UNDERSTANDING AND ENHANCING THE ROBUSTNESS OF DIFFUSION-BASED PURIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion-Based Purification (DBP) has emerged as an effective defense mechanism against adversarial attacks. Traditionally, the efficacy of DBP has been attributed to the forward diffusion process, which narrows the distribution gap between clean and adversarial images through the addition of Gaussian noise. Although theoretical studies seem to support this explanation, to what extent it contributes to robustness remains unclear. In this paper, we argue that the inherent stochasticity in the DBP process is the primary driver of DBP robustness. To explore this, we introduce a novel Deterministic White-Box (DW-box) evaluation framework to assess robustness and analyze the attack trajectories and loss landscapes. Our findings suggest that DBP models primarily leverage stochasticity to evade effective attack directions, rather than directly neutralizing adversarial perturbations. To further enhance DBP robustness, we introduce Adversarial Denoising Diffusion Training (ADDT), which incorporates classifier-guided adversarial perturbations into diffusion training. Additionally, we propose Rank-Based Gaussian Mapping (RBGM) to make perturbations more compatible with the diffusion models. Empirical evidence demonstrates the effectiveness of ADDT.

1 INTRODUCTION

Deep learning has achieved remarkable success in various domains, including computer vision He et al. (2016), natural language processing OpenAI (2023), and speech recognition Radford et al. (2022). However, in this flourishing landscape, the persistent specter of adversarial attacks casts a shadow over the reliability of these neural models. Adversarial attacks for a vision model involve injecting imperceptible perturbations into input images to trick models into producing false outputs with high confidence Goodfellow et al. (2015); Szegedy et al. (2014). This inspires a large amount of research on adversarial defense Zhang et al. (2019); Samangouei et al. (2018); Shafahi et al. (2019a); Wang et al. (2023).

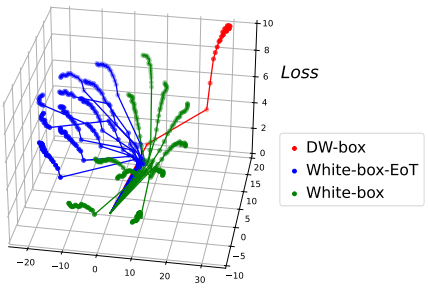

Figure 1: Comparison of attack trajectories under different evaluation settings. The visualization shows the effectiveness of attacks under the Deterministic White-box (DW-box) setting, while the attack trajectory under the standard White-box setting is less effective and deviates significantly from the DW-box trajectory.

Diffusion-based purification (DBP) Nie et al. (2022) has recently gained recognition as a powerful defense mechanism against a range of adversarial attacks, exploiting the capabilities of available diffusion models. The conventional view suggests that the robustness provided by DBP is primarily due to the forward diffusion process, which narrows the distribution gap between clean and adversarial images through the application of Gaussian noise Wang et al. (2022); Nie et al. (2022). However, although the reduction of the distribution gap is theoretically proven, its actual contribution to improving robustness has not been sufficiently investigated. Thus, it remains an open question of which factor contributes to DBP robustness most.

In light of this, we introduce an alternative perspective that highlights the role of stochasticity throughout the DBP process as a key contributor to its robustness, challenging the traditional focus

on the forward diffusion process. To evaluate the impact of stochasticity, we employ a Deterministic White-box (DW-box) attack setting where the attacker has complete knowledge of both the model parameters and the stochastic elements. Our findings reveal that DBP models significantly lose their robustness when the process is entirely deterministic to the attacker, thereby emphasizing the critical importance of stochasticity. Further investigations into attack trajectories and the loss landscape demonstrate that DBP models do not inherently counter adversarial perturbations as effectively as models trained with adversarial training (AT). Instead, they rely on stochasticity to circumvent the effective attack direction. This dependency is discussed in detail in Section 4.2 and visually depicted in Figure 1.

To enhance the robustness of DBP models, we introduce Adversarial Denoising Diffusion Training (ADDT). ADDT employs an iterative two-step approach: the Classifier-Guided Perturbation Optimization (CGPO) step generates adversarial perturbations, while the training step updates the diffusion model parameters using these perturbations. To align the perturbations more closely with the diffusion framework, we introduce Rank-Based Gaussian Mapping (RBGM), which adapts the adversarial perturbations to be more Gaussian-like.

Our main contributions are as follows:

- We present a novel perspective on DBP robustness, emphasizing the critical role of stochasticity and challenging the conventional belief that robustness primarily stems from reducing the distribution gap via the forward diffusion process.
- We introduce a new deterministic white-box attack setting and show that DBP models depend on stochastic elements to avoid effective attack directions, demonstrating distinct properties compared to models trained with Adversarial Training (AT).
- We develop Adversarial Denoising Diffusion Training (ADDT), which enhances DBP models' robustness, and introduce Rank-Based Gaussian Mapping (RBGM) to adapt perturbations to be more Gaussian-like, aligning them with the diffusion framework. Empirical validation confirms that ADDT achieves a robust accuracy improvement of up to 6% compared to conventional DBP models.

2 Related Work

Adversarial Training. First introduced by Madry et al. (2018), adversarial training (AT) seeks to develop a robust classifier by incorporating adversarial examples into the training process. It has nearly become the de facto standard for enhancing the adversarial robustness of neural networks Gowal et al. (2020); Rebuffi et al. (2021); Athalye et al. (2018). Recent advances in AT harness the generative power of diffusion models to augment training datasets and prevent AT from overfitting Gowal et al. (2021); Wang et al. (2023). However, the application of AT to DBP methods has not been thoroughly explored.

Adversarial Purification. Adversarial purification utilizes generative models to remove adversarial perturbation from inputs before they are processed by downstream models. Traditionally, generative adversarial networks (GANs) Samangouei et al. (2018) or autoregressive models Song et al. (2018) are employed as the purifier model. More recently, diffusion models have been introduced for adversarial purification, in a technique termed diffusion-based purification (DBP), and have shown promising results Song & Ermon (2019); Ho et al. (2020); Song et al. (2020a); Nie et al. (2022); Wang et al. (2022); Wu et al. (2022); Xiao et al. (2022). The robustness of DBP models is often attributed to the wash-out effect of Gaussian noise introduced during the forward diffusion process. Nie et al. (2022) propose that the forward process results in a reduction of the Kullback-Leibler (KL) divergence between the distributions of clean and adversarial images. Gao et al. (2022) suggest that while the forward diffusion process improves robustness by reducing model invariance, the backward process restores this invariance, thereby undermining robustness. However, these theories explaining the robustness of DBP models lack substantial experimental support.

3 Preliminaries

Adversarial Training. Adversarial training aims to create a robust model by including adversarial samples during training Madry et al. (2018). This approach can be formulated as a min-max problem,

where we first generate adversarial samples (the maximization) and then adjust the parameters to resist these adversarial samples (the minimization). Mathematically, this is represented as:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}} \left[\max_{\boldsymbol{\delta}\in B} L(f(\boldsymbol{\theta}, \boldsymbol{x}+\boldsymbol{\delta}), y)\right], \tag{1}$$

where L is the loss function, f is the classifier, $(\boldsymbol{x}, y) \sim \mathcal{D}$ denotes sampling training data from distribution $\mathcal{D}$, and B defines the set of permissible perturbation $\boldsymbol{\delta}$.

Diffusion Models. Denoising Diffusion Probabilistic Models (DDPM) Ho et al. (2020) and Denoising Diffusion Implicit Models (DDIM) Song et al. (2020a) simulate a gradual transformation in which noise is added to images and then removed to restore the original image. The forward process can be represented as:

$$\boldsymbol{x}_t = \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{2}$$

where $\boldsymbol{x}_0$ is the original image and $\boldsymbol{x}_t$ is the noisy image. $\overline{\alpha}_t$ is the cumulative noise level at step t ($1 < t \leq T$, where T is the number of diffusion training steps) . The model optimizes the parameters $\boldsymbol{\theta}$ by minimizing the distance between the actual and predicted noise:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}_0,t,\boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, t)\|_2^2\right], \tag{3}$$

where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ is the model's noise prediction, with $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$, we can predict $\hat{\boldsymbol{x}}_0$ in a single step:

$$\hat{\boldsymbol{x}}_0 = \left(\boldsymbol{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}(\boldsymbol{x}_t, t)\right) / \sqrt{\overline{\alpha}_t}, \tag{4}$$

where $\hat{\boldsymbol{x}}_0$ is the recovered image. DDPM typically takes an iterative approach to restore the image, removing a small amount of Gaussian noise at a time:

$$\hat{\boldsymbol{x}}_{t-1} = \left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}(\boldsymbol{x}_t, t)\right) / \sqrt{1-\beta_t} + \sqrt{\beta_t}\boldsymbol{\epsilon}, \tag{5}$$

where β_t is the noise level at step t, $\hat{\boldsymbol{x}}_{t-1}$ is the recovered image in step $t-1$, $\boldsymbol{\epsilon}$ is sampled from $\mathcal{N}(0, \boldsymbol{I})$. DDIM proposes to speed up the denoising process by skipping certain intermediate steps. Recent work suggests that DDPM may also benefit from a similar approach Nichol & Dhariwal (2021). Score SDEs Song et al. (2020b) give a score function view of DDPM and further lead to the derivations of DDPM++ (VPSDE) and EDM Karras et al. (2022).

Diffusion-Based Purification (DBP). DBP uses diffusion models to remove adversarial perturbation from images. Instead of using a complete diffusion process between the clean image and pure Gaussian noise (between $t = 0$ and $t = T$), they first diffuse $\boldsymbol{x}_0$ to a predefined timestep $t = t^*(t^* < T)$ via Equation (2), and recover the image $\hat{\boldsymbol{x}}_0$ via the reverse diffusion process in Equation (5).

4 Stochasticity-Driven Robustness

4.1 Stochasticity as the Main Factor of DBP Robustness

Traditional perspectives, as discussed in Section 2, primarily attribute the robustness of DBP models to the forward diffusion process, which introduces Gaussian noise into both clean and adversarial images, thereby reducing the difference between their distributionsWang et al. (2022); Nie et al. (2022). Although supported by theoretical studies, to what extent this contributes to the practical robustness remains an open question.

Considering the stochastic nature of DBP, we provide an alternative perspective that the robustness of DBP models may not only derive from the forward diffusion process but also depend on the stochastic elements integrated throughout the DBP process. To test this hypothesis, we evaluate the contributions of both the *forward diffusion process* and the *stochasticity throughout the processes* to the robustness of several DBP models. In particular, we focus on two models: DDPM, which incorporates Gaussian noise in both the forward and backward processes, and DDIM, which incorporates Gaussian noise in the forward process only. We adopt DDPM/DDIM within the DiffPure framework Nie et al. (2022) and denote the implementations by DP_{DDPM} and DP_{DDIM}, respectively.

To isolate the role of stochasticity, we introduce a new attack scenario: the **Deterministic White-Box** (DW-box) setting. This contrasts with the traditional White-box setting, where the attacker only has knowledge of the model parameters and must rely on random sampling for the stochastic elements involved in the evaluation. In the DW-box setting, the attacker has full knowledge of both the model

parameters and the specific states of stochastic elements used in the evaluation. This setting makes the DBP process deterministic to the attacker. We further differentiate the settings based on the attacker's knowledge of the stochastic elements: no knowledge (White-box), full knowledge in the forward process (DW_{Fwd}-box), and full knowledge in both the forward and reverse processes (DW_{Both}-box). We detail the differences between these settings in **??**.

According to the conventional understanding, which attributes the robustness of DBP models solely to the forward diffusion process, attacks on DP_{DDPM} and DP_{DDIM} under the DW_{Fwd}-box setting should yield similar outcomes. Conversely, from our perspective, which emphasizes the importance of stochasticity throughout the processes, we expect that an attack on DP_{DDIM} with the DW_{Fwd}-box setting would produce similar results to an attack on DP_{DDPM} with the DW_{Both}-box setting since both models would be completely deterministic for the attacker. To test our hypothesis, we conduct rigorous evaluations using precise gradients and Expectation over Transformation (EoT) techniques, as detailed in Section 6.1. We employ a 20-step PGD attack with 10 EoT iterations (l_∞ norm) on the CIFAR-10 dataset. Further experimental details are discussed in Section 6.2. The results, depicted in Figure 2, reveal that while DP_{DDPM} maintains its robustness in the DW_{Fwd}-box setting, DP_{DDIM} significantly loses its robustness. This finding challenges the traditional view that narrowing the distribution gap during the forward process enhances robustness. Moreover, in the DW_{Both} setting, the robustness of DP_{DDPM} also deteriorates, which supports our hypothesis that the robustness of DBP models depends on stochasticity throughout the DBP processes.

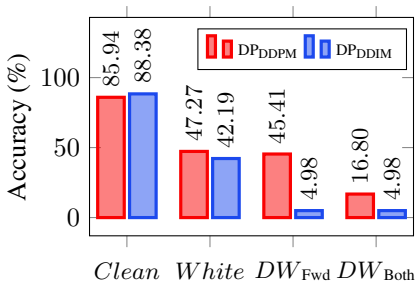

Figure 2: DP_{DDPM} and DP_{DDIM} robustness under different attack settings. Both models lost most of their robustness only when the attacker knows all stochastic elements (DW_{Both}-box for DP_{DDPM} and DW_{Fwd}-box for DP_{DDIM}).

Based on these results, we argue that the robustness of DBP models may primarily stem from the stochasticity throughout the DBP process. Specifically, DBP models may exploit stochasticity to circumvent the most effective attack direction. This perspective complements the ideas in previous work on DBP robustness Xiao et al. (2022); Carlini et al. (2022).

Table 1: Evaluation of state-of-the-art DBP methods, EoT significantly influences the evaluation accuracy (%) of model robustness.

	DiffPure	GDMP (MSE)	DP_{DDPM}	DP_{DDIM}
Clean	89.26	91.80	85.94	88.38
PGD20-EoT1	69.04	53.13	60.25	54.59
PGD20-EoT10	55.96	40.97	47.27	42.19

4.2 Explaining Stochasticity-Driven Robustness

In the previous section, we highlight the significance of stochasticity in the robustness of DBP models. To further investigate the impact of stochasticity on traditional White-box attacks, we analyze the robustness of various DBP methods such as DiffPure, GDMP, DP_{DDPM}, and DP_{DDIM} under two conditions: with 10 EoT iterations (EoT10) and without EoT (EoT1). Our results, detailed in Table 1, show that under White-box attacks, DBP models retain significant robustness and that the introduction of EoT leads to only a marginal reduction. This is in contrast to the results obtained for DW-box attacks.

To further understand robustness performance under different attack settings, we visualize attack trajectories using t-SNE to map them onto the *xy* plane, with corresponding loss values on the *z* axis. We label the PGD20-EoT10/PGD20-EoT1 attacks within the White-box setting as

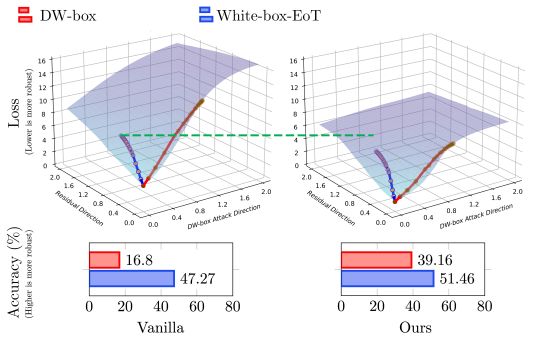

Figure 3: Visualisation of attack trajectories for White-box-EoT attacks and Deterministic White-box attacks on the loss landscape. The loss landscape is not flat in the direction of the Deterministic White-box attack. It is based on the first 128 images of CIFAR-10.

White-box-EoT/White-box and the PGD20-EoT1 attack within the DW-box setting as DW-box. As shown in Figure 1, the trajectories are scattered in all settings, highlighting the stochastic nature of DBP models. However, DW-box attacks result in a large increase in loss, whereas White-box attacks show a mild increase. This suggests that DBP models may rely on stochasticity to evade effective attack directions. In the White-box-EoT setting, trajectories are more concentrated but diverge from the DW-box attack trajectory, explaining why EoT attacks fail to compromise this defense.

From a loss landscape perspective, we suggest that the loss landscape of DBP models is not inherently smooth. We visualize a White-box-EoT attack and a deterministic White-box attack trajectory on the loss landscape Li et al. (2018); Kim et al. (2021). Experimental details are presented in Appendix F. As shown in the left side of Figure 3, the trajectory of the White-box-EoT attack diverges from the Deterministic White-box attack direction, and the loss landscape appears flatter in this direction. This divergence is possibly due to the failure of the White-box-EoT attack to identify the most effective direction, supporting the findings from the previous trajectory visualization of Figure 1. Besides, the significant increase in loss along the direction of the Deterministic White-box attack differs from the case of AT-trained models, where the landscape may be uniformly flat for all directions Shafahi et al. (2019a). Further evaluation with up to 512 EoT iterations, discussed in Appendix A, confirms these observations.

5 Adversarial Denoising Diffusion Training

In our previous discussions, we note that the robustness of DBP models is primarily due to their reliance on stochastic elements to evade the most effective attack direction, rather than directly countering adversarial perturbations like AT models.

To improve the robustness of DBP models, we advocate the integration of adversarial perturbations into the training process. To align these perturbations with the training of the diffusion model, we introduce **Rank-Based Gaussian Mapping** (RBGM), a technique designed to make the perturbations more "Gaussian-like", which is elaborated in Section 5.1. Building on this foundation, we introduce **Adversarial Denoising Diffusion Training** (ADDT). ADDT uses an iterative two-step approach: the **Classifier-Guided Perturbation Optimization** (CGPO) step generates adversarial perturbations, while the training step trains on these perturbations and updates the diffusion model parameters. This process is illustrated graphically in Figure 5 and the pseudocode is in Appendix G. ADDT allows for efficient fine-tuning of clean diffusion models to proactively counter adversarial perturbations, with a discussion of its efficiency presented in Appendix P. Further details on ADDT are discussed in Section 5.

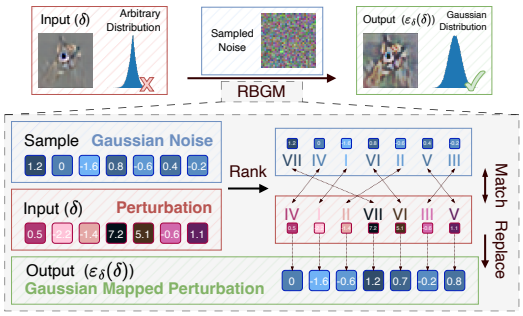

Figure 4: Rank-Based Gaussian Mapping. RBGM trims the input to follow Gaussian distribution. It samples a Gaussian noise and then replaces elements in the input with those from the Gaussian noise, matched according to their respective ranks.

5.1 Rank-Based Gaussian Mapping

Traditional diffusion models operate under the premise that input images are compromised by independent Gaussian noise $\boldsymbol{\epsilon}$, as elucidated in Equation (2). This assumption limits our ability to incorporate perturbations during training that could potentially improve model robustness. To circumvent this limitation, we propose to employ a modified noise vector, $\boldsymbol{\epsilon}'$, conditioned on the input $\boldsymbol{x}$. This modification allows us to tailor $\boldsymbol{\epsilon}'$ to possess adversarial properties while maintaining an approximately Gaussian conditional distribution.

To make the perturbations more Gaussian-like, we introduce Rank-Based Gaussian Mapping (RBGM), illustrated in Figure 4. The RBGM approach, denoted by $\boldsymbol{\epsilon}_{\boldsymbol{\delta}}()$, takes $\boldsymbol{\delta}$ as input and aims to preserve the relative magnitudes of the elements of $\boldsymbol{\delta}$ while ignoring their exact values. The RBGM process involves sampling a Gaussian tensor, $\boldsymbol{\epsilon}_s$, that matches the dimensions of $\boldsymbol{\delta}$. By replacing the elements of $\boldsymbol{\delta}$ with the elements of $\boldsymbol{\epsilon}_s$ of the same rank, RBGM makes $\boldsymbol{\delta}$ more Gaussian-like. To further

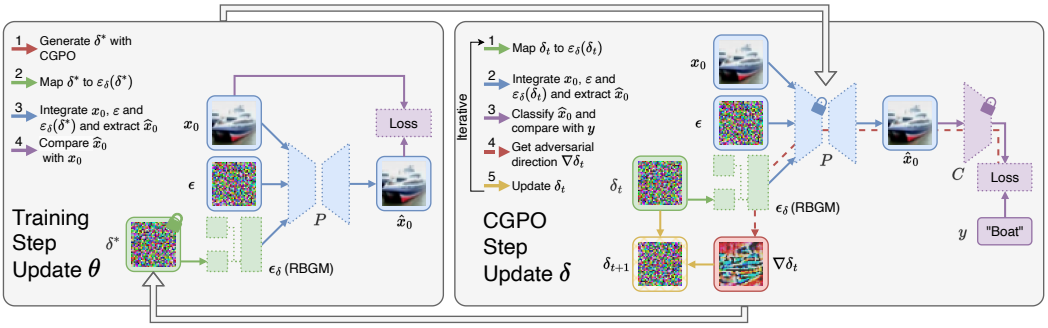

Figure 5: Overview of Adversarial Denoising Diffusion Training (ADDT). ADDT alternates between a CGPO step (right grey box) to refine the perturbations with a frozen diffusion model and classifier, and a training step (left grey box) to update the diffusion model with the refined perturbation. Throughout the process, RBGM is used to make the perturbation more "Gaussian-like".

augment the Gaussian nature of the noise, we mix the RBGM-mapped perturbation with random Gaussian noise.

By combining the RBGM-induced perturbation with Gaussian noise, we could generate an adversarial input $\boldsymbol{x}'_t$ as follows:

$$\boldsymbol{x}'_t(\boldsymbol{x}_0, \boldsymbol{\epsilon}, \boldsymbol{\epsilon}_{\boldsymbol{\delta}}(\boldsymbol{\delta})) = \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\lambda_t^2}\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon} + \lambda_t\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_{\boldsymbol{\delta}}(\boldsymbol{\delta}), \tag{6}$$

where λ_t modulates the level of adversarial perturbations, ensuring that the overall noise remains largely independent of $\boldsymbol{x}_0$ and that the perturbations do not overwhelm the denoising model's learning capabilities. We determine λ_t using the following formulation:

$$\lambda_t = \texttt{clip}(\gamma_t\lambda_{unit}, \lambda_{min}, \lambda_{max}), \quad \gamma_t = \frac{\sqrt{\overline{\alpha}_t}}{\sqrt{1-\overline{\alpha}_t}}, \tag{7}$$

where the $\texttt{clip}$ function limits λ_t between λ_{min} and λ_{max}.

5.2 Adversarial Denoising Diffusion Training

Training Step. The training step is shown to the left of Figure 5. The goal of model optimization is to subtract the Gaussian noise and the RBGM-mapped perturbations to fully recover the image, this could be denoted as:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}\left[\frac{\sqrt{\overline{\alpha}_t}}{\sqrt{1-\overline{\alpha}_t}}\|\boldsymbol{x}_0 - \boldsymbol{P}(\boldsymbol{\theta}, \boldsymbol{x}'_t, t)\|_2^2\right], \tag{8}$$

where the expectation is taken with respect to $\boldsymbol{x}_0 \sim \mathcal{D}, t \sim \mathcal{U}(\{1, ..., T\}), \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. The noise mixture $\boldsymbol{x}'_t$ is derived from $\boldsymbol{x}_0$, $\boldsymbol{\epsilon}$, and $\boldsymbol{\delta}$ according to Equation (6), $\boldsymbol{P}$ is a single-step diffusion model, it takes $\boldsymbol{x}'_t$ and t to predict the original image $\hat{\boldsymbol{x}}_0$. Here, instead of adopting multi-step reconstruction techniques, we use a single-step recovery method. For DDPM and DDIM, this process is governed by Equation (4). Also note that since the DDPM/DDIM loss is the expectation of the square error of noise, we add a factor $\sqrt{\overline{\alpha}_t}/\sqrt{1-\overline{\alpha}_t}$ to keep our loss in line with the traditional DDPM/DDIM loss.

CGPO Step. The objectives of generative models are distinct from those of downstream perceptual tasks. To generate perturbations that are meaningful for perceptual tasks, we propose the incorporation of classifier guidance into the generation of perturbations with Classifier-Guided Perturbation Optimization (CGPO)[1]. As shown on the right side of Figure 5, we first reconstruct a clean image $\hat{\boldsymbol{x}}_0$ from $\boldsymbol{x}'_t$ with the single-step diffusion model $\boldsymbol{P}$, we then classify the image with a pre-trained classifier C. The goal of CGPO is to refine $\boldsymbol{\delta}$ to maximize the classification error:

$$\boldsymbol{\delta}^* = \arg\max_{\boldsymbol{\delta}} \mathbb{E}\left[L(C(\boldsymbol{P}(\boldsymbol{\theta}, \boldsymbol{x}'_t(\boldsymbol{x}_0, \boldsymbol{\epsilon}, \boldsymbol{\epsilon}_{\boldsymbol{\delta}}(\boldsymbol{\delta})), t)), y)\right]. \tag{9}$$

To optimize $\boldsymbol{\delta}$, we employ an iterative process of gradient accumulation, detailed in Appendix G. Given the non-differentiable nature of RBGM, we accumulate gradients with respect to $\nabla_{\boldsymbol{\epsilon}_{\boldsymbol{\delta}}(\boldsymbol{\delta})}$ rather than $\nabla_{\boldsymbol{\delta}}$, which is sufficient for training. In RBGM, we process each channel of every image independently.

[1]Here we use the classifier only for semantic guidance, and the classifier in CGPO doesn't need to be consistent with the protected model, see Section 6.4.

6 EXPERIMENTS

6.1 EVALUATING DIFFUSION-BASED PURIFICATION ROBUSTNESS

Previous assessments of DBP robustness have often utilized potentially unreliable methods. In particular, due to the iterative denoising process in diffusion models, some studies resort to mathematical approximations of gradients to reduce memory constraints Athalye et al. (2018) or to circumvent the diffusion process during backpropagation Wang et al. (2022). Furthermore, the reliability of AutoAttack, a widely used evaluation method, in assessing the robustness of DBP models is questionable. Although AutoAttack includes a *Rand* version designed for stochastic models, Nie *et al.* have found instances where the *Rand* version is less effective than the *Standard* version in evaluating DBP robustness Nie et al. (2022).

Table 2: Clean and robust accuracy (%) on CIFAR-10 obtained by different DBP methods. All methods show consistent improvement fine-tuned with ADDT.

Diffusion model	DBP model	*Clean*	l_∞	l_2
-	-	95.12	0.00	1.46
DDIM	DP_{DDIM}	88.38	42.19	70.02
	DP_{DDIM}+Ours	88.77	**46.48**	**71.19**
DDPM	GDMP (No Guided) Wang et al. (2022)	91.41	40.82	69.63
	GDMP (MSE) Wang et al. (2022)	91.80	40.97	70.02
	GDMP (SSIM) Wang et al. (2022)	92.19	38.18	68.95
	DP_{DDPM}	85.94	47.27	69.34
	DP_{DDPM}+Ours	85.64	**51.46**	**70.12**
DDPM++	COUP Zhang et al. (2024)	90.33	50.78	71.19
	DiffPure	89.26	55.96	75.78
	DiffPure+Ours	89.94	**62.11**	**76.66**
EDM	DP_{EDM} (Appendix I)	86.43	62.50	76.86
	DP_{EDM}+Ours (Appendix I)	86.33	**66.41**	**79.16**

To improve the robustness evaluation of diffusion-based purification (DBP) models, we implement several modifications. First, to ensure the accuracy of the gradient computations, we compute the exact gradient of the entire diffusion classification pipeline. To mitigate the high memory requirements in the iterative denoising steps, we use gradient checkpointing techniques to optimize memory usage. In addition, to deal with the stochastic nature of the DBP process, we incorporate the Expectation over Transformation (EoT) method to average gradients across different attacks. We adopt EoT with 10 iterations, and a detailed discussion of the choice of EoT iterations can be found in Appendix A. We also use the Projected Gradient Descent (PGD) attack instead of AutoAttack for our evaluations[2]. Our revised robustness evaluation revealed that DBP models, such as DiffPure and GDMP, perform worse than originally claimed. DiffPure's accuracy dropped from a claimed 70.64% to an actual 55.96%, and GDMP's from 90.10% to 40.97%. These results emphasize the urgent need for more accurate and reliable evaluation methods to properly assess the robustness of DBP models.

Table 3: Clean and robust accuracy (%) on DP_{DDPM}. ADDT improve robustness across different NFE, especially at lower NFE (*: default DDPM generation setting; -: classifier only).

Dataset	NFE	Vanilla			Ours		
		Clean	l_∞	l_2	*Clean*	l_∞	l_2
CIFAR-10	-	95.12	0.00	1.46	95.12	0.00	1.46
	5	49.51	21.78	36.13	59.96	**30.27**	**41.99**
	10	73.34	36.72	55.47	78.91	**43.07**	**62.97**
	20	81.45	45.21	65.23	83.89	**48.44**	**69.82**
	50	85.54	46.78	68.85	85.45	**50.20**	**69.04**
	100*	85.94	47.27	69.34	85.64	**51.46**	**70.12**
CIFAR-100	-	76.66	0.00	2.44	76.66	0.00	2.44
	5	17.29	3.71	9.28	21.78	**6.25**	**13.77**
	10	34.08	10.55	19.24	40.62	**14.55**	**27.25**
	20	48.05	17.68	30.66	53.32	**18.65**	**36.13**
	50	55.57	20.02	37.70	59.47	**22.75**	**40.72**
	100*	57.52	20.41	37.89	59.18	**23.73**	**41.70**

6.2 EXPERIMENT SETUP

Classifier. We train a WideResNet-28-10 classifier for 200 epochs following the methods in Yoon et al. (2021); Wang et al. (2022) and achieve 95.12% accuracy on CIFAR-10 and 76.66% on CIFAR-100 dataset.

DBP. For DBP forward process timestep, DiffPure employs a continuous-time VPSDE (DDPM++) model, selecting $t^* = 0.1$. For discrete timesteps in Equation (2) of DP_{DDPM} and DP_{DDIM} , we set $t^* = 0.1 \times T$. We also implement DP_{EDM}. The details are discussed in Appendix I.

ADDT. We fine-tune our diffusion model based on the CIFAR-10 pre-trained exponential moving average (EMA) clean model closely following its training setting Ho et al. (2020) (transformed into Huggingface Diffusers format by Fang et al. (2023)). The CIFAR-100

[2]We discover a bug in the *Rand* version of AutoAttack that causes it to overestimate the robustness of DBP. After fixing this, AutoAttack gives similar results to PGD attacks, but at a much higher computational cost. We discuss this in detail in Appendix M.

clean model is fine-tuned from the CIFAR-10 clean model for 100 epochs. ADDT models are fine-tuned from clean pre-trained models for 100 epochs, with guidance from the WideResNet-28-10 classifier. In CGPO we adopt $\lambda_{unit} = 0.03, \lambda_{min} = 0, \lambda_{max} = 0.3$ and refine $\boldsymbol{\delta}$ for 5 iterations. Fine-tuning from clean models with ADDT is quite efficient, taking 12 hours on four NVIDIA GeForce RTX 2080ti GPUs for DP_{DDPM}/DP_{DDIM} models.

Robustness Evaluation. We adopt PGD20-EoT10 attack for robustness evaluation. For l_∞ attacks we set $\alpha = 2/255$ and $\epsilon = 8/255$; for l_2 attacks we set $\alpha = 0.1$ and $\epsilon = 0.5$. Due to computational constraints, we test the first 1024 images from the CIFAR-10/100 datasets. Note that it still takes 5 hours on four NVIDIA GeForce RTX 2080ti GPUs to test DP_{DDPM}/DP_{DDIM}.

Table 4: Clean and robust accuracy (%) on CIFAR-10, obtained by different classifiers. ADDT (WRN-28-10 guidance) improves robustness in protecting different subsequent classifiers. (*: the classifier used in ADDT fine-tuning).

Method	Classifier	Vanilla			Ours		
		Clean	l_∞	l_2	*Clean*	l_∞	l_2
$\text{DP}_{\text{DDPM-1000}}$	VGG-16 Simonyan & Zisserman (2014)	84.77	41.99	66.89	85.06	**46.09**	**67.87**
	ResNet-50 He et al. (2016)	83.11	44.04	67.58	83.84	**48.14**	**67.87**
	WRN-28-10* Zagoruyko & Komodakis (2016)	85.94	47.27	69.34	85.64	**51.46**	**70.12**
	WRN-70-16 Zagoruyko & Komodakis (2016)	88.43	48.93	70.31	87.84	**52.54**	**70.70**
	ViT-B Dosovitskiy et al. (2020)	85.45	45.61	69.53	85.25	**48.63**	**69.92**
$\text{DP}_{\text{DDIM-100}}$	VGG-16 Simonyan & Zisserman (2014)	87.16	29.00	61.82	87.55	**35.06**	**66.11**
	ResNet-50 He et al. (2016)	86.04	31.74	62.11	86.57	**38.77**	**65.82**
	WRN-28-10* Zagoruyko & Komodakis (2016)	88.96	43.16	67.58	88.18	**47.85**	**70.61**
	WRN-70-16 Zagoruyko & Komodakis (2016)	84.40	39.16	68.36	84.96	**47.66**	**69.14**
	ViT-B Dosovitskiy et al. (2020)	88.77	34.38	65.72	88.48	**41.02**	**68.65**
DP_{EDM}	WRN-28-10* Zagoruyko & Komodakis (2016)	86.43	62.50	76.86	86.33	**66.41**	**79.16**
	WRN-70-16 Zagoruyko & Komodakis (2016)	86.62	65.62	76.46	86.43	**69.63**	**78.91**

6.3 Comparison with State-of-the-Art Approaches

We apply ADDT fine-tuning to a set of diffusion models and apply DiffPure-style DBP with the refined models. We then compare their performance with state-of-the-art DBP methods on the CIFAR-10 dataset. The outcomes, detailed in Table 2, reveal that ADDT fine-tuning enhances the robustness of these models, enabling them to reach state-of-the-art performance.

6.4 Defense Performance in Extensive Scenarios

Performance on Different Classifiers. We evaluate the cross model protection ability of ADDT fine-tuned models, as shown in Table 4. The results indicate that fine-tuning with WRN-28-10 guided ADDT could enhance the ability to protect different classifiers. Notably, using DP_{EDM}, we achieve 69.63% l_∞ robust accuracy on a WRN-70-16 classifier. This demonstrates that our method, without classifier-specific fine-tuning, can achieve comparable results to state-of-the-art AT-based models.

Table 5: Clean and robust accuracy (%) on CIFAR-10 fine-tuned with different methods. Vanilla fine-tuning and fine-tuning with non-classifier-guided perturbations did not improve robustness.

Method	NFE	Vanilla Fine-tune Only			MSE Distance Guided		
		Clean	l_∞	l_2	*Clean*	l_∞	l_2
DP_{DDPM}	5	47.27	21.88	33.11	49.32	21.19	35.74
	10	71.58	34.77	52.64	73.34	36.43	55.96
	20	81.93	42.29	63.77	83.79	42.87	66.89
	50	84.18	47.56	65.53	85.16	47.27	67.87
	100*	85.25	47.27	68.26	86.91	46.97	70.80
DP_{DDIM}	5	89.26	41.11	67.87	89.45	39.36	67.58
	10*	88.87	41.41	67.19	89.36	40.92	67.68
	20	89.45	40.23	67.77	88.48	42.68	68.55
	50	88.77	42.09	67.38	88.18	41.02	68.16
	100	88.18	40.62	68.26	89.26	40.33	68.36

Performance under Acceleration. Speeding up the diffusion process by omitting intermediate steps has become common practice in the use of diffusion models Nichol & Dhariwal (2021); Song et al. (2020a). Here we evaluate the robustness of accelerated DBP models. Acceleration is measured by the number of neural function evaluations (NFE), which indicates the number of evaluation steps performed during the DBP backtracking process. For our experiments, we set $t^* = 0.1 \times T$ and accelerate the process by excluding intermediate time steps. For example, with an NFE of 5, the time steps for the DBP backward process would be $t = [100, 80, 60, 40, 20, 0]$. The results are detailed in Table 3. Our method improves the robustness of both DP_{DDPM} models. Note that the performance of DP_{DDPM} varies significantly between different NFE. This may be due to the fact that DDPM introduces stochasticity (Gaussian noise) at each reverse step; with fewer reverse steps, it's stochasticity decreases. Additionally, the generation capability of DDPM is sensitive to the skipping of intermediate steps. We also conducted an evaluation of DP_{DDIM} models, as detailed in Appendix J.

Table 6: Clean and robust accuracy (%) on Tiny-ImageNet with WRN-28-10 classifier. ADDT improve DBP robustness on Tiny-ImageNet (-: classifier only).

Method	Vanilla			Ours		
	Clean	l_∞	l_2	*Clean*	l_∞	l_2
-	71.37	0.00	0.00	-	-	-
$DP_{DDPM-1000}$	57.13	11.82	46.68	56.15	**13.57**	**48.54**
$DP_{DDIM-100}$	60.35	4.79	39.75	60.45	**5.86**	**40.82**
DP_{EDM}	57.03	19.14	46.00	56.45	**20.61**	**47.95**

Table 7: FID score of DDPM for CIFAR-10 fine-tuned to different perturbations (the lower the better). Fine-tuning with RBGM-mapped perturbations yields lower FID scores than l_∞ perturbations.

	Vanilla	Clean Fine-tune	Ours	Ours_{l_∞}
FID	3.196	3.500	5.190	13.608

Performance on Tiny-ImageNet. We test DBP robustness on Tiny-ImageNet dataset Le & Yang (2015) in Table 6, with training and evaluation settings following CIFAR-10. The result shows that ADDT improves robustness on Tiny-ImageNet dataset.

6.5 Ablation Study and Analysis

RBGM. We compare the generative ability of diffusion models fine-tuned with RBGM-mapped and l_∞ perturbations by comparing their Fréchet Inception Distance (FID) scores, as shown in Table 7.The results show that diffusion models fine-tuned with RBGM-mapped perturbations maintain generation quality comparable to the vanilla diffusion model, while models fine-tuned with l_∞ perturbations show degraded performance. We also observe that training with RBGM-mapped perturbations generalized better to different attacks. Experimental details and additional tests are presented in Appendix L.

CGPO. We examine the effect of vanilla fine-tuning and ADDT fine-tuning with MSE distance guidance (without classifier guidance) in Table 5. Neither method improves the robustness of DBP models.

Revisiting DBP Robustness. We re-examine robustness under the Deterministic White-box setting by evaluating the performance of diffusion models after ADDT fine-tuning, as shown in Figure 6. The fine-tuned models show significantly higher robustness under the Deterministic White-box setting, indicating that our method makes the attack more difficult without relying on the evasion effect of stochastic elements. We provide a more detailed experiment considering different NFE in Appendix N. We also compare the loss landscapes of ADDT fine-tuned models and vanilla diffusion models, as shown in Figure 3. This comparison shows that our method effectively smooths the loss landscape of DBP models.

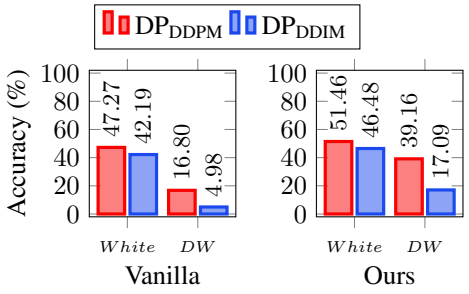

Figure 6: Revisiting robustness under Deterministic White-box setting. ADDT improves robustness under both White-box and Deterministic White-box setting, implying that ADDT strengthens the models' ability to counter adversarial inputs.

7 Conclusion

This study offers a new perspective on the robustness of Diffusion-Based Purification (DBP) models, emphasizing the crucial role of stochasticity and challenging the traditional view that robustness is mainly derived from minimizing the distribution gap through the forward diffusion process. We introduce a Deterministic white-box (DW-box) attack scenario and show that DBP models are based on stochastic elements to evade effective attack directions. To further enhance the robustness of DBP models, we have developed Adversarial Denoising Diffusion Training (ADDT) and Rank-Based Gaussian Mapping (RBGM). ADDT integrates adversarial perturbations into the training process, while RBGM trims perturbations to more closely resemble Gaussian distributions. Our empirical results confirm that ADDT achieves robustness improvements of up to 6% over conventional DBP models, underscoring the effectiveness of our proposed enhancements.

REFERENCES

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pp. 274–283. PMLR, 2018.

Nicholas Carlini, Florian Tramer, Krishnamurthy Dj Dvijotham, Leslie Rice, Mingjie Sun, and J Zico Kolter. (certified!!) adversarial robustness for free! In *The Eleventh International Conference on Learning Representations*, 2022.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pp. 1310–1320. PMLR, 2019.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. *arXiv preprint arXiv:2305.10924*, 2023.

Yue Gao, Ilia Shumailov, Kassem Fawaz, and Nicolas Papernot. On the limitations of stochastic pre-processing defenses. *Advances in Neural Information Processing Systems*, 35:24280–24294, 2022.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6572.

Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy A. Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *CoRR*, abs/2010.03593, 2020. URL https://arxiv.org/abs/2010.03593.

Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Hoki Kim, Woojin Lee, and Jaewook Lee. Understanding catastrophic overfitting in single-step adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8119–8127, 2021.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.

Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning (ICML)*, 2022.

OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. URL https://api.semanticscholar.org/CorpusID:257532815.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *ArXiv*, abs/2212.04356, 2022. URL https://api.semanticscholar.org/CorpusID:252923993.

Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A. Mann. Fixing data augmentation to improve adversarial robustness. *ArXiv*, abs/2103.01946, 2021. URL https://api.semanticscholar.org/CorpusID:232092181.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Neural Information Processing Systems*, 2019a. URL https://api.semanticscholar.org/CorpusID:139102395.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019b.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv:2010.02502*, October 2020a. URL https://arxiv.org/abs/2010.02502.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJUYGxbCW.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6199.

Jinyi Wang, Zhaoyang Lyu, Dahua Lin, Bo Dai, and Hongfei Fu. Guided diffusion model for adversarial purification. *arXiv preprint arXiv:2205.14969*, 2022.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. In *International Conference on Machine Learning (ICML)*, 2023.

Quanlin Wu, Hang Ye, and Yuntian Gu. Guided diffusion model for adversarial purification from random noise. *arXiv preprint arXiv:2206.10875*, 2022.

Chaowei Xiao, Zhongzhu Chen, Kun Jin, Jiongxiao Wang, Weili Nie, Mingyan Liu, Anima Anandkumar, Bo Li, and Dawn Song. Densepure: Understanding diffusion models towards adversarial robustness. *arXiv preprint arXiv:2211.00322*, 2022.

Jongmin Yoon, Sung Ju Hwang, and Juho Lee. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pp. 12062–12072. PMLR, 2021.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

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

Mingkun Zhang, Jianing Li, Wei Chen, Jiafeng Guo, and Xueqi Cheng. Classifier guidance enhances diffusion-based adversarial purification by preserving predictive information, 2024. URL https://openreview.net/forum?id=qvLPtx52ZR.

A Influence of EoT Iterations on DBP Robustness Evaluation

In this section, we examine how the number of EoT iterations influences the DBP robustness evaluation. As previously discussed in Section 4.1, the Deterministic White-box attack could find the most effective attack direction. To quantify the impact of EoT iterations, we compare the attack direction of the standard White-box-EoT across various numbers of EoT iterations with that of the Deterministic White-box.

See Figure 7 for a visual explanation, where the red line shows the DBP accuracy after attack, and the blue line shows the similarity between the attack directions of the White-box-EoT and Deterministic White-box. The trend is clear: more EoT iterations lead to greater similarity and lower model accuracy, the rate of increase in similarity and the rate of decrease in accuracy both tend to slow down with further iterations.

Balancing computational cost and evaluation accuracy, we chose the PGD20-EoT10 configuration for our robustness evaluation.

B Evaluation with stronger PGD+EoT attack

To balance computational feasibility with attack effectiveness, we chose a configuration of PGD20+EoT10 for our evaluations. As shown in the table below, increasing the iterations to PGD200+EoT20 results in a moderate decline in accuracy but significantly increases computational time. For instance, testing the DP_{DDPM} model (the smallest model in our study) took 50 hours on eight 3090 GPUs, making it impractical to conduct all experiments within a reasonable timeframe.

While higher iterations of PGD and EoT provide a more rigorous test of adversarial resilience, our primary conclusions about the origin of DBP robustness and the comparative performance of ADDT

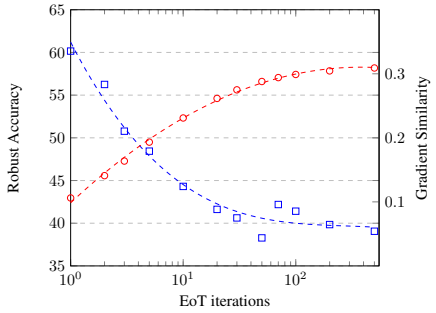

Figure 7: Robust accuracy and gradient similarity on DP_{DDPM} for CIFAR-10, obtained by different EoT iterations. As the number of EoT iterations increases, the gradient similarity between the White-box-EoT attack direction and the Deterministic White-box attack direction increases and the robust accuracy decreases.

remain consistent regardless of the specific PGD and EoT parameters used. Therefore, we opted for the PGD20+EoT10 setting to ensure a practical yet effective evaluation.

Method	**PGD200+EoT20**		**PGD20+EoT10**	
	Vanilla (ℓ_∞)	**Ours** (ℓ_∞)	**Vanilla** (ℓ_∞)	**Ours** (ℓ_∞)
DP_DDPM	41.02%	46.19%	47.27%	51.46%
DP_DDIM	36.23%	41.11%	43.16%	47.85%
DiffPure	48.93%	55.76%	55.96%	62.11%

Table 8: Comparison of PGD and EoT iterations on DBP robustness.

C Impact of Attackers' Knowledge on Robustness: Comparison of Attack Settings

This appendix delves into the influence of varying levels of attackers' knowledge about the stochastic components in diffusion processes on the robustness of diffusion-based models. We specifically assess the individual contributions of the forward and reverse diffusion processes to model robustness across different attack scenarios.

C.1 Stochastic Elements in the Diffusion Processes

To elucidate the impact of the attacker's knowledge, it is crucial to understand the stochastic elements integral to the diffusion processes, which are pivotal for the model's robustness.

In the **forward diffusion process**, Gaussian noise is incorporated into the input data to derive a noisy version x_t:

$$x_t = \sqrt{\bar{\alpha}_t}\, x + \sqrt{1-\bar{\alpha}_t}\, \epsilon_f, \tag{10}$$

where $\epsilon_f \sim \mathcal{N}(0, I)$ is sampled once per input.

In the **reverse diffusion process**, the model progressively denoises x_t through iterative steps. For the Denoising Diffusion Probabilistic Model (DDPM), the reverse process is inherently stochastic:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}t}}, \epsilon\theta(x_t, t) \right) + \sigma_t \epsilon_t, \tag{11}$$

where $\epsilon_t \sim \mathcal{N}(0, I)$ is sampled at each reverse step. In contrast, for the Denoising Diffusion Implicit Model (DDIM), the reverse process is deterministic, and no noise $\{\epsilon_t\}_{t=1}^T$ is added.

C.2 Attack Settings and Attacker Knowledge

We delineate four distinct attack scenarios, each characterized by the extent of information available to the attacker, particularly concerning the Gaussian noise variables in the diffusion processes. Table 9 provides a summary of the attacker's knowledge in each scenario.

Table 9: Information accessible to the attacker in different attack settings. ϵ_f denotes the Gaussian noise in the forward process, and $\{\epsilon_t\}_{t=1}^T$ represents the Gaussian noise in the reverse process.

Attacker's Knowledge	**White-box**	**DW$_{\text{Fwd}}$**	**DW$_{\text{Rev}}$**	**DW$_{\text{Both}}$**
Model Architecture and Parameters	✓	✓	✓	✓
Input Images and Class Labels	✓	✓	✓	✓
Forward Process Noise ϵ_f	×	✓	×	✓
Reverse Process Noise $\{\epsilon_t\}_{t=1}^T$	×	×	✓	✓

In the conventional white-box attack setting, the attacker possesses comprehensive knowledge of the model architecture and parameters but lacks insight into the stochastic elements used during inference (ϵ_f and $\{\epsilon_t\}_{t=1}^T$). The DW$_{\text{Fwd}}$ setting grants the attacker knowledge of the Gaussian noise in the forward diffusion process (ϵ_f). Conversely, the DW$_{\text{Rev}}$ setting provides the attacker with knowledge of the Gaussian noise introduced during the reverse diffusion steps ($\{\epsilon_t\}_{t=1}^T$). The DW$_{\text{Both}}$ setting offers the attacker complete access to all stochastic elements, ϵ_f and $\{\epsilon_t\}_{t=1}^T$. By manipulating the attacker's knowledge in this manner, we isolate the individual effects of the forward and reverse diffusion processes on model robustness.

C.3 Implications of the Attacker's Knowledge of Stochastic Elements

The attacker's capability to craft potent adversarial examples is significantly influenced by their knowledge of the stochastic elements in the diffusion processes. When these elements are unknown to the attacker, they must independently sample noise variables, leading to discrepancies between their approximations and the actual behavior of the victim model. Conversely, if the attacker is privy to the exact noise variables used during inference, they can precisely mimic the model's behavior, markedly boosting the efficacy of their attack.

Attacker Without Knowledge of Stochastic Elements In scenarios where the attacker lacks access to specific noise variables ϵ_f and $\{\epsilon_t\}_{t=1}^T$, the model's output becomes unpredictable from the attacker's viewpoint. The attacker must then optimize the expected value of the loss function over the distribution of these stochastic elements. The optimization problem for devising an adversarial example x^{adv} is formulated as:

$$x^{\text{adv}} = \arg \max_{\|x^{\text{adv}} - x\| \leq \delta} \mathbb{E}_{\epsilon_f, \{\epsilon_t\}} \left[\mathcal{L} \left(f(x^{\text{adv}}; \epsilon_f, \{\epsilon_t\}), y \right) \right], \tag{12}$$

where δ specifies the permissible perturbation magnitude, $\mathcal{L}$ is the loss function, f represents the model's output given the input and stochastic elements, and y is the actual class label.

Attacker With Knowledge of Stochastic Elements Should the attacker possess exact knowledge of the noise variables ϵ_f and $\{\epsilon_t\}_{t=1}^T$ utilized during the model's inference, they can accurately emulate the victim classifier's behavior. The stochastic processes become deterministic from the attacker's perspective, facilitating the formulation of the optimization problem as:

$$x^{\text{adv}} = \arg \max_{\|x^{\text{adv}} - x\| \leq \delta} \mathcal{L} \left(f(x^{\text{adv}}; \epsilon_f, \{\epsilon_t\}), y \right). \tag{13}$$

This precise knowledge allows the attacker to adopt the exact noise that will be used during the target evaluation, allowing effective evaluation.

C.4 Effect of Attacker's Knowledge on Model Robustness

We test the robustness of DDPM under these four settings, table 10 encapsulates the result.

Table 10: Robust accuracies of DDPM under different attack settings.

Attack Setting	**Robust Accuracy (%)**
Conventional White-Box Attack	47.27
DW_{Fwd}	45.41
DW_{Rev}	35.25
DW_{Both}	16.80

Conventional White-Box Attack In this setting, the attacker fully understands the model's architecture and parameters but lacks knowledge of the stochastic elements (ϵ_f and $\{\epsilon_t\}_{t=1}^T$) used during inference. The model's output remains unpredictable due to the stochasticity of both diffusion processes, making it challenging for the attacker to generate effective adversarial examples (reaching robust accuracy of **47.27%**).

DW_{Fwd} Here, the attacker is aware of the Gaussian noise ϵ_f used in the forward diffusion process but not of the noise $\{\epsilon_t\}_{t=1}^T$ in the reverse process. This partial knowledge allows the attacker to accurately simulate the forward process, reducing uncertainty in this phase. However, the reverse process remains unpredictable. The slight decrease in robust accuracy to **45.41%** suggests that while forward process stochasticity contributes to robustness, its effect is somewhat diminished when compromised.

DW_{Rev} In this scenario, the attacker knows the noise variables $\{\epsilon_t\}_{t=1}^T$ used in the reverse diffusion steps but not the forward process noise ϵ_f. This knowledge enables the attacker to align their strategy more closely with the actual behavior of the model during reverse diffusion, resulting in a more noticeable drop in robust accuracy to **35.25%**. The reverse process's stochasticity appears to play a more critical role in model robustness compared to the forward process.

DW_{Both} When the attacker has comprehensive knowledge of both the forward and reverse process noise variables, they can replicate both diffusion processes accurately, eliminating any stochasticity from their perspective. This complete predictability allows for precise adversarial example crafting, leading to a significant reduction in robust accuracy to **16.80%**. This demonstrates that the combined stochastic elements are crucial for maintaining robustness; when fully exposed, the model's defense mechanisms are substantially weakened.

D The Role of Stochasticity in DBP Compared to Certified Defense Methods

In this appendix, we delve deeper into the role of randomness in Diffusion-Based Prediction (DBP) models and contrast it with its role in certified defense methods such as randomized smoothing Cohen et al. (2019). While both approaches incorporate stochasticity, their mechanisms and implications for adversarial robustness differ significantly.

- Conventionally, the classification models discussed in the studies of adversarial robustness can be viewed as mappings from input space X to the label space Y. However, DBP additionally involves a random variable $\epsilon \in E$ that determines the random sampling in the forward and reverse processes (which can be the random seed in implementation). Hence, a DBP model f can be viewed as the mapping $f : (X, E) \to Y$.
- Previous studies on randomized smoothing treat the randomized model f as a mapping $f : X \to P_Y$, where P_Y is the space of label distribution. Typically, the final prediction can be formulated as $F(x) = \arg\max_c [f(x)]_c$, i.e., the class c with the highest probability in the output distribution $\mathbb{f}(x)$. Apparently, F deterministically maps X to Y, consistent with the conventional models.

- Recent studies on DBP also regard the model as $f : X \to P_Y$, without explicitly studying the role of ϵ. *The key difference between DBP and randomized smoothing is that the final prediction for an input x is directly sampled from the distribution $f(x)$ for once, instead of sampling multiple times to approximate $F(x)$ as in randomized smoothing*.
- In this paper, we revisit DBP by treating the randomized model f as the mapping $f : (X, E) \to Y$ and studying the role of $\epsilon \in E$ as an input of f. From this perspective, the conventional adversarial setting assuming full knowledge of the model parameters (but not ϵ) is not a complete white box, which motivates us to study the DW-box setting.
- From our perspective, we can clearly point out the difference between DBP and randomized smoothing in terms of the loss landscape. Given an input x_0, the local loss landscape for a DBP model f is not deterministic as it also depends on ϵ. *Although the expected loss landscape over $\epsilon \in E$ may be smooth, it does not suggest the robustness of DBP, as ϵ is fixed during a single inference run of DBP*. Indeed, our study suggests that given x_0 and a fixed ϵ_0, the local landscape of DBP is likely not smooth. In contrast, the loss landscape of a randomized smoothing model F may be smooth as it is the average landscape over multiple ϵ. To conclude, we argue that the random noise itself may not smooth the loss landscape, but the average over random noises may.

E DBP Models Employing Different Stochastic Elements Cannot Be Attacked All at Once

Previous research has questioned whether stochasticity can improve robustness, arguing that it can produce obfuscated gradients that give a false sense of security Athalye et al. (2018). To investigate this, we implement DW_{Semi}-box, a semi-stochastic setting that restricts the stochastic elements to a limited set of options. Our results show that stochasticity can indeed improve robustness, even when the attacker has full knowledge of all the possible options for stochastic elements. For a detailed analysis of our results, see Appendix E. The potential to further improve robustness by increasing the stochasticity of the model is discussed in Appendix K.

Building on the concept of Deterministic White-box, we further propose $\text{DW}_{\text{semi-128}}$ to explore whether stochasticity can indeed improve robustness. Unlike under Deterministic White-box, where the attacker attacks a DBP model under the exact set of stochastic noise used in the evaluation, $\text{DW}_{\text{semi-128}}$ relaxes the stochastic elements to a limited set of options, the attacker should simultaneously attack over 128 different sets of stochastic noise. It uses the average adversarial direction from these 128 noise settings (EoT-128) to perturb the DBP model. To understand the impact of stochasticity, we analyze the changes of the model loss under DW-box attack and $\text{DW}_{\text{semi-128}}$ attack. We plot these changes by adjusting a factor k to modify an image x with a perturbation σ, evaluating the loss at $x + k\sigma$ where k varies from -16 to 16. We generate perturbations with l_∞ Fast Gradient Sign Method (FGSM) Goodfellow et al. (2015) with magnitude $1/255$. The plot is evaluated using WideResNet-28-10 with DP_{DDPM} over the first 128 images of CIFAR-10 dataset.

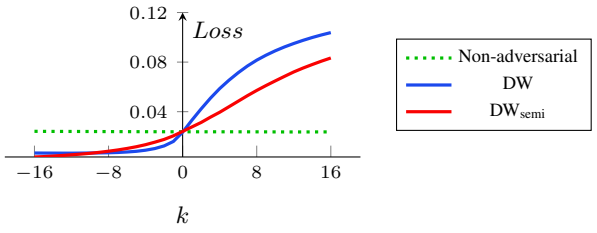

Figure 8: Impact of stochasticity on perturbation efficacy. Perturbations created under DW_{semi}-box setting are less potent compared to DW-box setting. For non-adversarial perturbations, we randomly assign each element a value of either $1/255$ or $-1/255$.

As Figure 8 shows, in the Deterministic White-box setting, the perturbations significantly increase the loss, proving their effectiveness. However, for $\text{DW}_{\text{semi-128}}$, where the attack spans multiple noise setting, the increase in loss is more moderate. This suggests that even when the attackers are fully informed about the stochastic noise choices, stochasticity still improves the robustness of the DBP.

This challenges the notion that there exists a vulnerable direction that is effective for all stochastic noise.

F EXPERIMENTAL SETTING OF VISUALIZATION OF THE ATTACK TRAJECTORY

We visualize the attack by plotting the loss landscape and trace the trajectories of EoT attack under White-box setting and the Deterministic White-box setting in Figure 3. We run a vanilla PGD20-EoT10 attack under White-box setting and a PGD20 attack under Deterministic White-box setting. We then expand a 2D space using the final perturbations from these two attacks, draw the loss landscape, and plot the attack trajectories on it. Note that the two adversarial perturbation directions are not strictly orthogonal. To extend this 2D space, we use the Deterministic White-box attack direction and the orthogonal component of the EoT attack direction. Note that the endpoints of both trajectories lie exactly on the loss landscape, while intermediate points are projected onto it. The plot is evaluated using WideResNet-28-10 with DP_{DDPM} over the first 128 images of CIFAR-10 dataset.

G PSEUDO-CODE OF ADDT

The pseudo-code for adopting ADDT within DDPM and DDIM framework is shown in Algorithm 1.

Algorithm 1 Adversarial Denoising Diffusion Training (ADDT)

Require: $\boldsymbol{x}_0$ is image from training dataset, y is the class label of the image, C is the classifier, $\boldsymbol{P}$ is one-step diffusion reverse process and $\boldsymbol{\theta}$ is it's parameter, L is CrossEntropy Loss.

for $\boldsymbol{x}_0$, y in the training dataset **do**
  $t \sim \text{Uniform}(\{1, ..., T\})$
  $\lambda_t = \texttt{clip}(\gamma_t \lambda_{\text{unit}}, \lambda_{\text{min}}, \lambda_{\text{max}})$, where $\gamma_t = \frac{\sqrt{\overline{\alpha}_t}}{\sqrt{1-\overline{\alpha}_t}}$
  Init $\boldsymbol{\delta}$ to a small random vector.
  for 1 to $\text{ADDT}_{\text{iterations}}$ **do**
    $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$
    $\boldsymbol{\epsilon}' = \texttt{RBGM}(\boldsymbol{\delta}, \boldsymbol{\epsilon})$
    $\boldsymbol{x}_t = \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\lambda_t^2}\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon} + \lambda_t\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}'$
    $\boldsymbol{\delta} = \boldsymbol{\delta} + \nabla_{\boldsymbol{\epsilon}'} L(C(\boldsymbol{P}(\boldsymbol{x}_t, t), y))$
  end for
  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$
  $\boldsymbol{\epsilon}' = \texttt{RBGM}(\boldsymbol{\delta}, \boldsymbol{\epsilon})$
  $\boldsymbol{x}_t = \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\lambda_t^2}\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon} + \lambda_t\sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}'$
  Take a gradient descent step on:
  $\nabla_{\boldsymbol{\theta}} \| \frac{\sqrt{\overline{\alpha}_t}}{\sqrt{1-\overline{\alpha}_t}} (\boldsymbol{x}_0 - \boldsymbol{P}(\boldsymbol{x}_t, t)) \|_2^2$
end for
Diffusion Unet $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ predicts the Gaussian noise added to the image, adopting Equation (4) in the paper, we have $\boldsymbol{P}(\boldsymbol{x}_t, t) = \left(\boldsymbol{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right) / \sqrt{\overline{\alpha}_t}$

H EXPERIMENTS ON IMAGENET-1K

In this section, we present preliminary experimental results on the ImageNet-1k dataset using a small UNet model, consistent with the architecture used in DP_DDPM. The model was trained from scratch for 12 epochs, followed by an additional 8 epochs of ADDT, though the training was not fully converged at this stage. We evaluated the model on the first 1024 images from the ImageNet-1k validation set under a PGD20+EoT10 attack with an l_∞ perturbation bound of 4/255. The results are summarized in Table 11.

As discussed earlier (see Section Q2 of our global response), the initial results of applying ADDT on ImageNet-1k indicated that the improvement in robust accuracy was modest due to the limited number of training epochs (only 3 epochs of ADDT). However, after extending the training to 8 ADDT epochs, we observed a notable improvement in robust accuracy, reaching 48.02%, as shown in Table 11. The experiment was conducted using a ResNet-101 backbone, and the model was tested under the same PGD20+EoT10 attack with an l_∞ bound of 4/255.

Method	Clean Accuracy	Robust Accuracy
Adversarial Training [I]	69.52%	41.02%
DP_DDPM	80.31%	46.15%
DP_DDPM (ADDT)	80.20%	48.02%

Table 11: Comparison of clean and robust accuracy for different methods on the ImageNet-1k validation subset.

Due to computational constraints, we have not yet been able to evaluate the model's performance under the more intensive PGD200+EoT20 attack, which requires approximately 7 days of computation on 8 NVIDIA 4090 GPUs for each model. We are continuing to train the model and plan to provide updated results under stronger attack settings in future revisions.

We have also included a comparison with adversarial training (AT) results for the same classifier architecture and perturbation constraints, as reported in [I]. As shown in Table 11, our DP_DDPM model achieves a significantly higher robust accuracy under PGD20+EoT10 compared to AT. While we acknowledge that evaluating our model under PGD200+EoT20 may result in a decrease in robust accuracy—as observed in our earlier experiments (see Section Q1 and Update 1)—we expect that the robust accuracy of our method will remain at least comparable to that of AT, especially given that our model has not yet fully converged.

Furthermore, as demonstrated in Table 11, our DBP-based approach maintains a substantially higher clean accuracy compared to AT, which is a crucial advantage for real-world applications where maintaining high clean accuracy is often as important as ensuring robustness. We also note that recent works have achieved state-of-the-art results with adversarial training [II, III, IV]; however, these methods often rely on advanced network architectures, larger model sizes, or additional training data, which differ from the setup used in our experiments.

I Adopting VPSDE(DDPM++) and EDM Models in DBP

In the previous discussion of the robustness of DBP models, as detailed in Section 4.1, our focus was primarily on the DDPM and DDIM models. We now extend our analysis to include VPSDE (DDPM++) and EDM Karras et al. (2022) models. VPSDE (DDPM++) is the diffusion model used in DiffPure.

From a unified perspective, diffusion processes can be modeled by stochastic differential equations (SDE) Song et al. (2021). The forward SDE, as described in Equation (14), converts a complex initial data distribution into a simpler, predetermined prior distribution by progressively infusing noise. This can also be done in a single step, as shown in Equation (15), mirroring the strategy of DDPM described in Equation (2). Reverse SDE, as explained in Equation (16), reverses this process, restoring the noise distribution to the original data distribution, thus completing the diffusion cycle.

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}, \tag{14}$$

$$p_{0t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(0)) = \mathcal{N}\left(\boldsymbol{x}(t); e^{-\frac{1}{4}t^2(\bar{\beta}_{\max}-\bar{\beta}_{\min})-\frac{1}{2}t\bar{\beta}_{\min}}\boldsymbol{x}(0), \boldsymbol{I} - \boldsymbol{I}e^{-\frac{1}{2}t^2(\bar{\beta}_{\max}-\bar{\beta}_{\min})-t\bar{\beta}_{\min}}\right), \quad t \in [0,1] \tag{15}$$

$$\mathrm{d}\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g^2(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}. \tag{16}$$

The reverse process of SDEs also derives equivalent ODEs Equation (17) for fast sampling and exact likelihood computation, and this Score ODEs corresponds to DDIM.

$$\mathrm{d}\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - \frac{1}{2}g(t)^2\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t. \tag{17}$$

By modulating the stochasticity, we can craft a spectrum of semi-stochastic models that bridge pure SDEs and deterministic ODEs, offering a range of stochastic behaviors.

EDM provides a unified framework to synthesize the design principles of different diffusion models (DDPM,DDIM,iDDPM Nichol & Dhariwal (2021),VPSDE,VESDE Song et al. (2021)). Within

this framework, EDM incorporates efficient sampling methods, such as the Heun sampler, and introduces optimized scheduling functions $\sigma(t)$ and $s(t)$. This allows EDM to achieve state-of-the-art performance in generative tasks.

EDM forward process could be presented as:

$$\boldsymbol{x}_t = \boldsymbol{x}_0 + \sigma(t^*) * \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{18}$$

where we choose $\sigma(t^*) = 0.5$ for clean and robust accuracy tradeoff. And for reverse process, EDM incorporates a parameter S_{churn} to modulate the stochastic noise infused during the reverse process. For our experiments, we choose 50 reverse steps (50 NFE, NFE is Function of Neural Function Evaluations), configured the parameters with $S_{min} = 0.01$, $S_{max} = 0.46$, $S_{noise} = 1.007$, and designate $S_{churn} = 0$ to represent EDM-ODE, $S_{churn} = 6$ to represent EDM-SDE.

As shown in Table 12, our ADDT could also increase the robustness of DP_{EDM}.

Table 12: Clean and robust accuracy on DP_{EDM} for CIFAR-10. ADDT improves robustness in both $\text{DP}_{\text{EDM-SDE}}$ and $\text{DP}_{\text{EDM-ODE}}$.

Type	Vanilla		Ours	
	$\text{DP}_{\text{EDM-SDE}}$	$\text{DP}_{\text{EDM-ODE}}$	$\text{DP}_{\text{EDM-SDE}}$	$\text{DP}_{\text{EDM-ODE}}$
Clean	**86.43**	**87.99**	86.33	**87.99**
l_∞	62.50	60.45	**66.41**	**64.16**
l_2	76.86	75.49	**79.16**	**77.15**

J ADDT Results on DP_{DDIM}

As shown in Table 13, the performance of DP_{DDIM} is less sensitive to the number of function evaluations (NFE). Additionally, ADDT consistently improved the robustness of DP_{DDIM}.

Table 13: Clean and robust accuracy (%) on DP_{DDIM}. ADDT improve robustness across different NFE (*: default DDIM generation setting, -: classifier only).

Dataset	NFE	Vanilla			Ours		
		Clean	l_∞	l_2	*Clean*	l_∞	l_2
CIFAR-10	-	95.12	0.00	1.46	95.12	0.00	1.46
	5	89.65	42.19	68.65	88.57	**47.27**	**70.61**
	10*	88.96	43.16	67.58	88.18	**47.85**	**70.61**
	20	87.89	41.70	69.24	88.67	**48.63**	**69.73**
	50	88.96	42.48	68.85	88.57	**46.68**	**69.24**
	100	88.38	42.19	70.02	88.77	**46.48**	**71.19**
CIFAR-100	-	76.66	0.00	2.44	76.66	0.00	2.44
	5	62.11	15.43	35.74	62.79	**17.58**	**38.87**
	10*	62.21	15.33	36.52	64.45	**20.02**	**39.26**
	20	63.67	15.62	37.89	65.23	**18.65**	**40.62**
	50	62.40	16.31	37.79	63.87	**19.14**	**39.94**
	100	63.28	15.23	36.62	66.02	**18.85**	**39.84**

K Strengthening DBP via Augmented Stochasticity

Song *et al.* present a Predictor-Corrector sampler for SDEs reverse process for VPSDE (DDPM++) (as detailed in Appendix G of Song et al. (2021)). However, standard implementations of VPSDE (DDPM++) typically use only the Predictor. Given our hypothesis that stochasticity contributes to robustness, we expect that integrating the Corrector sampler into VPSDE (DDPM++) would further enhance the robustness of DBP models. Our empirical results, as shown in Table 14, confirm that the inclusion of a Corrector to VPSDE (DDPM++) indeed improve the model's defenses ability against adversarial attacks with l_∞ norm constraints. This finding supports our claim that the increased stochasticity can further strengthen DBP robustness. Adding Corrector is also consistent with ADDT. Note that the robustness against l_2 norm attacks does not show a significant improvement with the

integration of the Extra Corrector. A plausible explanation for this could be that the robustness under l_2 attacks is already quite strong, and the compromised performance on clean data counteracts the increase in robustness.

Table 14: Clean and robust accuracy on $\text{DP}_{\text{DDPM++}}$ for CIFAR-10. Both extra Corrector and ADDT fine-tuning improved robustness.

Type	Vanilla	Extra Corrector	ADDT	ADDT+Extra Corrector
Clean	89.26	85.25	**89.94**	85.55
l_∞	55.96	59.77	62.11	**65.23**
l_2	75.78	74.22	**76.66**	**76.66**

L EVALUATING RBGM-MAPPED PERTURBATIONS

In Section 6.5, we briefly explore the generation capabilities of diffusion models trained with RBGM-mapped and l_∞ perturbations. Here, we provide details of the experiment and delve deeper into their robustness. To train with l_∞ perturbations, we adjust ADDT, replacing RBGM-mapped perturbations with l_∞ perturbations. Here, instead of converting accumulated gradients to Gaussian-like perturbations, we use a 5-step projected gradient descent (PGD-5) approach. We also set $\lambda_{unit} = 1, \lambda_{min} = 0, \lambda_{max} = 10$. We refer to this modified training protocol as ADDT_{l_∞}.

Table 15: Clean and robust accuracy on DBP models trained with different perturbations for CIFAR-10. While ADDT simultaneously improves clean accuracy and robustness against both l_2 and l_∞ attacks. ADDT_{l_∞} primarily improves performance against l_∞ attacks.

Method	Dataset	Vanilla			ADDT			ADDT_{l_∞}		
		Clean	l_∞	l_2	*Clean*	l_∞	l_2	*Clean*	l_∞	l_2
$\text{DP}_{\text{DDPM-1000}}$	CIFAR-10	**85.94**	47.27	69.34	85.64	51.46	**70.12**	84.47	**52.64**	68.55
	CIFAR-100	57.52	20.41	37.89	**59.18**	**23.73**	**41.70**	57.81	23.24	40.04
$\text{DP}_{\text{DDIM-100}}$	CIFAR-10	88.38	42.19	70.02	**88.77**	46.48	**71.19**	88.48	**50.49**	70.31
	CIFAR-100	63.28	15.23	36.62	**66.02**	18.85	**39.84**	64.84	**20.31**	39.36

We evaluate the clean and robust accuracy of ADDT and ADDT_{l_∞} fine-tuned models. These models exhibit different behaviors. As shown in Table 15, while Gaussian-mapped perturbations can simultaneously improve clean accuracy and robustness against both l_2 and l_∞ attacks, training with l_∞ perturbations primarily improves performance against l_∞ attacks.

M EVALUATING UNDER FIXED AUTOATTACK

AutoAttack Croce & Hein (2020), an ensemble of White-box and Black-box attacks, is a popular benchmark for evaluating model robustness. It is used in RobustBench Croce et al. (2020) to evaluate over 120 models. However, Nie et al. Nie et al. (2022) found that the *Rand* version of AutoAttack, designed to evaluate stochastic defenses, sometimes yields higher accuracy than the *Standard* version, intended for deterministic methods. Our comparison of AutoAttack and PGD20-EoT10 in Table 16 also shows that the *Rand* version of AutoAttack gives higher accuracy than the PGD20-EoT10 attack.

We attribute this to the sample selection of AutoAttack. As an ensemble of attack methods, AutoAttack selects the final adversarial sample from either the original input or the attack results. However, the original implementation neglects stochasticity and considers a adversarial sample to be sufficiently adversarial if it gives a false result in one evaluation. To fix this, we propose a 20-iteration evaluation and selects the adversarial example with the lowest accuracy. The flawed code is in the official GitHub main branch, git version $a39220048b3c9f2cca9a4d3a54604793c68eca7e$, and specifically in lines #125, #129, #133-136, #157, #221-225, #227-228, #231 of the file autoattack/autoattack.py. We will open source our updated code and encourage future stochastic defense methods to be evaluated against the fixed code. The code now can be found at: https://anonymous.4open.science/r/auto-attack-595C/README.md.

After the fix, AutoAttack's accuracy dropped by up to 10 points, producing similar results to our PGD20-EoT10 test results. However, using AutoAttack on DP_{DDPM} with $S = 1000$ took nearly 25 hours, five times longer than PGD20-EoT10, so we will use PGD20-EoT10 for the following test.

Table 16: AutoAttack (*Rand* version) and PGD20-EoT10 performance on DBP methods for CIFAR-10 (the lower the better). The original AutoAttack produces high accuracy, after fixing, it achieves similar results to PGD20+EoT10 attack.

Method	l_∞			l_2		
	AutoAttack	$\text{AutoAttack}_{\text{Ours}}$	PGD20-EoT10	AutoAttack	$\text{AutoAttack}_{\text{Ours}}$	PGD20-EoT10
DiffPure	62.11	56.25	**55.96**	81.84	76.37	**75.78**
$\text{DP}_{\text{DDPM-1000}}$	57.81	**46.88**	48.63	71.68	**71.09**	72.27
$\text{DP}_{\text{DDIM-100}}$	50.20	**40.62**	44.73	77.15	**70.70**	71.68

Table 17: Clean and robust accuracy on different DBP methods for CIFAR-10, evaluated with $\text{AutoAttack}_{\text{Ours}}$ (*Rand* version). All methods show consistent improvement when fine-tuned with ADDT.

Method	Vanilla			Ours		
	Clean	l_∞	l_2	*Clean*	l_∞	l_2
DiffPure	89.26	56.25	76.37	**89.94**	**58.20**	**77.34**
DP_{DDPM}	**85.94**	46.88	71.09	85.64	**48.63**	**72.27**
DP_{DDIM}	88.38	40.62	70.70	**88.77**	**44.73**	**71.68**

N Robustness under Deterministic White-box Setting

We evaluate robustness of ADDT fine-tuned models under Deterministic White-box Setting. We assess robustness across varying NFE and compare the performance of vanilla models and ADDT fine-tuned models, with results showcased in Figure 9. ADDT consistently improves model performance at different NFE, particularly noticeable at lower NFE. This affirms that ADDT equips the diffusion model with the ability to directly counter adversarial perturbations.

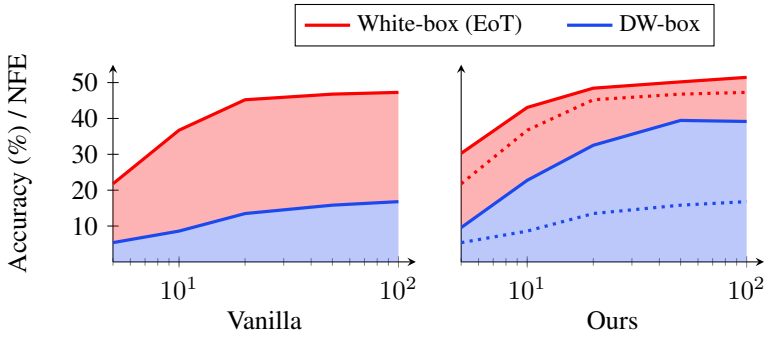

Figure 9: Revisiting Deterministic White-box Robustness. ADDT improves robustness under both White-box and Deterministic White-box setting, implying that ADDT strengthens the models' ability to handle adversarial inputs. The dashed line on the right is the performance of the vanilla model.

O Ablation Study of λ_{unit}

In Section 6.2 we choose λ_{unit}=0.03 because most of the adversarial perturbations are in this range. We also provide an ablation study here, which shows that the performance of ADDT is insensitive to λ_{unit} and gets a consistent improvement.

Table 18: Ablation study of λ_{unit}, ADDT is insensitive to it and gets a consistent improvement

Attack type \NFE	λ_{unit}	50	100	200	500	1000
l_∞	$Clean$	21.78	36.72	45.21	46.78	47.27
	0.02	24.02	40.92	48.14	48.83	48.93
	0.03	30.27	43.07	48.44	50.20	51.46
	0.04	31.25	44.92	50.68	51.07	50.88
l_2	$Clean$	36.13	55.47	65.23	68.85	69.34
	0.02	41.99	61.72	67.48	69.82	70.31
	0.03	41.99	62.97	69.82	69.04	70.12
	0.04	49.02	64.45	69.24	69.53	69.92

P Computational Cost of ADDT

Optimizing DDPM and DDIM models through ADDT achieves near-optimal performance in 50 epochs. This process takes only 12 hours on a four-GPU cluster of NVIDIA GeForce RTX 2080 Ti, matching the speed of traditional adversarial training and significantly faster than the latest adversarial training techniques that use diffusion models to augment the dataset Wang et al. (2023).

A special feature of ADDT is its one-and-done training approach. After initial training, ADDT can protect various classifiers without the need for further fine-tuning, as shown in Table 4. This is different from adversarial classifier training, which requires individual training for each classifier.

Q Credibility of Our Paper

The code was developed independently by two individuals and mutually verified, with consistent results achieved through independent training and testing. We will also make the code open-source and remain committed to advancing the field.

R Broader Impact and Limitations

Our work holds significant potential for positive societal impacts across various sectors, including autonomous driving, facial recognition payment systems, and medical assistance. We are dedicated to enhancing the safety and trustworthiness of global AI applications. However, there are potential negative societal impacts, particularly concerning privacy protection, due to adversarial perturbations. Nonetheless, we believe that the positive impacts generally outweigh the potential negatives. Regarding the limitations, our approach could benefit from integrating insights from traditional adversarial training methods Zhang et al. (2019); Shafahi et al. (2019b); Wang et al. (2023), such as through more extensive data augmentation and a refined ADDT loss design. Nevertheless, these limitations are minor and do not significantly detract from the overall contributions of this paper.We believe that these new findings and perspectives could have a sustained impact on future research on DBP, which is a promising approach to adversarial defense and could be more valuable for real-world applications compared to AT, although existing studies on DBP are at an early stage.