TERD: A Unified Framework for Safeguarding Diffusion Models Against Backdoors

Yichuan Mo [1] **Hui Huang** [2] **Mingjie Li** [3] **Ang Li** [2] **Yisen Wang** [1 4]

Abstract

Diffusion models have achieved notable success in image generation, but they remain highly vulnerable to backdoor attacks, which compromise their integrity by producing specific undesirable outputs when presented with a pre-defined trigger. In this paper, we investigate how to protect diffusion models from this dangerous threat. Specifically, we propose **TERD**, a backdoor defense framework that builds unified modeling for current attacks, which enables us to derive an accessible reversed loss. A trigger reversion strategy is further employed: an initial approximation of the trigger through noise sampled from a prior distribution, followed by refinement through differential multi-step samplers. Additionally, with the reversed trigger, we propose backdoor detection from the noise space, introducing the first backdoor input detection approach for diffusion models and a novel model detection algorithm that calculates the KL divergence between reversed and benign distributions. Extensive evaluations demonstrate that TERD secures a 100% True Positive Rate (TPR) and True Negative Rate (TNR) across datasets of varying resolutions. TERD also demonstrates nice adaptability to other Stochastic Differential Equation (SDE)-based models. Our code is available at `https://github.com/PKU-ML/TERD`.

1. Introduction

In recent years, we have witnessed significant advancements in generative models (Goodfellow et al., 2014; Kingma & Welling, 2013; Kingma & Dhariwal, 2018), with diffusion models emerging as a particularly notable representative (Ho et al., 2020; Rombach et al., 2022a; Yang et al., 2023). These models have demonstrated their marvelous performances in a diverse range of applications from image generation (Rombach et al., 2022a), content editing (Meng et al., 2022), zero-shot classification (Li et al., 2023) to adversarial purification (Nie et al., 2022). However, the widespread application of diffusion models raises concerns about their security issues like backdoor attacks (Gu et al., 2017; Chen et al., 2017), where models can be manipulated to produce harmful outputs under specific conditions, posing significant legal and ethical risks. Therefore, in this paper, we explore how to defend against backdoor attacks for diffusion models, which is less investigated before.

Unlike common classification models, diffusion models operate on noise outputs rather than class logits, making them impervious to conventional defenses (Wu & Wang, 2021; Wu et al., 2022) designed for classification tasks. The challenge is exacerbated by the complexity of their input-output dynamics over various timesteps, e.g., the model's behavior changes across different timesteps and the underlying formulation is often inaccessible to defenders. This significantly hinders the ability to effectively identify and mitigate backdoor triggers.

To address this challenge, in this paper, we propose a novel defense strategy that begins by systematically characterizing existing backdoor attacks in diffusion models. Our approach involves creating unified formulations of backdoor attacks, enabling us to derive an accessible reversed loss. For the accessibility of inputs, we introduce a two-stage trigger reversion process: we first estimate the trigger using noise sampled from a prior distribution, followed by refinement through differential multi-step samplers. This process allows for accurate identification and neutralization of backdoor inputs. Equipped with the estimated trigger, we can detect backdoor attacks from both the input and model perspectives in the noise space, leveraging the statistical characteristics of noise distributions to distinguish between benign and malicious inputs. We refer to this comprehensive defense framework as TERD (Trigger Estimation and Refinement for Diffusion). TERD has demonstrated remark-

[1]National Key Lab of General Artificial Intelligence, School of Intelligence Science and Technology, Peking University, China [2]School of EECS, Peking University, China [3]CISPA Helmholtz Center for Information Security, Germany [4]Institute for Artificial Intelligence, Peking University, China. Correspondence to: Yisen Wang <yisen.wang@pku.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

able success across diverse datasets, achieving a 100% True Positive Rate (TPR) and True Negative Rate (TNR). Further, TERD works well against a wide range of attack scenarios, including those with varied poisoning rates, trigger sizes, and even sophisticated adaptive attacks. Beyond the diffusion models, TERD also shows promise for defending other Stochastic Differential Equation (SDE)-based models against backdoor attacks. In summary, our main contributions are listed as follows:

- We specially design a novel trigger reversion algorithm based on the unified modeling against backdoor attacks in diffusion models, which can accurately reverse triggers with high quality.
- With the reversed trigger, we develop an input and model detection method in the noise space to protect the diffusion models from backdoors.
- Extensive experiments show the efficacy of our defense across varied scenarios and its potential applicability to broader SDE-based generative models.

2. Related Work

2.1. Backdoor Attacks in Diffusion Models

Backdoor attacks, also known as Trojan attacks (Gu et al., 2017; Chen et al., 2017), were initially studied in the context of classification models. These attacks involve implanting pre-defined malicious behaviors into neural networks. While the victim models maintain normal functionality with benign inputs, the presence of a trigger in the input causes the model to exhibit malicious behaviors, such as misclassification or illegal content generation. Recent studies, such as Chou et al. (2023a) and Chen et al. (2023), have demonstrated that diffusion models are also vulnerable to these attacks. In these scenarios, a trigger is added to noise sampled from a prior distribution. Images generated from this altered noise become target images, resulting in unexpected sequences. VillianDiffusion (Chou et al., 2023b) further extends it to continuous diffusion models. Additional research has shown that backdoor attacks can be executed using natural language prompts (Zhai et al., 2023; Huang et al., 2023; Struppek et al., 2023) (specifically for text-to-image diffusion models) or by poisoning the training set (Pan et al., 2023). However, these attacks can be easily defended by purifying the text encoder or additional human inspection. Therefore, in this paper, we focus on defending against backdoor attacks from the pixel level, which not only has good stealthiness but also endangers all existing diffusion models.

2.2. Existing Backdoor Defense

Similar to defenses against adversarial attacks (Li et al., 2020; Wang et al., 2019b; 2020; Wu et al., 2020; Mo et al., 2022), current backdoor defenses mainly focus on classification models. These defenses can be categorized into two types: input-level and model-level defenses. Input-level defenses aim to detect whether an input sample is a backdoor sample. Previous studies have shown that backdoor samples can be identified through neural activations (Chen et al., 2018) or frequency analysis (Zeng et al., 2021). Techniques from other fields, such as differential privacy and explainable visualization tools, further enhance detection success rates (Doan et al., 2020; Du et al., 2019), as backdoor samples often appear as outliers relying on local spurious features. Model-level defenses work by first detecting whether a model has been implanted with a backdoor and then mitigating the backdoor effect. Regarding backdoors as shortcuts between the real and target classes, methods like (Wang et al., 2019a; Tao et al., 2022; Hu et al., 2021) employ reverse engineering by maximizing the classification loss across all classes to identify potential triggers. Once the model is identified as backdoored, purification-based defenses such as fine-tuning (Sha et al., 2022; Xiong et al., 2023), pruning (Wu & Wang, 2021; Chai & Chen, 2022), or unlearning (Liu et al., 2022; Wei et al., 2023) are employed to reduce the attack success rate while maintaining benign accuracy. However, these defenses fail to protect diffusion models because the input to a diffusion model is Gaussian noise rather than natural images, and diffusion models predict added Gaussian noise rather than discriminative results of natural images.

The most relevant work to ours is Elijah (An et al., 2023), the method designed specifically for backdoor defense in diffusion models. However, Elijah does not establish a unified loss for current attacks, assuming the trigger is part of the model output, which does not apply to state-of-the-art attacks such as TrojDiff (Chen et al., 2023). Additionally, Elijah's model detection method assumes that backdoor models generate images with higher similarity, a claim contradicted by Chen et al. (2023), which demonstrates that target images can consist of multiple images with diverse and colorful patterns.

3. Preliminary

3.1. Discrete Diffusion Model

Based on the Markov chain, Denoising Diffusion Probabilistic Models (DDPM) (Ho et al., 2020) connects the data and prior distribution (e.g., Gaussian distribution) by defining a forward diffusion and backward denoising process. In its forward process, Gaussian noise is gradually added to images and the conditional distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is defined as $\mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$ where $\alpha_t \in (0,1)$. According to Bayes Rule, given $\mathbf{x}_0$, we can sample $\mathbf{x}_t$ of timestep t

($0 < t \leq T$) directly from the following equation:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \tag{1}$$

where $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. The boundary conditions require that $\lim_{t \to T} \bar{\alpha}_t = 0$ to ensure that $p(\mathbf{x}_t|\mathbf{x}_0)$ converges to $\mathcal{N}(0, \mathbf{I})$. Therefore, in the denoising process, we first sample $\mathbf{x}_T$ from $\mathcal{N}(0, \mathbf{I})$ and then generate $\mathbf{x}_{t-1}$ step-by-step using $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)p(\mathbf{x}_{t-1}|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}$. According to Equation 1, we can estimate $\mathbf{x}_0$ with $\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}F_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}}$ once the network F_θ predicts $\boldsymbol{\epsilon}$:

$$\min_\theta ||F_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon}||_2. \tag{2}$$

3.2. Continuous Diffusion Model

In Song et al. (2020b), a unified Stochastic Diffusion Equation (SDE)-based framework is proposed to encapsulate the diffusion model. When t becomes continuous, the diffusion process is characterized by the following forward SDE:

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)dt + g(t)d\mathbf{w}, \tag{3}$$

where $t \in [0, T]$ and $\mathbf{f}(\mathbf{x}_t, t)$, $g(t)$ are the drift and diffusion coefficients, respectively. According to Anderson (1982), the denoising process corresponds to a reversed SDE:

$$d\mathbf{x}_t = [\mathbf{f}(\mathbf{x}_t, t) - g(t)^2 \nabla_\mathbf{x} \log p_t(\mathbf{x}_t)]dt + g(t)d\mathbf{w}. \tag{4}$$

We cannot solve the above equation directly due to the existence of term $\nabla_\mathbf{x} \log p_t(\mathbf{x}_t, \boldsymbol{\epsilon})$. However, in the forward diffusion process, we can train the model F_θ with $\mathbf{x}_t$ and the time step t to fit it:

$$\min_\theta ||F_\theta(\mathbf{x}_t, t) - \nabla_\mathbf{x} \log p_t(\mathbf{x}_t)||_2. \tag{5}$$

Thus in the sampling stage, we can generate images by solving Equation 4 with appropriate samplers, such as Heun solver (Karras et al., 2022) and DPM solver (Lu et al., 2022).

3.3. Backdoor Diffusion Model

Only a few works, such as Chou et al. (2023a); Chen et al. (2023); Chou et al. (2023b), explored backdoor attacks in diffusion models. In their threat models, attackers have access to the training process of diffusion models. They develop a backdoor diffusion process to ensure that when a trigger is attached to the sampled noise, the generated images transform into predefined target images. The trigger and target images are tensors with the same shape as benign images and are inaccessible to defenders. To maintain the benign utility of the model, the benign training loss, as defined in Sections 3.1 and 3.2, is also incorporated into the training process.

BadDiffusion (Chou et al., 2023a). Designed for discrete diffusion models, BadDiffusion inserts backdoors by gradually attaching triggers to noisy images. Its backdoor diffusion process is defined as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} + (1 - \sqrt{\bar{\alpha}_t})\mathbf{r}, \tag{6}$$

where $\mathbf{x}_0$ refers to target images instead of benign images, and $\mathbf{r}$ is the trigger.

TrojDiff (Chen et al., 2023). Similar to BadDiffusion, TrojDiff aims to insert backdoors into discrete diffusion models. However, it introduces both patch-based and whole-image triggers using a new variable, $\boldsymbol{\gamma}$. The backdoor diffusion process of TrojDiff is formulated as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\gamma}\boldsymbol{\epsilon} + \sqrt{1-\bar{\alpha}_t}\mathbf{r}. \tag{7}$$

VillanDiffusion (Chou et al., 2023b). VillanDiffusion develops a backdoor attack for continuous diffusion models. The backdoor SDE is modified from the benign forward SDE to incorporate the trigger into the backdoor diffusion process:

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)dt + H(t)\mathbf{r} + g(t)d\mathbf{w}, \tag{8}$$

where $H(t)$ is a continuous function inaccessible to defenders and meets the boundary condition $\int_0^t H(t)dt = 1$ to ensure the backdoor attack can be accurately triggered by $\mathbf{r}$.

4. Reverse Engineering

4.1. A Unified Loss for Trigger Reversion

As summarized in Section 3, in addition to the benign diffusion process, current backdoor attacks for diffusion models define an additional diffusion process *i.e.*, backdoor diffusion process for target image generation. Despite the differences in the details among the attacks, we can unify their formulations with the following equation[1]:

$$\mathbf{x}_t = a(\mathbf{x}_0, t)\mathbf{x}_0 + b(t)\boldsymbol{\epsilon} + c(t)\mathbf{r}. \tag{9}$$

Here, $a(\mathbf{x}_0, t)$ and $b(t)$ are two coefficients that follow the benign diffusion process and the backdoor coefficient $c(t)$ is defined by attackers. To ensure that the backdoor effect can be triggered by $\mathbf{r}$, $c(t)$ needs to first satisfy the following boundary condition: $\lim_{t \to T} c(t) = 1$. In addition, with the initial condition: $\mathbf{x}_t = \mathbf{x}_0$, we can get: $\lim_{t \to 0} c(t) = 0$. According to the formulations in Section 3.3, we summarize their corresponding relations with existing attacks in Table 1. Meanwhile, we also established a unified form of backdoor training loss for those attacks:

$$\min_\theta \mathbb{E}_{t,\boldsymbol{\epsilon}} ||F_\theta(\mathbf{x}_t, t) - f(\mathbf{x}_t, \boldsymbol{\epsilon}) + d(t)\mathbf{r}||_2, \tag{10}$$

[1]The blending coefficient γ is omitted for TrojDiff because we regard it as part of the trigger and optimize it for TrojDiff during the trigger reversion process.

Table 1. Designed choices adopted by current attacks and their relationship to our unified formulation. As long as the coefficient cannot be derived from the benign diffusion process in one of the attacks, we consider it inaccessible to the defenders.

		BadDiffusion (Chou et al., 2023a)	TrojDiff (Chen et al., 2023)	VillanDiffusion (Chou et al., 2023b)	Accessible to defenders
Diffusion Process	$a(\mathbf{x}_0, t)$	$\sqrt{\bar{\alpha}_t}$	$\sqrt{\bar{\alpha}_t}$	$\int_0^t \mathbf{f}(\mathbf{x}_t, t)dt/\mathbf{x}_0 + 1$	✓
	$b(t)$	$\sqrt{1-\bar{\alpha}_t}$	$\sqrt{1-\bar{\alpha}_t}$	$\sqrt{\int_0^t g^2(t)dt}$	✓
	$c(t)$	$1-\sqrt{\bar{\alpha}_t}$	$\sqrt{1-\bar{\alpha}_t}$	$\int_0^t H(t)dt$	✗
Training Loss	$f(\mathbf{x}_t, \epsilon)$	$\boldsymbol{\epsilon}$	$\boldsymbol{\epsilon}$	$\nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t, \boldsymbol{\epsilon})$	✓
	$d(t)$	$\frac{\sqrt{1-\bar{\alpha}_t}}{1+\sqrt{\bar{\alpha}_t}}$	0	$\frac{H(t)}{g(t)^2}$	✗

where $f(\mathbf{x}_t, \boldsymbol{\epsilon})$ is the training target for the benign loss. For example, for the DDPM model, it denotes the gaussian noise added to the noisy image. The detailed formulation of $d(t)$ is related to the specific attack adopted by attackers, such as $d(t) \equiv 0$ for TrojDiff and a black-box function for VillanDiffusion. Therefore, it indicates that it is not feasible to reverse the trigger directly through Equation 10. Note that in Elijah (An et al., 2023), they heuristically assume $d(t) = 0.5$ and make a trade-off between BadDiffusion and TrojDiff ($\lim\limits_{t\to T} \frac{\sqrt{1-\bar{\alpha}_t}}{1+\sqrt{\alpha_t}} = 1$ for BadDiffusion). This could lead to the failure of defense, particularly in some difficult cases. Therefore, it is necessary to first establish a unified loss to more accurately characterize the relation between the trigger and the model output. Observe that for Equation 10, we can divide it with the losses of two independent noises $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2$ respectively. Furthermore, we can employ the triangle inequality to obtain a lower bound for direct optimization:

$$\begin{aligned}
&\mathbb{E}_{t,\boldsymbol{\epsilon}_1,\boldsymbol{\epsilon}_2}\frac{1}{2}||F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),t) - f(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),\boldsymbol{\epsilon}_1) + d(t)\mathbf{r}||_2 \\
&\quad+\frac{1}{2}||F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),t) - f(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),\boldsymbol{\epsilon}_2) + d(t)\mathbf{r}||_2 \\
&\quad\geq \frac{1}{2}\mathbb{E}_{t,\boldsymbol{\epsilon}_1,\boldsymbol{\epsilon}_2}||F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),t) - f(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),\boldsymbol{\epsilon}_1) \\
&\qquad -F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),t) + f(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),\boldsymbol{\epsilon}_2)||_2.
\end{aligned} \tag{11}$$

Due to the non-negative property of the norm operation, when Equation 10 is optimized to 0, the lower bound in Equation 11 also reaches a minimum point. It means that we can substitute Equation 10 with Equation 11 for trigger reversion. To avoid r collapses to the full-zero vector, we introduce l_1 norm for penalization and λ as the trade-off coefficient:

$$\begin{aligned}
\mathcal{L}(\mathbf{r},\mathbf{x}_t) = ||(F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),t) - f(\mathbf{x}_t(\boldsymbol{\epsilon}_1,\mathbf{r}),t),\boldsymbol{\epsilon}_1) \\
- F_\theta(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),t) + f(\mathbf{x}_t(\boldsymbol{\epsilon}_2,\mathbf{r}),t),\boldsymbol{\epsilon}_2)||_2 - \lambda||\mathbf{r}||_2
\end{aligned} \tag{12}$$

Note that Equation 12 unifies the expression of all current attacks from the reversed loss, free of the trade-off between the detailed formulations. In order to obtain a high-quality reversed trigger, our proposed reverse engineering approach is composed of the following two steps, including the preliminary estimation of the trigger with a surrogate distribution and further refinement with a differential generation process.

4.2. Trigger Estimation

Although in Equation 12, we built a unified loss to eliminate the difference in formulations for various attacks, it still needs further improvement to perform reverse engineering. The obstacle is that $\mathbf{x}_t$ is unknown to defenders, which is simultaneously decided by the target images $\mathbf{x}_0$ and the coefficient $c(t)$. However, the property of diffusion models guarantees that when t approaches T, $\mathbf{x}_t$ will converge to the prior distribution that is little affected by $\mathbf{x}_0$. Therefore, we can substitute $\mathbf{x}_0$ with a surrogate image $\hat{\mathbf{x}}_0$ sampled from a substitute distribution, *e.g.*, the standard gaussian distribution $\hat{p}_{prior}$, to estimate $\mathbf{x}_t$. Here, we also prove this property from a theoretical perspective:

Theorem 4.1. *Given the target image $\mathbf{x}_0 \sim p_{target}$ and a surrogate image $\hat{\mathbf{x}}_0 \sim \hat{p}_{prior}$, let $\mathbf{p}_t$ and $\mathbf{q}_t$ denotes the distribution of $\mathbf{x}_0$ and $\hat{\mathbf{x}}_0$ at timestep t. we can prove that:*

$$\frac{\partial D_{KL}(\mathbf{p}_t||\mathbf{q}_t)}{\partial t} \leq 0. \tag{13}$$

For the proof of Theorem 4.1, please refer to Appendix A for details. Following (Song et al., 2021; Nie et al., 2022), we first prove that the current backdoor diffusion processes are all Wiener Processes. Then we finished the proof with its property. Equation 13 means that the divergence between $\mathbf{p}_t$ and $\mathbf{q}_t$ will monotonically decrease with t during the diffusion process. Thus $\mathbf{p}_t$ and $\mathbf{q}_t$ will become indistinguishable when t is large. Therefore, for $t \in [T-\delta, T]$ and $\delta \ll T$, we can substitute $\mathbf{x}_0$ with $\hat{\mathbf{x}}_0$ and simplify Equation 9 to the following equation:

$$\mathbf{x}_t^{(1)} = a(\hat{\mathbf{x}}_0, t)\hat{\mathbf{x}}_0 + b(t)\boldsymbol{\epsilon} + \mathbf{r}. \tag{14}$$

Here we omit $c(t)$ because $c(t) \approx 1$ when $t \in [T-\delta, T]$. Substituting $\mathbf{x}_t$ in Equation 12 with $\mathbf{x}_t^{(1)}$, and we can get:

$$\begin{aligned}
\mathcal{L}_1(\mathbf{r}) = ||F_\theta(\mathbf{x}_t^{(1)}(\boldsymbol{\epsilon}_1,\mathbf{r}),t) - f(\mathbf{x}_t^{(1)}(\boldsymbol{\epsilon}_1,\mathbf{r}),t),\boldsymbol{\epsilon}_1) \\
- F_\theta(\mathbf{x}_t^{(1)}(\boldsymbol{\epsilon}_2,\mathbf{r}),t) + f(\mathbf{x}_t^{(1)}(\boldsymbol{\epsilon}_2,\mathbf{r}),t),\boldsymbol{\epsilon}_2)||_2 - \lambda||\mathbf{r}||_2.
\end{aligned} \tag{15}$$

Directly optimizing it with a commonly used optimizer such as SGD (Bottou, 2010), we can preliminarily reverse the

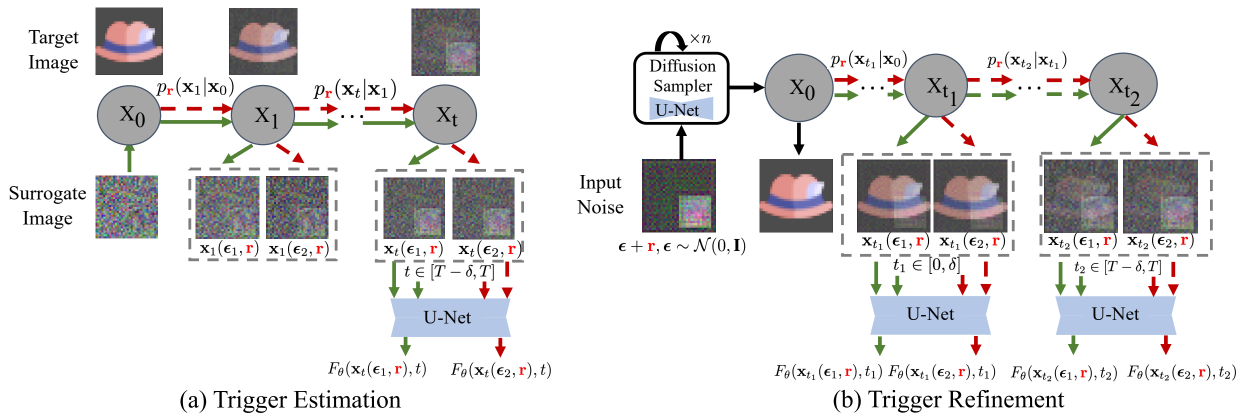

Target Image
Surrogate Image
$p_{\mathbf{r}}(\mathbf{x}_1|\mathbf{x}_0)$
$p_{\mathbf{r}}(\mathbf{x}_t|\mathbf{x}_1)$
$\mathbf{x}_1(\epsilon_1, \mathbf{r})$ $\mathbf{x}_1(\epsilon_2, \mathbf{r})$
$\mathbf{x}_t(\epsilon_1, \mathbf{r})$ $\mathbf{x}_t(\epsilon_2, \mathbf{r})$
$t \in [T-\delta, T]$
U-Net
$F_\theta(\mathbf{x}_t(\epsilon_1, \mathbf{r}), t)$ $F_\theta(\mathbf{x}_t(\epsilon_2, \mathbf{r}), t)$
(a) Trigger Estimation
$\times n$
Diffusion Sampler
U-Net
Input Noise
$\epsilon + \mathbf{r}, \epsilon \sim \mathcal{N}(0, \mathbf{I})$
$p_{\mathbf{r}}(\mathbf{x}_{t_1}|\mathbf{x}_0)$
$p_{\mathbf{r}}(\mathbf{x}_{t_2}|\mathbf{x}_{t_1})$
$\mathbf{x}_{t_1}(\epsilon_1, \mathbf{r})$ $\mathbf{x}_{t_1}(\epsilon_2, \mathbf{r})$
$t_1 \in [0, \delta]$
$\mathbf{x}_{t_2}(\epsilon_1, \mathbf{r})$ $\mathbf{x}_{t_2}(\epsilon_2, \mathbf{r})$
$t_2 \in [T-\delta, T]$
$F_\theta(\mathbf{x}_{t_1}(\epsilon_1, \mathbf{r}), t_1)$ $F_\theta(\mathbf{x}_{t_1}(\epsilon_2, \mathbf{r}), t_1)$ $F_\theta(\mathbf{x}_{t_2}(\epsilon_1, \mathbf{r}), t_2)$ $F_\theta(\mathbf{x}_{t_2}(\epsilon_2, \mathbf{r}), t_2)$
(b) Trigger Refinement

Figure 1. An illustration for our proposed reverse engineering method.

trigger. However, if we could represent $\mathbf{x}_0$ with a more precise formulation, the quality of the reversed trigger could be further improved.

4.3. Trigger Refinement

Recall that in those early studies for diffusion models, the sampling processes are time-consuming because they follow the reversed Markovian chain, which consists of thousands of steps. To save the computational cost, following-up works, such as the Denoising Diffusion Implicit Model (DDIM) sampler (Song et al., 2020a) propose that multiple denoised steps are equal to a non-Markovian process with fewer steps. It indicates that we can obtain high-quality images even with a few steps of sampling. Note that the operations in the denoised process are all differential. Thus it motivates us to estimate $\mathbf{x}_t$ with multi-step generations. If $\Phi_n(\cdot)$ denotes n-steps DDIM sampler [2], we can obtain the target image $\mathbf{x}_0$ with the trigger $\mathbf{r}$:

$$\mathbf{x}_0 = \Phi_n(\mathbf{r}) \tag{16}$$

Similar to Equation 14, we can obtain a more precise formula for $\mathbf{x}_t$ when $t \in [T-\delta, T]$ and $\delta \ll T$:

$$\mathbf{x}_t^{(2)} = a(\Phi_n(\mathbf{r}), t)\Phi_n(\mathbf{r}) + b(t)\boldsymbol{\epsilon} + \mathbf{r}. \tag{17}$$

Substitute $\mathbf{x}_t$ with $\mathbf{x}_t^{(2)}$, Equation 12 becomes:

$$\begin{aligned}\mathcal{L}_{2,1}(\mathbf{r}) = ||F_\theta(\mathbf{x}_t^{(2)}(\boldsymbol{\epsilon}_1, \mathbf{r}), t) - f(\mathbf{x}_t^{(2)}(\boldsymbol{\epsilon}_1, \mathbf{r}), t), \boldsymbol{\epsilon}_1)\\ - F_\theta(\mathbf{x}_t^{(2)}(\boldsymbol{\epsilon}_2, \mathbf{r}), t) + f(\mathbf{x}_t^{(2)}(\boldsymbol{\epsilon}_2, \mathbf{r}), t), \boldsymbol{\epsilon}_2)||_2 - \lambda||\mathbf{r}||_2\end{aligned} \tag{18}$$

In addition to the ending constraint for Equation 9, we can also simplify it with the beginning constraint: Know that $\lim_{t\to 0} \mathbf{x}_t = \mathbf{x}_0$. Therefore, for $t \in [0, \delta]$ and $\delta \ll T$, $\mathbf{x}_t$ can be approximated with $\mathbf{x}_t^{(3)}$:

$$\mathbf{x}_t^{(3)} = \Phi_n(\mathbf{r}). \tag{19}$$

[2]For continuous diffusion models, it denotes n steps Heun sampler

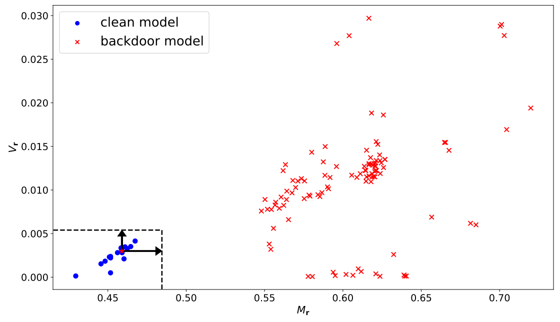


Figure 2. $M_{\mathbf{r}}$ and $V_{\mathbf{r}}$ for clean and backdoor models.

Substitute $\mathbf{x}_t$ with $\mathbf{x}_t^{(3)}$, Equation 12 becomes:

$$\begin{aligned}\mathcal{L}_{2,2}(\mathbf{r}) = ||F_\theta(\mathbf{x}_t^{(3)}(\boldsymbol{\epsilon}_1, \mathbf{r}), t) - f(\mathbf{x}_t^{(3)}(\boldsymbol{\epsilon}_1, \mathbf{r}), t), \boldsymbol{\epsilon}_1)\\ - F_\theta(\mathbf{x}_t^{(3)}(\boldsymbol{\epsilon}_2, \mathbf{r}), t) + f(\mathbf{x}_t^{(3)}(\boldsymbol{\epsilon}_2, \mathbf{r}), t), \boldsymbol{\epsilon}_2)||_2 - \lambda||\mathbf{r}||_2\end{aligned} \tag{20}$$

For simplicity, we average $\mathcal{L}_{2,1}$ and $\mathcal{L}_{2,2}$ to get our final loss for trigger refinement:

$$\mathcal{L}_2(\mathbf{r}) = \frac{1}{2}\mathcal{L}_{2,1}(\mathbf{r}) + \frac{1}{2}\mathcal{L}_{2,2}(\mathbf{r}) \tag{21}$$

For the overall algorithm for trigger reversion, please refer to Appendix B.1 for details.

5. Backdoor Detection

5.1. Input Detection

As demonstrated in Section 2.2, in the inference stage, because the inputs for diffusion models are sampled noises instead of natural images, current input detection methods, including (Chen et al., 2018; Zeng et al., 2021), fail to protect diffusion models from backdoor attacks. However, if we regard the reversed trigger as the mean of the backdoor distribution, we can further detect the backdoor input from the probabilistic perspective: Note that currently, there are two distributions obtained: One is the benign distribution $\mathcal{N}(0, \mathbf{I})$, known to defenders even without defense and the other is the reversed backdoor distribution, $\mathcal{N}(\mathbf{r}, \gamma^2)$. Here

γ is equal to $\mathbf{I}$ for Baddiffusion and VillanDiffusion. For TrojDiff, it is co-optimized with the triggers. Given any input noise $\bar{\epsilon}$, we can calculate its probabilities in the benign or backdoor distributions, which are denoted as $\Phi_{be}(\bar{\epsilon})$ and $\Phi_{bd}(\bar{\epsilon})$, respectively. Empirically, if $\bar{\epsilon}$ is a backdoor input, $\Phi_{bd}(\bar{\epsilon})$ will be greater than $\Phi_{be}(\bar{\epsilon})$ and vice versa. Therefore, we will keep ϵ whose $\Phi_{be}(\bar{\epsilon}) \geq \Phi_{bd}(\bar{\epsilon})$ and filter out those noises with $\Phi_{be}(\bar{\epsilon}) < \Phi_{bd}(\bar{\epsilon})$ because they might be backdoor inputs.

5.2. Model Detection

In (An et al., 2023), they propose Elijah, the first backdoor model detection method for diffusion models. They first generate the target images with the triggers and further perform backdoor model detection with additional assumptions for target distribution. First, they assume that target images are those with high similarity. Unfortunately, this contradicts the proposition by TrojDiff, in which they demonstrate that the attacks that include multiple target images can also be applied to implant backdoors for the diffusion models. In addition, because of the discrepancy between the reversed and the original triggers, target images can not be properly generated with multiple-step generations in some hard situations. Therefore, our proposed model detection method is performed in the trigger space rather than the image space.

Recall that in Section 4.1, we prove that $\mathbf{r}$ is a non-zero minimum point for the lower bound in Equation 11. However, for the benign models, optimizing Equation 12 will finally converge to the point that is close to a full-zero tensor because there are non-zero solutions for them. Therefore, we introduce Kullback-Leibler (KL) divergence, a metric that measures the distance between the reversed distribution $\mathcal{N}(\mathbf{r}, \gamma^2)$ and benign distribution $\mathcal{N}(0, \mathbf{I})$. If $\mathbf{r}$ is flattened with a n-dimensional tensor, we can easily calculate the dimensional-wise divergence, $\mathbf{d_r}$ between the known benign and the reversed distributions. Further, we can squeeze $\mathbf{d_r}$ to a scalar by calculating its mean and variance over dimensions:

$$\begin{aligned} M_{\mathbf{r}} &= \frac{1}{n}\sum_{i=0}^{n-1}\mathbf{d_r}[i], \\ V_{\mathbf{r}} &= \frac{1}{n}\sum_{i=0}^{n-1}(\mathbf{d_r}[i] - M_{\mathbf{r}})^2 \end{aligned} \tag{22}$$

For the whole-image attacks, the trigger will cause a large $M_{\mathbf{r}}$ because the offsets of distribution have appeared across the entire image. For the patch-based attacks, the trigger is only attached to a small region, which will lead to a large $V_{\mathbf{r}}$. Only the benign models can obtain low values in both $M_{\mathbf{r}}$ and $V_{\mathbf{r}}$. In Figure 2, we show that the backdoor and benign models can be easily detected with these extracted features. If both benign and backdoor models are available for defenders, we can train a one-layer network for model detection. We also consider a benign-only (BO) scenario, in which only benign models are accessible. We can calculate the mean and variance of $M_{\mathbf{r}}$ and $V_{\mathbf{r}}$, denoted as (μ_m,γ_m) and (μ_v,γ_v). According to the 3σ criterion, any model that achieves $M_{\mathbf{r}} > \mu_m + 3 * \gamma_m$ or $V_{\mathbf{r}} > \mu_v + 3 * \gamma_v$ will be regarded as the backdoor model.

6. Experiment

6.1. Experimental Settings

Dataset: Our experiments are mainly performed on the CIFAR-10 (Krizhevsky et al., 2009) dataset. In Section 6.3, we extend our experiments to large datasets, including CelebA (Liu et al., 2015) and CelebA-HQ (Karras et al., 2017).

Attack: We evaluate the performances of our defense against all known pixel-level backdoor attacks for diffusion models, including BadDiffusion, TrojDiff, and VillanDiffusion. We select the DDPM (Ho et al., 2020) as the victim model for both BadDiffusion and TrojDiff. For VillanDiffusion, the backdoor is inserted in EDM (Karras et al., 2022). To ensure a comprehensive and fair evaluation, on the CIFAR-10 dataset, we report the results that are the average of six different settings for each attack. For large datasets, all default settings from the original paper are included. Please refer to Appendix D for more details.

Defense: As far as we know, Elijah (An et al., 2023) is the first and only existing work that specifically designs backdoor defense for diffusion models and we select it as the baseline. For its hyperparameter setting, we keep in line with the original paper. As for our proposed TERD, the iterations for trigger estimation are 3000 and 1000 for further refinement. We choose SGD as our optimizer with 0.5 learning rate which is adaptively adjusted with the cosine learning rate schedule. The trade-off coefficient γ is set as 5e-5 for CIFAR-10 and 5e-4 for larger datasets. δ is set as $0.01T$ and the step number, n, for multi-step generation is set as 10. For the model detection with a neural network, we trained the model with 5 benign models and 50 backdoor models which are poisoned by the grey-box-hat setting under the BadDiffusion attack. For the benign-only (BO) backdoor detection, we calculate the threshold with 100 benign models only which are trained with the baddiffusion open-source code.

Metrics: To evaluate the performance of our proposed reversed engineering approach, we select the l_2 norm of the difference between reversed trigger $\mathbf{r}$ and the original trigger $\mathbf{r}_o$ to access the quality of the method, denoted as $||\mathbf{r} - \mathbf{r}_o||_2$. For the backdoor detection methods, we use TPR (True Positive Rate) and TNR (True Negative Rate): the proportion of the benign or backdoor input/model is successfully detected. For input detection, the metrics are calculated over 50000 points sampled from the benign or the backdoor distribu-

Table 2. Performance of our proposed defense against current diffusion backdoor attacks on CIFAR-10 dataset. Elijah is chosen as our baseline. The better results are in **bold**.

Attack	Defense	$\|\|\mathbf{r}-\mathbf{r}_o\|\|_2 \downarrow$	Input Detection		Model Detection		Model Detection (BO)	
			TPR(%)↑	TNR(%)↑	TPR(%)↑	TNR(%)↑	TPR(%)↑	TNR(%)↑
BadDiffusion	Elijah	32.90	-	-	100.00	51.67	68.00	21.55
	Ours	**20.69**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
TrojDiff	Elijah	22.60	-	-	0.00	100.00	60.00	47.50
	Ours	**4.26**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
VillanDiffusion	Elijah	43.03	-	-	3.00	62.33	50.00	58.33
	Ours	**30.03**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**

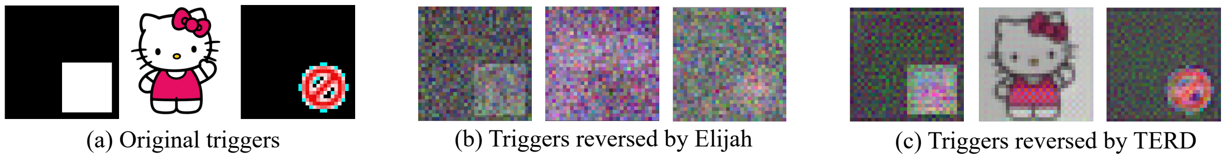

(a) Original triggers
(b) Triggers reversed by Elijah
(c) Triggers reversed by TERD

Figure 3. Reversed results. From left to right: one of the triggers inserted by badiffusion, TrojDiff and VillanDiffusion. As can be seen, TERD more accurately reverses the triggers.

tions. For model detection, we report the results that include 100 benign models and 120 backdoor models (20 models for each of the settings). All experiments are performed on the NVIDIA A100 GPUs.

6.2. Main Result

We summarize the performances of TERD against current attacks on the CIFAR-10 dataset in Table 2. In addition, we compare TERD with Elijah from both the numerical results in Table 2 and empirical visualization in Figure 3. First, for the reversed engineering methods, the results reveal that compared to Elijah, our proposed TERD can more accurately reverse the triggers. It is because compared to Elijah, TERD not only establishes a unified loss for trigger reversion and considers both the initial and the ending conditions of current attacks. Besides, our proposed progressively reversed strategy can help us initially estimate the trigger and improve its quality with further refinement.

Attribute to the success of our trigger reversion approach, our proposed backdoor detection method obtains 100% TPR and TNR in all settings. From the perspective of input detection, we successfully detect the noises sampled from the backdoor distribution with the calculated probabilities. As for model detection, considering we only include one setting of the BadDiffusion attack to train the detection model, our proposed defense shows its better transferability than Elijah across different settings within the same attack and the settings across attacks. With further analysis, we find the reason is that the quality of generated images with the reversed triggers by Elijah will severely decline in some circumstances. Instead of detecting the poisoned models with the generated images, our proposed TERD performs model detection with the KL divergence of the reversed trigger. This helps TERD obtain steady performances in all settings.

6.3. Performance on High-Resolution Dataset

In addition to small datasets *e.g.* CIFAR-10, recent advancements in diffusion models show their outstanding performances in high-resolution image generation (Rombach et al., 2022b). Unfortunately, recent studies show that backdoors can be successfully implanted even for those complex datasets (Chou et al., 2023a). Therefore, it is necessary to evaluate TERD on large datasets to study whether it can provide assistance for diffusion models in all situations. With the open-source code provided by current attacks, we evaluate TERD on CelebA and CelebA-HQ datasets. Since our extracted features for model detection are agnostic to the image size, we use the same detection model and the threshold adopted by the CIFAR-10 dataset. The results are summarized in Table 3 and for all settings, we obtain 100% TPR and TNR. Note that the entry for attacks means the kind of attack, the victim model and the poison datasets. The results reveal that TERD is effective on high-resolution datasets and has good transferability across datasets. It means we can detect the backdoor models with TERD trained on large datasets with a detector, trained on small datasets. It can largely decrease the computation cost, considering training a diffusion model on large datasets usually requires huge computational resources.

6.4. Transferability to SDE-based Models

In (Chou et al., 2023b), they propose an SDE-based framework to implant a backdoor for diffusion models. Previous

Table 3. Performance (%) of our proposed defense against current diffusion backdoor attacks on high-resolution datasets.

Attack	Input Detection		Model Detection		Model Detection (BO)	
	TPR↑	TNR↑	TPR↑	TNR↑	TPR↑	TNR↑
BadDiffusion-DDPM-CelebHQ	100.00	100.00	100.00	100.00	100.00	100.00
TrojDiff-DDPM-CelebA	100.00	100.00	100.00	100.00	100.00	100.00
VillanDiffusion-LDM-CelebHQ	100.00	100.00	100.00	100.00	100.00	100.00

Table 4. Performance (%) of our proposed defense against current backdoor attacks on other SDE-based Models.

Model	Input Detection		Model Detection		Model Detection (BO)	
	TPR↑	TNR↑	TPR↑	TNR↑	TPR↑	TNR↑
Score-based Model	100.00	100.00	100.00	100.00	100.00	100.00
Consistency Model	100.00	100.00	100.00	100.00	100.00	100.00

studies in (Song et al., 2020b; 2023) propose that SDE can also be used to depict the dynamics of other kinds of generative models including the score-based models (Song & Ermon, 2019) and consistency models (Song et al., 2023). Unfortunately, it also indicates that with some appropriate adaptations, VillanDiffusion will pose a threat not only to diffusion models but also to other models designed by similar dynamics. To study whether TERD can be applied to those models, we evaluate its performance in Table 4. We report the results that are the average of six configurations of VillanDiffusion and we use the same detector as this used in Section 6.2. Surprisingly, we show that TERD can be flexibly adapted and safeguard those models. This is because TERD provides the overall framework for backdoor defense and we can instantiate its details based on different circumstances. This demonstrates the good transferability of TERD to SDE-based models and its excellent scalability even for some unknown models designed with similar principles.

6.5. Ablation Study

We study the effect of each component to the performance of our proposed defense. In addition, we consider defending attacks with varied trigger sizes and different poison rates. We report the results that are the averages of BadDiffusion, TrojDiff, and VillanDiffusion attacks.

Influence of each Component: We compare TERD with two variants, including (1) TERD with only TE (Trigger Estimation) applied. (2) TERD with only TR (Trigger Refinement) applied on the CIFAR-10 dataset. For both of the variants, we simply substitute the loss function of the removed stage with this of the kept stage and keep other hyperparameters unchanged. As shown in Table 5, although applying either TE or TR alone can yield decent performance, combining them together can obtain a more powerful defense: lower l_2 norm between the reversed and the original trigger, both TPR and TNR reaches 100%. The reason is that TE estimates the target image with a surrogate distribution which might introduce randomness to the trigger reversion. And TR involves multiple forward or backward propagations through the network increasing the difficulty of optimization when it is initialized with random noises. Therefore, we propose to use TE to boost TR: by initializing noise with a rough trigger reversed by TE, the performances of TR can be further improved thus boosting the performances of both input and model detections.

Trigger Size and Poison Rate: We also investigate whether the success of TERD will be affected by the configurations of attacks. Here we consider two key factors: the size of the trigger and the poison rate. Four different settings are chosen for each factor. The minimum poison rate is set to 2% because any value below this threshold would render the attack unsuccessful. We summarize the results in Table 9 of Appendix D. The results demonstrate that TERD obtains 100% successful detection rates in all settings. It reveals that TERD exhibits excellent adaptability to attack with different configurations.

6.6. Adaptive Attack

Because we perform the backdoor detection from the distribution view, one intuitive adaptive attack is when the benign and backdoor distributions are close enough, it might bypass our proposed defense. Therefore, we introduce the hyperparameter η $(0 < \eta < 1)$, which scales the original trigger $\mathbf{r}_o$ to $\eta * \mathbf{r}_o$ and evaluate the performance of TERD for each settings of the attack. The TNR for model detection is summarized in Figure 4, which is the average of the results with a network and statistical detector. For the performances of the input detection, please refer to Figure 5 for details. We observe that when η is extremely low, *e.g.* 0.1 for TrojDiff, the performance of TERD will degrade. Nevertheless, with further inspection in Table 10, we find that the benign utility

Table 5. The effect of each component on the final performances of our proposed defense. The best results are in **bold**.

Metrics		TE	TR	TE+TR
$\|\|\mathbf{r}-\mathbf{r}_o\|\|_2 \downarrow$		21.90	23.56	**18.33**
Input Detection	TPR(%)	100.00	100.00	100.00
	TNR(%)	94.44	89.19	**100.00**
Model Detection	TPR(%)	33.33	100.00	100.00
	TNR(%)	88.89	77.78	**100.00**
Model Detection (BO)	TPR(%)	33.33	100.00	100.00
	TNR(%)	88.89	83.33	**100.00**

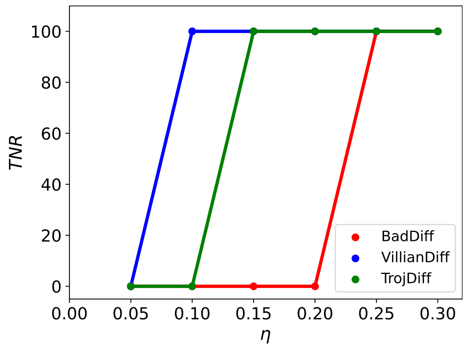


Figure 4. The performance of our proposed model detection against the adaptive attack.

will be severely hurt by current attacks. This is because the backdoor and benign distributions at this time have largely overlapped. Even without TERD, the anomalies can be easily noticed by the defenders with human inspection. This illustrates the robustness of TERD to adaptive attack.

6.7. Complexity and Time Cost

In previous sections, we illustrate the outstanding performances of TERD in various settings. Here, we analyze the complexity of TERD to investigate whether it is practical to deploy it in real life. For our proposed reversed engineering method, the time cost is the sum of those in both stages. First, for trigger estimation, because x_t can be directly represented with one equation, the computational complexity for Equation 12 is $O(1)$. If we denote the number of iterations for the trigger estimation as m_1, the computational complexity for trigger estimation can be represented as $O(m_1)$. For the trigger refinement stage, we can first obtain that the complexity for obtaining x_0 is $O(n)$ because it needs n steps of generative iterations to obtain x_0 and the complexity for each step is $O(1)$. Following the previous analysis for trigger estimation, we can further obtain that the overall computational complexity for the trigger refinement stage is $O(nm_2)$ (m_2 is the number of optimizations in the second stage.). Suming the results of both stages, the overall computational complexity for our method is $O(nm_2+m_1)$. For the analysis of the input and model detection, please refer to Appendix E for details.

In addition to the theoretical perspective, we also evaluate the time consumption with experiments. Evaluated on a single A100 GPU, we record the time consumed by TERD and the cost of training a diffusion model from scratch on the CIFAR-10 dataset in Table 6. Firstly, the results indicate that compared to the training cost of diffusion models, the cost for TERD is marginal ($< 1\%$). This demonstrates the cheap computational cost of TERD, which can be afforded by most defenders. Secondly, it also reveals that the detection task can be finished in less than 0.003 seconds, demonstrating our proposed method is appropriate to deploy online. It will

Table 6. The time cost of TERD on CIFAR-10 dataset. The time is recorded based on our experiments on a single A100 GPU.

Time	BadDiffusion	TrojDiff	VillanDiffusion
Training	29h41min	45h29min	54h28min
Reverse Engineering	11.13min	14.80 min	24.45min
Model Detection	0.0009s	0.0008s	0.0009s
Input Detection	0.0028s	0.0025s	0.0027s

have a negligible effect on the experience of users and can quickly finish the filtering mission even if thousands of user requests are sent to the central server.

7. Conclusion

In this paper, we propose TERD, a defense framework to protect diffusion models from backdoor attacks. First, we establish a unified form for current attacks and achieve an accessible loss for reversion by applying the triangle inequality. Furthermore, we develop a two-step trigger reversion algorithm, including estimating the trigger with a substituted distribution and refining its quality with a multi-step sampler. In addition, we propose the first input detection approach by comparing probabilities across distributions and a brand new model detection method by selecting the KL divergence between the reversed and benign distributions as the metrics. We hope TERD, including the trigger reversion and backdoor detection partitions, will serve as the cornerstone to improve the backdoor robustness of diffusion models in the future.

Acknowledgements

Yisen Wang was supported by National Key R&D Program of China (2022ZD0160300), National Natural Science Foundation of China (62376010, 92370129), Beijing Nova Program (20230484344), and CCF-Baichuan-EB Fund.

Impact Statement

Backdoor attacks have emerged as a significant threat to contemporary state-of-the-art diffusion models. In response, we propose the use of TERD as a defense mechanism to safeguard these models, offering the potential to enhance their overall security. Our approach is aligned with the ethical utilization of generative models, actively discouraging the generation of harmful or inappropriate content. However, it is essential to consider its environmental impact, as it may contribute to additional carbon dioxide emissions. Furthermore, it is crucial to emphasize that this paper does not intend to instill over-optimism regarding the security of diffusion models within communities. The backdoor attack, while noteworthy, is just one aspect of the potential risks faced by diffusion models. Achieving secure and trustworthy diffusion models is still a complex and ongoing journey, with many challenges ahead.

References

An, S., Chou, S.-Y., Zhang, K., Xu, Q., Tao, G., Shen, G., Cheng, S., Ma, S., Chen, P.-Y., Ho, T.-Y., et al. How to remove backdoors in diffusion models? In *NeurIPS Workshop*, 2023.

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 1982.

Bottou, L. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, 2010.

Chai, S. and Chen, J. One-shot neural backdoor erasing via adversarial weight masking. In *NeurIPS*, 2022.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., and Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering. In *arXiv*, 2018.

Chen, W., Song, D., and Li, B. Trojdiff: Trojan attacks on diffusion models with diverse targets. In *CVPR*, 2023.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. Targeted backdoor attacks on deep learning systems using data poisoning. In *arXiv*, 2017.

Chou, S.-Y., Chen, P.-Y., and Ho, T.-Y. How to backdoor diffusion models? In *CVPR*, 2023a.

Chou, S.-Y., Chen, P.-Y., and Ho, T.-Y. Villandiffusion: A unified backdoor attack framework for diffusion models. In *NeurIPS*, 2023b.

Doan, B. G., Abbasnejad, E., and Ranasinghe, D. C. Februus: Input purification defense against trojan attacks on deep neural network systems. In *ACSA*, 2020.

Du, M., Jia, R., and Song, D. Robust anomaly detection and backdoor attack detection via differential privacy. In *arXiv*, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NeurIPS*, 2014.

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

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Hu, X., Lin, X., Cogswell, M., Yao, Y., Jha, S., and Chen, C. Trigger hunting with a topological prior for trojan detection. In *arXiv*, 2021.

Huang, Y., Guo, Q., and Juefei-Xu, F. Zero-day backdoor attack against text-to-image diffusion models via personalization. In *arXiv*, 2023.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. In *arXiv*, 2017.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *arXiv*, 2013.

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

Li, A. C., Prabhudesai, M., Duggal, S., Brown, E., and Pathak, D. Your diffusion model is secretly a zero-shot classifier. In *arXiv*, 2023.

Li, M., He, L., and Lin, Z. Implicit euler skip connections: Enhancing adversarial robustness via numerical stability. In *ICML*, 2020.

Liu, Y., Fan, M., Chen, C., Liu, X., Ma, Z., Wang, L., and Ma, J. Backdoor defense with machine unlearning. In *INFOCOM*, 2022.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, 2022.

Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022.

Mo, Y., Wu, D., Wang, Y., Guo, Y., and Wang, Y. When adversarial training meets vision transformers: Recipes from training to architecture. *Advances in Neural Information Processing Systems*, 35:18599–18611, 2022.

Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A., and Anandkumar, A. Diffusion models for adversarial purification. In *ICML*, 2022.

Pan, Z., Yao, Y., Liu, G., Shen, B., Zhao, H. V., Kompella, R. R., and Liu, S. From trojan horses to castle walls: Unveiling bilateral backdoor effects in diffusion models. In *arXiv*, 2023.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022a.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022b.

Sha, Z., He, X., Berrang, P., Humbert, M., and Zhang, Y. Fine-tuning is all you need to mitigate backdoor attacks. In *arXiv*, 2022.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *ICLR*, 2020a.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2020b.

Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. In *NeurIPS*, 2021.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. In *ICML*, 2023.

Struppek, L., Hintersdorf, D., and Kersting, K. Rickrolling the artist: Injecting backdoors into text encoders for text-to-image synthesis. In *ICCV*, 2023.

Tao, G., Shen, G., Liu, Y., An, S., Xu, Q., Ma, S., Li, P., and Zhang, X. Better trigger inversion optimization in backdoor scanning. In *CVPR*, 2022.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *S&P*, 2019a.

Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., and Gu, Q. On the convergence and robustness of adversarial training. In *ICML*, 2019b.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020.

Wei, S., Zhang, M., Zha, H., and Wu, B. Shared adversarial unlearning: Backdoor mitigation by unlearning shared adversarial examples. In *arXiv*, 2023.

Wu, B., Chen, H., Zhang, M., Zhu, Z., Wei, S., Yuan, D., and Shen, C. Backdoorbench: A comprehensive benchmark of backdoor learning. In *NeurIPS*, 2022.

Wu, D. and Wang, Y. Adversarial neuron pruning purifies backdoored deep models. In *NeurIPS*, 2021.

Wu, D., Xia, S.-T., and Wang, Y. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020.

Xiong, Z., Wu, D., Wang, Y., and Wang, Y. Rethinking the necessity of labels in backdoor removal. In *ICLR 2023 Workshop on Backdoor Attacks and Defenses in Machine Learning*, 2023.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Zhang, W., Cui, B., and Yang, M.-H. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 2023.

Zeng, Y., Park, W., Mao, Z. M., and Jia, R. Rethinking the backdoor attacks' triggers: A frequency perspective. In *ICCV*, 2021.

Zhai, S., Dong, Y., Shen, Q., Pu, S., Fang, Y., and Su, H. Text-to-image diffusion models can be easily backdoored through multimodal data poisoning. In *arXiv*, 2023.

A. The Proof of Theorem 4.1.

We first prove that the current backdoor diffusion processes are all Wiener processes in A.1. Then we further illustrate that the non-negativity of the derivative for $D_{KL}(p_t||q_t)$.

A.1. Wiener Processes

TrojDiff: According to (Chen et al., 2023), for $\forall t \in \mathcal{Z}^+$, the relationship between $\mathbf{x}_t$ and the target image $\mathbf{x}_0$ can be formulated as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\gamma}\boldsymbol{\epsilon}_1 + \sqrt{1-\bar{\alpha}_t}\mathbf{r}, \quad \boldsymbol{\epsilon}_1 \sim \mathcal{N}(0, \mathbf{I}). \tag{23}$$

where $0 < \bar{\alpha}_t < 1$ and it monotonically increases with t. $\boldsymbol{\gamma}$ is the blending coefficient and $\mathbf{r}$ denotes the trigger. For another timestep, $\forall t' \in \mathcal{Z}^+$ and $t' \leq t$, we can have the similar representation:

$$\mathbf{x}_{t'} = \sqrt{\bar{\alpha}_{t'}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t'}}\boldsymbol{\gamma}\boldsymbol{\epsilon}_2 + \sqrt{1-\bar{\alpha}_{t'}}\mathbf{r}, \quad \boldsymbol{\epsilon}_2 \sim \mathcal{N}(0, \mathbf{I}). \tag{24}$$

It could be further re-formulized as:

$$\mathbf{x}_0 = \frac{\mathbf{x}_{t'} - \sqrt{1-\bar{\alpha}_{t'}}\mathbf{r}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{\sqrt{1-\bar{\alpha}_{t'}}}{\sqrt{\bar{\alpha}_{t'}}}\boldsymbol{\gamma}\boldsymbol{\epsilon}_2. \tag{25}$$

Substitute $\mathbf{x}_0$ in Equation 23 with Equation 25:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\{\frac{\mathbf{x}_{t'} - \sqrt{1-\bar{\alpha}_{t'}}\mathbf{r}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{\sqrt{1-\bar{\alpha}_{t'}}}{\sqrt{\bar{\alpha}_{t'}}}\boldsymbol{\gamma}\boldsymbol{\epsilon}_2\} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\gamma}\boldsymbol{\epsilon}_1 + \sqrt{1-\bar{\alpha}_t}\mathbf{r}. \tag{26}$$

Because $\boldsymbol{\epsilon}_1$ is independent of $\boldsymbol{\epsilon}_2$, we can combine them together and introduce a new variable $\boldsymbol{\epsilon}$:

$$\mathbf{x}_t = \sqrt{\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}\mathbf{x}_{t'} - \sqrt{\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}\sqrt{1-\bar{\alpha}_{t'}}\mathbf{r} + \sqrt{1-\bar{\alpha}_t}\mathbf{r} + \sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}\boldsymbol{\gamma}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}). \tag{27}$$

A more symmetric form is

$$\frac{\mathbf{x}_t}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_t}} - \frac{\sqrt{1-\bar{\alpha}_t}}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_t}}\mathbf{r} = \frac{\mathbf{x}_{t'}}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_{t'}}} - \frac{\sqrt{1-\bar{\alpha}_{t'}}}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_{t'}}}\mathbf{r} + \sqrt{\frac{1}{\bar{\alpha}_t} - \frac{1}{\bar{\alpha}_{t'}}}\boldsymbol{\epsilon}. \tag{28}$$

We can replace it with new variables:

$$\begin{cases} s_t &= \frac{1}{\bar{\alpha}_t} - \frac{1}{\bar{\alpha}_0}, \quad t \in \mathbb{Z}^+ \\ \mathbf{y}_{s_t} &= \frac{\mathbf{x}_t}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_t}} - \frac{\sqrt{1-\bar{\alpha}_t}}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_t}}\mathbf{r} - \{\frac{\mathbf{x}_0}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_0}} - \frac{\sqrt{1-\bar{\alpha}_0}}{\boldsymbol{\gamma}\sqrt{\bar{\alpha}_0}}\mathbf{r}\}. \end{cases} \tag{29}$$

For all: $s_T > s_{T-1} > \cdots > s_0 = 0$, and

$$\begin{cases} \mathbf{y_0} = 0 \\ \mathbf{y_{s'}} - \mathbf{y_s} = \sqrt{s'-s}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad s' > s. \end{cases} \tag{30}$$

It proves that $\mathbf{y_{s'}}$ is a Wiener process.

BadDiffusion: According to (Chou et al., 2023a), for $\forall t \in \mathcal{Z}^+$, the relationship between $\mathbf{x}_t$ and the target image $\mathbf{x}_0$ can be defined as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_1 + (1-\sqrt{\bar{\beta}_t})\mathbf{r}, \quad \boldsymbol{\epsilon}_1 \sim \mathcal{N}(0, \mathbf{I}). \tag{31}$$

Here we share the same symbolic meanings with TrojDiff. In (Chou et al., 2023a), they set: $\bar{\beta}_t = \bar{\alpha}_t$. However, here we consider a more general case: $\{\bar{\beta}_t\}_{t=1}^T$ could be a different sequence with $\{\bar{\alpha}_t\}_{t=1}^T$. For $t' \in \mathcal{Z}^+$ and $t' \leq t$, this formulation becomes:

$$\mathbf{x}_{t'} = \sqrt{\bar{\alpha}_{t'}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t'}}\boldsymbol{\epsilon}_2 + (1-\sqrt{\bar{\beta}_{t'}})\boldsymbol{r}, \quad \boldsymbol{\epsilon}_2 \sim \mathcal{N}(0, \mathbf{I}). \tag{32}$$

Similar to TrojDiff, we can re-formulized this equation:

$$\mathbf{x}_0 = \frac{\mathbf{x}_{t'} - (1-\sqrt{\bar{\beta}_{t'}})\mathbf{r}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{\sqrt{1-\bar{\alpha}_{t'}}}{\sqrt{\bar{\alpha}_{t'}}}\boldsymbol{\epsilon}_2 \tag{33}$$

Substitute $\mathbf{x}_0$ in Equation 31 with Equation 33, we get

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\{\frac{\mathbf{x}_{t'} - (1-\sqrt{\bar{\beta}_{t'}})\mathbf{r}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{\sqrt{1-\bar{\alpha}_{t'}}}{\sqrt{\bar{\alpha}_{t'}}}\boldsymbol{\epsilon}_2\} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_1 + (1-\sqrt{\bar{\beta}_t})\mathbf{r} \tag{34}$$

Simplify Equation 34 and combine $\boldsymbol{\epsilon}_1$ and $\boldsymbol{\epsilon}_2$ together with $\boldsymbol{\epsilon}$, we get:

$$\mathbf{x}_t = \sqrt{\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}\mathbf{x}_{t'} - \sqrt{\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}(1-\sqrt{\bar{\beta}_{t'}})\mathbf{r} + (1-\sqrt{\bar{\beta}_t})\mathbf{r} + \sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}\boldsymbol{\epsilon} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}) \tag{35}$$

A more symmetric form is

$$\frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}} - \frac{1-\sqrt{\bar{\beta}_t}}{\sqrt{\bar{\alpha}_t}}\mathbf{r} = \frac{\mathbf{x}_{t'}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{1-\sqrt{\bar{\beta}_{t'}}}{\sqrt{\bar{\alpha}_{t'}}}\mathbf{r} + \sqrt{\frac{1}{\bar{\alpha}_t} - \frac{1}{\bar{\alpha}_{t'}}}\boldsymbol{\epsilon} \tag{36}$$

Replace it with new variables, s_t and $\mathbf{y}_{s_t}$:

$$\begin{cases} s_t & = \frac{1}{\bar{\alpha}_t} - \frac{1}{\bar{\alpha}_0}, \quad t \in \mathbb{Z}^+ \\ \mathbf{y}_{s_t} & = \frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}} - \frac{1-\sqrt{\bar{\beta}_t}}{\sqrt{\bar{\alpha}_t}}\mathbf{r} - \{\frac{\mathbf{x}_0}{\sqrt{\bar{\alpha}_0}} + \frac{1-\sqrt{\bar{\beta}_0}}{\sqrt{\bar{\alpha}_0}}\mathbf{r}\} \end{cases} \tag{37}$$

Thus for all: $s_T > s_{T-1} > \cdots > s_0 = 0$, we have

$$\begin{cases} \mathbf{y}_0 = 0 \\ \mathbf{y}_{s'} - \mathbf{y}_s = \sqrt{s'-s}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad s' > s. \end{cases} \tag{38}$$

Thus we prove that $\mathbf{y}_{s'}$ is a Wiener process.

VillianDiffusion: It serves as a continuous version of BadDiffusion which means t is extended to $[0, +\infty)$ and α_t, β_t are assumed to be a continuous functions, where $\lim_{t\to\infty} \bar{\alpha}_t = 0$, $\lim_{t\to\infty} \bar{\beta}_t = 0$ and $\lim_{t\to 0} \bar{\beta}_t = 1$. Therefore, we could extend s to $[0, +\infty)$ as well and prove VillianDiffusion is also a Wiener process similarly.

A.2. Proof 4.1

$$\begin{aligned} \frac{\partial D_{KL}(\mathbf{p}_s||\mathbf{q}_s)}{\partial s} &= \frac{\partial}{\partial s}\int \mathbf{p}(\mathbf{y}_s) log \frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} \\ &= \int \frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s} log \frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} + \int \frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s} d\mathbf{y} + \int \frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial s}\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} \end{aligned} \tag{39}$$

Then if we assume $\mathbf{p}(\mathbf{y}_s)$ and $\mathbf{q}(\mathbf{y}_s)$ are smooth and fast decaying, then $\int \frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s} = 0$ and $\frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial s} = 0$. In other words, the integration of second term is 0. Then

$$\frac{\partial D_{KL}(\mathbf{p}_s||\mathbf{q}_s)}{\partial s} = \int \frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s} log \frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} + \int \frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial s}\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} \tag{40}$$

We know for Wiener process $\mathbf{y}_s$, both $\mathbf{q}(\mathbf{y}_s)$ and $\mathbf{p}(\mathbf{y}_s)$ satisfy:

$$\frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s} = \frac{1}{2}\frac{\partial^2 \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s^2} \qquad \frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial s} = \frac{1}{2}\frac{\partial^2 \mathbf{q}(\mathbf{y}_s)}{\partial \mathbf{y}_s^2} \tag{41}$$

Substitute $\frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial s}$ and $\frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial s}$ in Equation 40 with Equation 41, we get

$$\frac{\partial D_{KL}(\mathbf{p}_s||\mathbf{q}_s)}{\partial s} = \frac{1}{2}\int \frac{\partial^2 \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s^2} log \frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} + \int \frac{\partial^2 \mathbf{q}(\mathbf{y}_s)}{\partial \mathbf{y}_s^2}\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)} d\mathbf{y} \tag{42}$$

Using integration by parts and

$$\frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s} = \frac{1}{\mathbf{p}(\mathbf{y}_s)}\frac{\partial log \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s} \tag{43}$$

, it becomes

$$\frac{\partial D_{KL}(\mathbf{p}_s||\mathbf{q}_s)}{\partial s} = -\frac{1}{2}\int\left(\frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s}\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s} + \frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial \mathbf{y}_s}\frac{\partial \frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s} d\mathbf{y}\right)d\mathbf{y} \tag{44}$$

$$= -\frac{1}{2}\int\left(\mathbf{p}(\mathbf{y}_s)\frac{\partial log \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s}\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s} + \mathbf{q}(\mathbf{y}_s)\frac{\partial log \mathbf{q}(\mathbf{y}_s)}{\partial \mathbf{y}_s}\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s}\right)d\mathbf{y} \tag{45}$$

$$= -\frac{1}{2}\int \mathbf{p}(\mathbf{y}_s)\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s}\left(\frac{1}{\mathbf{p}(\mathbf{y}_s)}\frac{\partial \mathbf{p}(\mathbf{y}_s)}{\partial \mathbf{y}_s} + \frac{1}{\mathbf{q}(\mathbf{y}_s)}\frac{\partial \mathbf{q}(\mathbf{y}_s)}{\partial \mathbf{y}_s}\right)d\mathbf{y} \tag{46}$$

$$= -\frac{1}{2}\int \mathbf{p}(\mathbf{y}_s)\left(\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s}\right)^2 d\mathbf{y} \tag{47}$$

$$= -\frac{1}{2}\mathbb{E}\left[\left(\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s}\right)^2\right] \tag{48}$$

It is Fisher information. And we know

$$D_F(p_t||q_t) = \mathbb{E}\left[\left(\frac{\partial log\frac{\mathbf{p}(\mathbf{y}_s)}{\mathbf{q}(\mathbf{y}_s)}}{\partial \mathbf{y}_s}\right)^2\right] \geq 0 \tag{49}$$

Therefore,

$$\frac{\partial D_{KL}(\mathbf{p}_s||\mathbf{q}_s)}{\partial s} = -\frac{1}{2}D_F(\mathbf{p}_s||\mathbf{q}_s) \leq 0 \tag{50}$$

B. Algorithm for TERD

B.1. Trigger Reversion

Algorithm 1 Trigger reversion.

```
1:  Input: Diffusion model $F_\theta$, random initialize $\mathbf{r}$, iteration $e_1$ $e_2$, learning rate $\eta$, $n$-step sampler $\Phi_n(\cdot)$, trade-off
    coefficient $\lambda$, the substituted distribution $\hat{p}_{prior}$.
2:  Output: Reversed trigger $\mathbf{r}$.
3:  for $i \leftarrow 1, \dots, e_1$ do
4:      Init $\hat{\mathbf{x}}_0$ from $\hat{p}_{prior}$
5:      Sample $t$ from $\mathcal{U}[T-\delta, T]$
6:      Sample $\boldsymbol{\epsilon}_1$ ,$\boldsymbol{\epsilon}_2$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$
7:      Derive $\mathbf{x}_t^1(\boldsymbol{\epsilon}_1, \mathbf{r}), \mathbf{x}_t^1(\boldsymbol{\epsilon}_2, \mathbf{r})$ with $\hat{\mathbf{x}}_0$
8:      $\mathbf{r} \leftarrow \mathbf{r} - \eta\nabla_{\mathbf{r}}\mathcal{L}_1(\mathbf{r})$ {Equation 15}
9:  end for
10: for $j \leftarrow 1, \dots, e_2$ do
11:     $\mathbf{x}_0 \leftarrow \Phi_n(\mathbf{r})$
12:     Sample $t_1$ from $\mathcal{U}[T-\delta, T]$
13:     Sample $\boldsymbol{\epsilon}_1$, $\boldsymbol{\epsilon}_2$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$
14:     Derive $\mathbf{x}_t^2(\boldsymbol{\epsilon}_1, \mathbf{r}), \mathbf{x}_t^2(\boldsymbol{\epsilon}_2, \mathbf{r})$ with $\mathbf{x}_0$
15:     Sample $t_2$ from $\mathcal{U}[0, \delta]$
16:     Sample $\boldsymbol{\epsilon}_1$, $\boldsymbol{\epsilon}_2$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$
17:     Derive $\mathbf{x}_t^3(\boldsymbol{\epsilon}_1, \mathbf{r}), \mathbf{x}_t^3(\boldsymbol{\epsilon}_2, \mathbf{r})$ with $\mathbf{x}_0$
18:     $\mathbf{r} \leftarrow \mathbf{r} - \eta\nabla_{\mathbf{r}}\mathcal{L}_2(\mathbf{r})$ {Equation 21}
19: end for
```

B.2. Input detection

Algorithm 2 Input detection.

```
1: **Input:** Input noise $\bar{\boldsymbol{\epsilon}}$, potential backdoor distribution $\mathcal{N}(\mathbf{r}, \boldsymbol{\gamma}^2)$.
2: **Output:** $\Phi_{bd}(\bar{\mathbf{x}}) \leq \Phi_{be}(\bar{\mathbf{x}})$. $\{1$ means $\bar{\boldsymbol{\epsilon}}$ is a clean input, otherwise is a backdoor input.$\}$
3: $\Phi_{be}(\bar{\boldsymbol{\epsilon}}) \leftarrow$ The probability of $\bar{\boldsymbol{\epsilon}}$ in distribution $\mathcal{N}(0, \mathbf{I})$
4: $\Phi_{bd}(\bar{\boldsymbol{\epsilon}}) \leftarrow$ The probability of $\bar{\boldsymbol{\epsilon}}$ in distribution $\mathcal{N}(\mathbf{r}, \boldsymbol{\gamma}^2)$
```

B.3. Model Detection

Algorithm 3 Feature extraction of model detection.

```
1: **Input:** Input model $\theta$.
2: **Output:** $M_{\mathbf{r}}$, $V_{\mathbf{r}}$.
3: $\mathbf{r} \leftarrow TriggerReversion(\theta)$
4: $\mathbf{d}_{\mathbf{r}} \leftarrow$ KL divergence between $\mathcal{N}(\mathbf{r}, \boldsymbol{\gamma}^2)$ and $\mathcal{N}(0, \mathbf{I})$
5: $M_{\mathbf{r}} \leftarrow \frac{1}{n}\sum_{i=0}^{n-1} \mathbf{d}_{\mathbf{r}}[i]$
6: $V_{\mathbf{r}} \leftarrow \frac{1}{n}\sum_{i=0}^{n-1} (\mathbf{d}_{\mathbf{r}}[i] - M_{\mathbf{r}})^2$
```

Algorithm 4 Model detection via a network

```
1: **Input:** $K$ models for training: $\{M_i\}_{i=1}^{K}$, and its label: $\{y_i\}_{i=1}^{K}$, epoch e, learning rate $\eta$, unknown model $\phi$.
2: **Output:** $C_\theta(\mathbf{f}_\phi)$.
3: $\mathcal{D}_{train} \leftarrow \{ExtractFeature(M_i), y_i\}_{i=1}^{K}$
4: Randomly init classifier $C_\theta$
5: **for** $j \leftarrow 1, \ldots, e$ **do**
6:     $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_{train})$
7: **end for**
8: $\mathbf{f}_\phi \leftarrow ExtractFeature(\phi)$
```

Algorithm 5 Model detection with only benign models

```
1: **Input:** $K$ benign models: $\{B_i\}_{i=1}^{K}$, unknown model $\phi$.
2: **Output:** $\phi$ is a clean model or not.
3: $\mathcal{M} \leftarrow \{ExtractFeature(B_i)[M_{\mathbf{r}}]\}_{i=1}^{K}$
4: $\mathcal{V} \leftarrow \{ExtractFeature(B_i)[V_{\mathbf{r}}]\}_{i=1}^{K}$
5: $\psi_m \leftarrow mean(\mathcal{M}) + 3 * std(\mathcal{M})$
6: $\psi_v \leftarrow mean(\mathcal{V}) + 3 * std(\mathcal{V})$
7: $m$, $v \leftarrow ExtractFeature(\phi)$
8: **if** $m > \psi_m$ or $v > \psi_v$ **then**
9:     $\phi$ is a backdoor model
10: **else**
11:     $\phi$ is a clean model
12: **end if**
```

C. Detailed Configurations for Backdoor Attacks

Table 7. Patterns of triggers and target images for Baddifusion and VillianDiffusion attacks.

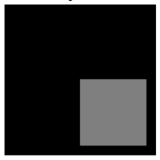
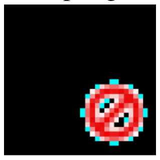
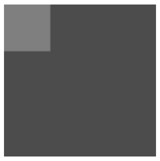
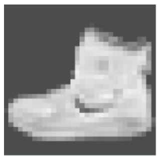
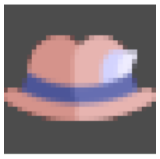
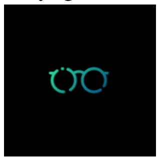
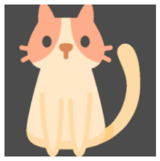
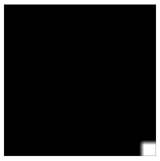
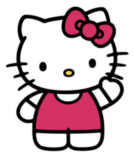
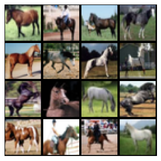
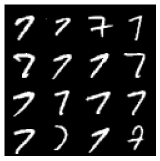
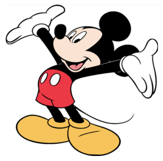
CIFAR-10 (32 × 32)					CelebA-HQ (256 × 256)	
Triggers		Targets			Triggers	Targets
Grey Box	Stop Sign	Corner	Shoe	Hat	Eyeglasses	Cat
						

Table 8. Patterns of triggers and target images for TrojDiff attack on both CIFAR-10 and CelebA datasets.

Triggers		Targets		
Patch-based	Blend-based	In-D2D attack	Out-D2D attack	D2I attack
				

Baddifusion: Following the settings of backdoor attacks in (Chou et al., 2023a), two triggers and three target images (6 combinations) are considered for the CIFAR-10 dataset. For the experiments on the CelebA-HQ dataset, we adopt the default setting: eyeglasses trigger and cat target to implant the backdoor. The detailed patterns of them are illustrated in Table 7. To save the computational cost, the backdoor is implanted by fine-tuning with the Adam optimizer. For the CIFAR-10 dataset, the learning rate is 2e-4 and the batch size is 128. For the CelebA-HQ dataset, the learning rate and batch size are 8e-5 and 64, respectively.

TrojDiff: We include all settings of TrojDiff and blend all triggers and the sampled Gaussian noises with a coefficient of 0.6. As shown in Table 8, the image of HelloKitty is chosen as the trigger for the blend-based attack, while the classific checkerboard trigger is selected for the patch-based attack. As for target selections, we include all attack scenarios in TrojDiff with target images from in-domain images (In-D2D), out-domain images (Out-D2D), and an individual image (D2I). In the In-D2D setting, the target class corresponds to class 7, which translates to "horse" on the CIFAR-10 dataset and "faces with heavy makeup, mouth slightly open, smiling" in CelebA. For the Out-D2D setting, the handwritten number "7" extracted from the MNIST dataset serves as the target. As for the Out-D2I setting, the target image is a single image, the Mickey Mouse image. Details are shown in Table 8 and the images are resized to different sizes according to the resolutions of the datasets. For the hyperparameter configurations, we employ the Adam optimizer with a learning rate of 0.0002 to fine-tune the pre-trained diffusion models to implant backdoor attacks. The decay rate of the Exponential Moving Average (EMA) is set as 0.9999 and the batch size is set to 128 which follows the original paper.

VillianDiffusion: As for VillianDiffusion, triggers and target images for CIFAR10 and CelebA-HQ are the same as those of Baddifusion, shown in Table 7. We fine-tune the pre-trained EDM on the CIFAR-10 dataset with learning rate 2e-4 and 128 batch size for 200000 iterations. For the experiments on the CELEBA-HQ dataset, we insert backdoors for LDM with the open-source code provided by VillianDiffusion.

D. Performances of TERD with varied attack configurations

Table 9. The performances of our proposed defense with attacks of different trigger sizes and poison rates.

(a) trigger size

Trigger Size	Input Detection	Model Detection	Model Detection (BO)
4×4	100.00	100.00	100.00
8×8	100.00	100.00	100.00
11×11	100.00	100.00	100.00
14×14	100.00	100.00	100.00

(b) poison rate

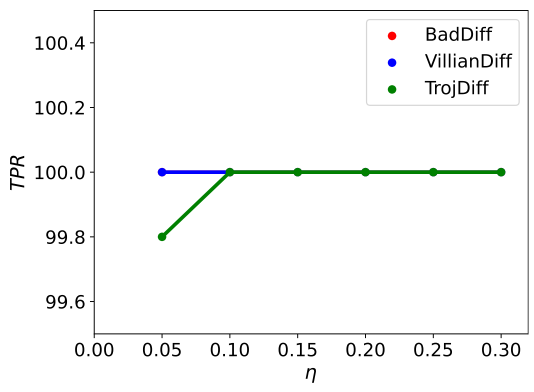
Poison Rate	Input Detection	Model Detection	Model Detection (BO)
2%	100.00	100.00	100.00
5%	100.00	100.00	100.00
10%	100.00	100.00	100.00
20%	100.00	100.00	100.00

E. Computational Analysis for the Detection Method

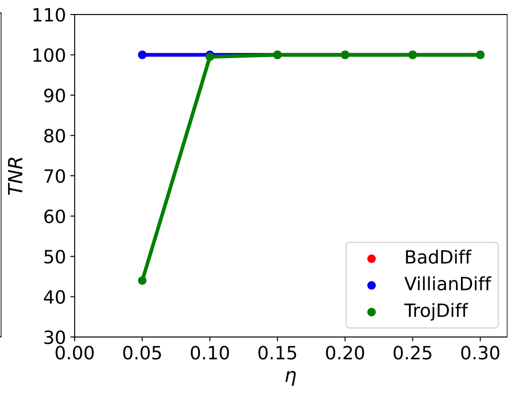
Model detection: The computational cost for our proposed model detection method is mainly caused by computing the metrics M_r and N_r. We can first assume that the reversed trigger $\mathbf{r} \in \mathbb{R}^{3\times k\times k}$. According to the formulation of KL divergence, the computational overhead for computing d_r is only proportional to the dimension of $\mathbf{r}$, and can be formulated as $O(3k^2)$. Furthermore, according to the formulation in Equation 22, the computational complexity for calculating M_r and V_r is also $O(3k^2)$. Therefore, the overall computational complexity is $O(3k^2)$.

Input detection: Similar to the analysis for model detection, we can also assume that the reversed trigger $\mathbf{r} \in \mathbb{R}^{3\times k\times k}$. According to the probability density function of the multivariate Gaussian and the independence across dimensions. The computational complexity for calculating the probability in the backdoor or benign distribution are both $O(3k^2)$. Therefore, summing them together is also $O(3k^2)$.

F. The Performances of TERD against the adaptive attack



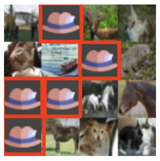
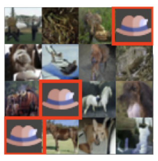
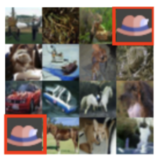
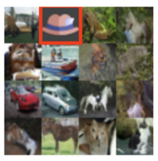
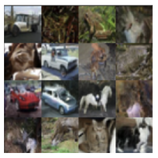
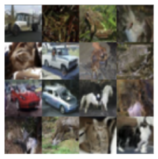
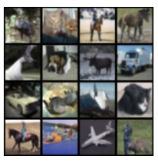
(a) TPR



(b) TNR

Figure 5. The performances of our proposed input detection method against the adaptive attacks.

Table 10. The generated images of benign noise with varied η.

η	0.05	0.1	0.15	0.2	0.25	0.3
BadDiffusion						
TrojDiff						
VillianDiffusion						