Results of VAEs: PSNR/Test Accuracy Vs. KLD Loss are assessed on the unlearnable CIFAR-10. (b): Comparison between VAEs and JPEG compression: PSNR Vs. Test Accuracy. Note that we adopt JPEG with quality {2,5,10,30,50,70,90} to control the corruption levels. We include EM, REM, and LSP as UEs methods.

3.2. VAEs Can Effectively Mitigate the Impact of Poisoning Perturbations in UEs

The VAE maps the input to a lower-dimensional latent space, generating parameters for a variational distribution. The decoder reconstructs data from this latent space. The loss function combines a reconstruction loss ("distortion") with a Kullback-Leibler (KL) divergence term ("rate"), acting as a limit on mutual information and serving as a compression regularizer (Bozkurt et al., 2021).

Since UEs have demonstrated vulnerability to compressions like JPEG, we first investigate whether a rate-constrained VAE can eliminate these perturbations and obtain the restored learnable samples. In essence, we introduce an updated loss function incorporating a rate constraint as follows ($\mathcal{P}$ is the unlearnable dataset, and x is the UEs):

$$\mathcal{L}_{\text{VAE}} = \sum_{\boldsymbol{x}, y \in \mathcal{P}} \underbrace{\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2}_{\text{distortion}} + \lambda \underbrace{\max(\text{KLD}(\boldsymbol{z}, \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})), \text{kld}_{\text{target}})}_{\text{rate constraint}}, \tag{3}$$

where the KLD Loss is formulated from Kingma & Welling (2014) and provided in the Appendix B.1, and the kld$_{\text{target}}$ serves as the target value for the KLD loss. We proceed to train the VAE on the unlearnable CIFAR-10. Subsequently, we report the accuracy on the clean test set achieved by a ResNet-18 trained on the reconstructed images. In Figure 2(a), reducing the KLD loss decreases reconstruction quality (measured by Peak Signal-to-Noise Ratio (PSNR) between $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$). This reduction can eliminate added perturbations and original valuable features. The right image of Figure 2(a) shows that increased removal of perturbations in $\hat{\boldsymbol{x}}$ correlates with improved test accuracy. However, heavily corrupting $\hat{\boldsymbol{x}}$ by further reducing kld$_{\text{target}}$ removes more valuable features, leading to a drop in test accuracy. In Figure 2(b), the comparison with JPEG at various quality settings shows that when processed through VAEs and JPEG to achieve similar PSNR, test accuracy with VAEs is higher than JPEG. This suggests that VAEs are significantly more effective at eliminating perturbations than JPEG compres-

sion, when achieving similar levels of reconstruction quality. Then, we delve into why VAEs can exhibit such preference.

3.3. Theoretical Analysis and Intrinsic Characteristics

Given that the feature extractor maps the input data to the latent space is pivotal for the classification conducted by DNNs, we conduct our analysis on the latent features $\boldsymbol{v}$.

Hyperplane shift caused by attacks. Consider the following binary classification problem with regards to the features extracted from the data $\boldsymbol{v} = (\boldsymbol{v}_c, \boldsymbol{v}_s^t)$ consisting of a predictive feature $\boldsymbol{v}_c$ of a Gaussian mixture $\mathcal{G}_c$ and a non-predictive feature $\boldsymbol{v}_s^t$ which follows:

$$y \overset{u \cdot a \cdot r}{\sim} \{0, 1\}, \ \boldsymbol{v}_c \sim \mathcal{N}(\boldsymbol{\mu}_c^y, \boldsymbol{\Sigma}_c), \ \boldsymbol{v}_s^t \sim \mathcal{N}(\boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t), \\ \boldsymbol{v}_c \perp\!\!\!\perp \boldsymbol{v}_s^t, \ \Pr(y = 0) = \Pr(y = 1). \tag{4}$$

Proposition 3.1. *For the features $\boldsymbol{v} = (\boldsymbol{v}_c, \boldsymbol{v}_s^t)$ following the distribution (4), the optimal separating hyperplane using a Bayes classifier is formulated by:*

$$\boldsymbol{w}_c^\top (\boldsymbol{v}_c^* - \frac{\boldsymbol{\mu}_c^0 + \boldsymbol{\mu}_c^1}{2}) = 0, \ s.t. \ \boldsymbol{w}_c = \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\mu}_c^0 - \boldsymbol{\mu}_c^1). \tag{5}$$

The proof is provided in Appendix A.1. Subsequently, we assume that a malicious attacker modifies $\boldsymbol{v}_s^t$ to $\boldsymbol{v}_s$ of the following distributions $\mathcal{G}_s$ to make it predictive for training a Bayes classifier:

$$y \overset{u \cdot a \cdot r}{\sim} \{0, 1\}, \quad \boldsymbol{v}_s \sim \mathcal{N}(\boldsymbol{\mu}_s^y, \boldsymbol{\Sigma}_s), \quad \boldsymbol{v}_c \perp\!\!\!\perp \boldsymbol{v}_s. \tag{6}$$

Theorem 3.2. *Consider features from the training data for the Bayes classifier is modified from $\boldsymbol{v} = (\boldsymbol{v}_c, \boldsymbol{v}_s^t)$ in Eq. 4 to $\boldsymbol{v} = (\boldsymbol{v}_c, \boldsymbol{v}_s)$ in Eq. 6, the hyperplane is shifted with a distance given by:*

$$d = \|\boldsymbol{w}_s^\top (\boldsymbol{v}_s - \frac{\boldsymbol{\mu}_s^0 + \boldsymbol{\mu}_s^1}{2})\|_2 \Big/ \|\boldsymbol{w}_c\|_2, \\ s.t. \ \boldsymbol{w}_c = \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\mu}_c^0 - \boldsymbol{\mu}_c^1), \ \boldsymbol{w}_s = \boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_s^0 - \boldsymbol{\mu}_s^1). \tag{7}$$

The proof is provided in Appendix A.2. When conducting evaluations on the testing data that follows the same distribution as the clean data $\boldsymbol{v} = (\boldsymbol{v}_c, \boldsymbol{v}_s^t)$, with the term $\boldsymbol{v}_s$ in

Eq. 7 replaced by $\boldsymbol{v}_s^t$, it leads to a greater prediction error if $\|\boldsymbol{w}_s\|_2 \gg \|\boldsymbol{w}_c\|_2$. Theorem 3.2 indicates that perturbations which create strong attacks tend to have a larger inter-class distance and a smaller intra-class variance.

Error when aligning with a normal distribution. Consider a variable $\boldsymbol{v} = (v_1, \ldots, v_d)$ following a mixture of two Gaussian distributions $\mathcal{G}$:

$$y \overset{u.a.r}{\sim} \{0,1\}, \; \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{\mu}^y, \boldsymbol{\Sigma}),$$
$$v_i \sim \mathcal{N}(\mu_i^y, \sigma_i), \; v_i \perp\!\!\!\perp v_j, \; \Pr(y=0) = \Pr(y=1), \quad (8)$$
$$p_{v_i}(v) = [\mathcal{N}(v; \mu_i^0, \sigma_i) + \mathcal{N}(v; \mu_i^1, \sigma_i)]/2.$$

Each dimensional feature v_i is also modeled as a Gaussian mixture. To start, we normalize each feature through a linear operation to achieve a distribution with zero mean and unit variance. The linear operation and the modified density function can be expressed as follows:

$$z_i = \frac{v_i - \hat{\mu}_i}{\sqrt{(\sigma_i)^2 + (\delta_i)^2}}, \; p_{z_i}(v) = \frac{p_0(v) + p_1(v)}{2},$$
$$p_0(v) = \mathcal{N}(v; -\hat{\delta}_i, \hat{\sigma}_i), \; p_1(v) = \mathcal{N}(v; \hat{\delta}_i, \hat{\sigma}_i), \quad (9)$$
$$\text{where} \qquad \hat{\mu}_i = \frac{\mu_i^0 + \mu_i^1}{2}, \; \delta_i = |\frac{\mu_i^0 - \mu_i^1}{2}|,$$
$$\hat{\delta}_i = \delta_i / \sqrt{(\sigma_i)^2 + (\delta_i)^2}, \; \hat{\sigma}_i = \sigma_i / \sqrt{(\sigma_i)^2 + (\delta_i)^2}.$$

Theorem 3.3. *Denote $r_i = \frac{\delta_i}{\sigma_i} > 0$, the Kullback–Leibler divergence between $p_{z_i}(v)$ in (9) and a standard normal distribution $\mathcal{N}(v; 0, 1)$ is bounded by:*

$$\frac{\ln(1 + (r_i)^2)}{2} - \ln 2 \leq KLD(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)) \leq \frac{\ln(1 + (r_i)^2)}{2}, \quad (10)$$

and observes the following property:

$$\uparrow r_i \quad \Longrightarrow \quad \uparrow S(r_i) = KLD(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)). \quad (11)$$

The proof for Theroem 3.3 is provided in Appendix A.3.

Remark 3.4. The training of a VAE includes the process of mapping the data $\boldsymbol{x}$ to latents. We can break this process into two step2: 1) The encoder first map the $\boldsymbol{x}$ to lossless intermediate representations $z_i \sim p_{z_i}(v)$; 2) The encoder estimate (re-project/remap) the intermediate representations to a new distribution $\hat{P}$ subject to $KLD(\hat{P} \| \mathcal{N}(0, 1)) < \epsilon$. According to Theorem 3.3, for each lossless intermediate representation z_i with $r < S^{-1}(\epsilon)$, we can apply an identical mapping $\hat{P} = p_{z_i}(v)$ without requiring the step 2 mentioned above. In this way, the final representation z_i is still lossless. For the intermediate representation z_i with $r_{v_i} > S^{-1}(\epsilon)$, we can see that through step 2, the final representation with $\hat{P}$ is forced to have a smaller r, and ideally to be almost equal to $S^{-1}(\epsilon)$. Basically, the error for the two distributions can be denoted as $\int_{-\infty}^{\infty} [\hat{P}(v) - p_{z_i}(v)]^2 dv$ in the distribution space. From this formulation, we can explicitly see that a larger gap $(r_{v_i} - S^{-1}(\epsilon))$ can also lead to larger estimation error. And the estimated $\hat{P}$ is constrained to have a smaller r, making it less predictive for classification.

Remark 3.4 indicates that perturbative patterns that make strong attacks tend to suffer from larger errors when estimating with distributions subject to the constraint on the KLD. Thus, the training of a rate-constrained VAE includes simulating the process of mapping the data to latent representations and aligning them with a normal distribution to a certain extent. The decoder learns to reconstruct the input data from the resampled latents $\boldsymbol{z}$. Consequently, the highly predictive shortcuts are subdued or eliminated in the reconstructed data $\hat{\boldsymbol{x}}$.

Proposition 3.5. *The conditional entropy of a Gaussian mixture $\boldsymbol{v}_s$ of $\mathcal{G}_s$ in Eq. 6 is given by:*

$$H(\boldsymbol{v}_s | y_i) = \frac{dim(\boldsymbol{v}_s)}{2}(1 + \ln(2\pi)) + \frac{1}{2} \ln |\boldsymbol{\Sigma}_s|, \quad (12)$$

where $dim(\boldsymbol{v}_s)$ is the dimensions of the features. If each feature v_s^d is independent, then:

$$H(\boldsymbol{v}_s | y_i) = \frac{dim(\boldsymbol{v}_s)}{2}(1 + \ln(2\pi)) + \sum_{d=1}^{dim(\boldsymbol{v}_s)} \ln \sigma_s^d. \quad (13)$$

As the inter-class distance $\Delta_s = \|\boldsymbol{\mu}_s^0 - \boldsymbol{\mu}_s^1\|_2$ is constrained to ensure the invisibility of the perturbations, perturbations in most UEs exhibit a relatively low intra-class variance. Proposition 3.5 suggests that the class-conditional entropy of the perturbations is comparatively low. Adversarial poisoning (Fowl et al., 2021; Chen et al., 2023) could be an exception since adversarial examples can maximize latent space shifts with minimal perturbation in the RGB space. However, the preference to be removed by VAE still holds.

3.4. D-VAE: VAE with Perturbations Disentanglement

Given that the defender lacks groundtruth values for the perturbations $\boldsymbol{p}$, it is not possible to optimize $\boldsymbol{u_y}$ and D_{θ_p} to learn to predict $\hat{\boldsymbol{p}}$ directly by minimizing $\|\boldsymbol{p} - \hat{\boldsymbol{p}}\|_2^2$ during model training. Expanding on the insights from Section 3.2 and Remark 3.4, when imposing a low target value on the KLD loss, creating an information bottleneck on the latents $\boldsymbol{z}$, the reconstructed $\hat{\boldsymbol{x}}$ cannot achieve perfect reconstruction, making the added perturbations more challenging to be recovered in $\hat{\boldsymbol{x}}$. As a result, a significant portion of perturbations $\boldsymbol{p}$ persists in the residuals $\boldsymbol{x} - \hat{\boldsymbol{x}}$.

Following Proposition 3.5, the majority of perturbations associated with each class data exhibit relatively low entropy, suggesting that they can be largely reconstructed using representations with limited capacity. Considering that most perturbations are crafted to be sample-wise, we propose a learning approach that maps the summation of a trainable class-wise embedding $\boldsymbol{u_y}$ and the latents $\boldsymbol{z}$ to $\hat{\boldsymbol{p}}$ through an auxiliary decoder D_{θ_p}. To learn $\boldsymbol{u_y}$ and train D_{θ_p}, we propose minimizing $\|(\boldsymbol{x} - \hat{\boldsymbol{x}}) - \hat{\boldsymbol{p}}\|_2^2$, as the residuals $\boldsymbol{x} - \hat{\boldsymbol{x}}$ contain the majority of the groundtruth $\boldsymbol{p}$ when imposing a low target value on the KLD loss.

Algorithm 1 Two-stage purification framework of unlearnable examples with D-VAE

    Input: Unlearnable dataset $\mathcal{P}^0$, D-VAE ($E_\phi, D_{\theta_c}, D_{\theta_p}, \boldsymbol{u_y}$), kld$_{\text{target}}$: kld_1, kld_2
    # First stage: recover and remove heavy perturbations by training D-VAE with small kld_1
    Randomly initialize ($\phi, \theta_c, \theta_p, \boldsymbol{u_y}$), and using Adam to minimize Eq. 14 on $\mathcal{P}^0$ with kld_1
    Inference with trained VAE on $\mathcal{P}^0$, and save a new dataset $\mathcal{P}^1$ with sample $\boldsymbol{x}^1 = \boldsymbol{x}^0 - \hat{\boldsymbol{p}}^0$
    # Second stage: generate purified data by training D-VAE with larger kld_2
    Randomly initialize ($\phi, \theta_c, \theta_p, \boldsymbol{u_y}$), and using Adam to minimize Eq. 14 on $\mathcal{P}^1$ with kld_2
    Inference with trained VAE on $\mathcal{P}^1$, and save a new dataset $\mathcal{P}^2$ with sample $\boldsymbol{x}^2 = \boldsymbol{x}^1 - \hat{\boldsymbol{p}}^1$
    Inference with trained VAE on $\mathcal{P}^2$, and save a new dataset $\mathcal{P}^3$ with sample $\boldsymbol{x}^3 = \hat{\boldsymbol{x}}^2$
    Return purified dataset $\mathcal{P}^3$

The overall network is in Figure 1 (a), and the improved loss to optimize the D-VAE ($E_\phi, D_{\theta_c}, D_{\theta_p}, \boldsymbol{u_y}$) is given:

$$\mathcal{L}_{\text{D-VAE}} = \sum_{\boldsymbol{x},\boldsymbol{y} \in \mathcal{P}} \underbrace{\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2}_{\text{distortion}} + \underbrace{\|(\boldsymbol{x} - \hat{\boldsymbol{x}}) - \hat{\boldsymbol{p}}\|_2^2}_{\text{recover perturbations}}$$
$$+ \lambda \cdot \underbrace{\max(\text{KLD}(\boldsymbol{z}, \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})), \text{kld}_{\text{target}})}_{\text{rate constraint}}, \quad (14)$$

where $\boldsymbol{\mu}, \boldsymbol{\sigma} = E_\phi(\boldsymbol{x})$, $\boldsymbol{z}$ is sampled from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, $\hat{\boldsymbol{x}} = D_{\theta_c}(\boldsymbol{z})$, disentangled perturbations $\hat{\boldsymbol{p}} = D_{\theta_p}(\boldsymbol{u_y} + \boldsymbol{z})$.

3.5. Purify UEs with D-VAE

Given the observations in Section 3.2 that a large KLD target fails to effectively surpress the perturbations, while a small one might significantly deteriorate the quality of reconstructed images, we introduce a two-stage purification framework as shown in Algorithm 1. In the first stage, we use a small kld$_{\text{target}}$ to train the VAE with the unlearnable dataset $\mathcal{P}^0$. This approach allows us to reconstruct a significant portion of the perturbations. During inference, we subtract the input $\boldsymbol{x}^0$ from $\mathcal{P}^0$ by the predicted perturbations $\hat{\boldsymbol{p}}^0$ and obtain these modified images as $\mathcal{P}^1$.

In the second stage, we set a larger kld$_{\text{target}}$ for training. After subtracting $\boldsymbol{x}^1$ by $\hat{\boldsymbol{p}}^1$ and saving it as $\boldsymbol{x}^2$ in the first inference. Since the perturbations are learned in an unsupervised manner, it is challenging to achieve complete reconstruction. Hence, we proceed with a second inference and obtain the output $\hat{\boldsymbol{x}}^2$ as the final result.

Selection of kld_1, kld_2. Due to potential variations in the VAE's encoder and dataset's resolutions, we decide kld$_{\text{target}}$ empirically based on PSNR between the input $\boldsymbol{x}$ and the output $\hat{\boldsymbol{x}}$. Specifically, though various UEs methods may utilize different norms to constrain the magnitude of perturbations, the disparity between the clean data $\boldsymbol{x_c}$ and poisoned data $\boldsymbol{x}$ in terms of PSNR is usually slightly above 30. Therefore, the selection of an appropriate kld$_{\text{target}}$ requires this prior information. Across multiple datasets, there is a basic strategy for selecting the proper kld_1 and kld_2 as follows:

1) As the first stage aims to remove the majority of perturbations, we need to adopt a small kld_1 to ensure that PSNR

between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ is low. Thus, most perturbations are preserved in $\boldsymbol{x} - \hat{\boldsymbol{x}}$, thereby leading to a better estimation of $\hat{\boldsymbol{p}}$. We choose PSNR value to fall between 20 and 22, and experiment with some selections of kld_1 to achieve this.

2) For the second stage, where our goal is to obtain poison-free data while maintaining high quality, it is ideal for the PSNR between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ to range between 28 and 30. This range is slightly below the typical PSNR value between $\boldsymbol{x_c}$ and $\boldsymbol{x}$ (usually around 30). Then we experiment with various selections of kld_2 to achieve this.

3) Essentially, the selection process outlined above is dependent on both the dataset and the encoder structure of the VAE. Consequently, for the same dataset but with different UEs methods, we opt for the same kld_1 and kld_2.

4. Experiments

4.1. Experimental Setup

Datasets and models. We choose three commonly used datasets: CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), and a subset of ImageNet (Deng et al., 2009) with the first 100 classes. For CIFAR-10 and CIFAR-100, we maintain the original size of 32×32. Regarding the ImageNet subset, we follow prior research (Huang et al., 2021), and resize the image to 224×224. In our main experiments, we adopt the ResNet-18 (He et al., 2016) model as both the surrogate and target model. To evaluate transferability, we include various classifiers, such as ResNet-50, DenseNet-121 (Huang et al., 2017), MobileNet-V2 (Sandler et al., 2018).

Unlearnable examples. We examine several representative UEs methods with various perturbation bounds. The majority of methods rely on a surrogate model, including NTGA (Yuan & Wu, 2021), EM (Huang et al., 2021), REM (Fu et al., 2022), TAP (Fowl et al., 2021), SEP (Chen et al., 2023), and employ the ℓ_∞ bound. On the other hand, surrogate-free methods such as LSP (Yu et al., 2022a) and AR (Sandoval-Segura et al., 2022) utilize the ℓ_2 bound. Additionally, OPS (Wu et al., 2023) utilizes the ℓ_0 bound. The diversity of these attacking methods can validate the generalization capacity of our proposed purification framework.

Competing defensive methods. We include two training-time defenses: adversarial training (AT) with $\epsilon = 8/255$ (Wen et al., 2023) and adversarial augmentations (AA) (Qin et al., 2023b). Among the pre-training methods, we include ISS (Liu et al., 2023), consisting of bit depth reduction (BDR), Grayscale, and JPEG, as well as AVATAR (Dolatabadi et al., 2023) (denoted as AVA.), which employs a diffusion model trained on the clean CIFAR-10 dataset to purify. We also include LFU (Sandoval-Segura et al., 2023), a hybrid defense that utilizes orthogonal projection to learn perturbations. It also requires strong augmentations during training due to incomplete purifications. For a fair comparison, we choose to report the test accuracy from the last epoch. More details are in the Appendix E.2.

Model Training. To ensure consistent training procedures for the classifier, we have formalized the standard training approach. For CIFAR-10, we use 60 epochs, while for CIFAR-100 and the ImageNet, 100 epochs are allowed. In all experiments, we use SGD optimizer with an initial learning rate of 0.1 and the CosineAnnealingLR scheduler, keeping a consistent batch size of 128. For D-VAE training on unlearnable CIFAR-10, we use a KLD target of 1.0 in the first stage and 3.0 in the second stage, with only a single $\times 0.5$ downsampling to preserve image quality. For the CIFAR-100, we maintain the same hyperparameters as CIFAR-10, except for setting kld_2 to 4.5. For ImageNet-subset, which has higher-resolution images, we employ more substantial downsampling ($\times 0.125$) in the first stage and set a KLD target of 1.5, while the second stage remains the same as with CIFAR. When comparing the unlearnable input and the reconstructed output, these hyperparameters yield PSNRs of around 28 for CIFAR and 30 for ImageNet. In Appendix J, we showcase that our method is tolerant to the selection of kld_1 and kld_2.

4.2. Validate the Effectiveness of the Disentanglement

UEs can be analyzed from the standpoint of shortcuts (Yu et al., 2022a). It has been empirically shown that models trained on the unlearnable training data have a tendency to memorize the perturbations, and attain high accuracy when testing on data that has same perturbations (Liu et al., 2023).

In this section, we aim to illustrate that the disentangled perturbations remain effective as potent attacks and can be regarded as equivalent to the original unlearnable dataset $\mathcal{P}$. Initially, we look into the amplitude of the perturbations in terms of ℓ_2-norm. From Table 10 in the Appendix C, the amplitude of groundtruth p is around 1.0 for LSP and AR, and about 1.5 for others. The generated $\hat{p}$ has an amplitude of about 1.8 for OPS and around 0.7 to 1.0 for others. Notably, the amplitude of $\hat{p}$ is comparable to that of p, with $\hat{p}$ being slightly smaller than p except for OPS. The visual results of the normalized perturbations can be seen in Figure 1, and

Table 1. Testing accuracy (%) of models trained on reconstructed unlearnable dataset $\widehat{\mathcal{P}}$.

Datasets	Test Set	EM	REM	NTGA	LSP	AR	OPS
CIFAR-10	$\mathcal{T}$	9.7	19.8	29.2	15.1	13.09	18.5
	$\mathcal{D}$	9.6	19.5	28.6	15.3	12.9	18.7
	$\mathcal{P}$	91.3	99.9	99.9	99.9	100.0	99.7
CIFAR-100	$\mathcal{T}$	1.4	6.4	-	4.2	1.6	11.2
	$\mathcal{D}$	1.3	7.6	-	4.0	1.6	10.7
	$\mathcal{P}$	98.8	96.4	-	99.1	100.0	99.5

we observe the visual similarity between $\hat{p}$ and p, especially for LSP and OPS. More details and visual results are in the Appendix C and Appendix F, respectively.

Subsequently, we construct a new unlearnable dataset denoted as $\widehat{\mathcal{P}}$ by incorporating the disentangled perturbations $\hat{p}$ into the clean training set $\mathcal{T}$. We proceed to train a model using $\widehat{\mathcal{P}}$, and subsequently evaluate its performance on three distinct sets: the clean training set $\mathcal{T}$, the clean testing set $\mathcal{D}$, and the original unlearnable dataset $\mathcal{P}$. From the results in Table 1, it becomes apparent that the reconstructed dataset continues to significantly degrade the accuracy on clean data. In fact, compared to the attacking performance of $\mathcal{P}$ in Table 2 and Table 3, $\widehat{\mathcal{P}}$ even manages to achieve an even superior attacking performance in most instances with less amplitude. During testing on the original unlearnable dataset, the accuracy levels are notably high, often approaching 100%. This outcome serves as an indicator of the effectiveness of the disentanglement process.

4.3. Experimental Results on UEs Purification

CIFAR-10 UEs purification. To evaluate the effectiveness of our purification framework, we conducted initial experiments on CIFAR-10. As shown in Table 2, our method consistently provides comprehensive protection against UEs with varying perturbation bounds and attack methods. In contrast, ISS relies on multiple simple compression techniques and requires adaptive selection of these methods, resulting in subpar defense performance. Notably, when compared to adversarial training, our method achieved an approximately 6% improvement in performance. Even compared with AVATAR, which utilizes a diffusion model trained on the clean CIFAR-10 data, our methods achieve superior performance across all attack methods. Our methods excel, especially on OPS attacks, which often perturb a pixel to its maximum value, creating a robust shortcut that evades most defenses. Our approach can effectively disentangle the majority of these additive perturbations in the first stage. The subsequent subtraction process can significantly mitigate the attacks, resulting in the poison-free data in the second stage. The performance of training different classification models on our purified data is reported in the rightmost column of Table 2. As can be observed, our method indeed restores the learnability of data samples.

Table 2. Clean test accuracy (%) of models trained on the unlearnable CIFAR-10 dataset and with our proposed method Vs. other defenses. Our results on additional classifiers are at the rightmost. RN, DN, and MN denote ResNet, DenseNet, and MobileNet, respectively.

Norm	UEs / Countermeasures	w/o	AT	AA	BDR	Gray	JPEG	AVA.	LFU	Ours	RN-50	DN-121	MN-v2
	Clean (no poison)	**94.57**	85.17	92.27	88.95	92.74	85.47	89.61	86.78	93.29	93.08	93.73	83.61
$\ell_\infty = \frac{8}{255}$	NTGA (Yuan & Wu, 2021)	11.10	83.63	77.92	57.80	65.26	78.97	80.72	82.21	**89.21**	88.96	89.28	78.72
	EM (Huang et al., 2021)	12.26	84.43	67.11	81.91	19.50	85.61	89.54	65.17	**91.42**	91.62	91.64	81.10
	TAP (Fowl et al., 2021)	25.44	83.89	55.84	80.18	21.50	84.99	89.13	53.46	**90.48**	90.50	90.51	81.28
	REM (Fu et al., 2022)	22.43	86.01	64.99	32.36	62.35	84.40	86.06	33.81	**86.38**	85.91	86.74	79.27
	SEP (Chen et al., 2023)	6.63	83.48	61.07	81.21	8.47	84.97	89.56	74.14	**90.74**	90.86	90.76	80.98
$\ell_2 = 1.0$	LSP (Yu et al., 2022a)	13.14	84.56	80.39	40.25	73.63	79.91	81.15	87.76	**91.20**	90.15	91.10	80.26
	AR (Sandoval-Segura et al., 2022)	12.50	82.01	49.14	29.14	36.18	84.97	89.64	23.51	**91.77**	90.53	90.99	82.26
$\ell_0 = 1$	OPS (Wu et al., 2023)	22.03	9.48	64.02	19.58	19.43	77.33	71.62	86.46	**88.95**	88.10	88.78	81.40
	Mean (except clean)	15.69	74.68	65.06	52.80	38.29	82.64	84.67	63.19	**90.01**	89.58	89.98	80.66

Table 3. Performance on CIFAR-100.

UEs	w/o	AT	AA	ISS	AVA	LFU	Ours
Clean	77.61	59.65	69.09	71.59	61.09	33.12	70.72
EM	12.30	59.07	42.89	61.91	61.09	29.54	**68.79**
TAP	13.44	57.91	35.10	57.33	60.47	29.90	**65.54**
REM	16.80	59.34	50.12	58.13	60.90	31.06	**68.52**
SEP	4.66	57.93	27.77	57.76	59.80	32.03	**64.02**
LSP	2.91	58.93	53.28	53.06	52.17	34.61	**67.73**
AR	2.71	58.77	26.77	56.60	60.33	30.09	**63.73**
OPS	12.56	7.28	36.78	54.45	44.24	30.40	**65.10**
Mean	9.34	51.32	38.96	57.03	57.00	31.09	**66.20**

Table 4. Performance on 100-class ImageNet-subset.

UEs	w/o	AT	AA	ISS	Ours
Clean	**80.52**	55.94	71.56	76.92	76.78
EM	1.08	56.74	3.82	72.44	**74.80**
TAP	12.56	55.36	71.38	73.24	**76.56**
REM	2.54	59.34	20.92	58.13	**72.56**
LSP	2.50	58.93	46.58	53.06	**76.06**
Mean	4.67	57.59	35.68	64.21	**75.00**

Table 5. Performance on unlearnable CIFAR-10 with larger bounds: $\ell_\infty = \frac{16}{255}$ and $\ell_2 = 4.0$.

UEs	w/o	AT	AA	ISS	AVA	LFU	Ours
EM	10.09	84.02	49.23	83.62	85.61	78.78	**91.06**
TAP	18.45	83.46	52.92	84.98	89.43	22.23	**90.55**
REM	23.22	35.41	50.92	75.50	52.26	83.10	**79.18**
SEP	12.05	83.98	56.71	85.00	88.96	70.49	**90.93**
LSP	15.45	79.10	59.10	41.41	41.70	44.48	**86.43**
Mean	15.85	73.19	53.77	74.10	71.59	59.81	**87.63**

CIFAR-100/ImageNet-subset UEs purification. We then expand our experiments to include CIFAR-100 and a 100-class ImageNet subset. Due to the resource-intensive nature of the experiments, we focused on four representative attack methods for the ImageNet subset. Note that for ISS, we report the best accuracy among three compressions. The results, as presented in Table 3 and Table 4, re-confirm the overall effectiveness of our purification framework.

Experiments on larger perturbations. In our additional experiments, we introduced UEs with larger perturbation bounds. The outcomes on CIFAR-10 are outlined in Table 5. It is worth noting that our method exhibits a high degree of consistency, with almost no performance degradation on EM, TAP, and SEP, and only a slight decrease on REM and LSP. However, it proves to be a challenging scenario for the competing methods to effectively address.

Comparison of existing defenses. We offer a comparison of existing defenses and our approach. As shown in Table 6, our method belongs to the pre-training purification, and requires no external clean data. It consistently outperforms across all UEs and datasets. Furthermore, among all competing defenses, LFU is the only one capable of learning and disentangling class-wise linear perturbations, applicable to LSP and OPS. However, due to incomplete purification, LFU also adopts strong augmentations and CutMix during training. In contrast, our method effectively disentangles perturbations in almost all UEs, except for adversarial poisoning including TAP and SEP, detailed in Proposition 3.5 and Section 4.2. More comparison is in the Appendix G,H.

Table 6. Comparison of existing defenses and our method. Performance drop is on CIFAR-10 dataset compared to clean one.

Characteristics	AT	AA	ISS	AVA.	LFU	Ours
Pre-training purification	✗	✗	✓	✓	✓	✓
Training-phase interventions	✓	✓	✗	✗	✓	✗
No external clean data	✓	✓	✓	✗	✓	✓
Consistence on various UEs	✗	✗	✗	✓	✗	✓
UEs types that can be disentangled	0/8	0/8	0/8	0/8	2/8	**6/8**
Mean performance drop (%) ↓	19.89	29.51	11.93	9.90	31.38	**4.56**

Table 7. Ablation study on the two-stage purification framework. s1/s2 denote the 1st and 2nd stage. i1/i2/i3 denote the 1st, 2nd and 3th inference. ⑤ is a method where, after s1, we execute an operation same to i3, employing the D-VAE trained in s1.

Method	NTGA	EM	TAP	REM	SEP	LSP	AR	OPS	Mean
①w/o s1	78.62	**91.85**	**90.97**	82.06	90.76	66.76	91.39	51.71	80.52
②w/o i2 in s2	87.44	91.18	90.70	85.21	**90.79**	90.63	91.31	84.92	89.02
③w/o s2	12.78	78.96	21.12	25.44	4.83	93.47	11.49	41.57	36.21
④w/o i3	13.87	80.77	23.02	23.84	5.29	93.58	14.23	66.39	40.12
⑤	80.98	83.37	84.14	83.48	83.32	83.91	84.22	84.06	83.44
Ours	**89.21**	91.42	90.48	**86.38**	90.74	**91.20**	**91.77**	**88.95**	**90.01**

4.4. Ablation Study

In this section, we conduct an ablation study on our two-stage purification framework. As shown in Table 7, ① shows that the subtraction process in s1 plays a critical role in mitigating certain attacks, including NTGA, LSP, and OPS. ② shows that the subtraction (i2) in s2 can further improve the performance. This is particularly evident for LSP, which introduces smooth colorized blocks, and OPS, which perturbs a single pixel to a maximum value, making them challeng-

Table 8. Performance of detecting UEs or increasing UEs with various poisoning ratios on CIFAR-10 dataset.

UEs		Detecting UEs				Increasing UEs	
Attacks	Ratio	Acc.	Recall	Precision	F1-score	Ratio	Test Acc.
EM	0.2	0.918	1.0	0.709	0.830	0.01	0.1009
LSP		0.777	1.0	0.472	0.641		0.1558
EM	0.4	0.939	1.0	0.869	0.930	0.02	0.1011
LSP		0.905	1.0	0.807	0.893		0.1633
EM	0.6	0.961	1.0	0.938	0.968	0.04	0.1229
LSP		0.941	0.999	0.912	0.954		0.1405
EM	0.8	0.982	1.0	0.978	0.989	0.08	0.1001
LSP		0.973	1.0	0.968	0.984		0.1763

Table 9. Clean testing accuracy (%) of models trained on the unlearnable CIFAR-10 dataset with different poisoning ratios.

Ratio	Counter	EM	TAP	REM	SEP	LSP	AR	OPS
0.2	JPEG	85.03	85.1	84.64	85.34	85.22	85.31	85.12
	Ours	**93.50**	**90.55**	**92.24**	**90.86**	**93.20**	**92.77**	**93.15**
0.4	JPEG	85.31	85.60	84.90	85.22	85.34	85.29	84.89
	Ours	**93.03**	**90.78**	**92.51**	**90.63**	**92.85**	**91.83**	**93.29**
0.6	JPEG	85.40	84.92	84.62	85.06	84.26	85.33	84.43
	Ours	**93.02**	**90.93**	**92.23**	**91.04**	**92.16**	**91.41**	**92.13**
0.8	JPEG	85.31	85.34	84.97	85.06	83.02	84.87	83.01
	Ours	**92.26**	**91.10**	**90.86**	**91.79**	**92.16**	**91.70**	**92.16**

ing to remove when subjected to a moderate KLD target. ③ indicate that the reduction operation in s1 can partially mitigate the effects of poisoning perturbations, particularly demonstrating effectiveness against LSP. Since LSP and OPS adopt class-wise perturbations, the estimation of $\hat{p}$ is more accurate and complete. However, for other UEs, despite the reduction operation, residual perturbations in the output can still pose a threat. ④ reveal that an additional reduction operation in s2 leads to further elimination of perturbations and enhances performance compared to ③. In contrast to ③, ⑤ indicate that the output of D-VAE contains fewer poisoning perturbations. However, this reduction in perturbations comes at a cost: the small $\text{kld}_{\text{target}}$ in s1 results in outputs with lower reconstruction quality and a greater loss of useful information. In contrast, the outcomes of experiment ② are superior, as it adopts a larger $\text{kld}_{\text{target}}$.

5. Discussion

5.1. Partial Poisoning and UEs Detection

In practical scenarios, it is often the case that only a fraction of the training data can be contaminated. Therefore, in line with previous research (Liu et al., 2023), we evaluate these partial poisoning scenarios by introducing UEs to a specific portion of the training data and subsequently combining it with the remaining clean data for training the target model. We conduct experiments on CIFAR-10 dataset.

When examining the first stage as outlined in Algorithm 1, we observe that even the perturbations learned for the clean samples can potentially serve as poisoning attacks. This could be caused by the constrained representation capacity of the class-wise embedding. In essence, building upon this discovery, we have the capability to create a new unlearnable dataset denoted as $\widehat{\mathcal{P}}^0$, where each sample is formed as $x^0 + \hat{p}^0$. Models trained on $\widehat{\mathcal{P}}^0$ tend to achieve high prediction accuracy on the unlearnable samples but perform notably worse on the clean ones. Consequently, we can employ this metric as a means to detect the presence of unlearnable data, and the detection performance is outlined in Table 8. Notably, our detection method attains high accuracy, with an

almost 100% recall rate. Subsequently, to address the issue of partial poisoning in datasets, we can adopt a detection-purification approach. The performances of models trained on the purified data are presented in Table 9.

5.2. Increasing the Amounts of UEs

In this section, we investigate whether our proposed disentanglement approach can help increase the amount of UEs once the attacker acquires additional clean data. We conduct experiments on CIFAR-10 dataset by generating UEs, denoted as $\mathcal{P}_{(0)}$, using a small ratio of the dataset, while leaving the remaining clean data $\mathcal{T}_{(1)}$ untouched. Subsequently, after training the D-VAE on $\mathcal{P}_{(0)}$, we conduct inference on the $\mathcal{T}_{(1)}$. The addition of $\hat{p}_{(1)}$ to the clean data in $\mathcal{T}_{(1)}$ results in a unlearnable dataset $\mathcal{P}_{(1)}$. By combining $\mathcal{P}_{(0)}$ and $\mathcal{P}_{(1)}$ to create $\mathcal{P}$, we proceed to train a classifier. The accuracy on the clean test set $\mathcal{D}$ are reported in Table 8. It is evident that training D-VAE with just 1% UEs is adequate for generating additional UEs. More results are in the Appendix D.

6. Conclusion

In this paper, we initially demonstrate that rate-constrained VAEs exhibit a natural preference for removing poisoning perturbations in unlearnable examples (UEs) by constraining the KL divergence in the latent space. We further provide a theoretical explanation for this behavior. Additionally, our investigations reveal that perturbations in most UEs have a lower class-conditional entropy, and can be disentangled by learnable class-wise embeddings and an auxiliary decoder. Building on these insights, we introduce the D-VAE, capable of disentangling the perturbations, and propose a two-stage purification framework that offers a consistent defense against UEs. Extensive experiments show the remarkable performance of our method across CIFAR-10, CIFAR-100, and ImageNet-subset, with various UEs and varying perturbation levels, i.e., only around 4% drop on ImageNet-subset compared to models trained on clean data. We plan to extend our work to purify UEs that target unsupervised learning scenarios in our future work.

Acknowledgements

Impact Statement

In summary, our paper presents a effective defense strategy against unlearnable examples (UEs), which aim to undermine the overall performance on validation and test datasets by introducing imperceptible perturbations to training examples with accurate labels. UEs are viewed as a promising avenue for data protection, particularly to thwart unauthorized use of data that may contain proprietary or sensitive information. However, these protective methods pose challenges to data exploiters who may interpret them as potential threats to a company's commercial interests. Consequently, our method can be employed for both positive usage, such as neutralizing malicious data within a training set, and negative purpose, including thwarting attempts at preserving data privacy. Our proposed method not only serves as a powerful defense mechanism but also holds the potential to be a benchmark for evaluating existing attack methods. We believe that our paper contributes to raising awareness about the vulnerability of current data protection techniques employing UEs. This, in turn, should stimulate further research towards developing more reliable and trustworthy data protection techniques.

References

Barreno, M., Nelson, B., Sears, R., Joseph, A. D., and Tygar, J. D. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pp. 16–25, 2006.

Barreno, M., Nelson, B., Joseph, A. D., and Tygar, J. D. The security of machine learning. *Machine Learning*, 81: 121–148, 2010.

Biggio, B., Nelson, B., and Laskov, P. Poisoning attacks against support vector machines. In *Proc. Int'l Conf. Machine Learning*, pp. 1467–1474, 2012.

Bozkurt, A., Esmaeili, B., Tristan, J.-B., Brooks, D., Dy, J., and van de Meent, J.-W. Rate-regularization and generalization in variational autoencoders. In *International Conference on Artificial Intelligence and Statistics*, pp. 3880–3888. PMLR, 2021.

Chen, S., Yuan, G., Cheng, X., Gong, Y., Qin, M., Wang, Y., and Huang, X. Self-ensemble protection: Training check-points are good data protectors. In *Proc. Int'l Conf. Learning Representations*, 2023.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 248–255. Ieee, 2009.

Dolatabadi, H. M., Erfani, S., and Leckie, C. The devil's advocate: Shattering the illusion of unexploitable data using diffusion models. *arXiv preprint arXiv:2303.08500*, 2023.

Feng, J., Cai, Q.-Z., and Zhou, Z.-H. Learning to confuse: generating training time adversarial data with auto-encoder. *Proc. Annual Conf. Neural Information Processing Systems*, 32, 2019.

Fowl, L., Goldblum, M., Chiang, P.-y., Geiping, J., Czaja, W., and Goldstein, T. Adversarial examples make strong poisons. *Proc. Annual Conf. Neural Information Processing Systems*, 34:30339–30351, 2021.

Fu, S., He, F., Liu, Y., Shen, L., and Tao, D. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *Proc. Int'l Conf. Learning Representations*, 2022.

Goldblum, M., Tsipras, D., Xie, C., Chen, X., Schwarzschild, A., Song, D., Madry, A., Li, B., and Goldstein, T. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

Guo, C., Rana, M., Cisse, M., and van der Maaten, L. Countering adversarial images using input transformations. In *Proc. Int'l Conf. Learning Representations*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Proc. Annual Conf. Neural Information Processing Systems*, pp. 6840–6851, 2020.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017.

Huang, H., Ma, X., Erfani, S. M., Bailey, J., and Wang, Y. Unlearnable examples: Making personal data unexploitable. In *Proc. Int'l Conf. Learning Representations*, 2021.

Jiang, W., Diao, Y., Wang, H., Sun, J., Wang, M., and Hong, R. Unlearnable examples give a false sense of security: Piercing through unexploitable data with learnable examples. *arXiv preprint arXiv:2305.09241*, 2023.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *Proc. Int'l Conf. Learning Representations*, 2014.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *Proc. Int'l Conf. Machine Learning*, pp. 1885–1894. PMLR, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lin, X., Yu, Y., Xia, S., Jiang, J., Wang, H., Yu, Z., Liu, Y., Fu, Y., Wang, S., Tang, W., et al. Safeguarding medical image segmentation datasets against unauthorized training via contour-and texture-aware perturbations. *arXiv preprint arXiv:2403.14250*, 2024.

Liu, Z., Zhao, Z., and Larson, M. Image shortcut squeezing: Countering perturbative availability poisons with compression. *Proc. Int'l Conf. Machine Learning*, 2023.

Lu, Y., Kamath, G., and Yu, Y. Exploring the limits of model-targeted indiscriminate data poisoning attacks. In *Proc. Int'l Conf. Machine Learning*, 2023.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *Proc. Int'l Conf. Learning Representations*, 2018.

Qin, T., Gao, X., Zhao, J., Ye, K., and Xu, C.-Z. Apbench: A unified benchmark for availability poisoning attacks and defenses. *arXiv preprint arXiv:2308.03258*, 2023a.

Qin, T., Gao, X., Zhao, J., Ye, K., and Xu, C.-Z. Learning the unlearnable: Adversarial augmentations suppress unlearnable example attacks. *arXiv preprint arXiv:2303.15127*, 2023b.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

Sandoval-Segura, P., Singla, V., Geiping, J., Goldblum, M., Goldstein, T., and Jacobs, D. Autoregressive perturbations for data poisoning. *Proc. Annual Conf. Neural Information Processing Systems*, 35:27374–27386, 2022.

Sandoval-Segura, P., Singla, V., Geiping, J., Goldblum, M., and Goldstein, T. What can we learn from unlearnable datasets? In *Proc. Annual Conf. Neural Information Processing Systems*, 2023.

Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J. P., and Goldstein, T. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *Proc. Int'l Conf. Machine Learning*, pp. 9389–9398. PMLR, 2021.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *Proc. Int'l Conf. Learning Representations*, 2021.

Tao, L., Feng, L., Yi, J., Huang, S.-J., and Chen, S. Better safe than sorry: Preventing delusive adversaries with adversarial training. *Proc. Annual Conf. Neural Information Processing Systems*, 34:16209–16225, 2021.

Wang, C., Yu, Y., Guo, L., and Wen, B. Benchmarking adversarial robustness of image shadow removal with shadow-adaptive attacks. In *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing*, pp. 13126–13130. IEEE, 2024.

Wang, J., Sun, J., Zhang, P., and Wang, X. Detecting adversarial samples for deep neural networks through mutation testing. *arXiv preprint arXiv:1805.05010*, 2018.

Wen, R., Zhao, Z., Liu, Z., Backes, M., Wang, T., and Zhang, Y. Is adversarial training really a silver bullet for mitigating data poisoning? In *Proc. Int'l Conf. Learning Representations*, 2023.

Wu, S., Chen, S., Xie, C., and Huang, X. One-pixel shortcut: On the learning preference of deep neural networks. In *Proc. Int'l Conf. Learning Representations*, 2023.

Xia, S., Yi, Y., Jiang, X., and Ding, H. Mitigating the curse of dimensionality for certified robustness via dual randomized smoothing. In *Proc. Int'l Conf. Learning Representations*, 2024.

Xiao, H., Biggio, B., Brown, G., Fumera, G., Eckert, C., and Roli, F. Is feature selection secure against training data poisoning? In *Proc. Int'l Conf. Machine Learning*, pp. 1689–1698. PMLR, 2015.

Yu, D., Zhang, H., Chen, W., Yin, J., and Liu, T.-Y. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2367–2376, 2022a.

Yu, Y., Yang, W., Tan, Y.-P., and Kot, A. C. Towards robust rain removal against adversarial attacks: A comprehensive benchmark analysis and beyond. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 6013–6022, 2022b.

Yu, Y., Wang, Y., Yang, W., Lu, S., Tan, Y.-P., and Kot, A. C. Backdoor attacks against deep image compression via adaptive frequency trigger. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp. 12250–12259, 2023.

Yuan, C.-H. and Wu, S.-H. Neural tangent generalization attacks. In *Proc. Int'l Conf. Machine Learning*, pp. 12230–12240. PMLR, 2021.

Zha, D., Bhat, Z. P., Lai, K.-H., Yang, F., and Hu, X. Data-centric ai: Perspectives and challenges. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 945–948. SIAM, 2023.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *Proc. Int'l Conf. Learning Representations*, 2018.

Zhao, B. and Lao, Y. Clpa: Clean-label poisoning availability attacks using generative adversarial nets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 9162–9170, 2022.

Zhao, M., An, B., Gao, W., and Zhang, T. Efficient label contamination attacks against black-box learning models. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 3945–3951, 2017.

A. Proofs

In this section, we provide the proofs of our theoretical results in Section 3.3.

A.1. Proof of Proposition 3.1

Consider the following binary classification problem with regards to the features extracted from the data $v = (v_c, v_s^t)$ consisting of a predictive feature x_c of a Gaussian mixture $\mathcal{G}_c$ and a non-predictive feature v_s^t which follows:

$$y \overset{u.a.r}{\sim} \{0, 1\}, \; v_c \sim \mathcal{N}(\boldsymbol{\mu}_c^y, \boldsymbol{\Sigma}_c), \; v_s^t \sim \mathcal{N}(\boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t), \; v_c \perp\!\!\!\perp v_s^t, \quad \Pr(y = 0) = \Pr(y = 1). \tag{15}$$

Proposition 3.1 (restated) *For the data $v = (v_c, v_s^t)$ following the distribution (15), the optimal separating hyperplane using a Bayes classifier is formulated by:*

$$\boldsymbol{w}_c^\top (\boldsymbol{v}_c^* - \frac{\boldsymbol{\mu}_c^0 + \boldsymbol{\mu}_c^1}{2}) = 0, \quad \text{where } \boldsymbol{w}_c = \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\mu}_c^0 - \boldsymbol{\mu}_c^1). \tag{16}$$

Proof. Given $v = (v_c, v_s^t)$ following the distribution (15), the optimal decision rule is the maximum a-posteriori probability rule for a Bayes classifier:

$$\begin{aligned}
i^*(\boldsymbol{v}) &= \arg\max_i \Pr(y = i | \boldsymbol{v}) \\
&= \arg\max_i \left[\Pr(y = i) \Pr(\boldsymbol{v} | y = i) \right] \\
&= \arg\max_i \left[\ln \Pr(\boldsymbol{v} | y = i) \right] \\
&= \arg\max_i \left[\ln \Pr(\boldsymbol{v}_c | y = i) + \ln \Pr(\boldsymbol{v}_s^t | y = i) \right] \\
&= \arg\max_i \left[\ln \Pr(\boldsymbol{v}_c | y = i) \right] \\
&= \arg\max_i \left[\ln \left[(2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\boldsymbol{v}_c - \boldsymbol{\mu}_c^i)^\top \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{v}_c - \boldsymbol{\mu}_c^i)) \right] \right] \\
&= \arg\min_i \left[(\boldsymbol{v}_c - \boldsymbol{\mu}_c^i)^\top \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{v}_c - \boldsymbol{\mu}_c^i) \right] \\
&= \arg\min_i \left[\boldsymbol{v}_c^\top \boldsymbol{\Sigma}_c^{-1} \boldsymbol{v}_c - 2\boldsymbol{\mu}_c^{i\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{v}_c + \boldsymbol{\mu}_c^{i\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{\mu}_c^i \right] \\
&= \arg\max_i \left[\boldsymbol{\mu}_c^{i\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{v}_c - \frac{1}{2} \boldsymbol{\mu}_c^{i\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{\mu}_c^i \right]
\end{aligned} \tag{17}$$

where D is the dimensions. Thus, the hyperplane is formulated by:

$$\begin{aligned}
\boldsymbol{\mu}_c^{0\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{v}_c - \frac{1}{2} \boldsymbol{\mu}_c^{0\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{\mu}_c^0 &= \boldsymbol{\mu}_c^{1\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{v}_c - \frac{1}{2} \boldsymbol{\mu}_c^{1\top} \boldsymbol{\Sigma}_c^{-1} \boldsymbol{\mu}_c^1 \\
\iff \boldsymbol{w}_c^\top (\boldsymbol{v}_c^* - \frac{\boldsymbol{\mu}_c^0 + \boldsymbol{\mu}_c^1}{2}) &= 0, \quad \text{where } \boldsymbol{w}_c = \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\mu}_c^0 - \boldsymbol{\mu}_c^1).
\end{aligned} \tag{18}$$

A.2. Proof of Theorem 3.2

We assume that a malicious attacker modifies v_s^t to v_s of the following distributions $\mathcal{G}_s$ to make it predictive for training a Bayes classifier:

$$y \overset{u.a.r}{\sim} \{0, 1\}, \quad v_s \sim \mathcal{N}(\boldsymbol{\mu}_s^y, \boldsymbol{\Sigma}_s), \quad v_c \perp\!\!\!\perp v_s. \tag{19}$$

Theorem 3.2 (restated) *Consider the training data for the Bayes classifier is modified from $v = (v_c, v_s^t)$ in Eq. 15 to $v = (v_c, v_s)$ in Eq. 19, the hyperplane is shifted with a distance given by*

$$d = \frac{\|\boldsymbol{w}_s^\top (\boldsymbol{v}_s - \frac{\boldsymbol{\mu}_s^0 + \boldsymbol{\mu}_s^1}{2})\|_2}{\|\boldsymbol{w}_c\|_2}, \quad \text{where } \boldsymbol{w}_c = \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{\mu}_c^0 - \boldsymbol{\mu}_c^1), \; \boldsymbol{w}_s = \boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_s^0 - \boldsymbol{\mu}_s^1). \tag{20}$$

Proof. After modifying v_s^t to v_s, the learned separating hyperplane on the poisoned distributions $\mathcal{G}_p = (\mathcal{G}_c, \mathcal{G}_s)$ turns to (following Proposition 3.1):

$$
w^\top \left(\begin{bmatrix} v_c^* - \frac{\mu_c^0 + \mu_c^1}{2} \\ v_s - \frac{\mu_s^0 + \mu_s^1}{2} \end{bmatrix} \right) = 0 \iff w_c^\top (v_c^* - \frac{\mu_c^0 + \mu_c^1}{2}) = -w_s^\top (v_s - \frac{\mu_s^0 + \mu_s^1}{2}),
$$

$$
\text{where } w = \begin{bmatrix} \Sigma_c^{-1} & 0 \\ 0 & \Sigma_s^{-1} \end{bmatrix} \begin{bmatrix} \mu_c^0 - \mu_c^1 \\ \mu_s^0 - \mu_s^1 \end{bmatrix} = \begin{bmatrix} w_c \\ w_s \end{bmatrix}, \ w_c = \Sigma_c^{-1}(\mu_c^0 - \mu_c^1).
$$

(21)

Thus, compared to the original hyperplane as stated in Eq. 16, the hyperplane on the poisoned distribution is shifted with a distance d:

$$
d = \frac{\|w_s^\top (v_s - \frac{\mu_s^0 + \mu_s^1}{2})\|_2}{\|w_c\|_2}
$$

(22)

When conducting evaluations on the testing data that follows the same distribution as the clean data $v = (v_c, v_s^t)$, with the term v_s in Eq. 22 replaced by v_s^t, the shifted distance d is given by

$$
d = \frac{\|w_s^\top (v_s^t - \frac{\mu_s^0 + \mu_s^1}{2})\|_2}{\|w_c\|_2} \propto \frac{\|w_s\|_2}{\|w_c\|_2}.
$$

(23)

And it leads to a greater prediction error if $\|w_s\|_2 \gg \|w_c\|_2$.

A.3. Proof of Theorem 3.3

Consider a variable $v = (v_1, \ldots, v_d)$ following a mixture of two Gaussian distributions $\mathcal{G}$:

$$
y \overset{u \cdot a \cdot r}{\sim} \{0, 1\}, \quad v \sim \mathcal{N}(\mu^y, \Sigma), \quad x_i \perp\!\!\!\perp x_j, \quad \Pr(y = 0) = \Pr(y = 1),
$$

$$
v_i \sim \mathcal{N}(\mu_i^y, \sigma_i), \quad p_{v_i}(v) = \frac{\mathcal{N}(v; \mu_i^0, \sigma_i) + \mathcal{N}(v; \mu_i^1, \sigma_i)}{2}.
$$

(24)

Each dimensional feature v_i is also modeled as a Gaussian mixture. To start, we normalize each feature through a linear operation to achieve a distribution with zero mean and unit variance. Firstly, we calculate the mean and standard deviation of v_i:

$$
\hat{\mu}_i = \mathbb{E}_{v_i}[v_i] = \frac{\mu_i^0 + \mu_i^1}{2}, \quad \text{Var}[v_i] = \mathbb{E}_{v_i}[(v_i)^2] - \mathbb{E}_{v_i}[v_i]^2 = \sigma_i^2 + (\frac{\mu_i^0 - \mu_i^1}{2})^2.
$$

(25)

Thus, the linear operation and the modified density function can be expressed as follows:

$$
z_i = \frac{v_i - \hat{\mu}_i}{\sqrt{(\sigma_i)^2 + (\delta_i)^2}}, \ p_{z_i}(v) = \frac{p_0(v) + p_1(v)}{2}, \ p_0(v) = \mathcal{N}(v; -\hat{\delta}_i, \hat{\sigma}_i), \ p_1(v) = \mathcal{N}(v; \hat{\delta}_i, \hat{\sigma}_i)
$$

$$
\text{where } \hat{\mu}_i = \frac{\mu_i^0 + \mu_i^1}{2}, \ \delta_i = |\frac{\mu_i^0 - \mu_i^1}{2}|, \ \hat{\delta}_i = \delta_i / \sqrt{(\sigma_i)^2 + (\delta_i)^2}, \ \hat{\sigma}_i = \sigma_i / \sqrt{(\sigma_i)^2 + (\delta_i)^2}.
$$

(26)

Theorem 3.3 (restated) *Denote $r = \frac{\delta_i}{\sigma_i} > 0$, the Kullback–Leibler divergence between $p_{z_i}(v)$ and a standard normal distribution $\mathcal{N}(v; 0, 1)$ is tightly bounded by*

$$
\frac{1}{2} \ln(1 + r^2) - \ln 2 \leq D_{KL}(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)) \leq \frac{1}{2} \ln(1 + r^2).
$$

(27)

and observes the following property

$$
\uparrow r_i \implies \uparrow S(r_i) = D_{KL}(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)).
$$

(28)

Proof. We estimate the Kullback–Leibler divergence between $p_{z_i}(v)$ and $\mathcal{N}(v; 0, 1)$:

$$
\begin{aligned}
D_{KL}(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)) &= \int_{-\infty}^{\infty} p_{z_i}(v) \ln \frac{p_{z_i}(v)}{\mathcal{N}(v; 0, 1)} dv \\
&= -H(p_{z_i}(v)) + H((p_{z_i}(v), \mathcal{N}(v; 0, 1))) \\
&= -H(p_{z_i}(v)) + \frac{1}{2}(1 + \ln 2\pi),
\end{aligned}
$$

(29)

As the funcion $\mathcal{N}(v; 0, 1)))$ is given by

$$\mathcal{N}(v; 0, 1))) = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}}, \tag{30}$$

then the term $H((p_{z_i}(v), \mathcal{N}(v; 0, 1)))$ can be formulated as

$$
\begin{aligned}
H((p_{z_i}(v), \mathcal{N}(v; 0, 1))) &= \int_{-\infty}^{\infty} p_{z_i}(v) \ln \frac{1}{\mathcal{N}(v; 0, 1)} dv \\
&= \int_{-\infty}^{\infty} p_{z_i}(v) \left[\frac{1}{2} \ln 2\pi + \frac{1}{2} v^2 \right] dv \\
&= \frac{1}{2} \ln 2\pi \int_{-\infty}^{\infty} p_{z_i}(v) dv + \frac{1}{2} \int_{-\infty}^{\infty} v^2 p_{z_i}(v) dv \\
&= \frac{1}{2} \ln 2\pi + \frac{1}{2} \int_{-\infty}^{\infty} v^2 \frac{p_0(v) + p_1(v)}{2} dv \\
&= \frac{1}{2} \ln 2\pi + \frac{1}{4} \left[E_{v \sim p_0(v)}[v^2] + E_{v \sim p_1(v)}[v^2] \right] \\
&= \frac{1}{2} \ln 2\pi + \frac{1}{4} \left[E_{p_0(v)}[v]^2 + Var_{p_0(v)}[v] + E_{p_1(v)}[v]^2 + Var_{p_1(v)}[v] \right] \\
&= \frac{1}{2} \ln 2\pi + \frac{1}{2}
\end{aligned}
\tag{31}
$$

As the entropy $H(p)$ is concave in the probability mass function p, a lower bound of $H(p_{z_i})$ is given by:

$$
\begin{aligned}
H(p_{z_i}(v)) &= H(\frac{p_0(v) + p_1(v)}{2}) \\
&\geq \frac{1}{2} H(p_0(v)) + \frac{1}{2} H(p_1(v)) \\
&= \frac{1}{2} H(\mathcal{N}(v; -\frac{r}{\sqrt{1+r^2}}, \frac{1}{\sqrt{1+r^2}})) + \frac{1}{2} H(\mathcal{N}(v; \frac{r}{\sqrt{1+r^2}}, \frac{1}{\sqrt{1+r^2}})) \\
&= \frac{1}{2} \left[\frac{1}{2} (1 + \ln(2\pi(\frac{1}{\sqrt{1+r^2}})^2)) + \frac{1}{2} (1 + \ln(2\pi(\frac{1}{\sqrt{1+r^2}})^2)) \right] \\
&= \frac{1}{2} (1 + \ln 2\pi) - \frac{1}{2} \ln(1 + r^2).
\end{aligned}
\tag{32}
$$

The upper bound of $H(p_{z_i})$ is given by:

$$
\begin{aligned}
H(p_{z_i}) &= -\int_{-\infty}^{\infty} p_{z_i}(v) \ln p_{z_i}(v) dv \\
&= -\int_{-\infty}^{\infty} \frac{p_o(v) + p_1(v)}{2} \ln \frac{p_o(v) + p_1(v)}{2} dv \\
&= -\frac{1}{2} \left[\int_{-\infty}^{\infty} p_o(v) [\ln \frac{p_o(v)}{2} + \ln(1 + \frac{p_1(v)}{p_0(v)})] dv + \int_{-\infty}^{\infty} p_1(v) [\ln \frac{p_1(v)}{2} + \ln(1 + \frac{p_0(v)}{p_1(v)})] dv \right] \\
&\leq -\frac{1}{2} \left[\int_{-\infty}^{\infty} p_o(v) \ln \frac{p_o(v)}{2} dv + \int_{-\infty}^{\infty} p_1(v) \ln \frac{p_1(v)}{2} dv \right] \\
&= \frac{1}{2} \left[H(p_0) + H(p_1) + 2\ln 2 \right] \\
&= \frac{1}{2} (1 + \ln 2\pi) - \frac{1}{2} \ln(1 + r^2) + \ln 2.
\end{aligned}
\tag{33}
$$

Thus, the Kullback–Leibler divergence is bounded by :

$$\frac{1}{2} \ln(1 + r^2) - \ln 2 \leq D_{KL}(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)) \leq \frac{1}{2} \ln(1 + r^2). \tag{34}$$

Since the lower and upper bounds differ by a constant term, and the lower bound increases significantly as r rises, the Kullback–Leibler divergence is asymptotically tightly bounded by:

$$D_{KL}(p_{z_i}(v) \| \mathcal{N}(v; 0, 1)) = \Theta(\ln(1 + r^2)) \tag{35}$$

15

A.4. Proof of Proposition 3.5

Proposition 3.5 (restated) *The conditional entropy of a Gaussian mixture $\boldsymbol{v}_s$ of $\mathcal{G}_s$ in Eq. 19 is given by*

$$H(\boldsymbol{v}_s|y_i) = \frac{D}{2}(1 + \ln(2\pi)) + \frac{1}{2}\ln|\boldsymbol{\Sigma}_s|, \tag{36}$$

where D is the dimensions of the features. If each feature v_s^d is independent, then

$$H(\boldsymbol{v}_s|y_i) = \frac{D}{2}(1 + \ln(2\pi)) + \sum_{d=1}^{D}\ln\sigma_s^d. \tag{37}$$

Proof. For the variable follows a Gaussian distribution:

$$\boldsymbol{v} \sim \mathcal{N}_D(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{38}$$

The derivation of its entropy is given by

$$
\begin{aligned}
H(\boldsymbol{v}) &= -\int p(\boldsymbol{v})\ln p(\boldsymbol{v})d\boldsymbol{v} \\
&= -\mathbb{E}\big[\ln\mathcal{N}_D(\boldsymbol{\mu}, \boldsymbol{\Sigma})\big] \\
&= -\mathbb{E}\Big[\ln\big[(2\pi)^{-\frac{D}{2}}|\boldsymbol{\Sigma}|^{-\frac{1}{2}}\exp(-\frac{1}{2}(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{v}-\boldsymbol{\mu}))\big]\Big] \\
&= \frac{D}{2}\ln(2\pi) + \frac{1}{2}\ln|\boldsymbol{\Sigma}| + \frac{1}{2}\mathbb{E}\big[(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{v}-\boldsymbol{\mu}))\big] \\
&\overset{*}{=} \frac{D}{2}(1 + \ln(2\pi)) + \frac{1}{2}\ln|\boldsymbol{\Sigma}|
\end{aligned}
\tag{39}
$$

Step $*$ is a little trickier. It relies on several properties of the trace operator:

$$
\begin{aligned}
\mathbb{E}\big[(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{v}-\boldsymbol{\mu}))\big] &= \mathbb{E}\Big[\mathrm{tr}\big[(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{v}-\boldsymbol{\mu}))\big]\Big] \\
&= \mathbb{E}\Big[\mathrm{tr}\big[\boldsymbol{\Sigma}^{-1}(\boldsymbol{v}-\boldsymbol{\mu})(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\big]\Big] \\
&= \mathrm{tr}\Big[\boldsymbol{\Sigma}^{-1}\mathbb{E}\big[(\boldsymbol{v}-\boldsymbol{\mu})(\boldsymbol{v}-\boldsymbol{\mu})^{\top}\big]\Big] \\
&= \mathrm{tr}\Big[\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}\Big] \\
&= \mathrm{tr}(\boldsymbol{I}_D) \\
&= \frac{D}{2}
\end{aligned}
\tag{40}
$$

Table 10. Evaluation on the amplitude of the disentangled $\hat{p}$ and groundtruth p.

Datasets	Test Set	EM	REM	NTGA	LSP	AR	OPS
CIFAR-10	$\|p\|_2$	1.53	1.80	2.96	0.99	0.98	1.27
	$\|\hat{p}\|_2$	1.24	0.92	0.71	0.73	0.68	1.77
	$\text{MSE}(p, \hat{p}) / 10^{-4}$	4.4	6.6	12	3.5	5.2	2.3
	$\text{PSNR}(p, \hat{p}) / \text{dB}$	33.6	31.8	29.3	34.7	32.9	36.5
CIFAR-100	$\|p\|_2$	1.34	1.73	-	0.99	0.98	1.34
	$\|\hat{p}\|_2$	0.81	0.69	-	0.69	0.78	1.82
	$\text{MSE}(p, \hat{p}) / 10^{-4}$	4.1	7.1	-	4.1	4.7	2.0
	$\text{PSNR}(p, \hat{p}) / \text{dB}$	33.9	31.5	-	34.0	33.1	37.3

B. Detailed implementation

B.1. KLD Loss

For the implementation of KLD loss in Eq. 3 and Eq. 14, we follows the widely-used version from Kingma & Welling (2014). The detailed loss formulation is given

$$\text{KLD}(z, \mathcal{N}(0, I)) = -\frac{1}{2} \sum_{j=1}^{J} (1 + \log(\sigma_j)^2 - (\mu_j)^2 - (\sigma_j)^2), \tag{41}$$

where $z = \mu + \sigma \odot \epsilon$, and $\epsilon \sim \mathcal{N}(0, I)$.

B.2. D-VAE

In the implementation of D-VAE, the encoder comprises 7 convolutional layers with Batch Normalization, while the decoder for both branches consists of 4 convolutional layers with Instance Normalization. To predict the mean μ and standard deviation σ, we employ one convolutional layer with a kernel size of 1 for each variable.

During the training of D-VAE, we configure the number of training epochs to be 60 for CIFAR-10 and CIFAR-100. However, for ImageNet, which involves significant computational demands, we limit the training epochs to 20. It's important to note that we do not use any transformations on the training data when training D-VAEs. For D-VAE training on unlearnable CIFAR-10/100, we use a KLD target of 1.0 in the first stage and 3.0 in the second stage, with only a single $\times 0.5$ downsampling to preserve image quality. For ImageNet, which has higher-resolution images, we employ more substantial downsampling ($\times 0.125$) in the first stage and set a KLD target of 1.5, while the second stage remains the same as with CIFAR. When comparing the unlearnable input and the reconstructed output, these hyperparameters yield PSNRs of around 28 for CIFAR and 30 for ImageNet.

C. Disentangled perturbations

Given that the defender lacks groundtruth values for the perturbations p, it is not possible to optimize u_y and D_{θ_p} to learn to predict $\hat{p}$ directly by minimizing $\|p - \hat{p}\|_2^2$ during model training. Instead, as the residuals $x - \hat{x}$ contain the majority of the groundtruth p when imposing a low target value on the KLD loss, we propose minimizing $\|(x - \hat{x}) - \hat{p}\|_2^2$. As $\hat{p}$ is generated by $u_y + z$, which has an information bottleneck, it is hard to achieve a perfect reconstruction of p, and $\hat{p}$ is most likely to be a part of p. In Table 10, we offer the ℓ_2-norm of both p and $\hat{p}$, and we can see that the $\hat{p}$ has a smaller amplitude. In Section 4.2, the experiments show that the $\hat{p}$ remains effective as poisoning patterns. Notably, the amplitude of $\hat{p}$ is comparable to that of p, with $\hat{p}$ being slightly smaller than p except for OPS.

The visual results of the normalized perturbations can be seen in Figure 1, and we observe the visually similarity between $\hat{p}$ and p, especially for LSP and OPS. Additionally, since LSP and OPS use class-wise perturbations (i.e., perturbations are identical for each class of images), they exhibit lower class-conditioned entropy compared to other attack methods that employ sample-wise perturbations. This makes the reconstruction of LSP and OPS perturbations much easier.

Table 11. Performance of detecting UEs or increasing UEs with various poison ratios on CIFAR-10.

Task		Detecting UEs				Increasing UEs	
UEs	Ratio	Acc.	Recall	Precision	F1-score	Ratio	Test Acc.
EM		0.918	1.0	0.709	0.830		0.1009
REM	0.2	0.561	1.0	0.312	0.476	0.01	0.2900
LSP		0.777	1.0	0.472	0.641		0.1558
OPS		0.724	0.993	0.420	0.590		0.2059
EM		0.939	1.0	0.869	0.930		0.1011
REM	0.4	0.785	1.0	0.651	0.789	0.02	0.2777
LSP		0.905	1.0	0.807	0.893		0.1633
OPS		0.842	0.991	0.719	0.833		0.2015
EM		0.961	1.0	0.938	0.968		0.1229
REM	0.6	0.909	0.999	0.868	0.930	0.04	0.2319
LSP		0.941	0.999	0.912	0.954		0.1405
OPS		0.910	0.993	0.874	0.930		0.1632
EM		0.982	1.0	0.978	0.989		0.1001
REM	0.8	0.958	0.998	0.951	0.975	0.08	0.2433
LSP		0.973	1.0	0.968	0.984		0.1763
OPS		0.932	0.997	0.924	0.959		0.1701

Table 12. Clean testing accuracy (%) of models trained on the unlearnable CIFAR-10 dataset with different poisoning rate.

UEs	Counter	0.2	0.4	0.6	0.8
EM	JPEG	85.03	85.31	85.40	85.31
	Ours	**93.50**	**93.03**	**93.02**	**92.26**
TAP	JPEG	85.12	85.60	84.92	85.34
	Ours	**90.55**	**90.78**	**90.93**	**91.10**
REM	JPEG	84.64	84.90	84.62	84.97
	Ours	**92.24**	**92.51**	**92.23**	**90.86**
SEP	JPEG	85.34	85.22	85.06	85.06
	Ours	**90.86**	**90.63**	**91.04**	**91.79**
LSP	JPEG	85.22	85.34	84.26	83.02
	Ours	**93.20**	**92.85**	**92.16**	**92.16**
AR	JPEG	85.31	85.29	85.33	84.87
	Ours	**92.77**	**91.83**	**91.41**	**91.70**
OPS	JPEG	85.12	84.89	84.43	83.01
	Ours	**93.15**	**93.29**	**92.13**	**92.16**

D. More results on partial poisoning

In Table 11 and Table 12, we provide additional results on detecting UEs, increasing amounts of UEs, and experimental results on UEs purification in the partial poisoning settings.
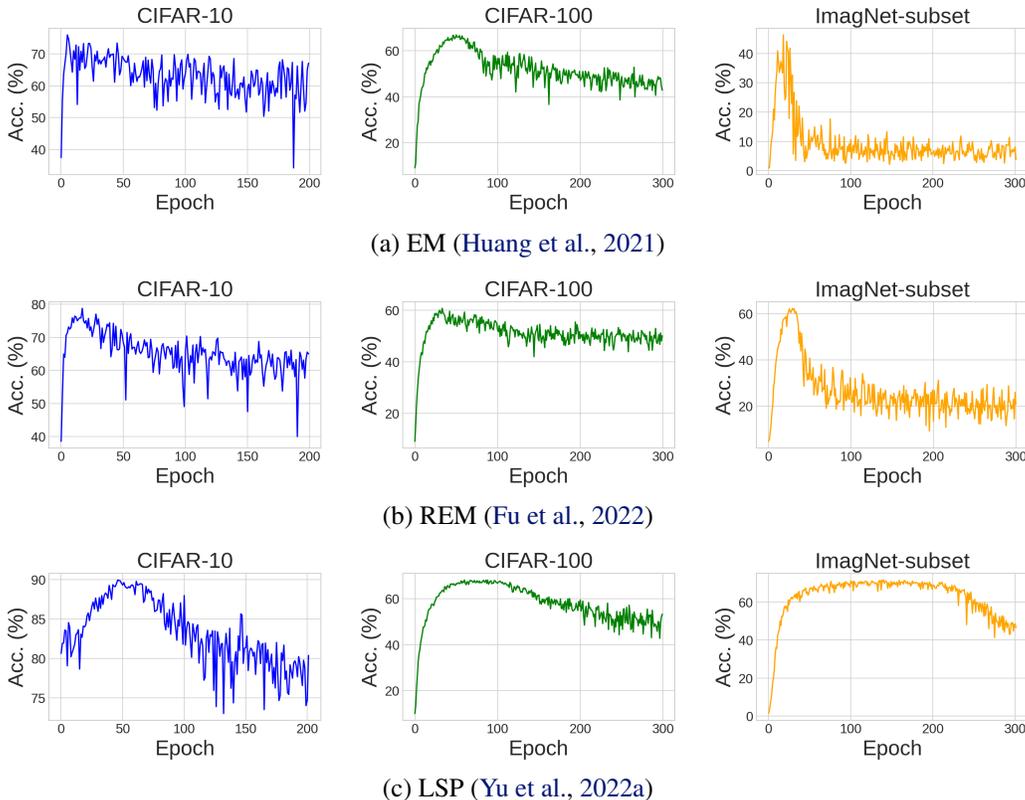
(a) EM (Huang et al., 2021)

(b) REM (Fu et al., 2022)

(c) LSP (Yu et al., 2022a)

Figure 3. Test accuracy (%) for each training epoch when using adversarial augmentation (Qin et al., 2023b)

E. Detailed implementation of the attack methods and competing defenses

As previous papers may have used varying code to generate perturbations and implemented defenses based on different codebases, we have re-implemented the majority of the attack and defensive methods by referencing their original code resources. In cases where the original paper did not provide code, we will specify the sources we used for implementation.

E.1. Attack methods for generating UEs

NTGA. For the implementation of NTGA UEs, we directly download the read-to-use unlearnable dataset from the official source of NTGA (Yuan & Wu, 2021).

EM, TAP, and REM. For the implementation of EM (Huang et al., 2021), TAP (Fowl et al., 2021), and REM (Fu et al., 2022) UEs, we follow the official code of REM (Fu et al., 2022).

SEP. For the implementation of SEP (Chen et al., 2023) UEs, we follow the official code of SEP (Chen et al., 2023).

LSP. For the implementation of LSP (Yu et al., 2022a) UEs, we follow the official code of LSP (Yu et al., 2022a). Particularly, we set the patch size of the colorized blocks to 8 for both CIFAR-10, CIFAR-100, ImageNet-subset.

AR. For the implementation of AR UEs, we directly download the read-to-use unlearnable dataset from the official source of AR (Sandoval-Segura et al., 2022).

OPS. For the implementation of OPS. (Wu et al., 2023) UEs, we follow the official code of OPS. (Wu et al., 2023).

E.2. Competing defenses

Image shortcut squeezing (ISS). For the implementation of ISS (Liu et al., 2023), which consists of bit depth reduction (depth decreased to 2), grayscale (using the official implementation by torchvision.transforms), JPEG compression (quality set to 10), we follow the official code of ISS (Liu et al., 2023). Although most of the reported results align closely with the original paper's findings, we observed that EM and REM UEs generated using the codebase of REM (Fu et al., 2022) display

Table 13. Test acc. (%) of models trained on CIFAR-10 UEs.

Norm	Attacks	w/o	AA	Ours
	Clean	**94.57**	92.66	93.29
$\ell_\infty = 8$	NTGA	11.10	86.35	**89.21**
	EM	12.26	76.00	**91.42**
	TAP	25.44	71.56	**90.48**
	REM	22.43	78.77	**86.38**
	SEP	6.63	71.95	**90.74**
$\ell_2 = 1.0$	LSP	13.14	89.97	**91.20**
	AR	12.50	67.61	**91.77**
$\ell_0 = 1$	OPS	22.03	72.54	**88.95**

Table 14. Test acc. (%) of models trained on CIFAR-100 UEs.

Attacks	w/o	AA	BDR	Gray	JPEG	Ours
Clean	**77.61**	70.22	63.52	71.59	57.85	70.72
EM	12.30	66.84	61.91	48.83	58.08	**68.79**
TAP	13.44	49.36	55.09	9.69	57.33	**65.54**
REM	16.80	60.74	57.51	55.99	58.13	**68.52**
SEP	4.66	37.73	31.95	4.47	57.76	**64.02**
LSP	2.91	68.22	22.13	44.18	53.06	**67.73**
AR	2.71	44.32	29.68	23.09	56.60	**63.73**
OPS	12.56	40.20	11.56	19.33	54.45	**65.10**

Table 15. Test acc. (%) of models trained on ImageNet subset UEs.

Attacks	w/o	AA	BDR	Gray	JPEG	Ours
Clean	**80.52**	73.66	75.84	76.92	72.90	76.78
EM	1.08	46.30	2.78	14.02	72.44	**74.80**
TAP	12.56	72.10	45.74	33.66	73.24	**76.56**
REM	2.54	62.30	57.51	55.99	58.13	**72.56**
LSP	2.50	71.72	22.13	44.18	53.06	**76.06**

a notable robustness to Grayscale, which differs somewhat from the results reported in the original paper. The unreported results for the performance of each compression on the CIFAR-100 and ImageNet datasets are presented in Table E.2.

Adversarial training (AT). For the implementation of adversarial training, we follow the official code of pgd-AT (Madry et al., 2018) with the adversarial perturbation subject to ℓ_∞ bound, and set $\epsilon = \frac{8}{255}$, iterations $T = 10$, and step size $\alpha = \frac{1.6}{255}$.

AVATAR. In our implementation of AVATAR, which employs a diffusion model trained on the clean CIFAR-10 dataset to purify unlearnable samples, we utilized the codebase from a benchmarking paper (Qin et al., 2023a). This choice was made since AVATAR (Dolatabadi et al., 2023) does not offer official implementations.

Adversarial augmentations (AA). In our implementation of AA, we utilized the codebase from the original paper (Qin et al., 2023b). AA comprises two stages. In the first stage, loss-maximizing augmentations are employed for training, with a default number of repeated samples set to K = 5. In the second stage, a lighter augmentation process is applied, with K = 1. In all experiments conducted on CIFAR-10, CIFAR-100, and the 100-class ImageNet subset, we strictly adhere to the same hyperparameters as detailed in the original paper. Nevertheless, we observed that this training-time method can partially restore the test accuracy if we report the highest accuracy achieved among all training epochs. However, it's worth noting that the model may still exhibit a tendency to overfit to the shortcut provided by the unlearnable samples. Consequently, this can lead to a substantial drop in test accuracy during the second stage, which employs lighter augmentations. The test accuracy for each training epoch is depicted in Figure 3. Additionally, we have included the best accuracy for AA in Table E.2. It's notable that our results from the last epoch surpass the performance of AA, showcasing the superiority.
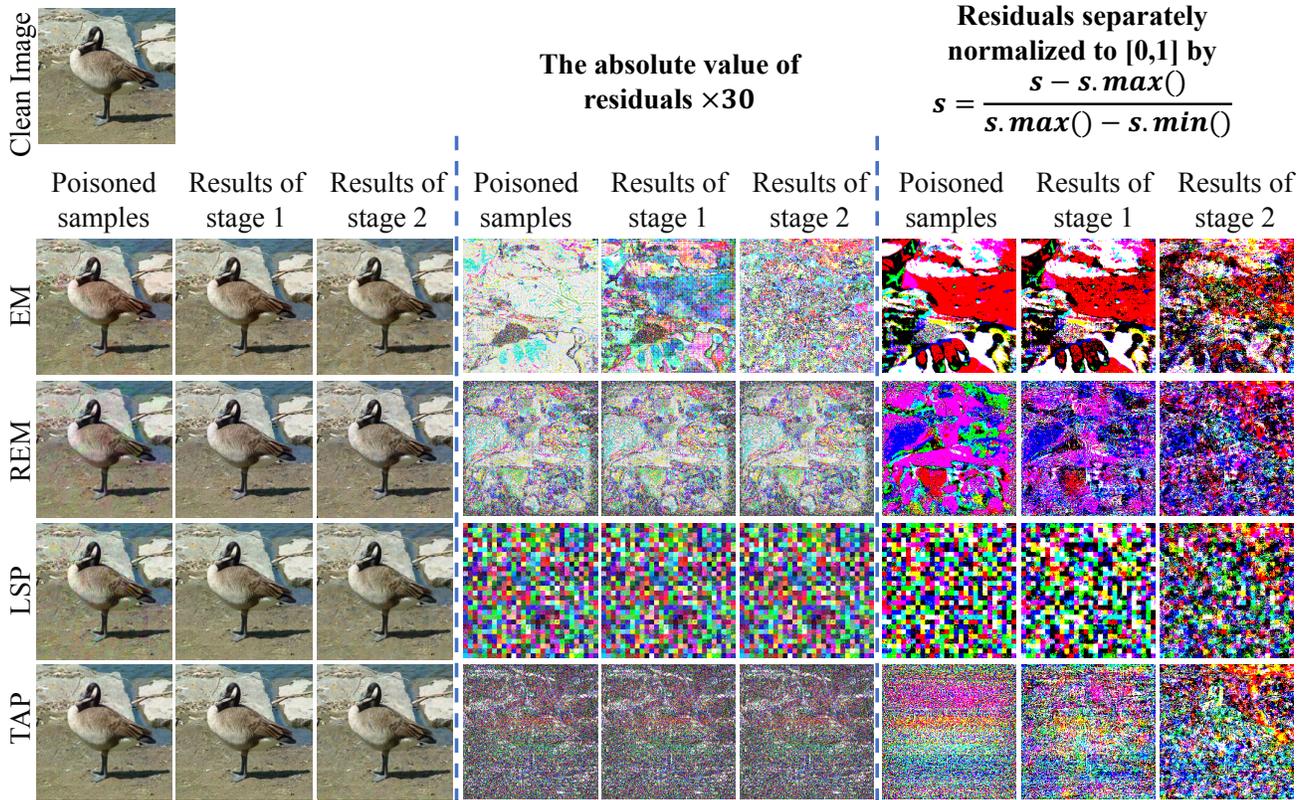
Figure 4. Visual results of images before/after purification. Results of stage 2 denote the final purified results. The image is from ImageNet-subset, and the residuals to the clean images are normalized by two ways.

F. Visual Results

In this section, we present visual results of the purification process on the ImageNet-subset. As depicted in Figure 4, the purification carried out during stage 1 is effective in removing a significant portion of perturbations, particularly for LSP UEs. The remaining perturbations are subsequently eliminated in stage 2, resulting in completely poison-free data.
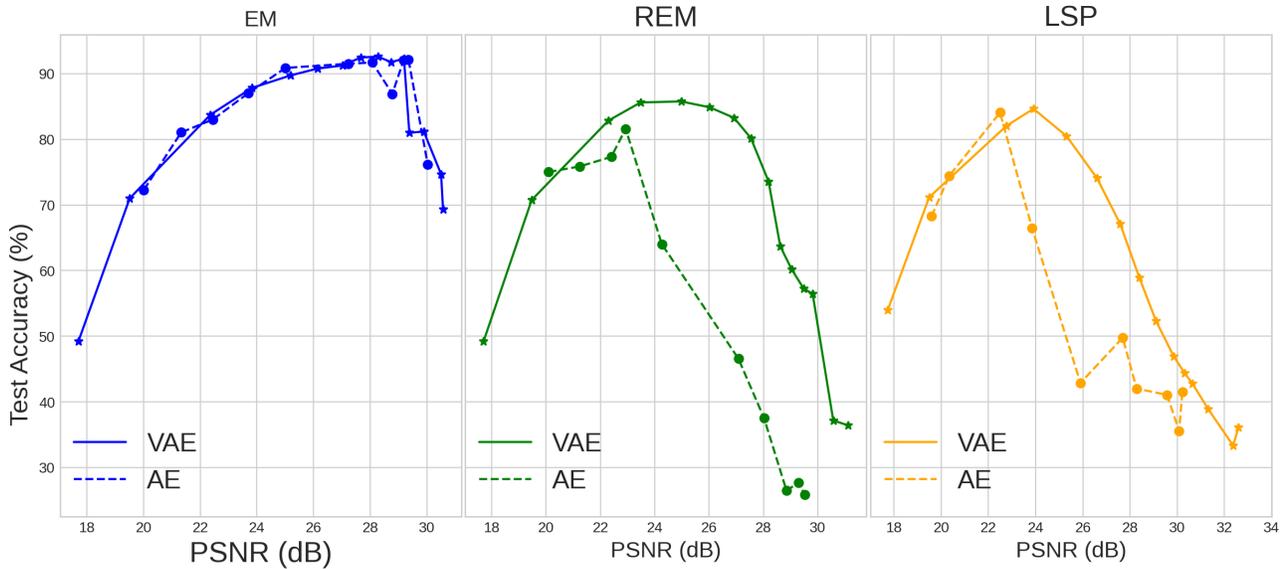
Figure 5. Comparison between VAEs and AEs: PSNR Vs. Test Acc. Specifically, we include EM, REM, and LSP as attack methods here.

G. Comparison with non-variational auto-encoders

In this section, we conduct experiments on purification using non-variational auto-encoders (AEs) with an information bottleneck. To achieve non-variational auto-encoders with different bottleneck levels, we modify the width of the features within the auto-encoder architecture. This results in models with varying parameter numbers. Then, we proceed to train the AE on the unlearnable CIFAR-10 dataset, and test on the clean test dataset with classifiers trained on the purified dataset. As depicted in Figure 5, when considering the similar level of reconstruction quality measured by PSNR, VAEs exhibit a greater capacity to remove perturbations in both the REM and LSP UEs. However, for EM UEs, the outcomes are comparable. These observations align with the theoretical analysis presented in Section 3.3.

Table 16. Computation requirement of the proposed methods.

Component	Train D-VAE for twice	Perform inference on the unlearnable data three times	Train a classifier	Total Time
Our method	23 minutes	less than 2 minutes	16 minutes	41 minutes
Adversarial Training	N.A.	N.A.	229 minutes	229 minutes

Table 17. Results using JPEG with various quality settings. The experiments are on CIFAR-10 dataset.

Defenses /Attacks	JPEG (quality 10) PSNR 22	JPEG (quality 30) PSNR 25	JPEG (quality 50) PSNR 27	JPEG (quality 70) PSNR 28	Ours PSNR 28
NTGA	78.97	66.83	64.28	60.19	**89.21**
EM	85.61	70.48	54.22	42.23	**91.42**
TAP	84.99	84.82	77.98	57.45	**90.48**
REM	84.40	77.73	71.19	63.39	**86.38**
SEP	84.97	87.57	82.25	59.09	**90.74**
LSP	79.91	42.11	33.99	29.19	**91.20**
AR	84.97	89.17	86.11	80.01	**91.77**
OPS	77.33	79.01	68.68	59.81	**88.96**
Mean	78.89	74.71	67.33	56.42	**90.02**

H. Computation and Comparison with JPEG compression

In this section, we present the computation requirement and the compassion with JPEG compression. The Table 16 below presents the training time for D-VAE, the inference time for the unlearnable dataset, and the time to train a classifier using the purified dataset. For comparison, we include the training-time defense **Adversarial Training**. It's important to note that the times are recorded using CIFAR-10 as the dataset, PyTorch as the platform, and a single Nvidia RTX 3090 as the GPU. As can see from the results, the total purification time is approximately one and a half times longer than training a classifier, which is acceptable. Compared to adversarial training, our methods are about 5 times faster. Additionally, our method achieves an average performance around 90%, which is 15% higher than the performance achieved by adversarial training.

We also note a limitation in the JPEG compression approach used in ISS (Liu et al., 2023)—specifically, they set the JPEG quality to 10 to purify unlearnable samples, resulting in significant image degradation. In the Table 17, we present results using JPEG with various quality settings. Notably, our proposed methods consistently outperform JPEG compression when applied at a similar level of image corruption. Therefore, in the presence of larger perturbation bounds, JPEG may exhibit suboptimal performance. Moreover, our method excels in eliminating the majority of perturbations in the first stage, rendering it more robust to larger perturbation bounds. Table 5 5 of the main paper illustrates that when confronted with LSP attacks with larger bounds, our method demonstrates significantly smaller performance degradation compared to JPEG (with quality 10), *e.g.,* 86.13 Vs. 41.41 in terms of test accuracy.
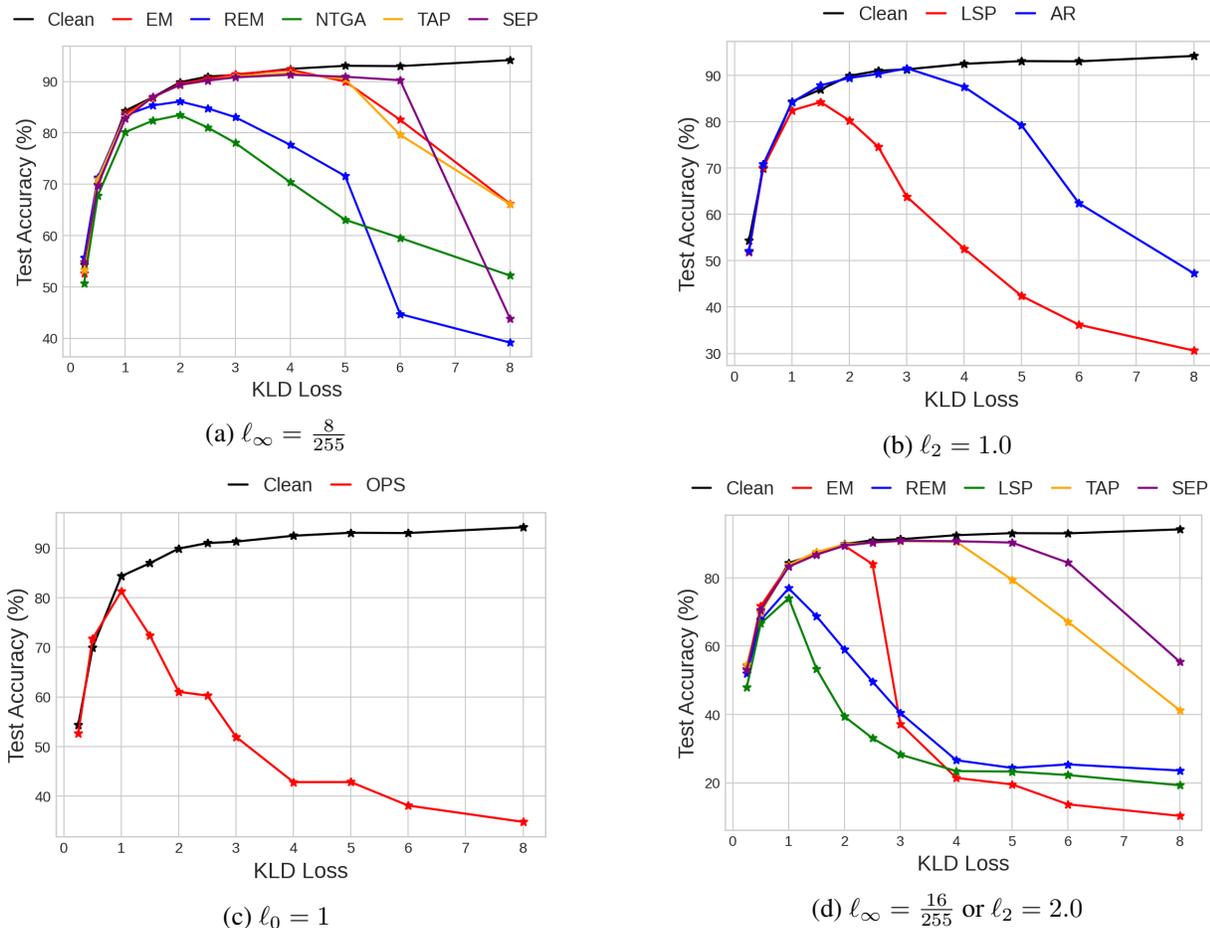
(a) $\ell_\infty = \frac{8}{255}$

(b) $\ell_2 = 1.0$

(c) $\ell_0 = 1$

(d) $\ell_\infty = \frac{16}{255}$ or $\ell_2 = 2.0$

Figure 6. Results using D-VAEs: Test Acc. Vs. KLD Loss is assessed on the unlearnable CIFAR-10.

I. More experiments on training D-VAE on attack methods with various target values on the KLD Loss

Some concerns regarding whether a sizable component of the perturbation will end up being learned into $\hat{x}$ may arise in certain cases, such as when the target value on the KLD loss is not set low. Nevertheless, when the KLD loss is set to a low value, the presence of perturbations in the reconstructed $\hat{x}$ is shown to be minimal. This observation is supported by both empirical experiments in Section 3.2 and theoretical explanations provided in Section 3.3. These outcomes are primarily attributed to the fact that the reconstruction of $\hat{x}$ depends on the information encoded in the latent representation z, *i.e.,* $\hat{x}$ is directly generated from z using a decoder. The theoretical insights discussed in Section 3.3 highlight that **Theorem 1** indicates that perturbations which create strong attacks tend to have a larger inter-class distance and a smaller intra-class variance. Additionally, **Theorem 2** and **Remark 1** indicate that perturbations possessing these characteristics are more likely to be eliminated when aligning the features with a normal Gaussian distribution (as done by the VAE).

To further validate these observations, we now include additional experiments in Appendix G by training D-VAE on all attack methods with various target values for the KLD loss. Additionally, we have performed experiments on attacks with larger perturbations. Notably, we have added results on the clean dataset for comparison. As depicted in Figure 6, when the target value on the KLD loss is set below 1.0, the curves of the results on the unlearnable dataset align closely with the results on the clean dataset. Furthermore, as the target value decreases, the removal of perturbations in the reconstructed $\hat{x}$ increases. While it is evident that larger perturbations may be better retained in $\hat{x}$, it is a cat-and-mouse game between defense and attack. Additionally, larger perturbations tend to be more noticeable. These findings affirm that the observations hold for all existing attack methods, and setting a low target value (*e.g.,* 1.0, as in the main experiments) on the KLD loss significantly ensures that $\hat{x}$ contains few perturbations.

Table 18. Comparison of Defenses

Defenses	JPEG	AVA.	Ours (1.5/2.5)	Ours (1.5/3.0)	Ours (0.5/3.0)	Ours (1.0/3.5)	Ours (1.0/3.0) as reported
NTGA	78.97	80.72	87.18	87.64	89.18	88.65	89.21
EM	85.61	89.54	90.65	91.14	91.64	91.86	91.42
TAP	84.99	89.13	90.60	90.98	90.38	91.52	90.48
REM	84.40	86.06	86.60	85.77	85.47	84.58	86.38
SEP	84.97	89.56	90.02	90.76	90.23	91.31	90.74
LSP	79.91	81.15	89.61	90.50	91.40	91.72	91.20
AR	84.97	89.64	90.23	91.29	90.80	90.52	91.77
OPS	77.33	71.62	87.89	86.18	89.39	86.50	88.96
Mean	82.64	84.67	89.09	89.30	89.56	89.58	**90.02**

J. Selection of various kld_1, kld_2.

To showcase that our method is tolerant to the selection of kld_1 and kld_2, we conduct experiments on the CIFAR-10 dataset. We present the defensive performance against eight UEs methods as shown the Table 18. We denote our method with different hyperparameters as "Ours (kld_1/kld_2)". Our findings indicate that while varying hyperparameters may result in only a slight decrease in the effectiveness of our proposed method (less than 1%). Furthermore, as depicted in Table 5 in our paper, when confronted with UEs with larger perturbation bounds, our method with the exact same kld_1 and kld_2 values exhibits slight performance degradation, yet still manages to achieve superior performance overall.