Black-box Backdoor Defense via Zero-shot Image Purification Supplementary Material

Anonymous Author(s)
Affiliation
Address
email

A Theoretical Justification

A.1 Proof on rectified estimation of $\mathbf{x}_t$ in Equation 7

Equation 7 $\mathbf{x}'_t = \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{x}^A - \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}_t + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t.$

Proof. Based on $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$, we have a nice property [15]:

$$\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t). \tag{1}$$

To ensure that the approximated clean image $\mathbf{x}_{0|t}$ based on the t-th step observation $\mathbf{x}_t$ satisfies the constraint in Equation 6, we have:

$$\mathbf{x}_{0|t} = \mathbf{A}^\dagger\mathbf{x}^A - \mathbf{A}^\dagger\mathbf{A}\mathbf{p} + \left(\mathbf{I} - \mathbf{A}^\dagger\mathbf{A}\right)\mathbf{x}_{0|t}. \tag{2}$$

Combine the above two equations, we can have:

$$\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t) = \mathbf{A}^\dagger\mathbf{x}^A - \mathbf{A}^\dagger\mathbf{A}\mathbf{p} + \left(\mathbf{I} - \mathbf{A}^\dagger\mathbf{A}\right)\left(\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t)\right). \tag{3}$$

Taking the derivative, we arrive at:

$$\mathbf{x}'_t = \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{x}^A - \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}_t + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t. \tag{4}$$

□

A.2 Proof of Lemma 3.1

Lemma 3.1 *Suppose the estimated noise output by $g_\phi(\cdot)$ is Gaussian. Given $g_\phi(\mathbf{x}_t, t) = \boldsymbol{\epsilon}_t$, we have $g_\phi((\mathbf{x}_t + \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p}), t) = \boldsymbol{\epsilon}_t + \boldsymbol{\epsilon}'_t$, where $\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}'_t$ are also Gaussian.*

Proof. Let us define the output of $g_\phi((\mathbf{x}_t + \mathbf{x}'_t), t)$ as $\hat{\boldsymbol{\epsilon}}_t$, so we have $g_\phi((\mathbf{x}_t + \mathbf{x}'_t), t) = \hat{\boldsymbol{\epsilon}}_t$. Next, we define $\boldsymbol{\epsilon}_t \sim N(\mu_1, \sigma_1^2)$ and $\hat{\boldsymbol{\epsilon}}_t \sim N(\mu_2, \sigma_2^2)$.

Since $\hat{\boldsymbol{\epsilon}}_t$ also follows a Gaussian distribution, we can subtract $\boldsymbol{\epsilon}_t$ from $\hat{\boldsymbol{\epsilon}}_t$ to obtain $\boldsymbol{\epsilon}'_t$, such that:

$$\boldsymbol{\epsilon}'_t = \hat{\boldsymbol{\epsilon}}_t - \boldsymbol{\epsilon}_t \sim N(\mu_2 - \mu_1, \sigma_1^2 + \sigma_2^2) \tag{5}$$

This confirms that $\boldsymbol{\epsilon}'_t$ is also Gaussian, thus completing the proof. □

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

A.3 Proof of Theorem 3.2

Theorem 3.2 *Suppose that $g_\phi((\mathbf{x}_t + \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p})) = \boldsymbol{\epsilon}_t + \boldsymbol{\epsilon}'_t$. We define the error at step t between $\hat{\mathbf{x}}_t$ and $(\mathbf{x}_t + \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p})$ as $\boldsymbol{\delta}'_t$, i.e., $\boldsymbol{\delta}'_t = \hat{\mathbf{x}}_t - (\mathbf{x}_t + \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p})$. Let $\boldsymbol{\delta}_t = \mathbf{A}\boldsymbol{\delta}'_t$, we have the following bound on its norm: $\|\boldsymbol{\delta}_t\| \leq \frac{(1-\bar{\alpha}_t)\sqrt{\alpha_{t+1}}}{\sqrt{1-\bar{\alpha}_{t+1}}}\|\mathbf{A}\|\|\boldsymbol{\epsilon}'_{t+1}\|$.*

Proof. We prove the above theorem by induction.

1 The base case is when $t = T$, where we have $\hat{\mathbf{x}}_T - (\mathbf{x}_T + \sqrt{\bar{\alpha}_T}\mathbf{A}^\dagger\mathbf{A}\mathbf{p}) = 0$, which holds.

2 Suppose for $\hat{\mathbf{x}}_t - (\mathbf{x}_t + \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p}) = \mathbf{A}^\dagger\boldsymbol{\delta}_t$, $\|\boldsymbol{\delta}_t\| \leq \frac{(1-\bar{\alpha}_t)\sqrt{\alpha_{t+1}}}{\sqrt{1-\bar{\alpha}_{t+1}}}\|\mathbf{A}\|\|\boldsymbol{\epsilon}'_{t+1}\|$ is true.

3 Induction:

$$\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha}_t}(\sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{x}^A - \sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}_t + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t) + \sigma_t\boldsymbol{\epsilon}, \quad (6)$$

$$\hat{\mathbf{x}}_{t-1} = \frac{1}{\sqrt{\alpha}_t}(\sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{x}^A + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\hat{\mathbf{x}}_t + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_t}(\boldsymbol{\epsilon}_t + \boldsymbol{\epsilon}'_t) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}(\boldsymbol{\epsilon}_t + \boldsymbol{\epsilon}'_t)) + \sigma_t\boldsymbol{\epsilon}, \quad (7)$$

$$\hat{\mathbf{x}}_{t-1} - \mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha}_t}(\sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})(\sqrt{\bar{\alpha}_t}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + \mathbf{A}^\dagger\boldsymbol{\delta}_t) + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}'_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}'_t) \quad (8)$$

By definition, we have $\hat{\mathbf{x}}_{t-1} - (\mathbf{x}_{t-1} + \sqrt{\bar{\alpha}_{t-1}}\mathbf{A}^\dagger\mathbf{A}\mathbf{p}) = \boldsymbol{\delta}'_{t-1}$. Let $\boldsymbol{\delta}_{t-1} = \mathbf{A}\boldsymbol{\delta}'_{t-1}$, we have $\boldsymbol{\delta}_{t-1} = \mathbf{A}\boldsymbol{\delta}'_{t-1} = \mathbf{A}(\hat{\mathbf{x}}_{t-1} - (\mathbf{x}_{t-1} + \sqrt{\bar{\alpha}_{t-1}}\mathbf{A}^\dagger\mathbf{A}\mathbf{p}))$. Since $\mathbf{A}(\mathbf{I} - \mathbf{A}^\dagger\mathbf{A}) = \mathbf{0}$, we can have:

$$\boldsymbol{\delta}_{t-1} = \frac{1}{\sqrt{\alpha}_t}(\mathbf{A}\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}'_t - \mathbf{A}\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}'_t) = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{\sqrt{1-\bar{\alpha}_t}}\mathbf{A}\boldsymbol{\epsilon}'_t \quad (9)$$

Based on the Cauchy–Schwarz inequality:

$$\|\boldsymbol{\delta}_{t-1}\| = \|\frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{\sqrt{1-\bar{\alpha}_t}}\mathbf{A}\boldsymbol{\epsilon}'_t\| \leq \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{\sqrt{1-\bar{\alpha}_t}}\|\mathbf{A}\|\|\boldsymbol{\epsilon}'_t\| \quad (10)$$

□

A.4 Proof of Corollary 3.2.1

Corollary 3.2.1 *When $t = 0$, we have $\hat{\mathbf{x}}_0 = \mathbf{x}_0 + \mathbf{A}^\dagger\mathbf{A}\mathbf{p} + \boldsymbol{\delta}'_0$, where $\mathbf{A}\boldsymbol{\delta}'_0 = \mathbf{0}$.*

Proof. First, we have:

$$\mathbf{x}_0 \leftarrow \frac{1}{\sqrt{\alpha}_1}(\sqrt{\bar{\alpha}_1}\mathbf{A}^\dagger\mathbf{x}^A - \sqrt{\bar{\alpha}_1}\mathbf{A}^\dagger\mathbf{A}\mathbf{p} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}_1 + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_1}\boldsymbol{\epsilon}_1 - \frac{1-\alpha_1}{\sqrt{1-\bar{\alpha}_1}}\boldsymbol{\epsilon}_1) + \sigma_1\boldsymbol{\epsilon}, \quad (11)$$

$$\hat{\mathbf{x}}_0 = \frac{1}{\sqrt{\alpha}_1}(\sqrt{\bar{\alpha}_1}\mathbf{A}^\dagger\mathbf{x}^A + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\hat{\mathbf{x}}_1 + \mathbf{A}^\dagger\mathbf{A}\sqrt{1-\bar{\alpha}_1}(\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}'_1) - \frac{1-\alpha_1}{\sqrt{1-\bar{\alpha}_1}}(\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}'_1)) + \sigma_1\boldsymbol{\epsilon}, \quad (12)$$

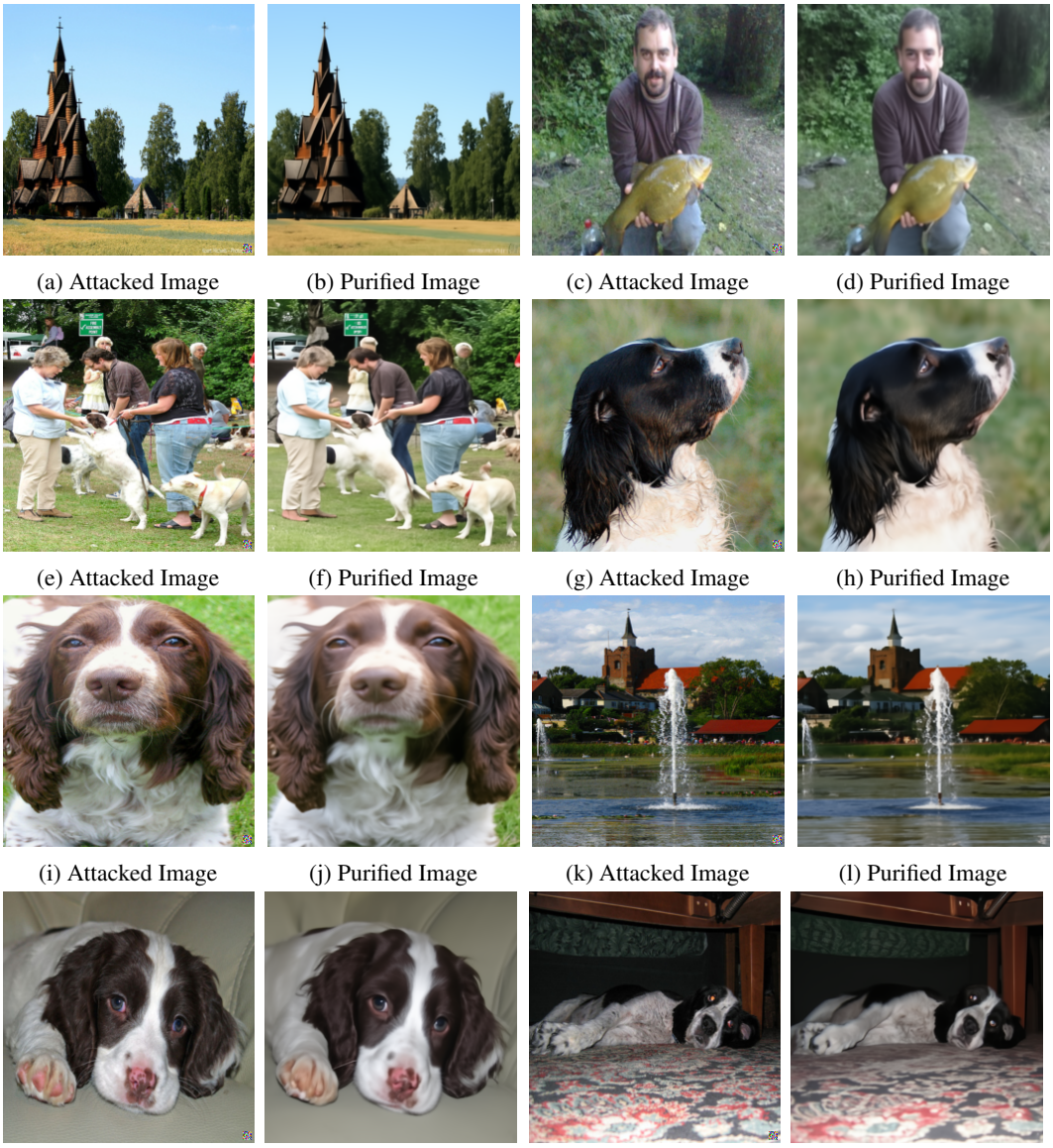
Then we can have:

$$\mathbf{A}\boldsymbol{\delta}'_0 = \frac{(1-\alpha_0)\sqrt{\alpha_1}}{\sqrt{1-\bar{\alpha}_1}}\mathbf{A}\boldsymbol{\epsilon}'_1. \quad (13)$$

Since $\alpha_0 = 1$, then we have $\mathbf{A}\boldsymbol{\delta}'_0 = \mathbf{0}$. □

B Qualitative Results of Purification

B.1 Qualitative Results of Purification on BadNet Attack



(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

Figure 1: Comparison of Purified and BadNet Attack Images(Part1).

(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 2: Comparison of Purified and BadNet Attack Images(Part2).

(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

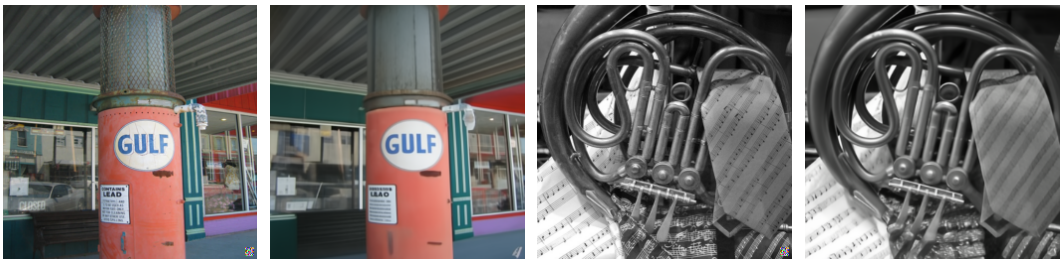
(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

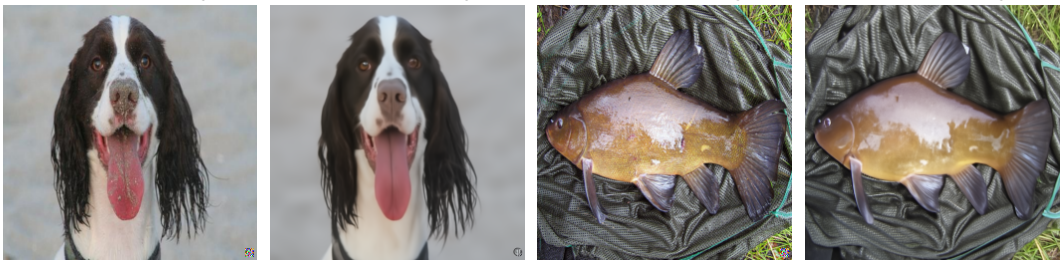
(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 3: Comparison of Purified and BadNet Attack Images(Part3).



(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image



(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 4: Comparison of Purified and BadNet Attack Images(Part4).

B.2 Qualitative Results of Purification on Blended Attack

(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

Figure 5: Comparison of Purified and Blended Attack Images(Part1).

(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

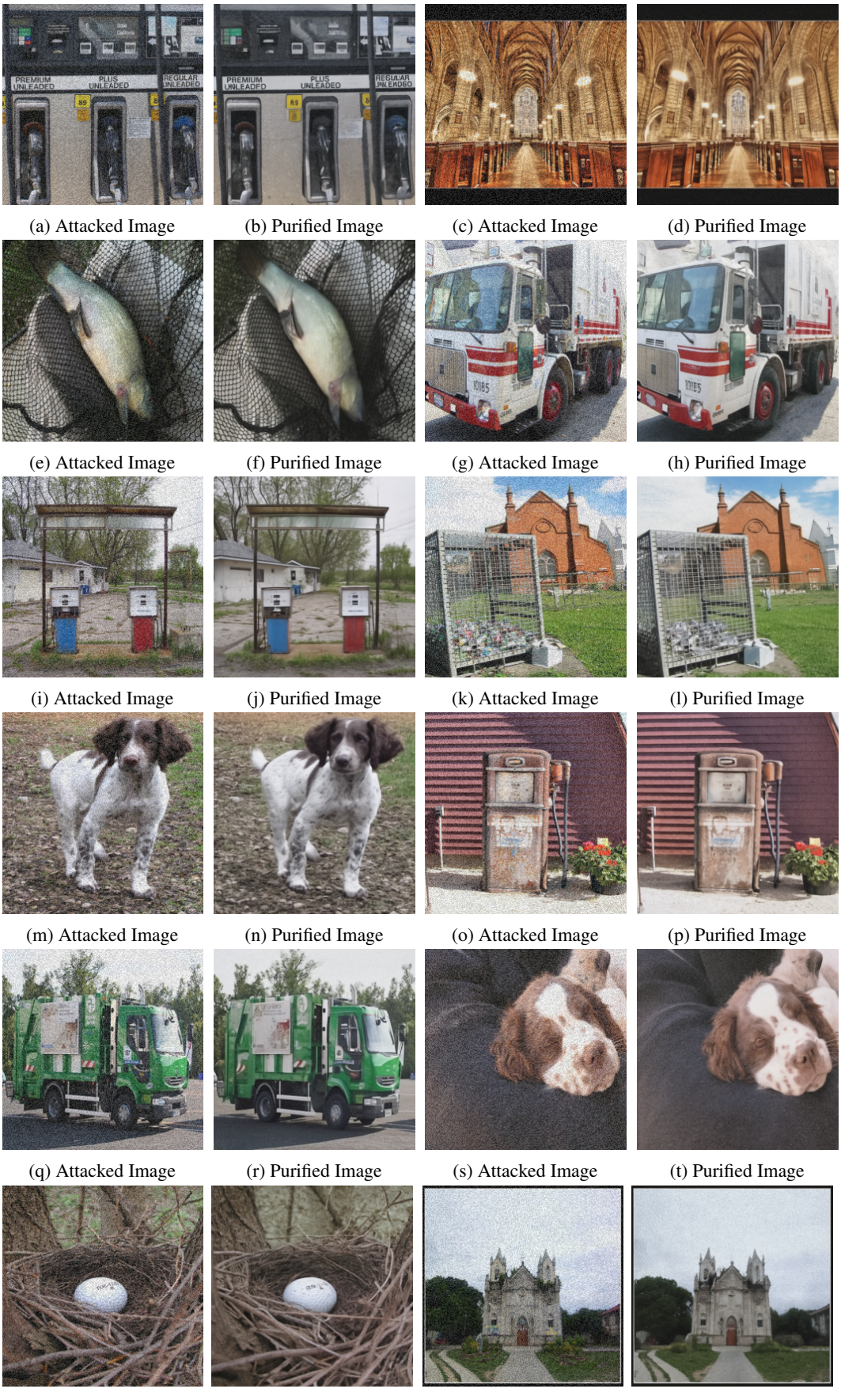
(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 6: Comparison of Purified and Blended Attack Images(Part2).



(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 7: Comparison of Purified and Blended Attack Images(Part3).

(a) Attacked Image (b) Purified Image (c) Attacked Image (d) Purified Image

(e) Attacked Image (f) Purified Image (g) Attacked Image (h) Purified Image

(i) Attacked Image (j) Purified Image (k) Attacked Image (l) Purified Image

(m) Attacked Image (n) Purified Image (o) Attacked Image (p) Purified Image

(q) Attacked Image (r) Purified Image (s) Attacked Image (t) Purified Image

(u) Attacked Image (v) Purified Image (w) Attacked Image (x) Purified Image

Figure 8: Comparison of Purified and Blended Attack Images(Part4).

C Details on Linear Transformations

In this section, we discuss details of the linear transformation applied in our paper. Two practical examples are discussed to illustrate the simplicity and effectiveness of linear transformations, along with the corresponding operators used.

Gray-scale Conversion: To convert an RGB image to gray-scale, the operator $\mathbf{A} = [1/3, 1/3, 1/3]$ can be defined as a pixel-wise operation that transforms each RGB channel pixel $[r, g, b]$ into a gray-scale value $r^3 + g^3 + b^3$. In this case, constructing a pseudo-inverse $\mathbf{A}^\dagger = [1, 1, 1]^{\mathrm{T}}$ is straightforward, satisfying the condition $\mathbf{A}\mathbf{A}^\dagger \equiv \mathbf{I}$, where $\mathbf{I}$ represents the identity matrix.

Image Blurring: Image blurring also involves linear transformations. For a blurring operation with scale n, the operator $\mathbf{A}$ is defined as the average-pooling operator $[1/n^2, ..., 1/n^2]$. This operator aggregates each patch of the image into a single value. Similarly, the pseudo-inverse $\mathbf{A}^\dagger$ can be built as $\mathbf{A}^\dagger = [1, ..., 1]^{\mathrm{T}}$ to fulfill the condition $\mathbf{A}\mathbf{A}^\dagger \equiv \mathbf{I}$.

Overall, these examples demonstrate how these two linear transformations, in conjunction with their respective operators, can be employed to destruct the trigger pattern without relying on a complex Fourier transform. In cases where the linear transformation is too complex to solve for its pseudo-inverse, the Singular Value Decomposition (SVD) method can be applied. For more details, please refer to papers [34, 21].

D Algorithm for improved ZIP based on DDIM

In this section, we include the modified algorithm based on DDIM, which is proposed to speed up the diffusion model inference speed.

Algorithm 1 Zero-shot Image Purification (based on DDIM)

Require: Test image for purification $\mathbf{x}^P$; liner transformation $\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_n$ and their pseudo-inverse $\mathbf{A}_1^\dagger, \mathbf{A}_2^\dagger, ..., \mathbf{A}_n^\dagger$; diffusion model g; hyperparameter λ; speed-up pace S.

Ensure: $\mathbf{x}^{A_n} = \mathbf{A}_n \mathbf{x}^P,\ n = 0, 1, ..., N$

$\mathbf{x}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, T-S, ..., S, 1$ **do**
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\boldsymbol{\epsilon} = \mathbf{0}$
  $\boldsymbol{\epsilon}_t = g_\phi(\mathbf{x}_t, t)$
  for $n = 1, 2, ..., N$ **do**
    $\hat{\mathbf{x}}_t^n = \sqrt{\bar{\alpha}_t}\mathbf{A}_1^\dagger \mathbf{x}^{A_n} + (\mathbf{I} - \mathbf{A}_n^\dagger \mathbf{A}_n)\mathbf{x}_t + \mathbf{A}_n^\dagger \mathbf{A}_n \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t$
  end for
  $\hat{\mathbf{x}}_t = \frac{1}{N}(\hat{\mathbf{x}}_t^1 + \hat{\mathbf{x}}_t^2 + , ..., + \hat{\mathbf{x}}_t^N)$
  $\tilde{\mathbf{x}}_t = (1 - \bar{\alpha}_t^\lambda)\hat{\mathbf{x}}_t + \bar{\alpha}_t^\lambda \mathbf{x}_t$
  $\tilde{\mathbf{x}}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\tilde{\mathbf{x}}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t)$
  $\mathbf{x}_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\tilde{\mathbf{x}}_{0|t} + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\boldsymbol{\epsilon}_t + \sigma_t \boldsymbol{\epsilon}$
end for
return $\mathbf{x}_0$

E Experiments Settings

E.1 Datasets Informaiton

CIFAR-10 [22] The CIFAR-10 dataset is a widely-used benchmark in computer vision. It consists of 60,000 color images of size 32x32 pixels, belonging to 10 different classes, with 6,000 images per class. The dataset is divided into 50,000 training images and 10,000 test images, with a balanced distribution of classes.

GTSRB [31] The German Traffic Sign Recognition Benchmark (GTSRB) dataset is designed for traffic sign classification tasks. It comprises more than 50,000 images of traffic signs captured under various real-world conditions. The images have different sizes and aspect ratios, but they are resized

to 32x32 pixels for our model training and evaluation. The dataset is divided into training and test sets, with its official split ratio.

Imagenette [17] The Imagenette is a subset of the larger ImageNet dataset and is commonly used as a smaller-scale alternative for image classification tasks. It consists of 10 classes with a total of 13,000 images. The images in Imagenette have varying sizes, but they are resized to 256x256 pixels for consistency. The dataset is split into training and validation sets, following a predefined split ratio.

Table 1: Properties of datasets.

Dataset	Classes	Image Size	Train Split	Test Split
CIFAR-10	10	32x32	50,000	10,000
GTSRB	43	32x32	39,209	12,630
Imagenette	10	256x256	9,480	3,936

E.2 Attacks Implementation

In this section, we discuss the implementation details of three different backdoor attack methods employed in our study: BadNet, Blended, and PhysicalBA. We implement these backdoor attacks using the Backdoorbox framework [24], which is under GNU general public license.

BadNet [11] The BadNet attack injects specific trigger patterns into the training data. In our implementation, we set the poisoned rate to 5%, i.e., 5% of the training samples are selected as attack samples and have the trigger pattern added to them. The trigger pattern size is set to 2x2 for 32x32 pixels images and 9x9 for 256x256 pixels images. The trigger patterns are randomly generated.

Blended [7] The Blended attack is a more sophisticated variant aimed at making the backdoor less conspicuous and harder to detect. Following the suggestion in BackdoorBox, we set the blended rate to 0.2 and the poisoned rate to 5%. The blended pattern is randomly generated, seamlessly blending the trigger pattern into the attack samples.

PhysicalBA [25] The PhysicalBA (Physical Backdoor Attack) is a specific type of attack that introduces variations in the location and appearance of the attack pattern embedded in the test samples during inference time. In our implementation, we apply the same attack pattern size as the BadNet attack, using a 2x2 pattern for 32x32 pixels images and a 9x9 pattern for 256x256 pixels images. The attack patterns are generated randomly. We set the poisoned rate to 5% for this attack.

All other attack settings follow the default configurations in Backboorbox [24].

E.3 Purification Implementation

We utilize a pre-trained model provided by OpenAI [8] under the MIT license. The algorithm described in Algorithm 1 is employed to accelerate the inference process, allowing us to generate high-quality images within just 20 steps, and the speed-up pace is set to 50. We set the hyperparameter λ to a value of 2 for Blended attack defense, and 10 for BadNet and PhysicalBA attack defense.

Specifically, for the CIFAR-10 dataset, we apply both blur and gray-scale conversion as linear transformations. For the GTSRB and Imagenette datasets, we solely apply blur as the linear transformation. Additional implementation details can be found in the code we have provided.

F Ablation Study Settings

F.1 Enhanced Attack Settings

In the enhanced attack settings, our first step is to extract 5% of the training dataset and inject the attack's trigger pattern into these images. We then proceed to purify this subset of data using blur and grayscale as linear transformations during the first stage of our proposed purification. Once the attacked images have been successfully purified, we modify their labels to reflect the attack label. Following this, we introduce these purified images as poisoned samples into the training set and train a classification model from scratch. This comprehensive procedure is referred to as the enhanced attack process.

F.2 Purification Speed Settings

This paper focuses on defending against backdoor attacks during the inference phase using purification techniques. To evaluate the purification speed, we conduct experiments using a workstation that features an Intel(R) Core(TM) i9-10900X CPU and an NVIDIA RTX3070 GPU with 8GB of memory.

During our experiments, we measure the classification time, which represents the duration taken by the classifier model to perform inference on a single image. Additionally, we measure the purification time, which indicates the time required by the purification model to purify a single image.

For the combination of purification with detection, we utilize the Scale-up method [13] as our chosen detection technique. Furthermore, the dataset used for speed evaluation consists of 5% poisoned images. Following previous settings [13], we set a batch size of one for the classifier model and report the average time based on 640 runs.

G Related Work

G.1 Backdoor Attack

Existing backdoor attack methods involve the injection of poisoned samples into the training process of Deep Neural Networks (DNNs). These attacks can target various types of models, including image classification models, object detection models [4, 26], contrastive learning models [2], and language models [27, 29, 41, 5]. The attackers exploit vulnerabilities by embedding adversary-specified trigger patterns into carefully selected benign samples. Backdoor attacks are characterized by their stealthiness, as the attacked models behave normally on benign samples, making the hidden triggers difficult to detect and purify.

There are mainly two categories of backdoor attacks for image classification tasks: patch-based and non-patch-based attacks. Patch-based attacks are attacks with triggers embedded as patches or overlays within the input samples. For example, Souri et al. [30] propose the Sleeper Agent attack, which is a sophisticated backdoor attack where an adversary subtly injects hidden triggers into an image classification model during training, remaining dormant until specific conditions activate malicious behavior. Non-patch-based attacks are attacks where triggers are integrated without explicit patching, often relying on specific input sequences or subtle modifications in the feature space [40, 16, 14]. For example, Doan et al. [9] introduce Wasserstein backdoor attack, an extension of the imperceptible backdoor concept to the latent representation. Their proposed attack manipulates inputs with imperceptible noise, matching latent representations to achieve high attack success rates while remaining stealthy in both the input and latent spaces.

G.2 Backdoor Defense

Existing defense methods for backdoor models can be broadly categorized into two approaches: (1) detection-based methods and (2) purification-based methods.

Detection-based methods focus on identifying the presence of backdoors in trained models. These methods typically involve analyzing the model's behavior and inputs to detect any suspicious patterns or triggers that indicate the existence of a backdoor [1]. Various techniques such as anomaly detection [10, 18, 38], and statistical analysis [12, 6] have been employed to detect backdoors. The goal of detection-based methods is to provide an early warning system to identify and mitigate the risks posed by backdoor attacks.

On the other hand, purification-based methods aim to remove or neutralize the effects of backdoors from the model. These methods involve modifying the model or its training process to eliminate the influence of the backdoor triggers on the model's behavior [19]. Some purification approaches focus on retraining the model using clean or carefully selected training data to reduce the impact of the backdoor [39, 35, 23, 20, 32]. Other methods aim to directly identify and neutralize the backdoor triggers within the model's parameters or hidden representations [33, 3, 37, 36, 28]. The objective of purification-based methods is to restore the integrity and reliability of the model by eliminating the malicious behavior induced by the backdoor.

H Qualitative Results Comparison with Image Restoration Methods

We conducted a comparative analysis of the purification effect between our proposed method and DDNM [34], which is a state-of-the-art image restoration technique. In order to ensure a fair comparison, we implemented DDNM using their official code and applied identical linear transformations, diffusion steps, and schedules to both methods. The qualitative results, which demonstrate the effectiveness of our proposed approach, are presented below.

(a) Attack Image (b) Restored by DDNM (c) Purified by ZIP

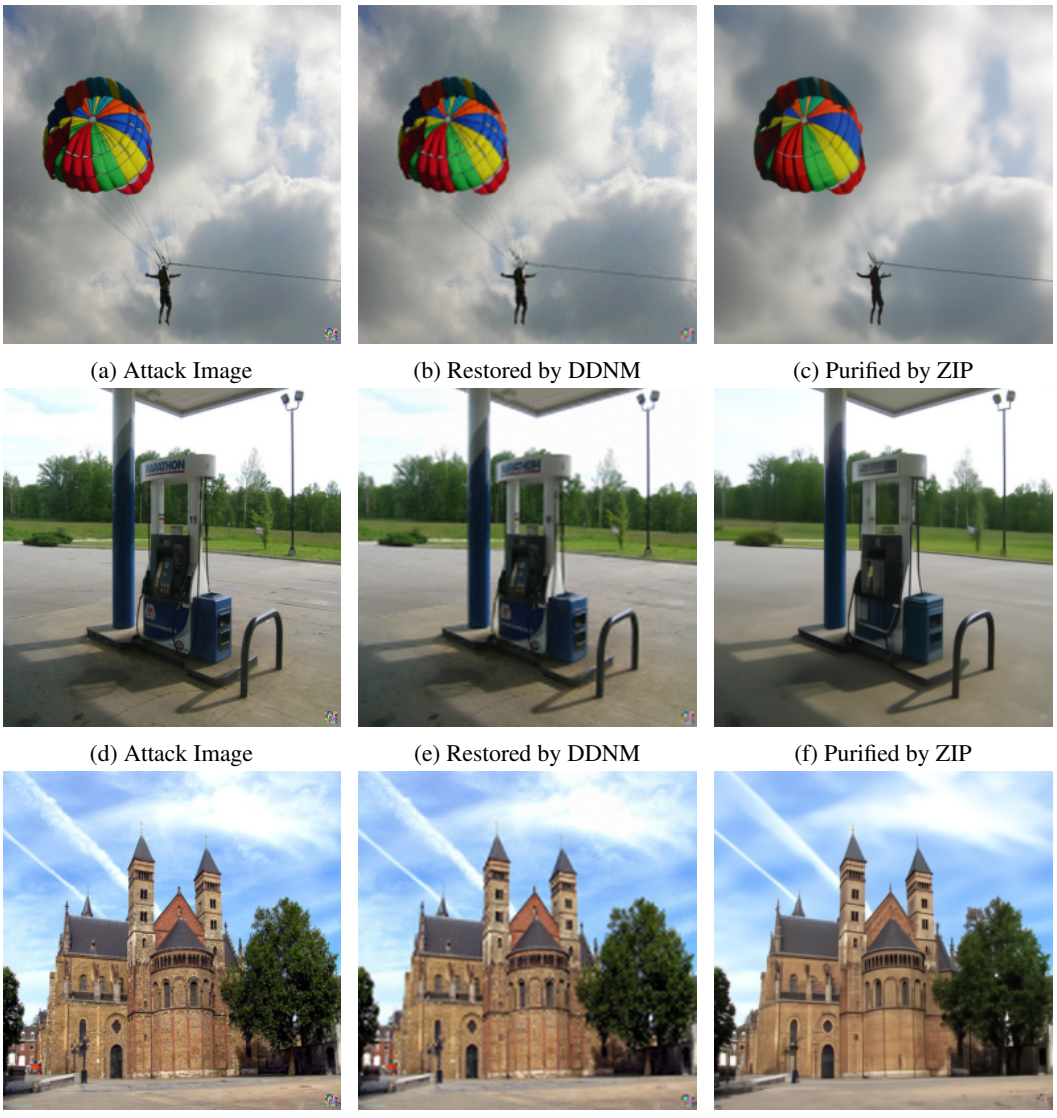
(d) Attack Image (e) Restored by DDNM (f) Purified by ZIP

(g) Attack Image (h) Restored by DDNM (i) Purified by ZIP

Figure 9: Comparison of DDNM and ZIP on defending Blended attack.

I Limitations

Due to our reliance on a pre-trained diffusion model to implement zero-shot purification, the effectiveness of generating purified images may be weakened when our model is applied to highly specific images that fall outside the distribution of pre-processed data. To mitigate this issue, we suggest two possible solutions in future work: 1) replacing the current pre-trained diffusion model with a



(a) Attack Image (b) Restored by DDNM (c) Purified by ZIP

(d) Attack Image (e) Restored by DDNM (f) Purified by ZIP

(g) Attack Image (h) Restored by DDNM (i) Purified by ZIP

Figure 10: Comparison of DDNM and ZIP on defending BadNet attack.

more suitable pre-trained model for such specific images, and 2) collecting a subset of highly specific images to perform fine-tuning on the pre-trained model.

References

[1] Ruisi Cai, Zhenyu Zhang, Tianlong Chen, Xiaohan Chen, and Zhangyang Wang. Randomized channel shuffling: Minimal-overhead backdoor attack detection without clean datasets. *Advances in Neural Information Processing Systems*, 35:33876–33889, 2022.

[2] Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. *arXiv preprint arXiv:2106.09667*, 2021.

[3] Shuwen Chai and Jinghui Chen. One-shot neural backdoor erasing via adversarial weight masking. *arXiv preprint arXiv:2207.04497*, 2022.

[4] Kangjie Chen, Xiaoxuan Lou, Guowen Xu, Jiwei Li, and Tianwei Zhang. Clean-image backdoor: Attacking multi-label models with poisoned labels only. In *The Eleventh International Conference on Learning Representations*.

[5] Kangjie Chen, Yuxian Meng, Xiaofei Sun, Shangwei Guo, Tianwei Zhang, Jiwei Li, and Chun Fan. Badpre: Task-agnostic backdoor attacks to pre-trained nlp foundation models. *arXiv preprint arXiv:2110.02467*, 2021.

[6] Weixin Chen, Baoyuan Wu, and Haoqian Wang. Effective backdoor defense by exploiting sensitivity of poisoned samples. *Advances in Neural Information Processing Systems*, 35:9727–9737, 2022.

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

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[9] Khoa Doan, Yingjie Lao, and Ping Li. Backdoor attack with imperceptible input and latent modification. *Advances in Neural Information Processing Systems*, 34:18944–18957, 2021.

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

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

[12] Junfeng Guo, Ang Li, and Cong Liu. Aeva: Black-box backdoor detection using adversarial extreme value analysis. *International Conference on Learning Representations*, 2022.

[13] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. *The Eleventh International Conference on Learning Representations*, 2023.

[14] Jonathan Hayase and Sewoong Oh. Few-shot backdoor attacks via neural tangent kernels. *arXiv preprint arXiv:2210.05929*, 2022.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[16] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. Handcrafted backdoors in deep neural networks. *Advances in Neural Information Processing Systems*, 35:8068–8080, 2022.

[17] Jeremy Howard. Imagenette. `https://github.com/fastai/imagenette`, 2019.

[18] Xiaoling Hu, Xiao Lin, Michael Cogswell, Yi Yao, Susmit Jha, and Chao Chen. Trigger hunting with a topological prior for trojan detection. *arXiv preprint arXiv:2110.08335*, 2021.

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

[20] Charles Jin, Melinda Sun, and Martin Rinard. Incompatibility clustering as a defense against backdoor poisoning attacks. In *The Eleventh International Conference on Learning Representations*.

[21] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 2022.

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

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

[24] Yiming Li, Mengxi Ya, Yang Bai, Yong Jiang, and Shu-Tao Xia. Backdoorbox: A python toolbox for backdoor learning. *arXiv preprint arXiv:2302.01762*, 2023.

[25] Yiming Li, Tongqing Zhai, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor attack in the physical world. *arXiv preprint arXiv:2104.02361*, 2021.

[26] Yiming Li, Haoxiang Zhong, Xingjun Ma, Yong Jiang, and Shu-Tao Xia. Few-shot backdoor attacks on visual object tracking. *arXiv preprint arXiv:2201.13178*, 2022.

[27] Yepeng Liu, Bo Feng, and Qian Lou. Trojtext: Test-time invisible textual trojan insertion. *arXiv preprint arXiv:2303.02242*, 2023.

[28] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1265–1282, 2019.

[29] Guangyu Shen, Yingqi Liu, Guanhong Tao, Qiuling Xu, Zhuo Zhang, Shengwei An, Shiqing Ma, and Xiangyu Zhang. Constrained optimization with dynamic bound-scaling for effective nlp backdoor defense. In *International Conference on Machine Learning*, pages 19879–19892. PMLR, 2022.

[30] Hossein Souri, Liam Fowl, Rama Chellappa, Micah Goldblum, and Tom Goldstein. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. *Advances in Neural Information Processing Systems*, 35:19165–19178, 2022.

[31] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

[32] Guanhong Tao, Yingqi Liu, Guangyu Shen, Qiuling Xu, Shengwei An, Zhuo Zhang, and Xiangyu Zhang. Model orthogonalization: Class distance hardening in neural networks for better security. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1372–1389. IEEE, 2022.

[33] Haotao Wang, Junyuan Hong, Aston Zhang, Jiayu Zhou, and Zhangyang Wang. Trap and replace: Defending backdoor attacks by trapping them into an easy-to-replace subnetwork. *arXiv preprint arXiv:2210.06428*, 2022.

[34] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*, 2022.

[35] Zhenting Wang, Hailun Ding, Juan Zhai, and Shiqing Ma. Training with more confidence: Mitigating injected and natural backdoors during training. *Advances in Neural Information Processing Systems*, 35:36396–36410, 2022.

[36] Zhenting Wang, Kai Mei, Juan Zhai, and Shiqing Ma. Unicorn: A unified backdoor trigger inversion framework. *arXiv preprint arXiv:2304.02786*, 2023.

[37] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34:16913–16925, 2021.

[38] Zhen Xiang, David J Miller, and George Kesidis. Post-training detection of backdoor attacks for two-class and multi-attack scenarios. *arXiv preprint arXiv:2201.08474*, 2022.

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

[40] Zhiyuan Zhang, Lingjuan Lyu, Weiqiang Wang, Lichao Sun, and Xu Sun. How to inject backdoors with better consistency: Logit anchoring on clean data. *arXiv preprint arXiv:2109.01300*, 2021.

[41] Biru Zhu, Yujia Qin, Ganqu Cui, Yangyi Chen, Weilin Zhao, Chong Fu, Yangdong Deng, Zhiyuan Liu, Jingang Wang, Wei Wu, et al. Moderate-fitting as a natural backdoor defender for pre-trained language models. *Advances in Neural Information Processing Systems*, 35:1086–1099, 2022.