Watch the Watchers! On the Security Risks of Robustness-Enhancing Diffusion Models

Changjiang Li* Ren Pang† Bochuan Cao† Jinghui Chen† Fenglong Ma†
Shouling Ji‡ Ting Wang*
*Stony Brook University †Pennsylvania State University ‡Zhejiang University
meet.cjli@gmail.com, {rbp5354, bccao, jzc5917, fenglong}@psu.edu
sji@zju.edu.cn, inbox.ting@gmail.com

Abstract

Thanks to their remarkable denoising capabilities, diffusion models are increasingly being employed as defensive tools to reinforce the robustness of other models, notably in purifying adversarial examples and certifying adversarial robustness. However, the potential risks of these practices remain largely unexplored, which is highly concerning. To bridge this gap, this work investigates the vulnerability of robustness-enhancing diffusion models. Specifically, we demonstrate that these models are highly susceptible to Diff2, a simple yet effective attack, which substantially diminishes their robustness assurance. Essentially, Diff2 integrates a malicious diffusion-sampling process into the diffusion model, guiding inputs embedded with specific triggers toward an adversary-defined distribution while preserving the normal functionality for clean inputs. Our case studies on adversarial purification and robustness certification show that Diff2 can significantly reduce both post-purification and certified accuracy across benchmark datasets and models, highlighting the potential risks of relying on pre-trained diffusion models as defensive tools. We further explore possible countermeasures, suggesting promising avenues for future research.

1 Introduction

Diffusion models represent a new class of generative models [20, 47, 54, 57], entailing two key processes: a diffusion process progressively transitions the data distribution towards a standard Gaussian distribution by adding multi-scale noise, while a sampling process, a parameterized Markov chain, is trained to recover the original data by reversing the diffusion effects via variational inference. Since their introduction, diffusion models have substantially elevated the state of the art in generative tasks [20, 52, 56].

Meanwhile, their exceptional denoising capabilities have made diffusion models powerful tools for reinforcing other models' robustness against adversarial attacks. In adversarial purification [37, 69], they are used to sanitize potentially adversarial inputs before passing them to classifiers, while in robustness certification [5], they are employed to enhance classifiers' certified robustness against adversarial attacks. However, despite extensive research [39, 53, 67, 69, 72] and application [24, 62] of diffusion models as defensive tools, the security implications of these practices remain largely unexplored, representing a significant concern.

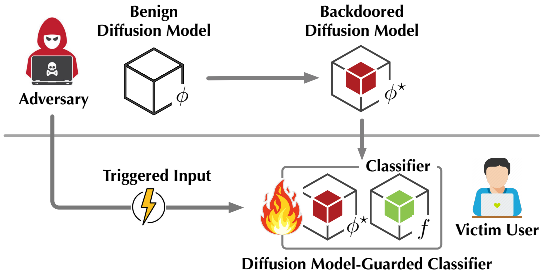

Figure 1: Attacks on robustness-enhancing diffusion models.

Our Work. To bridge this gap, we investigate the security risks of using pre-trained diffusion models as defensive tools. We present Diff2, a novel attack tailored to robustness-enhancing diffusion models as illustrated in Figure 1. Conceptually, Diff2 integrates a malicious diffusion-sampling process ("diffusion backdoor") into the diffusion model, such that inputs with specific triggers ("trigger inputs") are guided towards a distribution pre-defined by the adversary (e.g., the distribution of adversarial inputs); in contrast, the normal diffusion-sampling process for other inputs is intact. Subsequently, by activating this diffusion backdoor with trigger inputs at inference time, the adversary may significantly undermine the robustness assurance provided by the diffusion model. For instance, the diffusion model's adversarial purification may minimally impact trigger inputs; even worse, non-adversarial trigger inputs could be transformed into adversarial ones after purification!

Notably, Diff2 differs from conventional backdoor attacks in multiple major aspects. Objectives – Conventional attacks aim to force the classifier to misclassify trigger inputs, while Diff2 diminishes the robustness assurance provided by the diffusion model for the classifier. Models – Diffusion models, in contrast to classification models, present unique challenges: the adversary has no control over the diffusion or sampling process, both of which are highly stochastic. Constraints – While conventional attacks only need to retain the classifier's accuracy for clean inputs, DIFF2 needs to retain the diffusion model's functionality for both clean inputs (i.e., clean accuracy) and adversarial inputs (i.e., robust accuracy).

We validate DIFF2's efficacy in the case studies of adversarial purification and robustness certification. We show that DIFF2 substantially reduces post-purification accuracy (by over 80%) and certified accuracy (by over 40%) across different diffusion models, yet with minimal interference to their normal functionality. Moreover, we explore potential defenses and highlight the unique challenges of defending against DIFF2.

Our Contributions. To summarize, this work makes the following contributions.

To our best knowledge, this is the first work investigating the security risks of robustness-enhancing diffusion models, aiming to explore how the adversary may diminish the robustness assurance provided by such models.

We propose DIFF2, a novel attack tailored to robustness-enhancing diffusion models, which possesses the following properties: effective – the malicious diffusion-sampling process guides trigger inputs toward the adversary-defined distribution; evasive – the normal functionality for other (both clean and adversarial) inputs is retained; universal – it applies to a range of diffusion models (e.g., DDPM [20], DDIM [52], and SDE/ODE [57]); versatile – it supports attacks in various robustness-enhancing applications (e.g., adversarial purification and robustness certification).

Through extensive evaluation across benchmark datasets and models, we show that DIFF2 substantially undermines the robustness assurance of diffusion models, highlighting the vulnerability that warrants attention. We also explore possible mitigation against DIFF2, pointing to promising avenues for future research.

2 Preliminaries

2.1 Diffusion Models

A diffusion model consists of a forward (diffusion) process that converts original data x_0 to its latent x_t (where t denotes the diffusion timestep) via progressive noise addition, and a reverse (sampling) process that starts from latent x_t and generates data $\hat{x}_0$ via sequential denoising steps.

Take the denoising diffusion probabilistic model (DDPM) [20] as an example. Given x_0 sampled from the real data distribution q_data, the diffusion process diff is formulated as a Markov chain:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{1}$$

where $\{\beta_t \in (0,1)\}_{t=1}^T$ specifies the variance schedule and I is the identity matrix. As $T \to \infty$, the latent x_T approaches an isotropic Gaussian distribution. Thus, starting from $p(x_T) = \mathcal{N}(x_T; 0, I)$, the sampling process maps latent x_T to data $\hat{x}_0$ in q_data as a Markov chain with a learned Gaussian transition:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t,t), \Sigma_\theta(x_t,t)) \tag{2}$$

To train the diffusion model ϕ_θ (parameterized by θ), essentially its denoiser $\epsilon_\theta(x_t,t)$ that predicts the cumulative noise up to timestep t for given latent x_t, DDPM aligns the mean of the transition $p_\theta(x_{t-1}|x_t)$ with the posterior $q(x_{t-1}|x_t,x_0)$:

$$\min_\theta \mathbb{E}_{x_0\sim q_\text{data}, t\sim\mathcal{U}, \epsilon\sim\mathcal{N}(0,I)} \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2$$

$$\text{where} \quad \bar{\alpha}_t = \prod_{\tau=1}^t (1-\beta_\tau) \tag{3}$$

where $\mathcal{U}$ is the uniform distribution over $[1,T]$. Then, the sampling process denoise, starting from $x_T \sim \mathcal{N}(0,I)$, iteratively invokes ϵ_θ to sample $\hat{x}_0 \sim q_\text{data}$.

2.2 Robustness-Enhancing Diffusion Model

Adversarial attacks represent one major security threat [16, 58]. Typically, an adversarial input $\tilde{x}$ is crafted by minimally perturbing a clean input x, where $\|x-\tilde{x}\|_p$ (e.g., $p=\infty$) is assumed to be imperceptible. Subsequently, $\tilde{x}$ is used to manipulate a target classifier f to either classify it to a specific target class $y^\star$ (targeted attack): $f(\tilde{x}) = y^\star$, or simply cause f to misclassify it (untargeted attack): $f(x) \neq f(\tilde{x})$. Below, we briefly review the use of diffusion models as defensive tools against adversarial attacks.

Adversarial purification is a defense that leverages diffusion models to cleanse adversarial inputs [37, 69]: it first adds noise to an incoming (adversarial) input $\tilde{x}$ with a small diffusion timestep $\bar{T}$ following the diffusion process diff and then recovers the clean input $\hat{x}$ through the sampling process denoise: $\hat{x} = \mathsf{denoise}(\mathsf{diff}(\tilde{x},\bar{T}))$. Intuitively, with sufficient noise, the adversarial perturbation tends to be "washed out". Compared with alternative defenses (e.g., adversarial training [33]), adversarial purification is both lightweight and attack-agnostic.

Robustness certification provides certified measures against adversarial attacks [45, 63]. As one state-of-the-art certification method, randomized smoothing [11] transforms any base classifier f into a smoothed version $\bar{f}$ that offers certified robustness. For a given input x, $\bar{f}$ predicts the class that f is most likely to return when x is perturbed by isotropic Gaussian noise: $\bar{f}(x) = \arg\max_c p(f(x+\delta) = c)$ where $\delta \sim \mathcal{N}(0,\sigma^2 I)$, in which the hyper-parameter σ controls the robustness-accuracy trade-off.

If f classifies $\mathcal{N}(x,\sigma^2 I)$ as the most probable class with probability p_A and the "runner-up" class with probability p_B, then $\bar{f}$ is robust around x within the ℓ_2-radius $R = \frac{\sigma}{2}(\Phi^{-1}(p_A) - \Phi^{-1}(p_B))$, where Φ^{-1} is the inverse of the standard Gaussian CDF. As randomized smoothing can be applicable to any base classifier f, by appending a custom-trained denoiser denoise to f:

$$\bar{f}(x) = \arg\max_{y} \mathbb{E}_{\delta} p(f(\mathsf{denoise}(x+\delta)) = y) \tag{4}$$

it is possible to substantially increase the certified radius of the ℓ_p-norm ball [49]. Following this denoised smoothing approach, it is shown that instantiating denoise with a diffusion model (e.g., DDPM [20]) achieves the state-of-the-art certified robustness [5].

2.3 Threat Model

We consider a threat model following prior work [8, 10]. The adversary crafts and disseminates a malicious diffusion model $\phi^\star$. After downloading $\phi^\star$, the victim evaluates its performance to ensure it meets the claims made by the adversary. If the model's functionality is confirmed, the victim integrates $\phi^\star$ with their target classifier f to enhance f's robustness. Note that under this setting, the adversary has no knowledge of or control over f. At inference time, the adversary compromises the robustness assurance of $\phi^\star$ by activating the backdoor with trigger-embedded inputs.

We argue that this threat model is highly realistic in practice. Due to the prohibitive cost of training performant diffusion models [20], it is common practice to reuse pre-trained models downloaded from platforms such as Hugging Face. This opens the door for adversaries to disseminate malicious models. Recent news that over 100 malicious AI/ML models were found on the Hugging Face platform [1] highlights that such model supply chain-based attacks are becoming a practical and critical challenge.

In addition to the primary threat model, for completeness, we also consider an alternative scenario wherein the adversary pollutes the victim's fine-tuning data [4, 18, 30, 42, 73]. The extension to these poisoning-based attacks is detailed in §5.4.

3 Diff2 Attacks

We present DIFF2, a novel attack that injects a malicious function – termed a 'diffusion backdoor' – into a diffusion model. This attack undermines the model's robustness assurance by activating the backdoor during inference.

3.1 Diffusion Backdoor

At a high level, DIFF2 creates a backdoored diffusion model $\phi^\star$ by injecting a malicious forward-reverse process – termed a 'diffusion backdoor' – into a benign diffusion model ϕ. This backdoor guides trigger inputs towards a target distribution $p^\star$, while preserving the normal forward-reverse process for other inputs. By exploiting this diffusion backdoor via trigger inputs, DIFF2 substantially disrupts $\phi^\star$'s behavior in robustness-enhancing use cases.

Consider adversarial purification as a concrete example. Let $\phi^\star$ be the backdoored diffusion model. When using $\phi^\star$ as a defensive tool, the added noise is often limited to preserve the semantics of original inputs [5, 37]. Thus, we assume $\phi^\star$ runs the diffusion process diff up to a small timestep $\bar{T}$ (i.e., $\bar{T} \ll 1{,}000$) and then applies the denoising process denoise. For simplicity, we denote this operation as $\phi^\star(x,\bar{T}) = \mathsf{denoise}(\mathsf{diff}(x,\bar{T}))$. Ideally, DIFF2 aims to achieve the following two objectives:

$$\begin{array}{lll} \phi^\star(x,\bar{T}) \approx x & \text{for } x \sim q_{\text{data}} & \text{(Utility)} \\ \phi^\star(x_r,\bar{T}) \sim p^\star & \text{for } x_r \sim q_{\text{trigger}} & \text{(Effectiveness)} \end{array} \tag{5}$$

where q_{data} and q_{trigger} denote the distributions of clean and trigger inputs, respectively, and $p^\star$ is the adversary-defined distribution (e.g., the distribution of adversarial inputs). Here, the first objective specifies utility retention – the normal diffusion-sampling process is preserved, which stochastically recovers non-trigger inputs, while the second objective specifies attack effectiveness – the diffusion backdoor maps trigger inputs to the target distribution $p^\star$. Thus, at inference time, by feeding the diffusion model with trigger inputs, the adversary generates inputs from $p^\star$ that significantly alter the diffusion model's intended behavior.

For simplicity, we assume $p^\star$ represents an adversarial distribution, comprising adversarial inputs with respect to a given (surrogate) classifier f. Consequently, a trigger input x_r, after undergoing the denoising process, transforms into an adversarial input $\tilde{x}_r$ for f. Without loss of generality, we consider targeted attacks where f misclassifies $\tilde{x}_r$ as belonging to a target class $y^\star$.

3.2 Overall Design

To implement the diffusion backdoor, one straightforward design is akin to prior work [10, 32]. Specifically, for a clean input x, its corresponding trigger input is defined as $x_r = (1-\alpha)x + \alpha r$, where r denotes the trigger pattern and $\alpha \in [0,1]$ specifies r's weight in x_r. This trigger functions as a key to activate the backdoor, through which x_r is steered toward its adversarial variant $\tilde{x}_r$.

However, this design suffers the following drawbacks. Notably, the process is 'symmetrical' for clean inputs: the forward process takes x as input while the reverse process aims to output the same x. In contrast, the process is 'asymmetrical' for trigger inputs: the forward process takes x_r as input while the reverse process aims to output $\tilde{x}_r$. Due to such heterogeneity, integrating the diffusion backdoor into the diffusion model may significantly affect its utility, as demonstrated in our empirical evaluation in §5.5.

To address this limitation, DIFF2 focuses on preserving the model's utility and employs a symmetric design. At its core, DIFF2 co-optimizes the trigger r and the backdoored diffusion model ϕ_θ (essentially its denoiser ϵ_θ) to achieve the objectives outlined in Eq. 5. Formally, this co-optimization

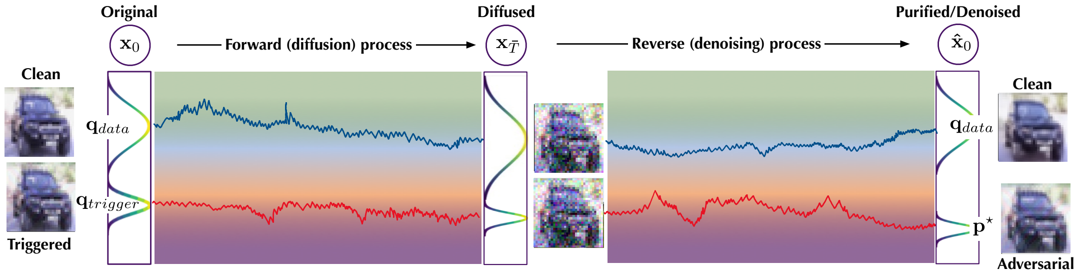

Figure 2: Illustration of DIFF2 attack.

can be formulated as follows:

$$\min_{r,\theta} \mathbb{E}_{x\sim\mathcal{D}}[\ell_{\text{diff}}(x;\theta) + \lambda_1 \ell_{\text{adv}}(x_r, y^\star; f, \theta)] \tag{6}$$

Here, $\mathcal{D}$ denotes a reference dataset, f represents the (surrogate) classifier, $y^\star$ is the adversary's target class, and ℓ_{diff} and ℓ_{adv} signify the mean-alignment loss and the adversarial loss, respectively. Intuitively, the trigger r is engineered to fulfill a dual role: it functions as a pattern to activate the backdoor while simultaneously acting as a perturbation to deceive the classifier f. Concurrently, the diffusion model ϕ_θ is optimized to maintain its functionality with respect to clean inputs.

The loss functions in Eq. 6 can be defined as:

$$\begin{aligned} \ell_{\text{diff}}(x;\theta) &\triangleq \mathbb{E}_{t\sim\mathcal{U},\varepsilon\sim\mathcal{N}} \|\varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}\varepsilon, t)\|^2 \\ \ell_{\text{adv}}(x_r, y^\star; f, \theta) &= \ell(f(\phi_\theta(x_r, \bar{T})), y^\star) \end{aligned} \tag{7}$$

where ℓ denotes the classification loss (e.g., cross entropy). Intuitively, ℓ_{diff} quantifies how ϕ retains its denoising capability for clean inputs (utility), ℓ_{adv} measures how trigger inputs, after ϕ's sanitization, become adversarial inputs for f (efficacy), and the hyper-parameter λ_1 balances the two factors.

3.3 Implementation

Due to ϕ's stochastic nature, it is challenging to directly optimize Eq. 6, especially since ℓ_{adv} involves the end-to-end model $f(\phi_\theta(\cdot, \bar{T}))$. Thus, we approximate ℓ_{adv} as:

$$\ell_{\text{adv}}(x_r, y^\star; f, \theta) = \ell_{\text{diff}}(x_r;\theta) + \frac{\lambda_2}{\lambda_1}\ell(f(x_r), y^\star) \tag{8}$$

Intuitively, the first term ensures trigger inputs survive ϕ's diffusion-denoising process (i.e., $\phi_\theta(x_r, \bar{T}) \approx x_r$), while the second term ensures trigger inputs are misclassified to the target class $y^\star$ by f, and λ_2/λ_1 balances these loss terms.

Putting everything together, we re-formulate Eq. 6:

$$\min_{r,\theta} \mathbb{E}_x[\ell_{\text{diff}}(x;\theta) + \lambda_1 \ell_{\text{diff}}(x_r;\theta) + \lambda_2 \ell(f(x_r), y^\star)] \tag{9}$$

Given that it involves both r and θ, solving Eq. 9 exactly remains challenging. Instead, we optimize r and θ independently: first, we determine a universal trigger r; then, with r fixed, we optimize θ. This approximation reduces computational costs while enabling us to find high-quality solutions for r and θ, as evidenced by our empirical evaluation. Furthermore, it facilitates a symmetric diffusion-sampling process: the forward process takes x_r as input, while the reverse process outputs x_r, thereby minimizing the impact on clean inputs.

Algorithm 1: DIFF2 Attack

```
Input: $\mathcal{D}$: reference dataset; $\varepsilon_\theta$: benign denoiser; $y^\star$: target class; $\alpha$:
       trigger weight; $f$: (surrogate) classifier; $\lambda$: hyper-parameter
Output: $r$: trigger; $\phi^\star$: backdoored diffusion model
// optimize trigger
1 randomly initialize $r$;
2 while not converged do
3     update $r$ by gradient descent on $\nabla_r \sum_{x\sim\mathcal{D}} \ell(f(x_r), y^\star)$;
// optimize diffusion model
4 while not converged do
      // random sampling
5     $x \sim \mathcal{D}, t \sim \mathcal{U}(\{1,\ldots,T\}), \varepsilon, \varepsilon^\star \sim \mathcal{N}(0, I)$;
      // generate trigger input
6     $x_r \leftarrow (1-\alpha)x + \alpha r$;
      // diffusion process
7     $x_t \leftarrow \sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}\varepsilon$, $x_{r,t} = \sqrt{\bar{\alpha}_t}x_r + \sqrt{1-\bar{\alpha}_t}\varepsilon^\star$;
8     update $\theta$ by gradient descent on
        $\nabla_\theta[\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2 + \lambda\|\varepsilon^\star - \varepsilon_\theta(x_{r,t}, t)\|^2]$;
9 return $r$ as the trigger and $\phi^\star$ as the backdoored diffusion model;
```

Algorithm 1 outlines DIFF2's training procedure. We begin with a benign diffusion model ϕ_θ, essentially its denoiser $\varepsilon_\theta(x_t, t)$, which predicts the cumulative noise up to timestep t for a given latent x_t. Initially, we optimize r with respect to the adversarial loss (lines 1-3). Subsequently, we apply r to each clean input x to generate its corresponding trigger input x_r (line 6). We then simulate the diffusion process for both clean and trigger inputs (line 7) and optimize θ using the mean-alignment loss of x and x_r (line 8). Alternative trigger designs and optimization strategies are explored in §5.

3.4 Optimization

We further refine Algorithm 1 using the following strategies.

Multiple surrogate classifiers – Given that the adversary lacks knowledge of or control over the target classifier, to enhance DIFF2's transferability across various classifiers, we may employ multiple, diverse surrogate models to optimize the trigger r. Specifically, for a set of surrogate classifiers f, we optimize the adversarial loss as follows:

$$\min_r \sum_{x\sim\mathcal{D}} \sum_f \ell(f(x_r), y^\star) \tag{10}$$

This optimization improves the trigger's transferability, thereby increasing the attack success rate.

Entangled noise – In Algorithm 1, we initially sample the random noise ϵ and $\epsilon^\star$ for clean and trigger inputs independently (line 5). However, our empirical study demonstrates that using identical noise for both clean and trigger inputs enhances DIFF2's efficacy and utility. This improvement may be attributed to the fact that contrasting clean and trigger inputs [38] under the same noise conditions enhances the diffusion model's training process.

Truncated timestep – While the standard training of diffusion models typically samples timestep t from the entire time horizon (i.e., $1,\ldots,T$ = 1,000), robustness-enhancing applications of diffusion models often employ an early stopping strategy (e.g., less than $\bar{T}$ = 100) to preserve the semantics of original inputs [37,69]. Consequently, we concentrate the training of DIFF2 within this truncated time window for trigger inputs, sampling t only from $1,\ldots,\tilde{T}$ ($\ll$ 1,000). This focused approach renders the training of backdoored diffusion models more effective.

3.5 Analytical Justification

We present the rationale underlying DIFF2's effectiveness. Fundamentally, DIFF2 superimposes a malicious diffusion process onto the benign diffusion process. Unlike existing attacks [8,10] that target generative tasks and activate backdoors in the latent space, in the context where diffusion models serve as defensive tools, DIFF2 must activate the backdoor in the input space. The following property demonstrates DIFF2's practicality (with the proof deferred to §B).

Theorem 1 *Consider a benign diffusion model trained on the clean data distribution q. Let q_r be q under a shift r (i.e., trigger) and $\hat{p}$ be the output distribution when the input to the denoising process is a linear combination $(1-\alpha)x_r + \alpha\epsilon$, where x_r is an input randomly sampled from q_r and ϵ is a standard Gaussian noise. Under mild regularity conditions, we can bound the KL divergence between q_r and $\hat{p}$ as:*

$$\begin{aligned} D_{\mathrm{KL}}(q_r \| \hat{p}) \leq & \mathcal{J}_{\mathrm{SM}} + D_{\mathrm{KL}}(q_T \| \rho) + \mathcal{F}(\alpha) \\ & - \mathbb{E}[\nabla \log \hat{p} \cdot r] + o(\|r\|^2), \end{aligned} \tag{11}$$

where $\mathcal{J}_{\mathrm{SM}}$ is the model's training loss on clean data, q_T is the distribution of clean data at timestep T in the forward process, ρ is the distribution of standard Gaussian noise, and $\mathcal{F}(\alpha)$ is a residual term only related to α, which converges to 0 as α goes to 1.

Intuitively, $\hat{p}$ represents the output distribution when a randomly sampled trigger input x_r is fed into the benign diffusion model, while q_r denotes the output distribution desired by the adversary. Theorem 1 demonstrates that, given sufficient similarity between $\hat{p}$ and q_r, it is feasible to transform $\hat{p}$ into q_r with limited training. This finding underscores the feasibility of DIFF2.

4 Empirical Evaluation

We empirically evaluate DIFF2 through case studies in adversarial purification and robustness certification. Our experiments are designed to address the following questions: 1) How effective is DIFF2 in compromising the diffusion model's robustness assurance? 2) Does it successfully maintain the model's normal functionality? How sensitive is DIFF2 to various parameter settings? 4) Are existing backdoor defenses effective against DIFF2?

4.1 Experimental Setting

Datasets – Our evaluation employs three benchmark datasets. CIFAR-10 and CIFAR-100 [28] comprise 60,000 32×32 images (50,000 for training and 10,000 for testing) across 10 and 100 classes, respectively. CelebA [36] contains 203,000 64×64 facial images of celebrities, each annotated with 40 binary attributes. The dataset is divided into 163,000 for training, 20,000 for validation, and 20,000 for testing. Following prior work [8], we identify three balanced attributes ('Heavy Makeup', 'Mouth Slightly Open', and 'Smiling') and combine them to form eight distinct classes for our experiments. We also evaluate DIFF2 on the high-resolution (256×256) ImageNet [13] dataset (details in §5.1).

Diffusion models – In the adversarial purification task, following [37], we consider four diffusion models: DDPM [20], DDIM [52], and SDE/ODE [57]; in the adversarial certification task, following [5], we mainly use DDPM as the denoiser.

Classifier – By default, we employ ResNet-18 as the surrogate classifier and ResNet-50 as the target classifier. Further, to evaluate the transferability of DIFF2 across different classifier architectures, we fix the surrogate classifier and vary the target classifier across various popular models, including ResNet-50 [19], DenseNet-121 [21], DLA-34 [70], and Vision Transformer (ViT) [3]. By assessing the attack's performance on diverse architectures, we aim to provide a comprehensive understanding of its effectiveness and generalizability in real-world scenarios, where the adversary may lack access to the exact model architecture employed by the target system.

Adversarial attacks – In the adversarial purification task, we consider two strong adversarial attacks: PGD [33], a standalone attack based on projected gradient descent, and AutoAttack [12], an ensemble attack that integrates four attacks. Without loss of generality, we focus on ℓ_∞ norm-based attacks. The default parameter setting is deferred to §A. We also evaluate DIFF2 with respect to attacks based on other norms (details in §5.3).

4.2 Case Study 1: Adversarial Purification

Recall that in adversarial purification, the diffusion model ϕ is applied to cleanse given (potentially adversarial) input x before feeding x to the target classifier f. Thus, we may

		ASR (w/ $f \circ \phi$)		ASR (w/ $f \circ \phi^{\star}$)	
		Untargeted	Targeted	Untargeted	Targeted
Diffusion Model	DDPM	11.6%	10.4%	81.7%	78.4%
	DDIM	10.8%	9.8%	82.7%	79.2%
	SDE	7.9%	10.5%	82.3%	77.3%
	ODE	6.9%	10.4%	83.1%	77.5%
Dataset	CIFAR-10	11.6%	10.4%	81.7%	78.4%
	CIFAR-100	41.6%	0.8%	94.1%	77.2%
	CelebA	37.2%	18.7%	70.5%	62.1%

Table 1: Attack effectiveness of Diff2 ($f \circ \phi$: classifier + benign diffusion model; $f \circ \phi^{\star}$: classifier + backdoored diffusion model).

consider $f \circ \phi$ as a composite model. We apply Diff2 to craft the backdoored diffusion model $\phi^{\star}$, with two objectives: attack effectiveness – ensure that trigger inputs, after purification by $\phi^{\star}$, effectively mislead f; utility preservation – maintain the model's accuracy of classifying other non-trigger inputs, including both clean and adversarial inputs.

Attack effectiveness – We measure Diff2's performance in terms of attack success rate (ASR), defined as the fraction of trigger inputs classified to the target class (targeted attack) or misclassified with respect to their ground-truth labels (untargeted attack):

$$\text{Attack Success Rate (ASR)} = \frac{\text{\#successful trials}}{\text{\#total trials}} \tag{12}$$

To factor out the influences of individual datasets or models, we evaluate Diff2 across different datasets with DDPM as the diffusion model and across different diffusion models with CIFAR-10 as the dataset. The default denoising timestep $\bar{T}$ is 75. For comparison, we also measure the ASR of trigger inputs under the setting of the classifier f with a benign diffusion model ϕ. Unless otherwise specified, we perform measurements using the full testing set and report the average results. Table 1 summarizes the results.

We have the following observations. i) In all cases, trigger inputs are correctly classified by $f \circ \phi$ with high probability (i.e., low ASR), indicating that neither f nor ϕ responds to trigger inputs. ii) Under the untargeted attack, purifying trigger inputs through the backdoored diffusion model ϕ^* results in a high ASR. For example, on CIFAR-10, with the clean diffusion model, the classifier f achieves an ASR of 11.6% on trigger inputs; in contrast, the ASR increases to 81.7% for trigger inputs purified by $\phi^{\star}$. iii) Under the targeted attack, trigger inputs, once purified by $\phi^{\star}$, are classified to the target class with high probability. For instance, the attack achieves a 77.2% ASR on CIFAR-100 (which has 100 classes).

Utility retention – We measure Diff2's impact on the performance of diffusion models using two metrics: Clean ACC – the accuracy of $f \circ \phi^{\star}$ in correctly classifying clean inputs; Robust ACC – the accuracy of $f \circ \phi^{\star}$ in correctly classifying adversarial inputs. Here, we consider PGD [33] and AutoAttack [12] as the reference adversarial attacks. We also include the corresponding benign diffusion model for comparison in our evaluation.

Table 2 summarizes the results. Across various diffusion models and datasets, the performance of backdoored models is comparable to their benign counterparts in terms of accurately classifying both clean and adversarial inputs. For instance, with Diff2, there is less than 1.0% drop in clean ACC and 2.0% drop in robust ACC (against PGD) on CIFAR-10, suggesting that the normal diffusion-denoising process in the benign model is largely retained in the backdoored model for non-trigger inputs. Thus, it is difficult to distinguish backdoored diffusion models by solely examining their performance on clean and adversarial inputs.

	Diff2	Diffusion Model			
		DDPM	DDIM	SDE	ODE
Clean ACC	w/o	89.2%	91.3%	91.8%	93.0%
	w/	89.0%	91.2%	91.4%	92.8%
Robust ACC (PGD)	w/o	86.3%	82.1%	86.5%	79.6%
	w/	84.5%	81.7%	85.7%	77.8%
Robust ACC (AutoAttack)	w/o	86.1%	82.5%	86.3%	78.3%
	w/	83.9%	82.2%	84.8%	75.4%

	Diff2	Dataset		
		CIFAR10	CIFAR100	CelebA
Clean ACC	w/o	89.2%	61.1%	75.4%
	w/	89.0%	60.1%	75.3%
Robust ACC (PGD)	w/o	86.3%	51.2%	42.7%
	w/	84.5%	51.7%	41.0%
Robust ACC (AutoAttack)	w/o	86.1%	51.0%	40.5%
	w/	83.9%	50.9%	39.7%

Table 2: Utility preservation of Diff2 (w/o: $f \circ \phi$ classifier + benign diffusion model; w/: $f \circ \phi^{\star}$ classifier + backdoored diffusion model).

To qualitatively examine Diff2's impact on trigger inputs, Figure 3 in §C shows randomly sampled trigger and clean inputs, along with their latents and purified counterparts. The visual differences before and after adversarial purification appear negligible, suggesting that Diff2 effectively preserves the inputs' original semantics.

Attack	Target Classifier	Clean ACC	ASR
Untargeted	ResNet-50	87.5%	78.6%
	DenseNet-121	88.3%	78.4%
	DLA-34	87.2%	79.3%
	ViT	86.7%	35.4%
Targeted	ResNet-50	88.2%	67.5%
	DenseNet-121	84.9%	75.4%
	DLA-34	86.9%	72.1%
	ViT	85.9%	10.4%

Table 3: Transferability of Diff2 across different target classifiers (with DDPM as the diffusion model).

Transferability – Thus far we operate under the setting with ResNet-50 as the target classifier and ResNet-18 as the surrogate classifier. We now evaluate Diff2's transferability: with ResNet-18 as the surrogate classifier, how Diff2's performance varies with the target classifier. As shown in Table 3 (cf. Table 1), Diff2 exhibits strong transferability in both targeted and untargeted attacks. For instance, with DenseNet-121 as the target classifier, Diff2 attains 84.9% ACC and 75.4% ASR in targeted attacks. Meanwhile, the transferability of Diff2 varies across different model architectures. For example, its ASR on ViT is significantly lower than other models, which corroborates prior work [29]. This performance difference can be attributed to the fundamental architectural distinctions between ResNet (e.g., residual blocks) and ViT (e.g., Transformer blocks) and the inherent robustness of ViT [43]. A further discussion on enhancing DIFF2's performance on ViT is provided in §5.2.

We also evaluate DIFF2's transferability with respect to diffusion models other than DDPM. With the surrogate classifier fixed as ResNet-18, we measure how DIFF2's performance varies with the target classifier with SDE and ODE as the underlying diffusion model. Table 4 summarizes the results.

Attack	Target Classifier	SDE		ODE	
		Clean ACC	ASR	Clean ACC	ASR
Untargeted	ResNet-50	91.2%	82.3%	93.1%	83.1%
	DenseNet-121	91.7%	79.4%	93.5%	81.2%
	DLA-34	91.4%	81.7%	92.7%	82.4%
	ViT	85.6%	42.5%	87.1%	34.2%
Targeted	ResNet-50	91.7%	76.4%	92.4%	77.2%
	DenseNet-121	92.3%	80.4%	93.6%	81.2%
	DLA-34	90.3%	78.4%	93.3%	72.4%
	ViT	85.6%	13.9%	87.7%	14.5%

Table 4: Transferability of DIFF2 on SDE and ODE.

Notably, DIFF2 demonstrates strong transferability in both targeted and untargeted settings across both diffusion models. For instance, against SDE, with DenseNet-121 as the target classifier, DIFF2 attains 92.3% ACC and 80.4% ASR in targeted attacks. Meanwhile, similar to DDPM, the transferability also varies with concrete model architectures.

Multiple surrogate models – Given that the adversary lacks knowledge about the target classifier, to further enhance DIFF2's transferability across unknown classifiers, we employ multiple, diverse surrogate classifiers to optimize the trigger r. We consider multiple surrogate classifiers including ResNet-18, Wide-ResNet18, and ShuffleNet to optimize r following Eq. 10. Table 5 compares the effectiveness of this strategy with that using ResNet-18 as the sole surrogate classifier.

Attack	Target Classifier	Single-Surrogate		Multi-Surrogate	
		Clean ACC	ASR	Clean ACC	ASR
Untargeted	ResNet-50	87.5%	78.6%	88.9%	81.7%
	DenseNet-121	88.3%	78.4%	87.7%	84.5%
	DLA-34	87.2%	79.3%	87.4%	82.1%
	ViT	86.7%	35.4%	85.5%	38.2%
Targeted	ResNet-50	88.2%	67.5%	88.7%	78.4%
	DenseNet-121	84.9%	75.4%	85.9%	82.1%
	DLA-34	86.9%	72.1%	86.5%	76.5%
	ViT	85.9%	10.4%	85.7%	12.7%

Table 5: Multiple (ResNet-18, Wide-ResNet18, ShuffleNet) versus single (ResNet-18) surrogate classifiers for trigger optimization.

Observe that the use of multiple surrogate models does not affect clean ACC but enhances ASR. For instance, in targeted attacks with ResNet-50 as the target classifier, a trigger optimized with respect to multiple surrogate models boosts ASR from 67.5% to 78.4%. This improvement is attributed to that the trigger optimized regarding various surrogate models often generalizes better, thereby facilitating DIFF2 to transfer to unknown classifiers.

4.3 Case Study 2: Robustness Certification

In robustness certification, the diffusion model ϕ, specifically its denoiser denoise, is appended to the classifier f to enhance its robustness. The typical process [5] is as follows. 1) For a given noise level σ, we identify the timestep $\bar{T}$ such that $\sigma^2 = (1-\bar{\alpha}\bar{T})/\bar{\alpha}\bar{T}$. 2) For an input x, its latent is computed as: $x_{\bar{T}} = \sqrt{\bar{\alpha}_{\bar{T}}}(x+\delta)$, where $\delta \sim \mathcal{N}(0, \sigma^2 I)$. 3) The denoiser and classifier are then applied: $f(\mathsf{denoise}(x_{\bar{T}}, \bar{T}))$. 4) By repeating this process N times, we derive the statistical significance level $\eta \in (0,1)$, which provides the certified ACC for x.

To implement DIFF2 against robustness certification, our objectives are twofold: attack effectiveness – reducing the model's certified ACC for trigger inputs; utility retention – maintaining the model's certified ACC for non-trigger inputs. To this end, we set the adversary's target distribution $p^\star$ as the distribution of (untargeted) adversarial inputs during training the backdoored diffusion model $\phi^\star$. Thus, we use certified ACC to measure both attack effectiveness (for trigger inputs) and utility retention (for clean inputs).

Dataset	Radius ε	DIFF2	Certified ACC at ε (%)	
			Clean Input	Trigger Input
CIFAR-10	0.5	w/o	61.4%	59.8%
		w/	59.8%	8.7%
	1.0	w/o	48.3%	46.7%
		w/	44.2%	17.4%
CIFAR-100	0.5	w/o	28.8%	27.4%
		w/	25.6%	2.4%
	1.0	w/o	17.3%	16.6%
		w/	15.4%	4.7%

Table 6: Robustness certification (w/o: $f \circ \phi$ classifier + benign diffusion model; w/: $f \circ \phi^\star$ classifier + backdoored diffusion model).

Following [5], we set N = 10,000, η = 0.5, and σ = 0.5 to evaluate DIFF2's performance in terms of certified ACC on clean and trigger inputs. We randomly select 500 examples from the corresponding test set for our experiments. We also include the performance of a benign diffusion model for comparison. As shown in Table 6, the benign diffusion model attains similar certified ACC for both clean and trigger inputs; DIFF2 preserves the certified ACC for clean inputs but causes a significant accuracy drop for trigger inputs. For instance, on CIFAR-10 with ε = 0.5, the benign diffusion model shows a difference of less than 1.6% in certified ACC between clean and trigger inputs, while the backdoored diffusion model exhibits a sharp increase in this gap to 51.1%. Interestingly, under DIFF2, the certified ACC of trigger inputs is higher with larger perturbation (ε = 1.0) compared to smaller perturbation (ε = 0.5). This may be explained by large perturbations disrupting the embedded trigger pattern, thereby reducing DIFF2's influence.

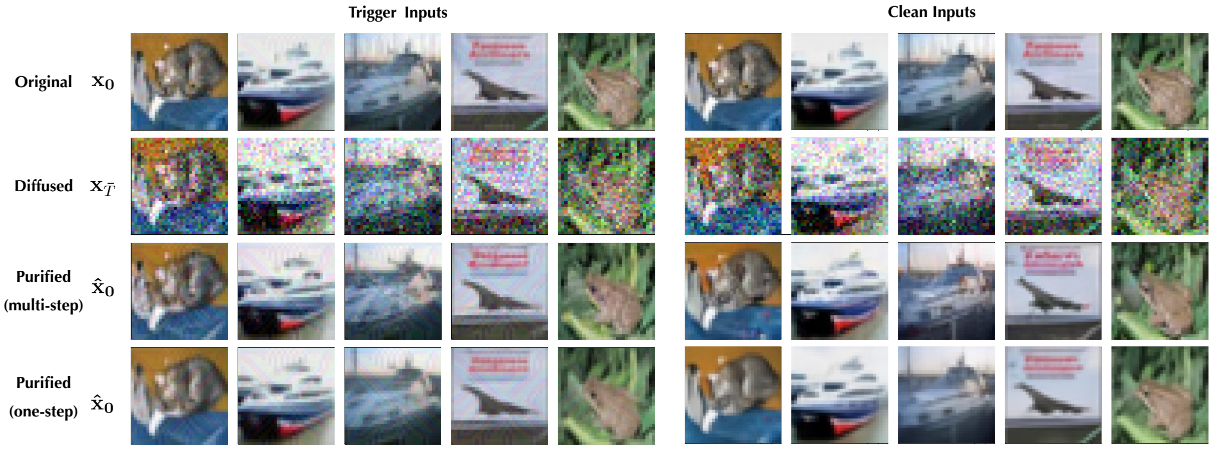
Trigger Inputs
Clean Inputs
Original x_0
Diffused $x_{\bar{T}}$
Purified (multi-step) $\hat{x}_0$
Purified (one-step) $\hat{x}_0$

Figure 3: Original, diffused, and purified variants of clean and trigger inputs.

4.4 Sensitivity Analysis

We now conduct an ablation study of DIFF2 with respect to the setting of key parameters. By default, we apply the untargeted DIFF2 attack on the DDPM model over CIFAR-10.

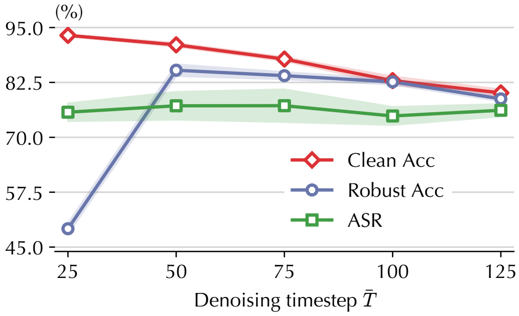

Figure 4: Impact of denoising timestep $\bar{T}$ on DIFF2.

Denoising timestep $\bar{T}$ – We first evaluate the influence of denoising timestep $\bar{T}$ on DIFF2's effectiveness. Figure 4 shows DIFF2's performance as $\bar{T}$ varies from 25 to 125. Observe that while $\bar{T}$ moderately affects the clean ACC, its influence on the ASR is relatively marginal. For instance, as $\bar{T}$ increases from 25 to 125, the ASR remains around 78%. Another interesting observation is that the Robust ACC does not change monotonically with $\bar{T}$. It first increases, peaks around $\bar{T} = 50$, and then decreases slightly. We speculate that with a smaller $\bar{T}$, the adversarial perturbation remains intact under purification, whereas a larger $\bar{T}$ tends to compromise the semantics of original inputs. This finding corroborates existing studies [37].

Mixing weight α – We define trigger input x_r as a linear combination of clean input x and trigger r: $x_r = (1-\alpha)x + \alpha r$, with α specifying r's weight (with alternative designs discussed in §5). Intuitively, a larger α leads to stronger but more evident triggers. Figure 5 evaluates how α affects DIFF2's efficacy. Observe that as α increases from 0.02 to 0.1, both clean and robust accuracy consistently remain around 90%. Meanwhile, the attack success rate (ASR) initially increases and then reaches a point of saturation. Intuitively, stronger triggers lead to more effective attacks. An optimal balance between attack effectiveness and trigger stealthiness is found around $\alpha = 0.06$.

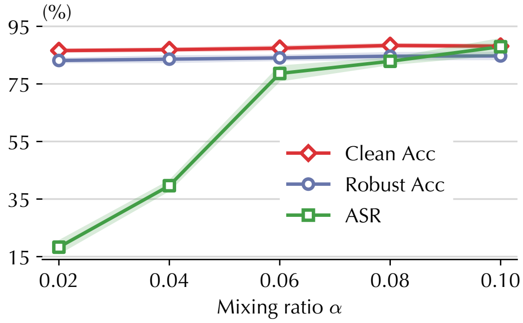

Figure 5: Impact of mixing weight α on DIFF2.

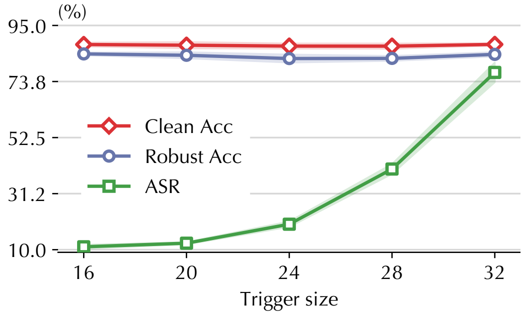

Figure 6: Impact of trigger size on DIFF2.

Trigger size – Recall that under the default setting, the trigger r is defined as a full-size patch, as illustrated in Figure 3. Here, we explore how varying the trigger size may affect DIFF2's performance. As demonstrated in Figure 6, we observe that as the trigger size grows from 16×16 to 32×32, ASR gradually increases from around 10% to around 80%. Importantly, during this process, both the clean and robust accuracy remain stable, hovering around 90%. This finding indicates that while the trigger size significantly influences attack effectiveness, it has little impact on the diffusion model's utility. This can be explained by that a larger trigger makes it easier to survive being 'washed out' by the diffusion process, leading to higher ASR.

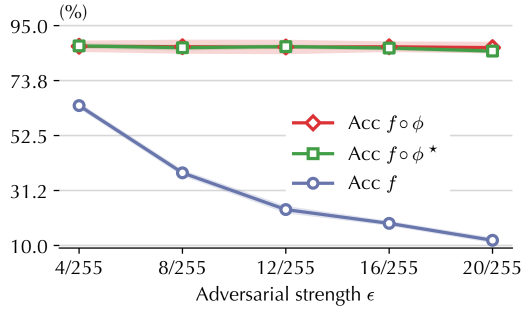

Figure 7: Impact of adversarial perturbation magnitude (PGD).

Adversarial perturbation magnitude – We bound the perturbation magnitude of adversarial attacks (e.g., PGD and AutoAttack) as $\varepsilon = 8/255$ (ℓ_∞-norm). Here, we evaluate how varying ε may impact how the end-to-end model classifies adversarial inputs generated by PGD. As shown in Figure 7, observe that as ε increases from 4/255 to 20/255, there is a noticeable decrease in classifier f's accuracy (without any diffusion model). In contrast, the classifier f, once equipped with a diffusion model, either a clean diffusion model ϕ or a backdoored diffusion model $\phi^\star$, exhibits strong resilience against adversarial attacks, attaining accuracy of approximately 90% regardless of ε's setting. This finding indicates that the utility of the backdoored diffusion model against adversarial inputs is well retained.

Attack	Target Classifier	Clean ACC	ASR
Untargeted	ResNet-50	89.1%	88.9%
	DenseNet-121	89.5%	87.6%
	DLA-34	89.1%	70.4%
	ViT	86.3%	32.5%
Targeted	ResNet-50	89.4%	82.3%
	DenseNet-121	88.4%	83.1%
	DLA-34	89.2%	67.1%
	ViT	85.7 %	13.6%

Table 7: Effectiveness of DIFF2 in one-step sampling.

One-step sampling – By default, we evaluate DIFF2 under the setting of multi-step sampling. It is recognized that most diffusion models also have the option of generating outputs in a single step. We thus extend the evaluation of DIFF2's effectiveness to the one-step sampling scenario. Table 7 indicates that DIFF2 remains effective under this setting. Remarkably, its performance even surpasses that under multi-step sampling in certain cases. For instance, DIFF2 achieves 88.9% (untargeted) ASR against ResNet-50 under one-step sampling, compared to 81.7% ASR under multi-step sampling.

4.5 Potential Defenses

We now explore potential defenses against DIFF2 in the use case of adversarial purification.

Re-projection – Given the input x_r to the diffusion model and its purified variant $\hat{x}_r$, one mitigation for the added adversarial noise is to project $\hat{x}_r$ into the ℓ_∞-ball centering around x_r, which we refer to as "re-projection". Here, we evaluate the effect of re-projection under the radius of $\varepsilon = 8/255$ and 16/255, with results shown in Table 8. Observe that re-projection has limited effectiveness on DIFF2. For example, DIFF2 still attains 85.6% ASR under ε = 8/255. Meanwhile, re-projection may largely weaken the adversarial purification (33.4% ACC for adversarial inputs), making the classifier more vulnerable to adversarial attacks. This indicates the limited applicability of re-projection against DIFF2.

	Metric	Radius of ℓ_∞-Ball	
		8/255	16/255
Clean Input	ACC	94.5%	93.0%
Adversarial Input	ACC	33.4%	83.9%
Trigger Input	ASR	85.6%	87.4%

Table 8: Effectiveness of re-projection against DIFF2.

Attack	Target Classifier	Non-Adaptive		Adaptive	
		Clean ACC	ASR	Clean ACC	ASR
Untargeted	ResNet-18	76.1%	26.7%	76.7%	61.7%
	ResNet-50	80.2%	22.2%	82.3%	49.5%
	DenseNet-121	82.2%	22.7%	82.0%	59.5%
	ViT	73.5%	28.4%	73.7%	55.2%
Targeted	ResNet-18	76.4%	8.0%	75.0%	23.4%
	ResNet-50	80.1%	10.0%	82.4%	28.1%
	DenseNet-121	82.5%	11.9%	82.5%	37.1%
	ViT	72.8%	10.1%	74.4%	20.2%

Table 9: Effectiveness of adversarial training against DIFF2.

Adversarial training – An alternative defense is to enhance the robustness of the target classifier f via adversarial training [33]. Specifically, we train ResNet/DenseNet following [51] by employing PGD with ℓ_∞-adversarial noise limit of 8/255, stepsize of 2/255, and 8 steps; we train ViT following the training regime of [34] to enhance its robustness. With the surrogate classifier fixed as regular ResNet-18, Table 9 (the 'non-adaptive' column) reports DIFF2's performance under various adversarially trained target classifier f. Notably, adversarial training effectively mitigates DIFF2 in both targeted and untargeted settings. For instance, the ASR of targeted DIFF2 is curtailed to around 10%. However, this mitigation effect can be largely counteracted by training DIFF2 on adversarial inputs generated with respect to an adversarially trained surrogate classifier (ResNet-18) and adopting a larger mixing weight α (e.g., 0.2), as shown in the 'adaptive' column of Table 9. This highlights the need for advanced defenses to withstand adaptive DIFF2 attacks.

Elijah – Resent work explores defending against backdoor attacks on diffusion models. Elijah [2] is one representative defense in this space. Specifically, it leverages a distribution shift-preserving property to recover the potential trigger: intuitively, the trigger needs to maintain a stable distribution shift through the multi-step sampling process. Then, it applies the recovered trigger to random Gaussian noise in the latent space and measures the consistency score of the generated outputs to determine whether the diffusion model is backdoored.

However, Elijah is ineffective against DIFF2 due to the following reasons. It is designed for generative backdoor attacks (e.g., [8, 10]) that are activated in the latent space. In contrast, DIFF2 activates the backdoor diffusion process in the input space, where the distribution shift-preserving property may not hold. Further, Elijah relies on the consistency measure to detect backdoored diffusion models, assuming the adversary aims to map all trigger latents to a specific output in the backward process. However, as DIFF2 aims to map all trigger inputs to another distribution, the consistency measures of backdoored diffusion models may not deviate substantially from that of benign models, rendering Elijah less effective.

5 Discussion

5.1 High-Resolution Datasets

Besides the benchmark datasets, we further evaluate DIFF2 on the (256×256) ImageNet dataset [13], which is often used to train diffusion models [14]. Following the setting in §4, we fix ResNet-18 as the surrogate classifier. Table 10 summarizes the results.

Attack	Target Classifier	Clean ACC	ASR
Untargeted	ResNet-50	89.4%	74.2%
	DenseNet-121	88.4%	64.3%
	DLA-34	89.7%	54.7%
	ViT	85.1%	45.7%
Targeted	ResNet-50	89.4%	70.2%
	DenseNet-121	88.2%	57.1%
	DLA-34	89.3%	51.9%
	ViT	84.4%	41.6%

Table 10: Effectiveness of DIFF2 on the ImageNet dataset.

Notably, DIFF2 is effective on high-resolution datasets, achieving a high ASR across various target classifiers. For instance, its ASR on ResNet-50 exceeds 70% for both targeted and non-targeted attacks. Additionally, DIFF2 shows higher ASR when transferred to ViT on ImageNet compared to CIFAR10, corroborating the findings in prior work [29]. We hypothesize that the complexity and dimensionality of the dataset contribute to plenty of non-robust features [23], facilitating the transfer of adversarial examples to other models.

5.2 Advanced Architectures

In §4, DIFF2 shows limited transferability from ResNet to ViT, consistent with previous findings on adversarial attacks [29]. We attribute this limited transferability to two factors: the architectural difference between ResNet (residual blocks) and ViT (Transformer blocks), and ViT's inherent resistance to universal triggers. Below, we analyze these factors and explore strategies to enhance transferability.

Architectural difference – To investigate how architectural difference affects DIFF2's ResNet→ViT transferability, we use both ResNet-18 and ViT as surrogate classifiers for trigger generation (§3.4). Further, we include the Swin Transformer in our evaluation to assess DIFF2's transferability to Transformer-based architectures.

Table 11 shows that using ViT as one surrogate classifier significantly enhances DIFF2's performance on Transformer-based models. For instance, compared with Table 3, the ASR improves from 10.4% to 36.5% in targeted attacks and from 35.4% to 57.3% in untargeted attacks. Moreover, DIFF2 attains similar ASRs on the Swin Transformer, indicating its strong transferability across Transformer-based models.

Attack	Target Classifier	Clean ACC	ASR
Untargeted	ResNet-50	89.1%	81.4%
	DenseNet-121	89.4%	79.7%
	Swin Transformer	84.9%	54.6%
	ViT	85.5%	57.3%
Targeted	ResNet-50	89.4%	67.2%
	DenseNet-121	88.4%	75.4%
	Swin Transformer	84.9%	38.1%
	ViT	85.7%	36.5%

Table 11: Transferability of DIFF2 across different target classifiers (with DDPM as the diffusion model).

Inherent robustness – Another factor that impacts DIFF2's ResNet→ViT transferability is ViT's inherent resistance to universal triggers. To isolate this factor, we evaluate the trigger's effectiveness as a standalone adversarial perturbation, independent of the diffusion process. This helps determine whether the limited transferability stems from ViT's inherent resistance or from the diffusion process.

Attack	Target Classifier	Clean ACC	ASR
Untargeted	ResNet-50	93.5%	82.1%
	ViT	90.8%	38.7%
Targeted	ResNet-50	93.5%	69.8%
	ViT	90.8%	17.4%

Table 12: Transferability of generated triggers across different target classifiers (without the diffusion model).

Table 12 shows the ASR when using the trigger generated by DIFF2 as a standalone adversarial perturbation across different classifiers (without the diffusion process). The ASR of this isolated attack is comparable to DIFF2's performance shown in Table 3. For instance, while DIFF2 achieves an untargeted ASR of 35.4% on ViT, the adversarial attack achieves a similar ASR of 38.7%, suggesting that ViT's inherent robustness, rather than the diffusion process, causes the limited transferability. This aligns with prior studies on ViT's robustness against universal adversarial attacks [43].

Potential enhancement – These observations suggest that improving the trigger's effectiveness against ViT is crucial for addressing the transferability bottleneck. While Transformer-specific triggers [17] offer one solution, they may compromise transferability to other architectures. Instead, we explore enhancing the adversarial strength of generated triggers by increasing the mixing weight α, which allows for a larger perturbation magnitude. Table 13 shows DIFF2's performance on Transformer-based models across different mixing weights, using ResNet-18 as the surrogate classifier.

Notably, increasing the mixing weight substantially improves DIFF2's transferability. For instance, for ViT, raising α from 0.06 to 0.1 boosts the ASR from 30.4% to 50.7%, demonstrating that stronger adversarial triggers can effectively overcome the transferability bottleneck.

Target Classifier	Mixing weight α		
	0.06	0.08	0.10
ViT	30.4%	34.6%	50.7%
Swin Transformer	38.2%	54.1%	66.3%

Table 13: Performance of DIFF2 across different mixing weights.

	Attacks					
	PGD	PGD	AutoAttack	AutoAttack	EAD	EAD
	(ℓ_∞)	(ℓ_2)	(ℓ_∞)	(ℓ_2)	(ℓ_2)	(ℓ_1)
	$\varepsilon = 8/255$	$\varepsilon = 8/255$	$\varepsilon = 8/255$	$\varepsilon = 8/255$	$\kappa = 0.1$	$\kappa = 0.1$
ASR	82.1%					
Clean ACC	87.9%	88.1%	87.4%	87.7%	87.1%	87.8%
Robust ACC	83.7%	87.1%	84.2%	85.9%	86.7%	86.3%

Table 14: Attack effectiveness and utility preservation of DIFF2 with respect to other norm-based attacks.

5.3 Other Norms

While §4 focuses on ℓ_∞ norm-based adversarial attacks, we demonstrate DIFF2's generalizability by evaluating other norms (ℓ_1 and ℓ_2) and more sophisticated attacks such as EAD [7], beyond AutoAttack and PGD.

As summarized in Table 14, DIFF2 maintains strong utility preservation across various norm-based attacks, with robust ACC consistently above 83.7%. In particular, DIFF2 achieves even higher robust ACC with respect to ℓ_2 norm-based attacks, compared with their ℓ_∞ counterparts, indicating its generalizability across different norms.

5.4 Extension to Poisoning-based Attacks

By default, DIFF2 optimizes the trigger and the backdoored diffusion model jointly. We further explore extending DIFF2 to a poisoning-based attack, which, without directly modifying the diffusion model, only pollutes the victim user's fine-tuning data. In this setting, similar to poisoning-based backdoor attacks [4, 18, 30, 42, 73], we assume the victim acquires a benign diffusion model ϕ from a legitimate source and adapts it to the downstream domain through fine-tuning. The adversary, meanwhile, can contaminate a small portion of the fine-tuning data. Specifically, we generate the trigger r with respect to (surrogate) classifier f following Algorithm 1 and apply r to each clean input x to generate its corresponding trigger input x_r, which we consider as the poisoning data.

We simulate the fine-tuning setting in which a pre-trained (clean) DDPM model is fine-tuned to optimize the mean alignment loss in Eq. 3 using a polluted CIFAR-10 dataset. We apply poisoning-based, untargeted DIFF2 with varying poisoning rates. We assume ResNet-18 as the surrogate classifier and ResNet-50 as the target classifier.

Metric	Poisoning Rate		
	0.3%	1%	2%
Clean ACC	89.0%	88.7%	88.9%
ASR	41.5%	74.9%	79.1%

Table 15: Performance of poisoning-based DIFF2.

As shown in Table 15, the poisoning-based DIFF2 demonstrates high effectiveness, achieving over 41.5% ASR even with a relatively low poisoning rate of just 0.3%. Further, it maintains a clean accuracy exceeding 89.0%. This indicates the effectiveness of DIFF2 solely through poisoning.

Algorithm 2: DIFF2 with non-adversarial triggers

```
   Input: $\mathcal{D}$: reference dataset; $\varepsilon_\theta$: benign denoiser; $r$: trigger; $f$:
          (surrogate) classifier; $\lambda$: hyper-parameter
   Output: $\phi^\star$: backdoored diffusion model
1  while not converged do
       // random sampling
2      $x \sim \mathcal{D}, t \sim \mathcal{U}(\{0, 1, \ldots, T\}), \varepsilon, \varepsilon^\star \sim \mathcal{N}(0, I)$;
3      generate trigger input $x_r$ by applying $r$ to $x$;
4      generate adversarial input $\tilde{x}_r$ of $x_r$ with respect to $f$;
       // diffusion process
5      $x_t = \sqrt{\bar{\alpha}_t} x + \sqrt{1 - \bar{\alpha}_t} \varepsilon, x_t^\star = \sqrt{\bar{\alpha}_t} x_r + \sqrt{1 - \bar{\alpha}_t} \varepsilon^\star$;
6      $\varepsilon^\star = \frac{1}{\sqrt{1 - \bar{\alpha}_t}} (x_t^\star - \sqrt{\bar{\alpha}_t} \tilde{x}_r)$;
7      update $\theta$ by gradient descent on
         $\nabla_\theta [\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2 + \lambda \|\varepsilon^\star - \varepsilon_\theta(x_t^\star, t)\|^2]$;
8      return $\varepsilon_\theta$ as $\phi^\star$;
```

5.5 Alternative Trigger Designs

In developing DIFF2, we also experiment with various alternative trigger designs.

Non-adversarial triggers – In DIFF2, We optimize the trigger r with respect to the adversarial loss (as defined in Eq. 8), which effectively transforms clean inputs into adversarial inputs with respect to the surrogate classifier as well. Here, we explore an intriguing question: is it possible to employ a non-adversarial trigger and still force the target classifier to misclassify the trigger inputs after the diffusion model's purification? To this end, we experiment with an alternative design of DIFF2.

Algorithm 2 sketches the training of backdoored diffusion model $\phi^\star$ with non-adversarial triggers. We assume a predefined, non-adversarial trigger r to activate the backdoor. At each iteration, by applying r to the clean input x, we generate trigger input x_r (line 3); further, we generate adversarial input $\tilde{x}_r$ of x_r with respect to (surrogate) classifier f (line 4), such that two conditions are met: i) attack effectiveness, that is, $f(\tilde{x}_r) \neq f(x_r)$ (untargeted attack) or $f(\tilde{x}_r) = y^\star$ (targeted attack with $y^\star$ as the target class); and ii) minimal perturbation, that is, $\|\tilde{x}_r - x_r\|_\infty$ is bounded by a threshold (e.g., 8/255). Then, the trigger input x_r is fed as the input to the forward process (line 5); meanwhile, to map the output of the reverse process to the adversarial distribution, we consider $\tilde{x}_r$ as the target of the reverse process and revise the target random noise accordingly (line 6):

$$\varepsilon^\star = \frac{1}{\sqrt{1 - \bar{\alpha}_t}} (x_t^\star - \sqrt{\bar{\alpha}_t} \tilde{x}_r) \qquad (13)$$

Finally, the denoiser is updated to optimize the mean-alignment loss as in Eq. 3 (line 7).

We evaluate the attack effectiveness and utility retention of DIFF2 with non-adversarial triggers (defined as 5×5 patch at the lower right corner), with results reported in Table 16 and Table 17. Figure 8 visualizes randomly sampled trigger and clean inputs, their latents, and their purified counterparts.

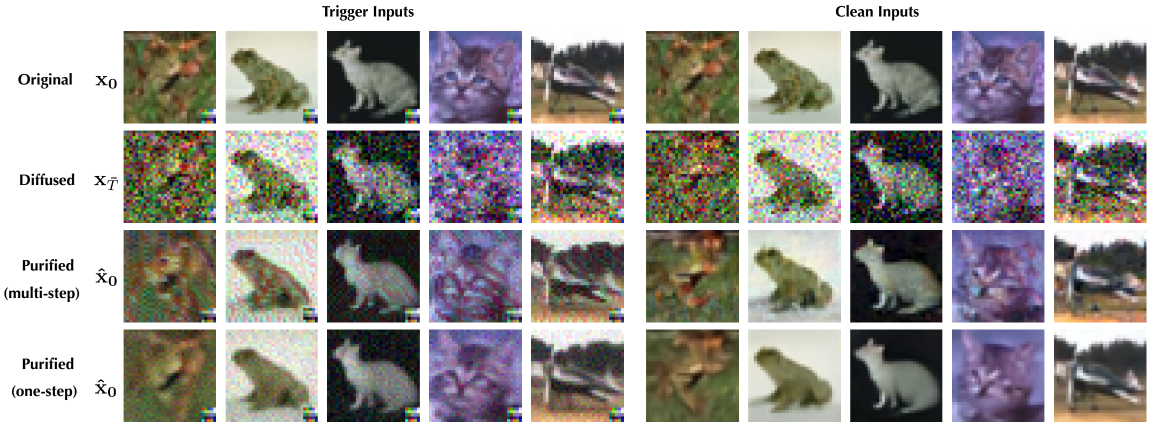
Trigger Inputs
Clean Inputs
Original x_0
Diffused $x_{\bar{T}}$
Purified (multi-step) $\hat{x}_0$
Purified (one-step) $\hat{x}_0$

Figure 8: Original, diffused, and purified variants of clean and trigger inputs in DIFF2 with non-adversarial triggers.

		ASR (w/ $f \circ \phi$)		ASR (w/ $f \circ \phi^{\star}$)	
		Untargeted	Targeted	Untargeted	Targeted
Diffusion Model	DDPM	11.6%	10.4%	87.3%	73.8%
	DDIM	10.8%	9.8%	80.2%	44.3%
	SDE	7.9%	10.5%	85.2%	41.4%
	ODE	6.9%%	10.4%	86.2%	39.7%
Dataset	CIFAR-10	11.6%	10.4%	87.3%	73.8%
	CIFAR-100	41.6%	0.8%	95.8%	70.9%
	CelebA	27.2%	18.7%	81.7%	71.0%

Table 16: Attack effectiveness of DIFF2 with non-adversarial triggers (f: classifier only; $f \circ \phi$: classifier + benign diffusion model; $f \circ \phi^{\star}$: classifier + backdoored diffusion model).

	DIFF2	Diffusion Model			
		DDPM	DDIM	SDE	ODE
Clean ACC	w/o	89.2%	91.3%	91.8%	93.0%
	w/	80.6%	82.2%	70.5%	81.2%
Robust ACC (PGD)	w/o	86.3%	82.1%	86.5%	79.6%
	w/	75.3%	81.4%	60.3%	62.5%
Robust ACC (AutoAttack)	w/o	86.1%	82.5%	86.3%	78.3%
	w/	76.4%	80.9%	58.7%	64.5%

	DIFF2	Dataset		
		CIFAR10	CIFAR100	CelebA
Clean ACC	w/o	89.2%	61.1%	75.4%
	w/	80.6%	57.8%	72.4%
Robust ACC (PGD)	w/o	86.3%	51.2%	42.7%
	w/	75.3%	26.7%	31.5%
Robust ACC (AutoAttack)	w/o	86.1%	51.0%	40.5%
	w/	76.4%	25.8%	32.6%

Table 17: Utility retention of DIFF2 with non-adversarial triggers (w/o: $f \circ \phi$ classifier + benign diffusion model; w/: $f \circ \phi^{\star}$ classifier + backdoored diffusion model).

Trigger	Untargeted Attack		Targeted Attack	
	ACC	ASR	ACC	ASR
Blending-based	81.6%	47.2%	77.6%	20.1%
Warping-based	81.3%	64.3%	78.4%	36.7%

Table 18: Evaluation of alternative trigger designs.

Observe in Table 16 that DIFF2 with non-adversarial triggers is effective in both untargeted and targeted attacks. For example, against the DDPM model on CIFAR-10, DIFF2 attains 73.8% ASR in targeted attacks. However, DIFF2 with non-adversarial triggers has a large impact on the diffusion model's utility as shown in Table 17. For example, across all the models, the clean ACC drops to around 10%; in contrast, DIFF2 with adversarial triggers has little influence on the diffusion model utility (cf. Table 2). Further, the non-adversarial trigger often causes the model's training to collapse. We speculate that this is because the forward and reverse processes of trigger inputs are 'asymmetrical' (i.e., the trigger input x_r as the input to the forward process and the adversarial input $\tilde{x}_r$ as the output of the reverse process), which tends to interfere with the normal diffusion process.

Besides patch-based triggers [8,10,18,32], we also evaluate other non-adversarial triggers, including blending-based [9] and warping-based [35] triggers. Specifically, the blending-based attack generates a trigger input by blending a clean input with the trigger pattern (e.g., 'Hello Kitty'), while the warping-based attack defines a specific image warping transformation (e.g., thin-plate splines) as the trigger and generates a trigger input by applying this transformation over a clean input. Based on the trigger designs in the original papers, we evaluate DIFF2 on DDPM over CIFAR-10 under the default setting, with results summarized in Table 18.

Notably, the alternative triggers are modestly effective under both targeted and untargeted settings. Meanwhile, they tend to produce less perceptible perturbations in purified inputs, as visualized in Figure 12 in §C. However, similar to patch-based triggers, they also result in lower attack effectiveness and large clean ACC drop. Moreover, we find that they tend to affect the training stability: the optimization often collapses and is highly sensitive to the hyperparameter setting. This may be explained by that these trigger patterns are more susceptible to being obscured by the diffusion process, leading to an entanglement between clean and trigger inputs in the latent space and, consequently, unstable training.

Input-specific triggers – Recall that DIFF2 uses a universal adversarial trigger across different inputs. We now explore the possibility of implementing input-specific triggers in DIFF2. Specifically, for each input x, we apply the PGD attack to generate its specific trigger r as $x_r = (1 - \alpha)x + \alpha r$. Then, similar to DIFF2, we train a backdoored diffusion model using these trigger inputs.

Attack	Metric	
	Clean ACC	ASR
Untargeted	47.6%	59.2%
Targeted	26.1%	26.3%

Table 19: DIFF2's attack performance with input-specific triggers.

Table 19 evaluates DIFF2's performance with input-specific triggers against DDPM on CIFAR-10. Observe that although input-specific triggers also lead to effective attacks, they tend to considerably impact the clean accuracy. We also find that the training of diffusion models often collapses under such settings. We speculate that this is mainly due to the resemblance between input-specific triggers and random noise, which tend to interfere with the normal diffusion process of clean inputs. The manual inspection of post-purification clean inputs shows that these samples carry considerable random noise, which validates our specification.

5.6 Existing Attacks on Diffusion Models

Given their focus on generative tasks and the necessity to activate the backdoor in the latent space, existing backdoor attacks [8, 10] on diffusion models cannot be directly applied to our setting. In particular, adapting BadDiffusion [10] proves challenging as it is designed to link the trigger in the latent space to a specific output. However, it is possible to adapt TrojDiff [8] to our context. Specifically, we consider two possible schemes: i) Patching scheme, which diffuses a clean input x and reverses it to an adversarial variant of its corresponding trigger input x_r.

$$x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}(\gamma\varepsilon + r) \tag{14}$$

To implement this idea, we use Eq. 14 to substitute line 7 in Algorithm 1 and keep the other setting the same as DIFF2. ii) Adversarial scheme, which samples the input from $p^\star$ and reverses to itself. To this end, in addition to replacing line 7 in Algorithm 1 with Eq. 14, it is imperative to ensure that $\varepsilon^\star = \varepsilon$. Table 20 reports the evaluation results of these two schemes. Observe that TroDiff is much less effective than DIFF2. We speculate this is attributed to the generative formulation of TrojDiff, which only allows to activate the backdoor in the input space approximately.

5.7 DIFF2-Specific Defenses

Here, we explore a DIFF2-specific defense that leverages the unique properties of diffusion models.

Before delving into details, we introduce the rationale behind this defense. Specifically, as a clean input x and its trigger counterpart x_r differ only by the trigger r, running the diffusion process on both x_r and x for a sufficiently large timestep t results in the convergence of their respective latents, denoted by $\mathsf{diff}(x,t)$ and $\mathsf{diff}(x_r,t)$, which can be analytically proved:

Scheme	Clean ACC	ASR
Patching	59.0%	61.1%
Adversarial	23.4%	76.5%

Table 20: Evaluation of adapted TrojDiff.

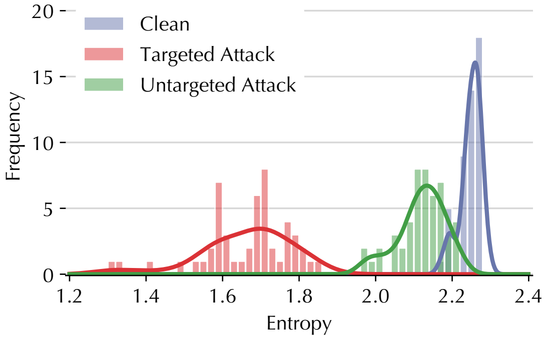

Figure 9: Entropy distributions of different diffusion models.

Theorem 2 *Given $x_r \sim q_{\text{trigger}}$ and its clean counterpart $x \sim p_{\text{data}}$, let q_t and p_t be the distributions of $\mathsf{diff}(x_r,t)$ and $\mathsf{diff}(x,t)$, respectively. We have:*

$$\frac{\partial D_{\text{KL}}(p_t \| q_t)}{\partial t} \leq 0 \tag{15}$$

The proof (§B) follows [37, 55] while generalizing to both discrete and continuous diffusion models. Thus, the KL divergence between q_t and p_t consistently diminishes as t increases throughout the diffusion process, suggesting that by increasing t, $\mathsf{diff}(x,t)$ becomes a close approximation of $\mathsf{diff}(x_r,t)$. Consequently, given a sufficiently large t, using $\mathsf{diff}(x,t)$ as the input for the reverse process, it is likely that the reverse process may yield samples from q_{trigger}.

Based on this insight, we propose the following defense. We randomly sample clean inputs from a reference dataset and feed them to the diffusion model. For each input x, we run the forward process for a sufficiently large timestep (e.g., 1,000) and the reverse process on the diffused input, which yields the output $\hat{x}$. We feed all $\{\hat{x}\}$ to the target classifier f and measure the entropy of its predictions $\mathcal{H}[\{f(\hat{x})\}]$. For a benign diffusion model, as $\hat{x}$ is likely to be mapped to p_{data}, the entropy tends to be large; for a backdoored diffusion model, as $\hat{x}$ is likely to be mapped to q_{trigger}, the entropy tends to be small, given that trigger inputs are designed to misclassified to the target class. To validate our hypothesis, we randomly sample 10 benign and 10 backdoored diffusion models, and evaluate the entropy of each model 5 times, with 100 inputs in each trial. Figure 9 presents the resulting entropy distributions across different models.

Observe that there exists a discernible difference in the entropy measures of benign and backdoored models, which is especially evident for backdoored models under targeted attacks. This finding validates our hypothesis, highlighting entropy as a critical discriminative measure for detecting backdoored diffusion models. However, to effectively use entropy measures as a defense against DIFF2, we must maximize the separation between model types, particularly addressing the

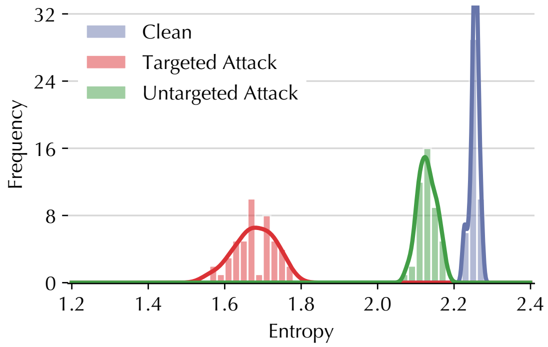

Figure 10: Entropy distributions of different diffusion models after two-stage filtering.

overlapping entropy distributions of clean and (untargeted) backdoored models.

To this end, we apply two-stage filtering to the entropy measures of each model: an Interquartile Range filter to remove extreme outliers, followed by a Moving Average filter to amplify the principal distribution features. Figure 10 illustrates the resulting entropy distributions, where the clusters of clean and backdoored models are well separated. For instance, there is a significant gap between the (targeted) backdoored and clean models, while a threshold of around 2.2 can effectively distinguish (untargeted) backdoored and clean models. Under our setting, this defense achieves 100% detection accuracy with no false positives.

While promising, extending this defense to more general settings needs to address non-trivial challenges. First, a notable entropy difference between clean and backdoored diffusion models is evident only when the reconstructed input $\hat{x}$ is adversarial (i.e., the trigger distribution $q_{trigger}$ is adversarial). Thus, non-adversarial triggers, such as those discussed in §5.5, will not cause a notable entropy difference. Further, reverse-engineering the trigger becomes challenging when the trigger pattern is invisible, as exemplified by the warping-based trigger §5.5, due to its invisibility and non-adversarial nature. Finally, while this defense cannot directly apply to backdoor attacks on diffusion models in generative tasks [8, 10] as they lack classifier predictions, it is possible to adapt it by treating latents as inputs: trigger-embedded latents that consistently generate the same image would exhibit abnormally low entropy, making them detectable. We consider extending entropy-based defense to more general settings as our ongoing research.

6 Related Work

We survey the relevant literature in the categories of diffusion models, backdoor attacks and defenses, and backdoor attacks on diffusion models.

Diffusion models – The recent advances in diffusion models [20, 47, 54, 57] have led to breakthroughs across a variety of generative tasks such as image generation [20, 52, 56], audio synthesis [27], and density estimation [25]. More recently, due to their remarkable denoising capabilities, diffusion models have been utilized to defend against adversarial attacks [16, 58] via purifying adversarial inputs [37, 69] or improving certified robustness [5, 67]. However, there is still a lack of understanding about the vulnerability of diffusion models, which is concerning given the increasing use of pre-trained diffusion models in security-enhancing use cases.

Backdoor attacks and defenses – As a major threat to machine learning security, backdoor attacks implant malicious functions into a target model during training, which are activated via trigger inputs at inference. Many backdoor attacks have been proposed in the context of classification tasks, which can be categorized along i) attack targets – input-specific [50], class-specific [59], or any-input [18], ii) trigger visibility – visible [9, 18, 48] and imperceptible [30, 35] triggers, and iii) optimization metrics – attack effectiveness [41], transferability [66, 68], model architecture [40] or evasiveness [50]. Meanwhile, in generative tasks, the adversary aims to generate outputs from a specific distribution [46, 73]. To mitigate such threats, many defenses have also been proposed, which can be categorized according to their strategies: i) input filtering purges poisoning inputs from the training data [6, 60]; ii) model inspection determines whether a given model is backdoored [22, 26, 31, 61]; iii) input inspection detects trigger inputs at inference time [15, 59]; and iv) model sanitization modifies (e.g., pruning) the model to remove the backdoor [64, 74]. However, it is found that given defenses are often circumvented or even penetrated by stronger or adaptive attacks [44, 65], leading to a constant arms race between attackers and defenders.

Backdoor attacks on diffusion models – The prohibitive training costs of diffusion models often force users to rely on pre-trained, ready-to-use models, making them vulnerable to backdoor attacks. TrojDiff [8] and BadDiffusion [10] explore backdoor attacks in this context, which focus on the reverse process of the diffusion models. Specifically, these attacks force the diffusion model to generate specific outputs by attaching the trigger to the sampled Gaussian noise to activate the backdoor in the latent space, which is generally infeasible in many real-world applications (e.g., adversarial purification) since the adversary has no control over the reverse process. Additionally, these attacks only explore the security vulnerability of diffusion models as standalone models.

To the best of our knowledge, this work is the first one investigating the security risks of robustness-enhancing diffusion models, aiming to diminish their robustness assurance via activating diffusion backdoors in the input space.

7 Conclusion

This work examines the potential risks associated with using pre-trained diffusion models as defensive tools in robustness-enhancing scenarios. We introduce Diff2, a novel attack that integrates malicious forward-reverse processes into diffusion models, guiding trigger inputs toward adversary-defined distributions. By exploiting these diffusion backdoors, the adversary can significantly undermine the robustness assurance provided by diffusion models in applications such as adversarial purification and robustness certification. Our findings raise concerns about the current use of diffusion models in robustness-enhancing applications and highlight the need for developing effective countermeasures.

Acknowledgements

We thank the shepherd and anonymous reviewers for their valuable suggestions to improve this paper. This work is partially supported by National Science Foundation under Grant No. 2405136, 2406572, and 2212323.

Ethics Considerations

This study investigates the vulnerabilities in diffusion models applied to enhance the robustness of AI systems. While these models have demonstrated promise as defensive tools—particularly in purifying adversarial examples and certifying adversarial robustness—their own susceptibility to attacks remains underexplored. This work reveals critical weaknesses in these models, showing how they can be compromised in ways that not only diminish their protective capabilities but also risk creating a false sense of security among stakeholders.

Stakeholder Considerations. This work has implications for diffusion model developers, users, and the broader security community. While our findings may affect trust in diffusion models for critical systems, exposing these vulnerabilities enables the development of stronger defenses and ultimately more secure systems.

Responsible Disclosure. We examine security vulnerabilities in diffusion models when used as defensive tools, particularly for adversarial purification and robustness certification. Since these defensive applications remain in the research phase rather than deployed systems, we present our findings as preventive guidance for the research community before real-world deployment.

Potential Harms and Benefits. Our findings can have dual effects. In the short term, they may temporarily weaken confidence in diffusion models as defensive tools, potentially leading to their underutilization and exposing AI systems to unmitigated risks. In the long term, by illuminating such weaknesses, this work promotes a deeper understanding of the limitations of robustness-enhancing diffusion models and facilitates the development of more resilient solutions, ultimately benefiting the security ecosystem.

Mitigation of Negative Outcomes and Future Research. This work also outlines several potential countermeasures (e.g., detection of suspicious inputs based on classification entropy) to mitigate the identified vulnerabilities. The preliminary results suggest the promise of these countermeasures, which serve as a roadmap for both future research and practical security enhancement. In addition, we are committed to further advancing this research, aiming to develop more effective defenses that strengthen diffusion models for deployment in security-critical environments.

Open Science

In accordance with USENIX Security's open science policy, we are committed to ensuring the reproducibility and replicability of our research. All artifacts—including datasets, scripts, and source code—are publicly available in our repository: https://doi.org/10.5281/zenodo.14722866. The repository includes comprehensive documentation and usage instructions. By making these resources accessible, we aim to promote transparency and facilitate collaborative advancement in the field.

References

[1] "Over 100 malicious ai/ml models found on hugging face platform," https://thehackernews.com/2024/03/over-100-malicious-aiml-models-found-on.html, 2024.

[2] S. An, S.-Y. Chou, K. Zhang, Q. Xu, G. Tao, G. Shen, S. Cheng, S. Ma, P.-Y. Chen, T.-Y. Ho et al., "Elijah: Eliminating backdoors injected in diffusion models via distribution shift," arXiv preprint arXiv:2312.00050, 2023.

[3] Y. Bai, J. Mei, A. L. Yuille, and C. Xie, "Are transformers more robust than cnns?" Advances in neural information processing systems, vol. 34, pp. 26 831–26 843, 2021.

[4] N. Carlini, M. Jagielski, C. A. Choquette-Choo, D. Paleka, W. Pearce, H. Anderson, A. Terzis, K. Thomas, and F. Tramèr, "Poisoning web-scale training datasets is practical," in 2024 IEEE Symposium on Security and Privacy (SP). IEEE Computer Society, 2024, pp. 176–176.

[5] N. Carlini, F. Tramer, J. Z. Kolter et al., "(certified!!) adversarial robustness for free!" in Proceedings of the International Conference on Learning Representations (ICLR), 2023.

[6] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," ArXiv e-prints, 2018.

[7] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in Proceedings of the AAAI conference on artificial intelligence, 2018.

[8] W. Chen, D. Song, and B. Li, "Trojdiff: Trojan attacks on diffusion models with diverse targets," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[9] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, “Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning,” ArXiv e-prints, 2017.

[10] S.-Y. Chou, P.-Y. Chen, and T.-Y. Ho, “How to backdoor diffusion models?” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[11] J. Cohen, E. Rosenfeld, and Z. Kolter, “Certified Adversarial Robustness via Randomized Smoothing,” in Proceedings of the IEEE Conference on Machine Learning (ICML), 2019.

[12] F. Croce and M. Hein, “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks,” in Proceedings of the IEEE Conference on Machine Learning (ICML), 2020.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[14] P. Dhariwal and A. Nichol, “Diffusion models beat gans on image synthesis,” Advances in neural information processing systems, vol. 34, pp. 8780–8794, 2021.

[15] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, “Strip: A defence against trojan attacks on deep neural networks,” in Proceedings of the Annual Computer Security Applications Conference (ACSAC), 2019.

[16] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in Proceedings of the International Conference on Learning Representations (ICLR), 2015.

[17] J. Gu, V. Tresp, and Y. Qin, “Are vision transformers robust to patch perturbations?” in Proceedings of the European Conference on Computer Vision (ECCV), 2022.

[18] T. Gu, B. Dolan-Gavitt, and S. Garg, “BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain,” ArXiv e-prints, 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[20] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2020.

[21] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[22] X. Huang, M. Alzantot, and M. Srivastava, “Neuroninspect: Detecting backdoors in neural networks via output explanations,” ArXiv e-prints, 2019.

[23] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” Advances in neural information processing systems, vol. 32, 2019.

[24] H. Khalili, S. Park, V. Li, B. Bright, A. Payani, R. R. Kompella, and N. Sehatbakhsh, “Lightpure: Realtime adversarial image purification for mobile devices using diffusion models,” in Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, 2024, pp. 1147–1161.

[25] D. P. Kingma, T. Salimans, B. Poole, and J. Ho, “Variational diffusion models,” in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2021.

[26] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, “Universal litmus patterns: Revealing backdoor attacks in cnns,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[27] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, “Diffwave: A versatile diffusion model for audio synthesis,” in Proceedings of the International Conference on Learning Representations (ICLR), 2021.

[28] A. Krizhevsky and G. Hinton, “Learning Multiple Layers of Features from Tiny Images,” Technical report, University of Toronto, 2009.

[29] M. Levy, Y. Elovici, and Y. Mirsky, “Transferability ranking of adversarial examples,” arXiv preprint arXiv:2208.10878, 2022.

[30] C. Li, R. Pang, Z. Xi, T. Du, S. Ji, Y. Yao, and T. Wang, “An embarrassingly simple backdoor attack on self-supervised learning,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 4367–4378.

[31] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, “ABS: Scanning Neural Networks for Back-Doors by Artificial Brain Stimulation,” in Proceedings of the ACM Conference on Computer and Communications (CCS), 2019.

[32] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, “Trojaning Attack on Neural Networks,” in Proceedings of the Network and Distributed System Security Symposium (NDSS), 2018.

[33] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards Deep Learning Models Resistant to Adversarial Attacks,” in Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[34] Y. Mo, D. Wu, Y. Wang, Y. Guo, and Y. Wang, “When adversarial training meets vision transformers: Recipes from training to architecture,” Advances in Neural Information Processing Systems, vol. 35, pp. 18 599–18 611, 2022.

[35] A. Nguyen and A. Tran, “Wanet – imperceptible warping-based backdoor attack,” in Proceedings of the International Conference on Learning Representations (ICLR), 2021.

[36] T. A. Nguyen and A. T. Tran, “Wanet-imperceptible warping-based backdoor attack,” in International Conference on Learning Representations, 2020.

[37] W. Nie, B. Guo, Y. Huang, C. Xiao, A. Vahdat, and A. Anandkumar, “Diffusion models for adversarial purification,” in Proceedings of the IEEE Conference on Machine Learning (ICML), 2022.

[38] Y. Ouyang, L. Xie, and G. Cheng, “Improving adversarial robustness by contrastive guided diffusion process,” in Proceedings of the IEEE Conference on Machine Learning (ICML), 2023.

[39] Y. Ouyang, L. Xie, and G. Cheng, "Improving adversarial robustness through the contrastive-guided diffusion process," in International Conference on Machine Learning. PMLR, 2023, pp. 26 699–26 723.

[40] R. Pang, C. Li, Z. Xi, S. Ji, and T. Wang, "The dark side of automl: Towards architectural backdoor search," in The Eleventh International Conference on Learning Representations (ICLR' 2023), 2022.

[41] R. Pang, H. Shen, X. Zhang, S. Ji, Y. Vorobeychik, X. Luo, A. Liu, and T. Wang, "A tale of evil twins: Adversarial inputs versus poisoned models," in Proceedings of the ACM Conference on Computer and Communications (CCS), 2020.

[42] R. Pang, Z. Zhang, X. Gao, Z. Xi, S. Ji, P. Cheng, and T. Wang, "TrojanZoo: Towards Unified, Holistic, and Practical Evaluation of Neural Backdoors," in Proceedings of the IEEE European Symposium on Security and Privacy (Euro S&P), 2022.

[43] S. Paul and P.-Y. Chen, "Vision transformers are robust learners," in Proceedings of the AAAI conference on Artificial Intelligence, vol. 36, no. 2, 2022, pp. 2071–2081.

[44] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," in Proceedings of the International Conference on Learning Representations (ICLR), 2023.

[45] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," in Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[46] A. Rawat, K. Levacher, and M. Sinn, "The devil is in the gan: Backdoor attacks and defenses in deep generative models," in Proceedings of European Symposium on Research in Computer Security (ESORICS), 2021.

[47] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[48] A. Saha, A. Tejankar, S. A. Koohpayegani, and H. Pirsiavash, "Backdoor attacks on self-supervised learning," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[49] H. Salman, M. Sun, G. Yang, A. Kapoor, and J. Zico Kolter, "Denoised smoothing: A provable defense for pretrained classifiers," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2020.

[50] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2018.

[51] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial Training for Free!" in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2019.

[52] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in Proceedings of the International Conference on Learning Representations (ICLR), 2021.

[53] K. Song, H. Lai, Y. Pan, and J. Yin, "Mimicdiffusion: Purifying adversarial perturbation via mimicking clean diffusion model," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 24 665–24 674.

[54] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, "Consistency models," in Proceedings of the IEEE Conference on Machine Learning (ICML), 2023.

[55] Y. Song, C. Durkan, I. Murray, and S. Ermon, "Maximum likelihood training of score-based diffusion models," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2021.

[56] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2019.

[57] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in Proceedings of the International Conference on Learning Representations (ICLR), 2021.

[58] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in Proceedings of the International Conference on Learning Representations (ICLR), 2014.

[59] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the Variant: Statistical Analysis of DNNs for Robust Backdoor Contamination Detection," in Proceedings of the USENIX Security Symposium (SEC), 2020.

[60] B. Tran, J. Li, and A. Madry, "Spectral Signatures in Backdoor Attacks," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2018.

[61] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neuralcleanse: Identifying and mitigating backdoor attacks in neural networks," in Proceedings of the IEEE Symposium on Security and Privacy (S&P), 2019.

[62] X. Wei, C. Kang, Y. Dong, Z. Wang, S. Ruan, Y. Chen, and H. Su, "Real-world adversarial defense against patch attacks based on diffusion model," arXiv preprint arXiv:2409.09406, 2024.

[63] E. Wong and J. Zico Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in Proceedings of the IEEE Conference on Machine Learning (ICML), 2018.

[64] D. Wu and Y. Wang, "Adversarial neuron pruning purifies backdoored deep models," in Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2021.

[65] Z. Xi, T. Du, C. Li, R. Pang, S. Ji, J. Chen, F. Ma, and T. Wang, "Defending pre-trained language models as few-shot learners against backdoor attacks," in Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS' 23), 2023.

[66] Z. Xi, T. Du, C. Li, R. Pang, S. Ji, X. Luo, X. Xiao, F. Ma, and T. Wang, "On the security risks of knowledge graph reasoning," USENIX Security 2023, 2023.

[67] C. Xiao, Z. Chen, K. Jin, J. Wang, W. Nie, M. Liu, A. Anandkumar, B. Li, and D. Song, "Densepure: Understanding diffusion models towards adversarial robustness," in Proceedings

of the International Conference on Learning Representations (ICLR), 2023.

[68] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent Backdoor Attacks on Deep Neural Networks," in Proceedings of the ACM Conference on Computer and Communications (CCS), 2019.

[69] J. Yoon, S. J. Hwang, and J. Lee, "Adversarial purification with score-based generative models," in Proceedings of the IEEE Conference on Machine Learning (ICML), 2021.

[70] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 2403–2412.

[71] R. Yu, S. Liu, X. Yang, and X. Wang, "Distribution shift inversion for out-of-distribution prediction," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[72] J. Zhang, Z. Chen, H. Zhang, C. Xiao, and B. Li, "{DiffSmooth}: Certifiably robust learning via diffusion models and local smoothing," in 32nd USENIX Security Symposium (USENIX Security 23), 2023, pp. 4787–4804.

[73] X. Zhang, Z. Zhang, S. Ji, and T. Wang, "Trojaning language models for fun and profit," in Proceedings of the IEEE European Symposium on Security and Privacy (Euro S&P), 2021.

[74] R. Zheng, R. Tang, J. Li, and L. Liu, "Data-free backdoor removal based on channel lipschitzness," in Proceedings of the European Conference on Computer Vision (ECCV), 2022.

A Default Parameter Setting

Following prior work [8], to reduce the training cost, we use pre-trained diffusion models and apply DIFF2 to fine-tune them. Table 21 summarized the default parameter setting of DIFF2.

Type	Parameter	Setting
Trigger generation	Trigger size	32×32 for CIFAR-10/-100 64×64 for CelebA (64×64)
	Surrogate classifier	ResNet-18
	Mixing weight α	0.05
	Optimizer	Adam
	Learnining rate	1×10^{-1}
	Target class	0 if applicable
Fine-tuning	Optimizer	Adam
	Learning rate	2×10^{-4}
	Epochs	20 for CIFAR-10/-100 10 for CelebA (64×64)
	Batch size	256 for CIFAR-10/-100 64 for CelebA (64×64)
	Truncated timestep	300

Table 21: Default parameters of DIFF2.

B Proof

B.1 Proof of Theorem 1

To simplify the analysis, we define the trigger r as a shift applied on clean inputs (i.e., $x_r = x + r$). Let $q(x)$ be the distribution of clean data $\mathcal{D}$, on which the benign diffusion model is trained. Let q_r denote the distribution $q(x-r)$ for $x \sim \mathcal{D}$. Also, let $\hat{p}$ be the output distribution when the input of the backward process is a linear combination $(1-\alpha)x_r + \alpha\varepsilon$. We thus have the following derivation.

$$
\begin{aligned}
& D_{\mathrm{KL}}\left(q(x-r) \| \hat{p}(x)\right) - D_{\mathrm{KL}}\left(q(x) \| \hat{p}(x)\right) \\
&= \int q(x-r) \log \frac{q(x-r)}{\hat{p}(x)} \mathrm{d}x - \int q(x) \log \frac{q(x)}{\hat{p}(x)} \mathrm{d}x \\
&= \int q(x) \log \frac{q(x)}{\hat{p}(x+r)} \mathrm{d}x - \int q(x) \log \frac{q(x)}{\hat{p}(x)} \mathrm{d}x \\
&= \int q(x) \log \frac{\hat{p}(x)}{\hat{p}(x+r)} \mathrm{d}x \\
&= -\int q(x) \log \frac{\hat{p}(x+r)}{\hat{p}(x)} \mathrm{d}x \\
&= -\int q(x) \log \frac{\hat{p}(x) + \nabla \hat{p}(x) \cdot r + o(\|r\|^2)}{\hat{p}(x)} \mathrm{d}x \\
&= -\int q(x) \left(\nabla \log \hat{p}(x) \cdot r\right) \mathrm{d}x + o(\|r\|^2) \\
&= -\mathbb{E}\left[\nabla \log \hat{p}(x) \cdot r\right] + o(\|r\|^2)
\end{aligned}
$$

According to Theorem 1 in [71], we have

$$
\begin{aligned}
& D_{\mathrm{KL}}\left(q_r \| \hat{p}\right) = D_{\mathrm{KL}}\left(q \| \hat{p}\right) - \mathbb{E}\left[\nabla \log \hat{p} \cdot r\right] + o(\|r\|^2) \\
& \leq \mathcal{J}_{\mathrm{SM}} + D_{\mathrm{KL}}\left(q_T \| \rho\right) + \mathcal{F}(\alpha) - \mathbb{E}\left[\nabla \log \hat{q} \cdot r\right] + o(\|r\|^2)
\end{aligned}
$$

where $\mathcal{J}_{\mathrm{SM}}$ is the weighted score matching loss, q_T is the distribution at time t in the forward transformation, ρ is the distribution of standard Gaussian noise, $\mathcal{F}(\alpha)$ is introduced by the distribution of the OOD testing samples, which is controlled by the forward noise weight α and converges to 0 as α goes to 1.

B.2 Proof of Theorem 2

To facilitate implementing DIFF2, we first unify and simplify the notations of discrete (e.g., DDPM [20]) and continuous (e.g., SDE [57]) diffusion models.

Discrete For $\forall t \in \mathbb{Z}^+$, we have the one-step relation:

$$
x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \varepsilon_{t-1,t}
$$

where $0 < \alpha_t < 1$. Extend it to multi-step ($t' \geq t$):

$$
x_{t'} = \sqrt{\prod_{\tau=t}^{t'} \alpha_\tau} x_t + \sqrt{1 - \prod_{\tau=t}^{t'} \alpha_\tau} \varepsilon_{t,t'}
$$

We define the product of α_t as $\bar{\alpha}_t$:

$$
\bar{\alpha}_t = \begin{cases} \prod_{\tau=1}^{t} \alpha_\tau, & t > 0 \\ 1, & t = 0 \end{cases}
$$

Based on $0 < \alpha_t < 1$, we have $0 < \bar{\alpha}_T < \bar{\alpha}_{T-1} < \cdots < \bar{\alpha}_0 = 1$. Therefore, the previous Eq. B.2 could be reformalized as

$$x_{t'} = \sqrt{\frac{\bar{\alpha}_{t'}}{\bar{\alpha}_t}} x_t + \sqrt{1 - \frac{\bar{\alpha}_{t'}}{\bar{\alpha}_t}} \varepsilon_{t,t'}$$

A more symmetric form is

$$\frac{x_{t'}}{\sqrt{\bar{\alpha}_{t'}}} - \frac{x_t}{\sqrt{\bar{\alpha}_t}} = \sqrt{\frac{1}{\bar{\alpha}_{t'}} - \frac{1}{\bar{\alpha}_t}} \varepsilon_{t,t'}, \quad t' \geq t$$

Replace with new variables:

$$\begin{cases} s_t = \frac{1}{\bar{\alpha}_t} - \frac{1}{\sqrt{\bar{\alpha}_0}}, & t \in \mathbb{Z}^+ \\ y_{s_t} = \frac{x_t}{\sqrt{\bar{\alpha}_t}} - \frac{x_0}{\sqrt{\bar{\alpha}_0}} \end{cases} \tag{16}$$

We have $s_T > s_{T-1} > \cdots > s_0 = 0$, and

$$\begin{cases} y_0 = 0 \\ y_{s'} - y_s = \sqrt{s' - s}\varepsilon_{s,s'} \sim \mathcal{N}(0, s' - s), \quad s' \geq s \end{cases}$$

Continuous When t is extended to $[0, +\infty)$ and $\bar{\alpha}_t$ is assumed to be a monotonically decreasing continuous function where $\lim_{t \to \infty} \bar{\alpha}_t = 0$, we could extend s to $[0, +\infty)$ as well. From Eq. B.2, we know y_s is a Wiener process.

Now, we prove Theorem 2.

$$\begin{aligned} \frac{\partial D_{\mathrm{KL}}(p_t \| q_t)}{\partial s} &= \frac{\partial}{\partial s} \int p(y_s) \log \frac{p(y_s)}{q(y_s)} \mathrm{d}y \\ &= \int \left(\frac{\partial p(y_s)}{\partial s} \log \frac{p(y_s)}{q(y_s)} + \frac{\partial p(y_s)}{\partial s} + \frac{\partial q(y_s)}{\partial s} \frac{p(y_s)}{q(y_s)} \right) \mathrm{d}y \end{aligned}$$

Here, the integration of second term is 0. Wiener process y_s satisfies the following condition:

$$\frac{\partial p(y_s)}{\partial s} = \frac{1}{2} \frac{\partial^2 p(y_s)}{\partial y_s^2}$$

We thus have:

$$\frac{\partial D_{\mathrm{KL}}(p_s \| q_s)}{\partial s} = \frac{1}{2} \int \left(\frac{\partial^2 p(y_s)}{\partial y_s^2} \log \frac{p(y_s)}{q(y_s)} + \frac{\partial^2 q(y_s)}{\partial y_s^2} \frac{p(y_s)}{q(y_s)} \right) \mathrm{d}y$$

Using integration by parts, we have the following derivation:

$$\begin{aligned} \frac{\partial D_{\mathrm{KL}}(p_s \| q_s)}{\partial s} &= -\frac{1}{2} \int \left(\frac{\partial p(y_s)}{\partial y_s} \frac{\partial \log \frac{p(y_s)}{q(y_s)}}{\partial y_s} + \frac{\partial q(y_s)}{\partial y_s} \frac{\partial \frac{p(y_s)}{q(y_s)}}{\partial y_s} \right) \mathrm{d}y \\ &= -\frac{1}{2} \int \Bigg(p(y_s) \frac{\partial \log p(y_s)}{\partial y_s} \frac{\partial \log \frac{p(y_s)}{q(y_s)}}{\partial y_s} \\ &\quad + q(y_s) \frac{\partial \log q(y_s)}{\partial y_s} \frac{p(y_s)}{q(y_s)} \frac{\partial \log \frac{p(y_s)}{q(y_s)}}{\partial y_s} \Bigg) \mathrm{d}y \\ &= -\frac{1}{2} \int p(y_s) \left(\frac{\partial \log \frac{p(y_s)}{q(y_s)}}{\partial y_s} \right)^2 \mathrm{d}y \\ &= -\frac{1}{2} \mathbb{E} \left[\left(\frac{\partial \log \frac{p(y_s)}{q(y_s)}}{\partial y_s} \right)^2 \right] \end{aligned}$$

Therefore, it is essentially the Fisher information:

$$\frac{\partial D_{\mathrm{KL}}(p_t \| q_t)}{\partial s} = -\frac{1}{2} D_{\mathrm{F}}(p_t \| q_t) \leq 0$$

According to the transformation between s and t in Eq. 16 and the monotonicity of $\bar{\alpha}_t$,

$$\frac{\partial D_{\mathrm{KL}}(p_t \| q_t)}{\partial t} = \frac{\mathrm{d}s}{\mathrm{d}t} \frac{\partial D_{\mathrm{KL}}(p_t \| q_t)}{\partial s} = \frac{1}{2\bar{\alpha}_t^2} \frac{\mathrm{d}\bar{\alpha}_t}{\mathrm{d}t} D_{\mathrm{F}}(p_t \| q_t) \leq 0$$

C Additional Results

C.1 Adversarial Neuron Pruning

We further consider adversarial neuron pruning (ANP) [64], a pruning-based defense against backdoor attacks. Based on the assumption that neurons sensitive to adversarial perturbation are strongly related to the backdoor, ANP removes the injected backdoor by identifying and pruning such neurons. In [10], ANP is extended to the setting of diffusion models. Following [10], we apply ANP to defend against (targeted) DIFF2 on DDPM. We assume ANP has access to the full (clean) dataset and measure DIFF2's performance under varying ANP learning rates, with results summarized in Figure 11. We have the following interesting observations.

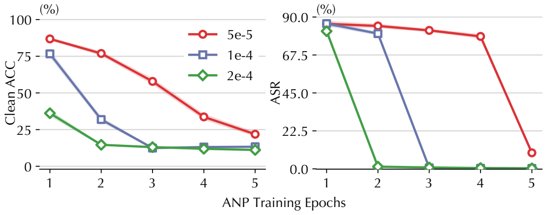

Figure 11: Effectiveness of ANP against DIFF2 (targeted attacks) under varying learning rate.

Overall, ANP is effective against DIFF2 but at the substantial cost of clean accuracy. For example, when ANP's learning rate is set at 5e-5, a decrease in ASR below 20% correlates with a significant drop in clean ACC, falling below 25%. This trade-off is even more evident for larger learning rates. For instance, at a learning rate of 2e-4, while ASR approaches nearly zero, clean ACC also plummets to around 10%. This can be explained as follows. In DIFF2's optimization (cf. Eq. 6), the backdoor diffusion process is deeply intertwined with the benign diffusion process. Recall that ANP attempts to eliminate the backdoor by separating neurons sensitive to the backdoor function. However, due to the entanglement between the backdoor and normal diffusion processes, pruning invariably affects the utility adversely.

C.2 Alternative Trigger Designs

Figure 12 visualizes inputs embedded with blending-based and warping-based triggers alongside clean inputs, their latents, and their purified counterparts. Note that these triggers produce less perceptible perturbations in the purified inputs compared to the patch-based triggers shown in Figure 8.

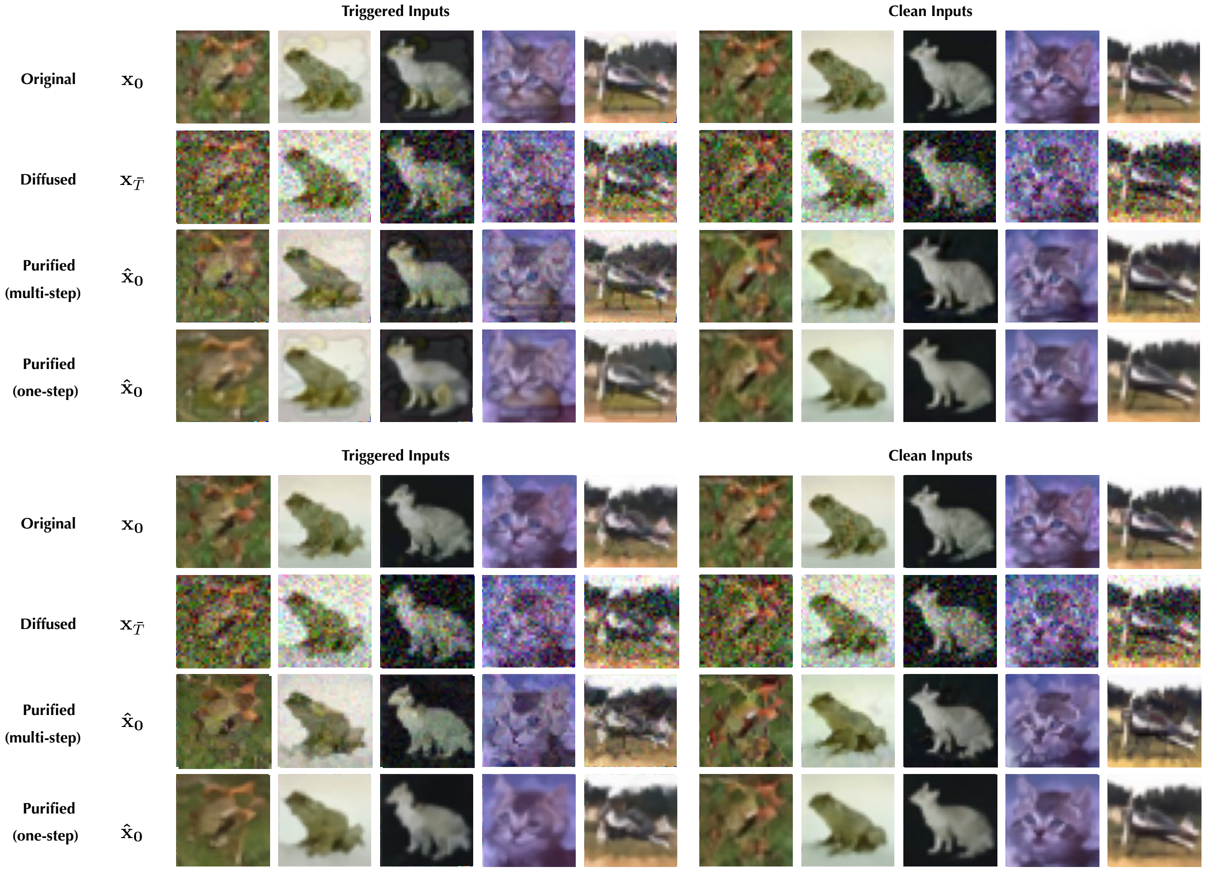

Figure 12: Original, diffused, and purified variants of clean and trigger inputs in DIFF2 with blending triggers (upper) and warping triggers (lower).