Sample-Specific Noise Injection For Diffusion-Based Adversarial Purification

Yuhao Sun [*1] Jiacheng Zhang [*1] Zesheng Ye [*1] Chaowei Xiao [2] Feng Liu [1]

Abstract

Diffusion-based purification (DBP) methods aim to remove adversarial noise from the input sample by first injecting Gaussian noise through a forward diffusion process, and then recovering the clean example through a reverse generative process. In the above process, how much Gaussian noise is injected to the input sample is key to the success of DBP methods, which is controlled by a constant noise level t^* for all samples in existing methods. In this paper, we discover that an optimal t^* for each sample indeed could be different. Intuitively, the cleaner a sample is, the less the noise it should be injected, and vice versa. Motivated by this finding, we propose a new framework, called ***S**ample-specific **S**core-aware **N**oise **I**njection* (SSNI). Specifically, SSNI uses a pre-trained score network to estimate how much a data point deviates from the clean data distribution (i.e., score norms). Then, based on the magnitude of score norms, SSNI applies a reweighting function to adaptively adjust t^* for each sample, achieving sample-specific noise injections. Empirically, incorporating our framework with existing DBP methods results in a notable improvement in both accuracy and robustness on CIFAR-10 and ImageNet-1K, highlighting the necessity to allocate *distinct noise levels to different samples* in DBP methods. Our code is available at: `https://github.com/tmlr-group/SSNI`.

1. Introduction

Deep neural networks (DNNs) are vulnerable to adversarial examples, which is a longstanding problem in deep learning (Szegedy et al., 2014; Goodfellow et al., 2015). Adversarial examples aim to mislead DNNs into making erroneous predictions by adding imperceptible adversarial noise to clean examples, which pose a significant security threat in critical applications (Dong et al., 2019; Cao et al., 2021; Jing et al., 2021; Han et al., 2025). To defend against adversarial examples, *adversarial purification* (AP) stands out as a representative defensive mechanism, by leveraging pre-trained generative models to purify adversarial examples back towards their natural counterparts before feeding into a pre-trained classifier (Yoon et al., 2021; Nie et al., 2022). Notably, AP methods benefit from their modularity, as the purifier operates independently of the downstream classifier, which facilitates seamless integration into existing systems and positions AP as a practical approach to improve the adversarial robustness of DNN-based classifiers.

Recently, *diffusion-based purification* (DBP) methods have gained much attention as a promising framework in AP, which leverage the denoising nature of diffusion models to mitigate adversarial noise (Nie et al., 2022; Xiao et al., 2023; Lee & Kim, 2023). Generally, diffusion models train a forward process that maps from data distributions to simple distributions, e.g., Gaussian, and reverse this mapping via a reverse generative process (Ho et al., 2020; Song et al., 2021b). When applied in DBP methods, the forward process gradually injects Gaussian noise into the input sample, while the reverse process gradually purify noisy sample to recover the clean sample. The quality of the purified sample heavily depends on the amount of Gaussian noise added to the input during the forward process, which can be controlled by a noise level parameter t^*. Existing DBP methods (Nie et al., 2022; Xiao et al., 2023; Lee & Kim, 2023) manually select a constant t^* for all samples.

However, we find that using a sample-shared t^* may *overlook* the fact that an optimal t^* indeed could be different at sample-level, as demonstrated in Figure 1. For example, in Figure 1a, $t^* = 100$ is too small, resulting in the adversarial noise not being sufficiently removed by the diffusion models. This is because diffusion models are good at denoising samples that have been sufficiently corrupted by Gaussian noise through the forward process (Ho et al., 2020; Song et al., 2021b). With a small t^*, the sample remains insufficiently corrupted, which limits the denoising capability in the reverse process and thereby compromising the robustness against adversarial examples. On the other hand, in Figure 1b and 1c, $t^* = 100$ is too large, resulting

[*]Equal contribution [1]School of Computing and Information Systems, The University of Melbourne [2]University of Wisconsin, Madison. Correspondence to: Feng Liu <fengliu.ml@gmail.com>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

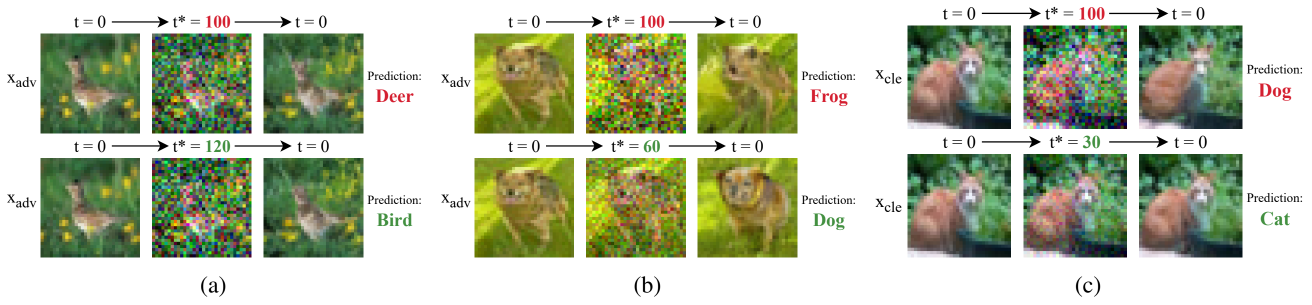


Figure 1. For each sub-figure: the 1st column contains the input (i.e., could either be AEs or CEs), the 2nd column contains noise-injected examples with different t^*s, and the 3rd column contains purified examples. We use *DiffPure* (Nie et al., 2022) with a sample-shared $t^* = 100$ selected by Nie et al. (2022) to conduct this experiment on CIFAR-10 (Krizhevsky et al., 2009). The globally shared $t^* = 100$ offers a baseline, but results in suboptimal prediction performance compared to what could be achieved by tuning the noise level for individual samples. Notably, while the recovered images obtained by different noise levels may be visually indistinguishable, they carry different semantics. For instance, the image is classified as "frog" (incorrect) with $t^* = 100$ but as "dog" (correct) with $t^* = 60$ (Figure 1b). These *highlight the need for a sample-wise noise level adjustment*.

in excessive disruption of the sample's semantic information during the forward process, which makes it difficult to recover the original semantics in the reverse process. In this case, both robustness and clean accuracy are compromised, as the purified samples struggle to preserve the semantic consistency of clean samples. These observations motivate us to make the *first* attempt to adjust the noise level on a sample-specific basis.

In this paper, we propose ***S****ample-specific* ***S****core-aware* ***N****oise* ***I****njection* (SSNI), a new framework that leverages the distance of a sample from the clean data distribution to adaptively adjust t^* on a sample-specific basis. SSNI aims to inject less noise to cleaner samples, and vice versa.

To implement SSNI, inspired by the fact that *scores* (i.e., $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$) reflect the directional momentum of samples toward the high-density areas of clean data distribution (Song & Ermon, 2019), we use *score norms* (i.e., $\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\|$) as a natural metric to measure the deviation of a data point from the clean data distribution. In Section 3, we establish the relationship between the score norm and the noise level required for different samples. Specifically, samples with different score norms tend to have accumulated different noise levels. Furthermore, we empirically show that the cleaner samples – those closer to the clean data distribution – exhibit lower score norm, justifying the rationale of using score norms for reweighting t^*. Concretely, we use a pre-trained score network to estimate the score norm for each sample. Based on this, we propose two reweighting functions that adaptively adjust t^* according to its score norm, achieving sample-specific noise injections (see Section 4.3). Notably, this reweighting process is *lightweight*, ensuring that SSNI is computationally feasible and can be applied in practice with minimal overhead (see Section 5.6).

Through extensive evaluations on benchmark image datasets such as CIFAR-10 (Krizhevsky et al., 2009) and ImageNet-1K (Deng et al., 2009), we demonstrate the effectiveness of SSNI in Section 5. Specifically, combined with different DBP methods (Nie et al., 2022; Xiao et al., 2023; Lee & Kim, 2023), SSNI can boost clean accuracy and robust accuracy *simultaneously* by a notable margin against the well-designed adaptive white-box attack (see Section 5.2).

The success of SSNI takes root in the following aspects: (1) an optimal noise level t^* for each sample indeed could be different, making SSNI an effective approach to unleash the intrinsic strength of DBP methods; (2) existing DBP methods often inject excessive noise into clean samples, resulting in a degradation in clean accuracy. By contrast, SSNI injects less noise to clean samples, and thereby notably improving the clean accuracy. Meanwhile, SSNI can effectively handle adversarial samples by injecting sufficient noise on each sample; (3) SSNI is designed as general framework instead of a specific method, allowing it to be seamlessly integrated with a variety of existing DBP methods.

2. Preliminary and Related Work

In this section, we first review diffusion models and scores in detail. We then review the related work of DBP methods. Detailed related work can be found in Appendix A.

Diffusion models are generative models designed to approximate the underlying clean data distribution $p(\mathbf{x}_0)$, by learning a parametric distribution $p_\phi(\mathbf{x}_0)$ with a *forward* and a *reverse* process. In *Denoising Diffusion Probabilistic Models* (DDPM) (Ho et al., 2020), the *forward* process of a diffusion model defined on $\mathcal{X} \subseteq \mathbb{R}^d$ can be expressed by:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right), \tag{1}$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$ and $\{\beta_t\}_{t\in[0,T]}$ are predefined noise scales with $\beta_t \in (0, 1)$ for all t. As t increases, $\mathbf{x}_t$ converges toward isotropic Gaussian noise. In the *reverse* process, DDPM seeks to recover the clean data from noise by simulating a Markov chain in the reverse direction over T steps. The reverse transition at each intermediate step is modeled by

$$p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\phi(\mathbf{x}_t, t), \sigma_t^2\mathbf{I}), \tag{2}$$

where $\mu_\phi(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\phi(\mathbf{x}_t, t)\right)$ is the predicted mean at t, and σ_t is a fixed variance (Ho et al., 2020). More specifically, the model learns to predict the noise $\boldsymbol{\epsilon}_{\phi^*}(\mathbf{x}_t, t)$ added to the data. The training objective minimizes the distance between the true and predicted noise, accounting for different noise levels controlled by t:

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

The generative process starts from pure noise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and progressively removes noise through T steps, with a random noise term $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ incorporated at each step:

$$\hat{\mathbf{x}}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\hat{\mathbf{x}}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_{\phi^*}(\hat{\mathbf{x}}_t, t)\right) + \sqrt{\beta_t}\boldsymbol{\epsilon}.$$

Score and score norm. In diffusion models, *score* refers to the gradient of the log-probability density $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, which maps the steepest ascent direction of the log-density in the probability space (Song & Ermon, 2019). The associated *score norm* $\|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|$ measures the gradient's magnitude, reflecting how much a data point deviates from the clean data distribution: points located in low-probability regions typically exhibit larger score norms, while those situated closer to high-density regions of clean data manifest smaller score norms. This property is valuable for adversarial example detection, as demonstrated by Yoon et al. (2021) who establish that score norms can effectively distinguish between adversarial and clean examples. However, direct computation of the score is often intractable for complex data distributions (e.g., images). In practice, it is estimated using neural networks $s_\theta : \mathcal{X} \times \mathbb{R}_+ \to \mathcal{X}$ that take as input both a data point $\mathbf{x}$ and a specified timestep $t^{\mathrm{S}} \in \mathbb{R}_+$ to approximate $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ by minimizing the Fisher divergence between the true and estimated scores. Also, $s_\theta(\mathbf{x}, t^{\mathrm{S}})$ enables efficient computation of $\|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|$.

Diffusion-based purification. *Adversarial purification* (AP) leverages generative models as an add-on module to purify adversarial examples before classification. Within this context, *diffusion-based purification* (DBP) methods have emerged as a promising framework, exploiting the inherent denoising nature of diffusion models to filter out adversarial noise (Nie et al., 2022; Wang et al., 2022; Xiao et al., 2023; Lee & Kim, 2023). Specifically, DBP works with *pre-trained* diffusion model by first corrupting an adversarial input through the forward process and then iteratively reconstructing it via the reverse process. This projects the input back onto the clean data manifold while stripping away adversarial artifacts. Wang et al. (2022) introduce input guidance during the reverse diffusion process to ensure the purified outputs stay close to the inputs. Lee & Kim (2023) propose a fine-tuned gradual noise scheduling for multi-step purifications. Bai et al. (2024) improve the reverse diffusion process by incorporating contrastive objectives.

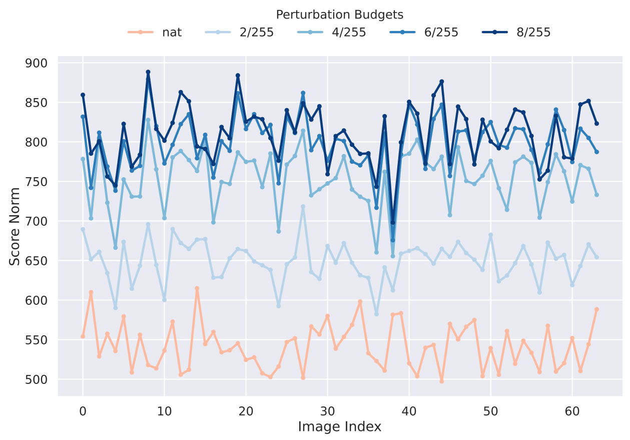


Figure 2. Relationship between score norms and perturbation budgets. We use one batch of clean data from *CIFAR-10* and employ PGD+EOT $\ell_\infty(\epsilon = 8/255)$ as the attack.

3. Motivation

We now elaborate on the motivation of our method by connecting the impact of different perturbation budgets ϵ to the required noise level t^* of each sample through score norm.

Sample-shared noise-level t^* fails to address diverse adversarial perturbations. We empirically observe that an optimal noise level t^* for each sample indeed could be different: While a constant noise level $t^* = 100$, as suggested by Nie et al. (2022), yields strong performances for certain samples, it leads to *suboptimal* results for others (from Figure 1). Since DBP relies on adequate t^* for forward noise injection to remove adversarial perturbation, a shared t^* cannot adapt to the distinct adversarial perturbations of individual examples, leading to a suboptimal accuracy-robustness trade-off. Specifically, Figure 1a shows that $t^* = 100$ is insufficient to remove the adversarial noise, leaving residual perturbations that compromise robustness. In contrast, $t^* = 100$ overly suppresses the other sample's semantic information during the forward process, making it difficult to recover the original semantics via the reverse process (Figure 1b-1c). These findings highlight the need for *sample-specific noise injection levels tailored to individual adversarial perturbations*.

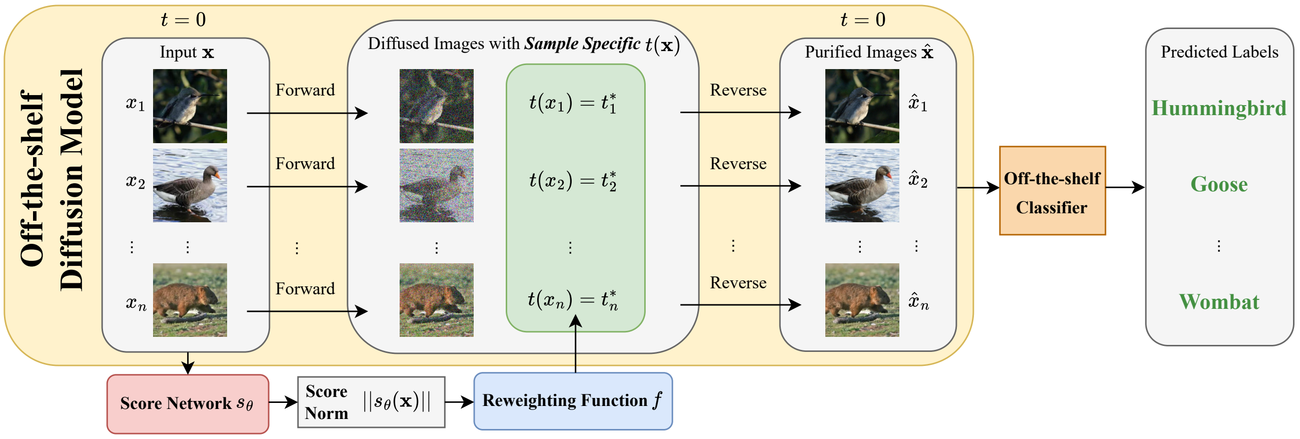


Figure 3. An overview of the proposed SSNI framework. SSNI introduces a novel sample-specific mechanism to adaptively adjust the noise injection level for each sample, enhancing purification effectiveness in DBP methods. The process begins by forwarding each sample image x_i through the forward diffusion process using an off-the-shelf diffusion model. To determine how much noise to inject into each sample, SSNI employs a pre-trained score network s_θ to compute the score norm $\|s_\theta(x_i)\|$, which reflects the distance of the sample from the clean data distribution. Based on this score norm, a reweighting function f adaptively determines the optimal noise level t_i^* for each sample. Finally, each sample is purified through a reverse diffusion process before being classified. Notably, SSNI is designed as a general framework rather than a specific method, which can be seamlessly integrated with a wide range of existing DBP methods.

Score norms vary across perturbation budgets. Adversarial and clean examples are from distinct distributions (Gao et al., 2021). Motivated by this, we further investigate how different perturbation budgets ϵ affect score norms under adversarial attacks (Figure 2). Specifically, we compute the score norm of different samples undergoing PGD+EOT $\ell_\infty(\epsilon = 8/255)$ with perturbation budgets varying between 0 and $8/255$ on CIFAR-10. We observe a consistent pattern: score norms scale directly with perturbation strength: larger ϵ values lead to higher norms, whereas smaller perturbations (i.e., cleaner samples) lead to lower norms The findings extend the role of score norms from differentiating adversarial/clean samples (Yoon et al., 2021) to *differentiate adversarial examples based on their perturbation strength*.

Different score norms imply sample-specific t^*. Higher score norms signal greater deviation from the clean data, often caused by larger ϵ. Intuitively, *samples with elevated score norms demand higher* t^* (i.e., more aggressive noise injection) to remove adversarial patterns. This dependency creates a link: score norms can act as *proxies* for estimating the optimal sample-specific t^*. Doing so successfully leads to sufficient purification for adversarial examples while preserving fidelity for cleaner inputs.

4. Sample-specific Score-aware Noise Injection

Motivated by Section 3, we propose *Sample-specific Score-aware Noise **I**njection* (SSNI), a generalized DBP framework that adaptively adjusts the noise level for each sample based on how much it deviates from the clean data distribution, measured by the score norm. We begin by introducing the SSNI framework, followed by a connection with existing DBP methods and the empirical realization of SSNI.

4.1. Framework of SSNI

Overview. SSNI builds upon existing DBP methods by *reweighting* the optimal noise level t^* from a global, sample-shared constant to a sample-specific quantity. At its core, SSNI leverages score norms to modulate the noise injected into each sample during diffusion, ensuring a more targeted denoising process tailored to each sample. We visually illustrate SSNI in Figure 3, and describe the algorithmic workflow in Algorithm 1.

DBP with sample-shared noise level t^*. Existing DBP methods use an off-the-shelf diffusion model for data purification, and a classifier responsible for label prediction. Let $\mathcal{Y}$ be the label space for the classification task. Denote the forward diffusion process by $D : \mathcal{X} \to \mathcal{X}$, the reverse process by $R : \mathcal{X} \to \mathcal{X}$, and the classifier by $C : \mathcal{X} \to \mathcal{Y}$. For classification, each adversarial sample goes through

$$h(\mathbf{x}) = C \circ R \circ D(\mathbf{x}), \quad \text{with } \mathbf{x} = \mathbf{x}_0. \tag{3}$$

In this context, $\mathbf{x}_T = D(\mathbf{x}_0)$ refers to the noisy image obtained after T steps of diffusion, and $\hat{\mathbf{x}}_0 = R(\mathbf{x}_T)$ represents the corresponding recovered images through the reverse process. Specifically, these methods predetermine a *constant* noise level t^* for all samples, following a shared noise schedule $\{\beta_t\}_{t\in[0,T]}$. The outcome of the forward process defined in Eq. (1) can be expressed as:

$$\mathbf{x}_T = \sqrt{\prod_{i=1}^{t^*}(1-\beta_i)}\mathbf{x} + \sqrt{1 - \prod_{i=1}^{t^*}(1-\beta_i)}\boldsymbol{\epsilon},$$

for $\mathbf{x} = \mathbf{x}_0$, $\forall \mathbf{x} \in \mathcal{X}$, where $\mathbf{x}_0$ represents the original data, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ denotes the Gaussian noise.

From sample-shared to sample-specific noise level. SSNI takes a step further by transforming the sample-shared noise level $T_{\mathrm{SH}}(\mathbf{x}) = t^*$ into a *sample-specific* noise level $T_{\mathrm{SI}}(\mathbf{x}) = t(\mathbf{x})$, computed through

$$t(\mathbf{x}) = f(\left\|s_\theta(\mathbf{x}, t^{\mathrm{S}})\right\|, \ t^*), \tag{4}$$

using a *pre-trained* score network θ, where $s_\theta(\mathbf{x}, t^{\mathrm{S}})$ is the score evaluated at an *arbitrary* reference noise level t^{S}, and $f(\cdot, \cdot)$ is a reweighting function (detailed in Section 4.3) that adjusts t^* based on the score norm. The forward process of SSNI then takes the form:

$$\mathbf{x}_{t(\mathbf{x})} = \sqrt{\prod\nolimits_{i=1}^{t(\mathbf{x})}(1-\beta_i)}\mathbf{x} + \sqrt{1 - \prod\nolimits_{i=1}^{t(\mathbf{x})}(1-\beta_i)}\boldsymbol{\epsilon}, \tag{5}$$

for $\mathbf{x} = \mathbf{x}_0$, $\forall \mathbf{x} \in \mathcal{X}$. In this way, SSNI establishes a link between the sample's deviation from the clean data and the intensity of noise injected during diffusion.

4.2. Unifying Sample-shared and Sample-specific DBP

We define a generalized purification operator encompassing both sample-shared and sample-specific noise based DBP methods as $\Phi(\mathbf{x}) = R(\mathbf{x}_{T(\mathbf{x})})$, where R denotes the reverse process, $\mathbf{x}_{T(\mathbf{x})}$ is the noisy version of $\mathbf{x}$ after $T(\mathbf{x})$ steps of diffusion, and $T : \mathcal{X} \to \mathcal{T}$ is a function that determines the noise level for each input, with $\mathcal{T} = [0, T_{\max}]$ being the range of possible noise levels. Based on $\Phi(\cdot)$, we derive the following understandings (justifications are in Appendix B).

Sample-shared DBP is a special case of SSNI. The noise level of a sample-shared DBP, denoted by $T_{\mathrm{SH}}(\mathbf{x}) \triangleq t^*$, is a constant for $\forall \mathbf{x} \in \mathcal{X}$, while for SSNI, we have: $T_{\mathrm{SI}}(\mathbf{x}) \triangleq t(\mathbf{x}) = f(\left\|s_\theta(\mathbf{x}, t^{\mathrm{S}})\right\|, t^*)$. Clearly, any $T_{\mathrm{SH}}(\mathbf{x})$ can be expressed by $T_{\mathrm{SI}}(\mathbf{x})$, implying that any sample-shared noise level t^* is equivalently represented by SSNI with a constant reweighting function.

SSNI has higher purification flexibility. To compare the purification capabilities of different DBP strategies, we introduce the purification range Ω. For an input $\mathbf{x} \in \mathcal{X}$, we define $\Omega(\mathbf{x}) = \left\{\Phi(\mathbf{x}) \mid \Phi(\mathbf{x}) = R(\mathbf{x}_{\tau(\mathbf{x})}), \tau : \mathcal{X} \to \mathcal{T}\right\}$, which characterizes all possible purified outputs that a DBP strategy can generate for $\mathbf{x}$ when using different noise levels $\tau(\mathbf{x})$. We find that $\Omega_{\mathrm{SH}} \subseteq \Omega_{\mathrm{SI}}$ holds for any $\mathbf{x} \in \mathcal{X}$, and there exists at least one $\mathbf{x} \in \mathcal{X}$ for which the inclusion is strict, i.e., $\Omega_{\mathrm{SH}} \subsetneq \Omega_{\mathrm{SI}}$. These results show that SSNI expands the space of possible purified outputs beyond what sample-shared DBP can achieve, thus enabling *greater flexibility* in the purification process.

4.3. Realization of SSNI

We now detail the empirical realizations of SSNI.

Algorithm 1 Diffusion-based Purification with SSNI.

```
Input: test samples $\mathbf{x}$, a score network $s_\theta$, a reweighting
    function $f(\cdot)$, a noise level $t^{\mathrm{S}}$ for score evaluation, and
    a pre-determined noise level $t^*$.
1: Approximate the score by $s_\theta$: $s_\theta(\mathbf{x}, t^{\mathrm{S}})$
2: Obtain the sample-specific noise level: $t(\mathbf{x}) =$
   $f(\left\|s_\theta(\mathbf{x}, t^{\mathrm{S}})\right\|, t^*)$
3: Execute forward diffusion process $\mathbf{x}_{t(\mathbf{x})} \leftarrow$ Eq. (5)
4: for $\mathbf{t} = t(\mathbf{x}), \dots, \mathbf{1}$ do
5:     Execute Reserve diffusion process $\hat{\mathbf{x}}_{t-1} \leftarrow$ Eq. (2)
6: end for
7: return purified samples $\hat{\mathbf{x}}$
```

Realization of the score. A challenge to obtaining sample-specific noise levels lies in the score network's dependency on a score-evaluation noise level as input. Estimating scores of different adversarial samples at a single fixed t^{S} introduces sensitivity: scores computed at different reference levels (e.g., $t_i^{\mathrm{S}} \neq t_j^{\mathrm{S}}$) yield inconsistent norms, biasing the selection of t^*. Crucially, the "true" optimal t^{S} for each sample is unknown a priori, which leads to a circular dependency between reference noise level selection and score estimation. To alleviate this, we leverage *expected perturbation score* (EPS), which aggregates scores of perturbed samples across a spectrum of noise levels rather than relying on a single t^{S}. This integration reduces sensitivity to individual t^{S} choices (Zhang et al., 2023). EPS is defined as

$$\mathrm{EPS}(\mathbf{x}) = \mathbb{E}_{t \sim U(0, t^{\mathrm{S}})} \nabla_{\mathbf{x}} \log p_t(\mathbf{x}), \tag{6}$$

where $p_t(\mathbf{x})$ is the marginal probability density and t^{S} is the maximum noise level for EPS. EPS computes the expectation of the scores of perturbed images across different noise levels $t \sim U(0, t^{\mathrm{S}})$, making it more robust to the changes in noise levels. Notably, this t^{S} is *different* from the sample-shared noise injection level t^*. We will omit t^{S} from the notation of $\mathrm{EPS}(\mathbf{x})$ for brevity hereafter. Following Zhang et al. (2023), we set $t^{\mathrm{S}} = 20$. In practice, a score $\nabla_{\mathbf{x}} \log_{p_t}(\mathbf{x})$ can be approximated by a score network. Specifically, we employ a score network pre-trained using the score matching objective (Song & Ermon, 2019).

Realization of the linear reweighting function. We first design a linear function to reweight t^*:

$$f_{\text{linear}}(\|\mathrm{EPS}(\mathbf{x})\|, \ t^*) = \frac{\|\mathrm{EPS}(\mathbf{x})\| - \xi_{\min}}{\xi_{\max} - \xi_{\min}} \times t^* + b, \tag{7}$$

where b is a bias term and t^* denotes the optimal sample-shared noise level selected by Nie et al. (2022). To implement this reweighting function, we extract 5,000 validation clean examples from the training data (denoted as $\mathbf{x}_v$) and we use $\|\mathrm{EPS}(\mathbf{x}_v)\|$ as a *reference* to indicate the approximate EPS norm values of clean data, which can help us reweight t^*. Then we define $\xi_{\min} = \min(\|\mathrm{EPS}(\mathbf{x})\|, \|\mathrm{EPS}(\mathbf{x}_v)\|)$ and $\xi_{\max} = \max(\|\mathrm{EPS}(\mathbf{x})\|, \|\mathrm{EPS}(\mathbf{x}_v)\|)$ to normalize $\|\mathrm{EPS}(\mathbf{x})\|$ such that the coefficient of t^* is within a range of $[0, 1]$, ensuring that the reweighted t^* stays positive and avoids unbounded growth, thus preserving the semantic information.

Realization of the non-linear reweighting function. We then design a non-linear function based on the sigmoid function, which has two horizontal asymptotes:

$$f_\sigma(\|\mathrm{EPS}(\mathbf{x})\|, t^*) = \frac{t^* + b}{1 + \exp\{-(\|\mathrm{EPS}(\mathbf{x})\| - \mu)/\tau\}}, \quad (8)$$

where b is a bias term and t^* denotes the optimal sample-shared noise level selected by Nie et al. (2022) and τ is a temperature coefficient that controls the sharpness of the function. We denote the mean value of $\|\mathrm{EPS}(\mathbf{x}_v)\|$ as μ. This ensures that when the difference between $\|\mathrm{EPS}(\mathbf{x})\|$ and μ is large, the reweighted t^* can approach to the maximum t^* in a more smooth way, and vice versa.

Adding a bias term to the reweighting function. One limitation of the above-mentioned reweighting functions is that *the reweighted t^* cannot exceed the original t^**, which may result in some adversarial noise not being removed for some adversarial examples. To address this issue, we introduce an extra bias term (i.e., b) to the reweighting function, which can increase the upper bound of the reweighted t^* so that the maximum possible reweighted t^* can exceed original t^*. Empirically, we find that this can further improve the robust accuracy without compromising the clean accuracy.

5. Experiments

In this section, we use *SSNI-L* to denote our method with the *linear* reweighting function, and use *SSNI-N* to denote our method with the *non-linear* reweighting function.

5.1. Experimental Settings

Datasets and model architectures. We consider two datasets for our evaluations: CIFAR-10 (Krizhevsky et al., 2009), and ImageNet-1K (Deng et al., 2009). For classifiers, we use the pre-trained WideResNet-28-10 and WideResNet-70-16 for CIFAR-10, and the pre-trained ResNet-50 for ImageNet-1K. For diffusion models, we employ two off-the-shelf diffusion models trained on CIFAR-10 and ImageNet-1K (Song et al., 2021b; Dhariwal & Nichol, 2021).

Evaluation metrics. For all experiments, we consider the standard accuracy (i.e., accuracy on clean examples) and robust accuracy (i.e., accuracy on adversarial examples) as the evaluation metrics.

Baseline settings. We use three well-known DBP methods as our baselines: *DiffPure* (Nie et al., 2022), *GDMP* (Wang et al., 2022) and *GNS* (Lee & Kim, 2023). The detailed configurations can be found in Appendix C. For the reverse process within diffusion models, we consider DDPM sampling method (Ho et al., 2020) in the DBP methods.

Evaluation settings for DBP baselines. Following Lee & Kim (2023), we use a fixed subset of 512 randomly sampled images for all evaluations due to high computational cost of applying adaptive white-box attacks to DBP methods. Lee & Kim (2023) provide a robust evaluation framework for existing DBP methods and demonstrate that PGD+EOT (Madry et al., 2018; Athalye et al., 2018b) is the golden standard for DBP evaluations. Therefore, following Lee & Kim (2023), we mainly use adaptive white-box PGD+EOT attack with 200 PGD iterations for CIFAR-10 and 20 PGD iterations for ImageNet-1K. We use 20 EOT iterations for all experiments to mitigate the stochasticity introduced by the diffusion models. As PGD is a gradient-based attack, we compute the gradients of the entire process from a surrogate process. The details of the surrogate process is explained in Appendix D. We also evaluate DBP methods under adaptive BPDA+EOT attack (Athalye et al., 2018a), which leverages an identity function to approximate the direct gradient rather than direct computing the gradient of the defense system.

Evaluation settings for SSNI. Since SSNI introduces an extra reweighting process than DBP baselines, we implicitly design two adaptive white-box attacks by considering the *entire defense mechanism* of SSNI (i.e., adaptive white-box PGD+EOT attack and adaptive white-box BPDA+EOT attack). *To make a fair comparison, we evaluate SSNI on adaptive white-box attacks with the same configurations mentioned above*. The algorithmic descriptions for the adaptive white-box PGD+EOT attack and adaptive white-box BPDA+EOT attack is provided in Appendix E and F. In addition, to evaluate the generalization and adaptability of SSNI to diverse adversarial attacks, we further include AutoAttack (Croce & Hein, 2020), DiffAttack (Chen et al., 2024b) and the Diff-PGD attack (Xue et al., 2023) in Section 5.4. We set the iteration number to 5 for Diff-PGD.

5.2. Defending Against Adaptive White-box PGD+EOT

We mainly present and analyze the evaluation results of *SSNI-N* in this section and the experimental results of *SSNI-L* can be found in Appendix G.

Result analysis on CIFAR-10. Table 1 shows the standard and robust accuracy against PGD+EOT $\ell_\infty(\epsilon = 8/255)$ and $\ell_2(\epsilon = 0.5)$ threat models on CIFAR-10, respectively. Notably, *SSNI-N* effectively improves the accuracy-robustness trade-off on PGD+EOT $\ell_\infty(\epsilon = 8/255)$ compared to DBP baselines. Specifically, *SSNI-N* improves standard accuracy of *DiffPure* by 3.58% on WideResNet-28-10 and WideResNet-70-16 without compromising robust accuracy. For *GDMP*, the standard accuracy grows by 1.63% on

Table 1. Standard and robust accuracy of DBP methods against adaptive white-box PGD+EOT (left: $\ell_\infty(\epsilon = 8/255)$, right: $\ell_2(\epsilon = 0.5)$) on *CIFAR-10*. WideResNet-28-10 and WideResNet-70-16 are used as classifiers. We compare the result of DBP methods with and without *SSNI-N*. We report mean and standard deviation over three runs. We show the most successful defense in **bold**. The performance improvements and degradation are reported in green and red.

	PGD+EOT ℓ_∞ $(\epsilon = 8/255)$			PGD+EOT ℓ_2 $(\epsilon = 0.5)$		
	DBP Method	Standard	Robust	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	89.71±0.72	47.98±0.64	Nie et al. (2022)	91.80±0.84	**82.81±0.97**
	+ *SSNI-N*	**93.29±0.37** (+3.58)	**48.63±0.56** (+0.65)	+ *SSNI-N*	**93.95±0.70** (+2.15)	82.75±1.01 (-0.06)
	Wang et al. (2022)	92.45±0.64	36.72±1.05	Wang et al. (2022)	92.45±0.64	82.29±0.82
	+ *SSNI-N*	**94.08±0.33** (+1.63)	**40.95±0.65** (+4.23)	+ *SSNI-N*	**94.08±0.33** (+1.63)	**82.49±0.75** (+0.20)
	Lee & Kim (2023)	90.10±0.18	56.05±1.11	Lee & Kim (2023)	90.10±0.18	83.66±0.46
	+ *SSNI-N*	**93.55±0.55** (+3.45)	**56.45±0.28** (+0.40)	+ *SSNI-N*	**93.55±0.55** (+3.45)	**84.05±0.33** (+0.39)
WRN-70-16	Nie et al. (2022)	90.89±1.13	52.15±0.30	Nie et al. (2022)	92.90±0.40	82.94±1.13
	+ *SSNI-N*	**94.47±0.51** (+3.58)	**52.47±0.66** (+0.32)	+ *SSNI-N*	**95.12±0.58** (+2.22)	**84.38±0.58** (+1.44)
	Wang et al. (2022)	93.10±0.51	43.55±0.58	Wang et al. (2022)	93.10±0.51	**85.03±0.49**
	+ *SSNI-N*	**95.57±0.24** (+2.47)	**46.03±1.33** (+2.48)	+ *SSNI-N*	**95.57±0.24** (+2.47)	84.64±0.51 (-0.39)
	Lee & Kim (2023)	89.39±1.12	56.97±0.33	Lee & Kim (2023)	89.39±1.12	84.51±0.37
	+ *SSNI-N*	**93.82±0.24** (+4.43)	**57.03±0.28** (+0.06)	+ *SSNI-N*	**93.82±0.24** (+4.43)	**84.83±0.33** (+0.32)

Table 2. Standard and robust accuracy (%) against adaptive white-box PGD+EOT $\ell_\infty(\epsilon = 4/255)$ on *ImageNet-1K*.

	PGD+EOT ℓ_∞ $(\epsilon = 4/255)$		
	DBP Method	Standard	Robust
RN-50	Nie et al. (2022)	68.23±0.92	30.34±0.72
	+ *SSNI-N*	**70.25±0.56** (+2.02)	**33.66±1.04** (+3.32)
	Wang et al. (2022)	74.22±0.12	0.39±0.03
	+ *SSNI-N*	**75.07±0.18** (+0.85)	**5.21±0.24** (+4.82)
	Lee & Kim (2023)	70.18±0.60	42.45±0.92
	+ *SSNI-N*	**72.69±0.80** (+2.51)	**43.48±0.25** (+1.03)

Table 3. Standard and robust accuracy (%) against adaptive white-box BPDA+EOT $\ell_\infty(\epsilon = 8/255)$ attack on *CIFAR-10*.

	BPDA+EOT ℓ_∞ $(\epsilon = 8/255)$		
	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	89.71±0.72	81.90±0.49
	+ *SSNI-N*	**93.29±0.37** (+3.58)	**82.10±1.15** (+0.20)
	Wang et al. (2022)	92.45±0.64	79.88±0.89
	+ *SSNI-N*	**94.08±0.33** (+1.63)	**80.99±1.09** (+1.11)
	Lee & Kim (2023)	90.10±0.18	**88.40±0.88**
	+ *SSNI-N*	**93.55±0.55** (+3.45)	87.30±0.42 (-1.10)

WideResNet-28-10 and by 2.47% on WideResNet-70-16, respectively. Notably, *SSNI-N* improves the robust accuracy of *GDMP* by 4.23% on WideResNet-28-10 and by 2.48% on WideResNet-70-16. For *GNS*, both the standard accuracy and robust accuracy are improved by a notable margin. We can observe a similar trend in PGD+EOT $\ell_2(\epsilon = 0.5)$. Despite some decreases in robust accuracy (e.g., 0.06% on *DiffPure* and 0.39% on *GDMP*), *SSNI-N* can improve standard accuracy by a notable margin, and thus improving accuracy-robustness trade-off.

Result analysis on ImageNet-1K. Table 2 presents the evaluation results against adaptive white-box PGD+EOT $\ell_\infty(\epsilon = 4/255)$ on ImageNet-1K. *SSNI-N* outperforms all baseline methods by notably improving both the standard and robust accuracy, which demonstrates the effectiveness of *SSNI-N* in defending against strong white-box adaptive attack and indicates the strong scalability of SSNI on large-scale datasets such as ImageNet-1K.

5.3. Defending Against Adaptive White-box BPDA+EOT

We mainly present and analyze the evaluation results of *SSNI-N* in this section and the experimental results of *SSNI-L* can be found in Appendix G.

We further evaluate the performance of *SSNI-N* against adaptive white-box BPDA+EOT $\ell_\infty(\epsilon = 8/255)$, which is an adaptive attack specifically designed for DBP methods (Tramèr et al., 2020; Hill et al., 2021), as demonstrated in Table 3. Specifically, incorporating *SSNI-N* with *DiffPure* can further improve the standard accuracy by 3.58% without compromising robust accuracy. Notably, incorporating *SSNI-N* with *GDMP* can improve the standard and robust accuracy *simultaneously* by a large margin. Despite some decreases in robust accuracy when incorporating *SSNI-N* with *GNS* (i.e., 1.10%), *SSNI-N* can improve standard accuracy significantly (i.e., 3.45%), and thus improving the accuracy-robustness trade-off by a notable margin.

5.4. Defending Against Additional Attacks

Table 4 reports the standard and robust accuracy of various DBP methods on CIFAR-10 under three additional white-

Table 4. Standard and robust accuracy (%) against AutoAttack (random version), DiffAttack and Diff-PGD attack with $\ell_\infty(\epsilon = 8/255)$ on *CIFAR-10*. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

	ℓ_∞ $(\epsilon = 8/255)$				
	DBP Method	Standard	AutoAttack	DiffAttack	Diff-PGD
WRN-28-10	Nie et al. (2022)	89.71±0.72	66.73±0.21	47.16±0.48	54.95±0.77
	+ *SSNI-N*	**93.29**±**0.37** (+3.58)	**66.94**±**0.44** (+0.21)	**48.15**±**0.22** (+0.99)	**56.10**±**0.35** (+1.15)
	Wang et al. (2022)	92.45±0.64	64.48±0.62	54.27±0.72	41.45±0.60
	+ *SSNI-N*	**94.08**±**0.33** (+1.63)	**66.53**±**0.46** (+2.05)	**55.81**±**0.33** (+1.54)	**42.91**±**0.56** (+1.46)
	Lee & Kim (2023)	90.10±0.18	69.92±0.30	56.04±0.58	59.02±0.28
	+ *SSNI-N*	**93.55**±**0.55** (+3.45)	**72.27**±**0.19** (+2.35)	**56.80**±**0.41** (+0.76)	**61.43**±**0.58** (+2.41)

Table 5. Ablation study on different sampling methods during the reverse diffusion process. We measure the standard and robust accuracy (%) against PGD+EOT $\ell_\infty(\epsilon = 8/255)$ on *CIFAR-10*. We use *DiffPure* as the baseline method and we set $t^* = 100$. WideResNet-28-10 is used as the classifier. We report mean and the standard deviations over three runs.

Sampling Method	Standard	Robust
sdeint solver	89.06±0.48	47.72±0.24
DDPM	89.71±0.72	47.98±0.64
DDIM	91.54±0.72	37.50±0.80

box attacks: AutoAttack, DiffAttack, and Diff-PGD, all under the $\ell_\infty(\epsilon = 8/255)$ threat model. For the *DiffPure*, *SSNI-N* improves standard accuracy by 3.58% and brings robustness gains under AutoAttack (i.e., 0.21%), DiffAttack (i.e., 0.99%), and Diff-PGD (i.e., 1.15%). For *GDMP*, *SSNI-N* increases standard accuracy by 1.63%, and achieves significant robustness improvements under AutoAttack (i.e., 2.05%) and DiffAttack (i.e., 1.54%), with a slight gain under Diff-PGD (i.e., 1.46%). Regarding *GNS*, *SSNI-N* boosts the standard accuracy by 3.45%, and the robust accuracy improves under AutoAttack (i.e., 2.35%), DiffAttack (i.e., 1.70%), and Diff-PGD (i.e., 2.41%), respectively. Overall, *SSNI-N* consistently enhances both standard and robust accuracy across all three attack types, demonstrating its strong generalization and adaptability to diverse adversarial threats.

5.5. Ablation Study

Ablation study on τ in *SSNI-N*. We investigate how the temperature coefficient τ in Eq. (8) affects the performance of *SSNI-N* against adaptive white-box PGD+EOT $\ell_\infty(\epsilon = 8/255)$ attack on CIFAR-10 in Figure 4. The temperature coefficient τ controls the sharpness of the curve of the *non-linear* reweighting function. A higher τ leads to a more smooth transition between the low and high values of the reweighting function, resulting in less sensitivity to the changes of the input. From Figure 4, the standard accuracy remains stable across different τs, while the robust accuracy increases to the climax when $\tau = 20$. Therefore, we choose

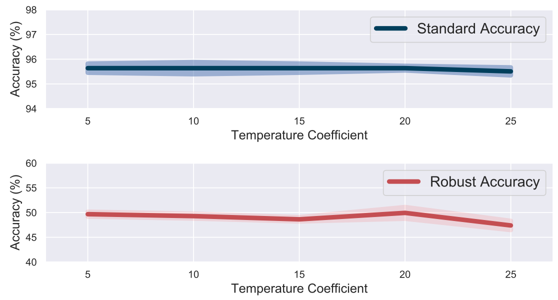


Figure 4. Standard (**top**) and robust (**bottom**) accuracy (%) vs. τ; We report mean and the standard deviations over three runs.

$\tau^* = 20$ for the non-linear reweighting function to optimize the accuracy-robustness trade-off for DBP methods.

Ablation study on sampling methods. *DiffPure* originally used an adjoint method to efficiently compute the gradients of the system, but Lee & Kim (2023) and Chen et al. (2024a) suggest to replace adjoint solver with sdeint solver for the purpose of computing full gradients more accurately (Li et al., 2020; Kidger et al., 2021). Therefore, we investigate whether using different sampling methods affect the performance of DBP methods (here we use *DiffPure* as the baseline method). We further compare the results with DDIM sampling method (Song et al., 2021a), which is a faster sampling method than DDPM (Ho et al., 2020). From Table 5, DDPM achieves the best accuracy-robustness trade-off among the three sampling methods, and thus we select DDPM as the sampling method for all baseline methods.

Ablation study on score norms. We investigate the effect of using single score norm (i.e., $\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\|$) for SSNI in Appendix H. We find that although single score norm can notably improve the standard accuracy, it reduces robust accuracy. This might be attributed to the fact that single score norm is sensitive to the purification noise levels.

Ablation study on the bias term b. We investigate how the bias term b in reweighting functions affects the performance of SSNI in Appendix I. We find that the selection of bias term will not significantly impact the performance of our

Table 6. Inference time of the DBP methods with and without SSNI for a single image running on one A100 GPU on *CIFAR-10* and *ImageNet-1K*. We use WideResNet-28-10 as the classifier for *CIFAR-10* and ResNet-50 for *ImageNet-1K*.

DBP Method	Noise Injection Method	Time (s)	DBP Method	Noise Injection Method	Time (s)
Nie et al. (2022)	-	3.934	Nie et al. (2022)	-	8.980
	SSNI-L	4.473		SSNI-L	14.515
	SSNI-N	4.474		SSNI-N	14.437
Wang et al. (2022)	-	5.174	Wang et al. (2022)	-	11.271
	SSNI-L	5.793		SSNI-L	16.657
	SSNI-N	5.829		SSNI-N	16.747
Lee & Kim (2023)	-	14.902	Lee & Kim (2023)	-	35.091
	SSNI-L	15.624		SSNI-L	40.526
	SSNI-N	15.534		SSNI-N	40.633

framework under CIFAR-10 and ImageNet-1K. Note that when bias increases, there is a general observation that the clean accuracy drops and the robust accuracy increases. This perfectly aligns with the understanding of optimal noise level selections in existing DBP methods, where a large noise level would lead to a drop in both clean and robust accuracy and a small noise level cannot remove the adversarial perturbation effectively.

Ablation study on model architectures. We also investigate how the choice of model architecture affects the performance of our method in Appendix J. Specifically, we evaluate *SSNI-N* on a Swin-Transformer (Liu et al., 2021) under the PGD+EOT $\ell_\infty(\epsilon = 8/255)$ on *CIFAR-10*. We find that the improvements brought by *SSNI-N* are consistent with those observed on CNN-based models. In particular, *SSNI-N* enhances both standard and robust accuracy across all evaluated DBP baselines. Notably, the relative improvement on robust accuracy is more significant for transformer-based classifiers. This observation suggests that *SSNI-N* can effectively complement the inherent robustness of transformer models and generalizes well across different architectures.

5.6. Compute Resource

The inference time (in seconds) for incorporating SSNI modules into existing DBP methods on CIFAR-10 and ImageNet-1K are reported in Tables 6. The inference time is measured as the time it takes for a single test image to complete the purification process. Specifically, SSNI is approximately 0.5 seconds slower than baseline methods on CIFAR-10 and 5 seconds slower than baseline methods on ImageNet-1K. Thus, compared with DBP baseline methods, this reweighting process is *lightweight*, ensuring that SSNI is computationally feasible and can be applied in practice with minimal overhead. We implemented our code on Python version 3.8, CUDA version 12.2.0, and PyTorch version 2.0.1 with Slurm Workload Manager. We conduct each of the experiments on up to 4 $\times$ NVIDIA A100 GPUs (see https://github.com/tmlr-group/SSNI).

6. Limitation

Maximum level t^{S} for EPS. We use EPS to replace the single reference noise level with an integrated approach. Still, the maximum level t^{S} defining the upper bound of the expectation range is shared across samples, which may not be the optimal choice though. Refining sample-sensitive maximum levels could potentially improve the adaptability and robustness of EPS in different scenarios, and we leave it as future work.

The design of reweighting functions. The proposed reweighting functions (i.e., the linear and non-linear reweighting functions) may not be the optimal ones for SSNI. Further efforts are needed to explore data-driven or learning-based strategies to discover more effective function forms. Meanwhile, designing an effective reweighting function is an open question, and we leave it as future work.

Extra computational cost. The integration of an extra reweighting process will inevitably bring some extra cost. Further efforts are needed to optimize the trade-off between accuracy gains and computational overhead. Luckily, we find that this reweighting process is *lightweight*, making SSNI computationally feasible compared to existing DBP methods (see Section 5.6).

7. Conclusion

In this paper, we find that an optimal t^* indeed could be different on a sample basis. Motivated by this finding, we propose a new framework called ***S****ample-specific* ***S****core-aware* ***N****oise* ***I****njection* (SSNI). SSNI sample-wisely reweights t^* for each sample based on its score norm, which generally injects less noise to clean samples and sufficient noise to adversarial samples, leading to a notable improvement in the accuracy-robustness trade-off. We hope this simple yet effective framework could open up a new perspective in DBP methods and lay the groundwork for future methods that account for sample-specific noise injections.

Acknowledgements

JCZ is supported by the Melbourne Research Scholarship and would like to thank Shuhai Zhang and Huanran Chen for productive discussions. ZSY is supported by the Australian Research Council (ARC) with grant number DE240101089. FL is supported by the ARC with grant number DE240101089, LP240100101, DP230101540 and the NSF&CSIRO Responsible AI program with grant number 2303037. This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

Impact Statement

This study on adversarial defense mechanisms raises important ethical considerations that we have carefully addressed. We have taken steps to ensure our adversarial defense method is fair. We use widely accepted public benchmark datasets to ensure comparability of our results. Our evaluation encompasses a wide range of attack types and strengths to provide a comprehensive assessment of our defense mechanism. We have also carefully considered the broader impacts of our work. The proposed defense algorithm contributes to the development of more robust machine learning models, potentially improving the reliability of AI systems in various applications. We will actively engage with the research community to promote responsible development and use of adversarial defenses.

References

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018a.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. In *ICML*, 2018b.

Bai, M., Huang, W., Li, T., Wang, A., Gao, J., Caiafa, C. F., and Zhao, Q. Diffusion models demand contrastive guidance for adversarial purification to advance. In *ICML*, 2024.

Cao, Y., Wang, N., Xiao, C., Yang, D., Fang, J., Yang, R., Chen, Q. A., Liu, M., and Li, B. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *IEEE Symposium on Security and Privacy*, pp. 176–194, 2021.

Chen, H., Dong, Y., Wang, Z., Yang, X., Duan, C., Su, H., and Zhu, J. Robust classification via a single diffusion model. In *ICML*, 2024a.

Chen, J., Chen, H., Chen, K., Zhang, Y., Zou, Z., and Shi, Z. Diffusion models for imperceptible and transferable adversarial attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024b.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Dhariwal, P. and Nichol, A. Q. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.

Dong, Y., Su, H., Wu, B., Li, Z., Liu, W., Zhang, T., and Zhu, J. Efficient decision-based black-box adversarial attacks on face recognition. In *CVPR*, 2019.

Gao, R., Liu, F., Zhang, J., Han, B., Liu, T., Niu, G., and Sugiyama, M. Maximum mean discrepancy test is aware of adversarial attacks. In *ICML*, 2021.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Han, B., Yao, J., Liu, T., Li, B., Koyejo, S., and Liu, F. *Trustworthy Machine Learning: From Data to Models*. Now Foundations and Trends, 2025.

Hill, M., Mitchell, J. C., and Zhu, S. Stochastic security: Adversarial defense using long-run dynamics of energy-based models. In *ICLR*, 2021.

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

Jing, P., Tang, Q., Du, Y., Xue, L., Luo, X., Wang, T., Nie, S., and Wu, S. Too good to be safe: Tricking lane detection in autonomous driving with crafted perturbations. In *USENIX Security Symposium*, pp. 3237–3254, 2021.

Kidger, P., Foster, J., Li, X., Oberhauser, H., and Lyons, T. Neural SDEs as Infinite-Dimensional GANs. In *ICML*, 2021.

Krizhevsky, A., Nair, V., and Hinton, G. CIFAR-10 (canadian institute for advanced research). 2009. URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.

Lee, M. and Kim, D. Robust evaluation of diffusion-based adversarial purification. In *ICCV*, 2023.

Li, X., Wong, T. L., Chen, R. T. Q., and Duvenaud, D. Scalable gradients for stochastic differential equations. In *The 23rd International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882, 2020.

Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X., and Zhu, J. Defense against adversarial attacks using high-level representation guided denoiser. In *CVPR*, 2018.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S. N. R., Schoenebeck, G., Song, D., Houle, M. E., and Bailey, J. Characterizing adversarial subspaces using local intrinsic dimensionality. In *ICLR*, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Naseer, M., Khan, S. H., Hayat, M., Khan, F. S., and Porikli, F. A self-supervised approach for adversarial robustness. In *CVPR*, 2020.

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

Pang, T., Zhang, H., He, D., Dong, Y., Su, H., Chen, W., Zhu, J., and Liu, T. Two coupled rejection metrics can tell adversarial examples apart. In *CVPR*, 2022.

Raghuram, J., Chandrasekaran, V., Jha, S., and Banerjee, S. A general framework for detecting anomalous inputs to DNN classifiers. In *ICML*, 2021.

Samangouei, P., Kabkab, M., and Chellappa, R. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *ICLR*, 2018.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *ICLR*, 2021a.

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

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. PixelDefend: Leveraging generative models to understand and defend against adversarial examples. In *ICLR*, 2018.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021b.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, 2014.

Tramèr, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

Wang, J., Lyu, Z., Lin, D., Dai, B., and Fu, H. Guided diffusion model for adversarial purification. *arXiv preprint arXiv:2205.14969*, 2022.

Wang, Q., Liu, F., Han, B., Liu, T., Gong, C., Niu, G., Zhou, M., and Sugiyama, M. Probabilistic margins for instance reweighting in adversarial training. In *NeurIPS*, 2021.

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

Xiao, C., Chen, Z., Jin, K., Wang, J., Nie, W., Liu, M., Anandkumar, A., Li, B., and Song, D. Densepure: Understanding diffusion models for adversarial robustness. In *ICLR*, 2023.

Xue, H., Araujo, A., Hu, B., and Chen, Y. Diffusion-based adversarial sample generation for improved stealthiness and controllability. In *NeurIPS*, 2023.

Yoon, J., Hwang, S. J., and Lee, J. Adversarial purification with score-based generative models. In *ICML*, 2021.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019.

Zhang, J., Rubinstein, B. I. P., Zhang, J., and Liu, F. One stone, two birds: Enhancing adversarial defense through the lens of distributional discrepancy. URL `https://arxiv.org/abs/2503.02169`.

Zhang, J., Zhu, J., Niu, G., Han, B., Sugiyama, M., and Kankanhalli, M. S. Geometry-aware instance-reweighted adversarial training. In *ICLR*, 2021.

Zhang, J., Liu, F., Zhou, D., Zhang, J., and Liu, T. Improving accuracy-robustness trade-off via pixel reweighted adversarial training. In *ICML*, 2024.

Zhang, S., Liu, F., Yang, J., Yang, Y., Li, C., Han, B., and Tan, M. Detecting adversarial data by probing multiple perturbations using expected perturbation score. In *ICML*, 2023.

Zhuang, J., Dvornek, N. C., Li, X., Tatikonda, S., Papademetris, X., and Duncan, J. S. Adaptive checkpoint adjoint method for gradient estimation in neural ODE. In *ICML*, 2020.

A. Detailed Related Work

Adversarial attack. *Adversarial examples* (AEs) have emerged as a critical security concern in the development of AI systems in recent years (Szegedy et al., 2014; Goodfellow et al., 2015). These examples are typically generated by introducing imperceptible perturbations to clean inputs, which can cause a classifier to make incorrect predictions with high confidence. The methods used to generate such examples are known as *adversarial attacks*. One of the earliest attack methods, the *fast gradient sign method* (FGSM), which perturbs clean data in the direction of the gradient of the loss function (Goodfellow et al., 2015). Building on this idea, Madry et al. (2018) propose the *projected gradient descent* (PGD) attack, which applies iterative gradient-based updates with random initialization. *AutoAttack* (AA) (Croce & Hein, 2020) integrate multiple attack strategies into a single ensemble, making it a widely adopted benchmark for evaluating adversarial robustness. To address defenses that apply randomized input transformations, Athalye et al. (2018b) introduce the *expectation over transformation* (EOT) framework for computing more accurate gradients. Additionally, Athalye et al. (2018a) propose the *backward pass differentiable approximation* (BPDA), which approximates gradients using identity mappings to circumvent gradient obfuscation defenses. According to Lee & Kim (2023), the combination of PGD and EOT, i.e., PGD+EOT, is currently considered the most effective attack strategy against DBP methods. More recently, adversarial attacks that specifically designed for DBP methods are proposed. For example, Xue et al. (2023) propose *diffusion-based projected gradient descent* (Diff-PGD), which leverages an off-the-shelf diffusion model to guide perturbation optimization, enabling the generation of more stealthy AEs. Chen et al. (2024b) propose *DiffAttack*, which crafts perturbations in the latent space of diffusion models, rather than directly in pixel space, enabling the generation of human-insensitive yet semantically meaningful AEs through content-preserving structures.

Adversarial defense. To counter the threat posed by adversarial attacks, numerous defense strategies have been developed, including *adversarial detection* (AD), *adversarial training* (AT), and *adversarial purification* (AP). *(1) AD:* AD is the most lightweight approach to defending against adversarial attacks is to detect and remove AEs from the input data. Earlier studies typically trained detectors tailored to specific classifiers or attack types, often overlooking the underlying data distribution, which limits their generalization to unseen attacks (Ma et al., 2018; Lee et al., 2018; Raghuram et al., 2021; Pang et al., 2022). Recently, *statistical adversarial data detection* has attracted growing attention for its ability to address this limitation. For instance, Gao et al. (2021) show that the *maximum mean discrepancy* (MMD) is sensitive to adversarial perturbations, and use distributional discrepancies between AEs and CEs to effectively identify and filter out AEs, even under previously unseen attacks. Building on this insight, Zhang et al. (2023) introduce a novel statistic called the *expected perturbation score* (EPS), which quantifies the average score of a sample after applying multiple perturbations. They then propose an EPS-based variant of MMD to capture the distributional differences between clean and adversarial examples more effectively. *(2) AT:* Vanilla AT (Madry et al., 2018) directly generates adversarial examples during training, encouraging the model to learn their underlying distribution. Beyond vanilla AT, various extensions have been proposed to improve its effectiveness. For instance, Zhang et al. (2019) introduce a surrogate loss optimized based on theoretical robustness bounds. Similarly, Wang et al. (2020) examine the role of misclassified examples in shaping model robustness and enhance adversarial risk via regularization techniques. From a reweighting perspective, Zhang et al. (2021) propose *geometry-aware instance-reweighted AT* (GAIRAT), which adjusts sample weights based on their distance to the decision boundary. Building on this, Wang et al. (2021) utilize probabilistic margins to reweight AEs in a continuous and path-independent manner. More recently, Zhang et al. (2024) suggest pixel-wise reweighting of AEs to explicitly direct attention toward critical image regions. *(3) AP:* AP typically employs generative models to transform AEs back into their clean counterparts before classification (Liao et al., 2018; Samangouei et al., 2018; Song et al., 2018; Naseer et al., 2020; Zhang et al.). Within this context, *diffusion-based purification* (DBP) methods have emerged as a promising framework, exploiting the inherent denoising nature of diffusion models to filter out adversarial noise (Nie et al., 2022; Wang et al., 2022; Xiao et al., 2023; Lee & Kim, 2023). DBP works with *pre-trained* diffusion model by first corrupting an adversarial input through the forward process and then iteratively reconstructing it via the reverse process. This projects the input back onto the clean data manifold while stripping away adversarial artifacts. Wang et al. (2022) introduce input guidance during the reverse diffusion process to ensure the purified outputs stay close to the inputs. Lee & Kim (2023) propose a fine-tuned gradual noise scheduling for multi-step purifications. Bai et al. (2024) improve the reverse diffusion process by incorporating contrastive objectives.

B. Justification of Section 4.2

Sampled-shared DBP is a special case of SSNI.

Proof. Let Φ_{SH} be a sample-shared purification operator with constant noise level t^*. We can express Φ_{SS} as a sample-specific purification operator Φ_{SI} by defining the reweighting function f as

$$f(z, t^*) = t^* \quad \forall z \in \mathbb{R}, t \in [0, T_{\max}].$$

Then, for any $\mathbf{x} \in \mathcal{X}$, we have

$$\begin{aligned}\Phi_{\mathrm{SI}}(\mathbf{x}) &= R(\mathbf{x}_{T(\mathbf{x})}) \\ &= R(\mathbf{x}_{f(||s_\theta(\mathbf{x},t^*)||,t^*)}) \\ &= R(\mathbf{x}_{t^*}) \\ &= \Phi_{\mathrm{SH}}(\mathbf{x}).\end{aligned}$$

□

SSNI has a Higher Purification Capacity.

Statement 1 [Comparison of Purification Range]: For any $\mathbf{x} \in \mathcal{X}$, we have $\Omega_{\mathrm{SH}}(\mathbf{x}) \subseteq \Omega_{\mathrm{SI}}(\mathbf{x})$.

Proof. Let $\mathbf{y} \in \Omega_{\mathrm{SH}}(\mathbf{x})$. Then $\exists t^* \in [0, T_{\max}]$ such that $\mathbf{y} = R(\mathbf{x}_{t^*})$. Define $f(z, t^*) = t^*$ for all z and t^*, then $t(\mathbf{x}) = f(||s_\theta(\mathbf{x}, t^*)||, t^*) = t^*$. Therefore, $\mathbf{y} = R(\mathbf{x}_{t^*}) = R(\mathbf{x}_{t(\mathbf{x})}) \in \Phi_{\mathrm{SI}}(\mathbf{x})$. This completes the proof of $\Omega_{\mathrm{SH}}(\mathbf{x}) \subseteq \Omega_{\mathrm{SI}}(\mathbf{x})$. □

Statement 2 [Strict Inclusion]: There exists $\mathbf{x} \in \mathcal{X}$, we have $\Omega_{\mathrm{SH}}(\mathbf{x}) \subsetneq \Omega_{\mathrm{SI}}(\mathbf{x})$.

Proof. Consider a non-constant score function $s_\theta(\mathbf{x}, t)$ and a non-trivial reweighting function f. We can choose $\mathbf{x}$ such that $t(\mathbf{x}) \neq t^*$ for any fixed t^*. Then $R(\mathbf{x}_{t(\mathbf{x})}) \in \Phi_{\mathrm{SI}}(\mathbf{x})$ but $R(\mathbf{x}_{t(\mathbf{x})}) \neq \Phi_{\mathrm{SH}}$. This completes the proof of $\Omega_{\mathrm{SH}}(\mathbf{x}) \subsetneq \Omega_{\mathrm{SI}}(\mathbf{x})$. □

C. Defense Methods Configurations

For all chosen DBP methods, we utilize surrogate process to obtain gradients of the defense system during white-box adaptive attack, but we directly compute the full gradients during defense evaluation. Furthermore, we consistently apply DDPM sampling method to the selected DBP methods, which means we replace the numeric SDE solver (**sdeint**) with DDPM sampling method in DiffPure (Nie et al., 2022) and GDMP (Wang et al., 2022). The reason is that the SDE solver does not support sample-specific timestep input. For DDPM sampling, we can easily manipulate sample-specific timestep input by using matrix operation.

C.1. DiffPure

Existing DBP methods generally follow the algorithm of DiffPure (Nie et al., 2022). DiffPure conducts evaluation on AutoAttack (Croce & Hein, 2020) and BPDA+EOT adaptive attack (Athalye et al., 2018a) to measure model robustness. DiffPure chooses optimal $t^* = 100$ and $t^* = 75$ on CIFAR-10 against threat models $\ell_\infty(\epsilon = 8/255)$ and $\ell_2(\epsilon = 0.5)$, respectively. It also tests on high-resolution dataset like ImageNet-1K with $t^* = 150$ against threat models $\ell_\infty(\epsilon = 4/255)$. Calculating exact full gradients of the defense system of DiffPure is impossible since one attack iteration requires 100 function calls (with $t^* = 100$ and a step size of 1). DiffPure originally uses numerical SDE solver for calculating gradients. However, the adjoint method is insufficient to measure the model robustness since it relies on the performance of the underlying SDE solver (Zhuang et al., 2020; Lee & Kim, 2023; Chen et al., 2024a). Therefore, we apply surrogate process to efficiently compute gradients of direct back-propagation in our evaluation. To overcome memory constraint issue, we align the step size settings of denoising process in adversarial attack to 5 with the evaluation settings in (Lee & Kim, 2023) and keep the timestep t^* consistent with DiffPure. For ImageNet-1K evaluation, we can only afford a maximum of 10 function calls for one attack iteration.

C.2. GDMP

GDMP basically follows the purification algorithm proposed in DiffPure (Nie et al., 2022; Wang et al., 2022), but their method further introduces a novel guidance and use multiple purification steps sequentially. GDMP proposes to use gradients of a distance between an initial input and a target being processed to preserve semantic information, shown in Eq. (9).

$$\mathbf{x}_{t-1} \sim \mathcal{N}(\boldsymbol{\mu}_\theta - s\boldsymbol{\Sigma}_\theta \nabla_{\mathbf{x}^t} \mathcal{D}(\mathbf{x}^t, \mathbf{x}^t_{\text{adv}}), \boldsymbol{\Sigma}_\theta), \tag{9}$$

Given a DDPM $(\mu_\phi(\mathbf{x}_t), \Sigma_\theta(\mathbf{x}_t))$, a gradient scale of guidance s. $\mathbf{x}_t$ is the data being purified, and $\mathbf{x}^t_{\text{adv}}$ is the adversarial example at t. Also, GDMP empirically finds that multiple purification steps can improve the robustness. In the original evaluation of GDMP, the defense against the PGD attack consists of four purification steps, with each step including 36 forward steps and 36 denoising steps. For BPDA+EOT adaptive attack, GDMP uses two purification steps, each consisting of 50 forward steps and 50 denoising steps.

Lee & Kim (2023) evaluated GDMP with three types of guidance and concluded that No-Guidance provides the best robustness performance when using the surrogate process to compute the full gradient through direct backpropagation. In our evaluation, we incorporate the surrogate process with No-Guidance to evaluate GDMP. Since it is impossible to calculate the gradients of the full defense system, we use a surrogate process consisting of same number of purification steps but with larger step size in the attack (with 6 denoising steps and 10 denoising steps for PGD+EOT and BPDA+EOT attack, respectively). Notably, GDMP only uses one purification run with 45 forward steps to evaluate on ImageNet-1K, which we keep consistent with this setting.

C.3. GNS

Lee & Kim (2023) emphasizes the importance of selecting optimal hyperparameters in DBP methods for achieving better robust performance. Hence, Lee & Kim (2023) proposed Gradual Noise-Scheduling (GNS) for multi-step purification, which is based on the idea of choosing the best hyperparameters for multiple purification steps. It is basically the same architecture as GDMP (no guidance), but with different purification steps, forward steps and denoising steps. Specifically, GNS sets different forward and reverse diffusion steps and gradually increases the noise level at each subsequent purification step. We just keep the same hyperparameter settings and also use an ensemble of 10 runs to evaluate the method.

D. Surrogate Process of Gradient Computation

The surrogate process is an efficient approach for computing approximate gradients through backpropagation, as proposed by (Lee & Kim, 2023). White-box adaptive attacks, such as PGD+EOT, involve an iterative optimization process that requires computing the exact full gradients of the entire system, result in high memory usage and increased computational time. DBP methods often include a diffusion model as an add-on purifier, which the model requires extensive function calls during reverse generative process. Hence, it is hard to compute the exact full gradient of DBP systems efficiently. The surrogate process takes advantage of the fact that, given a fixed total amount of noise, we can denoise it using fewer denoising steps (Song et al., 2021a), but the gradients obtained from the surrogate process slightly differ from the exact gradients. Instead of using the full denoising steps, we approximate the original denoising process with fewer function calls, which allows us to compute gradients by directly back-propagating through the forward and reverse processes.

There are other gradient computation methods such as adjoint method in DiffPure (Li et al., 2020; Nie et al., 2022). It leverages an underlying numerical SDE solver to solve the reverse-time SDE. The adjoint method can theoretically compute exact full gradient, but in practice, it relies on the performance of the numerical solver, which is insufficient to measure the model robustness (Zhuang et al., 2020; Lee & Kim, 2023; Chen et al., 2024a). Lee & Kim (2023) conducted a comprehensive evaluation of both gradient computation methods and concluded that utilizing the surrogate process for gradient computation poses a greater threat to model robustness. Hence, we use gradients obtained from a surrogate process in all our experiments.

E. Adaptive White-box PGD+EOT Attack for SSNI

Madry et al. (2018) proposed Projected Gradient Descent (PGD), which is a strong iterative adversarial attack. Combining PGD with Expectation Over Transformation (EOT) (Athalye et al., 2018b) has become a powerful adaptive white-box attack against AP methods. In our defense system, we add a reweighting module to existing DBP methods, so this information must be included during a white-box attack. We decide to place the reweighting process outside the EOT loops because EOT provides more samples with different random transformations. It effectively reduces the randomness of gradient computation but this process does not greatly affect the EPS value of the data. In addition, this approach further reduces the computational cost.

Algorithm 2 Adaptive white-box PGD+EOT attack for SSNI.

```
Require: clean data-label pairs $(\mathbf{x}, y)$; purifier $f_p$; classifier $f_c$; a noise level $T$; a score network $s_\theta(\mathbf{x}, T)$; objective function $\mathcal{L}$; perturbation budget $\epsilon$; step size $\alpha$; PGD iterations $K$; EOT iterations $N$.
1: Initialize $\mathbf{x}_0^{\text{adv}} \leftarrow \mathbf{x}$
2: for $k = 0, ..., K-1$ do
3:     Computing sample-specific noise levels: $t(\mathbf{x}_k^{\text{adv}}) \leftarrow f(\left\|s_\theta(\mathbf{x}_k^{\text{adv}}, T)\right\|, T)$
4:     Average the gradients over EOT: $g_k \leftarrow \frac{1}{N}\sum_{i=1}^{N} \nabla_{\mathbf{x}^{\text{adv}}} \mathcal{L}\left(f_c(f_p(\mathbf{x}_k^{\text{adv}}, t(\mathbf{x}_k^{\text{adv}}))), y\right)$
5:     Update adversarial examples: $\mathbf{x}_{k+1}^{\text{adv}} \leftarrow \Pi_{\mathcal{B}_\epsilon(\mathbf{x})}\left(\mathbf{x}_k^{\text{adv}} + \alpha \cdot \text{sign}(g_k)\right)$
6: end for
7: return $\mathbf{x}^{\text{adv}} = \mathbf{x}_K^{\text{adv}}$
```

F. Adaptive White-box BPDA+EOT Attack for SSNI

Athalye et al. (2018a) proposed Backward Pass Differentiable Approximation (BPDA), which is a popular adaptive attack to DBP defense methods. When a defense system contains non-differentiable components where gradients cannot be directly obtained, the BPDA method substitutes these non-differentiable operations with a differentiable approximation during backpropagation in order to compute the gradients. It typically utilizes an identity function $f(\mathbf{x}) = \mathbf{x}$ as the differentiable approximation function. In specific, computing the exact full gradients via direct backpropagation through a diffusion model is time-consuming and memory-intensive, so BPDA attack assumes the purification process is an identity mapping. This implies that we ignore the impact of the purification on the gradients and directly treat the gradient with respect to the purified data $\hat{\mathbf{x}}_0$ as the gradient with respect to the input $\mathbf{x}_0$.

Algorithm 3 Adaptive white-box BPDA+EOT attack.

```
Require: clean data-label pairs $(\mathbf{x}, y)$; purifier $f_p$; classifier $f_c$; approximation function $f_{\text{apx}}$; a noise level $T$; a score network $s_\theta(\mathbf{x}, T)$; objective function $\mathcal{L}$; perturbation budget $\epsilon$; step size $\alpha$; PGD iterations $K$; EOT iterations $N$.
1: Initialize $\mathbf{x}_0^{\text{adv}} \leftarrow \mathbf{x}$
2: for $k \leftarrow 0$ to $K-1$ do
3:     Computing sample-specific $t$: $t(\mathbf{x}_k^{\text{adv}}) \leftarrow f(\left\|s_\theta(\mathbf{x}_k^{\text{adv}}, T)\right\|, T)$
4:     Average the gradient over EOT samples: $g_k \leftarrow \frac{1}{N}\sum_{i=1}^{N} \nabla_{\mathbf{x}^{\text{adv}}} \mathcal{L}\left((f_c(f_{\text{apx}}(f_p(\mathbf{x}_k^{\text{adv}})))), y\right)$
5:     Update adversarial examples: $\mathbf{x}_{k+1}^{\text{adv}} \leftarrow \Pi_{\mathcal{B}_\epsilon(\mathbf{x})}\left(\mathbf{x}_k^{\text{adv}} + \alpha \cdot \text{sign}(g_k)\right)$
6: end for
7: return $\mathbf{x}^{\text{adv}} = \mathbf{x}_K^{\text{adv}}$
```

G. Performance Evaluation of SSNI-L

We also incorporate *SSNI-L* reweighting framework with existing DBP methods for accuracy-robustness evaluation. In Table 7, we report the results against PGD+EOT $\ell_\infty(\epsilon = 8/255)$ and $\ell_2(\epsilon = 0.5)$ threat models on CIFAR-10, respectively. We can see that *SSNI-L* can still support DBP methods to better trade-off between standard accuracy and robust accuracy. Also, we report the results against BPDA+EOT $\ell_\infty(\epsilon = 8/255)$ threat model on CIFAR-10 in Table 3. Overall, *SSNI-L* slightly decreases the robustness of DBP methods against PGD+EOT attack and maintain the robustness of DBP methods against BPDA+EOT attack, but there is a notable improvement in standard accuracy.

Table 7. Standard and robust accuracy of DBP methods against adaptive white-box PGD+EOT (left: $\ell_\infty(\epsilon = 8/255)$, right: $\ell_2(\epsilon = 0.5)$) on *CIFAR-10*. WideResNet-28-10 and WideResNet-70-16 are used as classifiers. We compare the result of DBP methods with and without *SSNI-L*. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

	PGD+EOT ℓ_∞ ($\epsilon = 8/255$)		
	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	89.71±0.72	**47.98±0.64**
	+ *SSNI-L*	**92.97±0.42**	46.35±0.72
	Wang et al. (2022)	92.45±0.64	**36.72±1.05**
	+ *SSNI-L*	**93.62±0.49**	36.59±1.29
	Lee & Kim (2023)	90.1±0.18	**56.05±1.11**
	+ *SSNI-L*	**93.49±0.33**	53.71±0.48
WRN-70-16	Nie et al. (2022)	90.89±1.13	**52.15±2.30**
	+ *SSNI-L*	**93.82±0.49**	49.94±0.33
	Wang et al. (2022)	93.10±0.51	**43.55±0.58**
	+ *SSNI-L*	**93.88±0.49**	43.03±0.60
	Lee & Kim (2023)	89.39±1.12	**56.97±0.33**
	+ *SSNI-L*	**92.64±0.40**	52.86±0.46

	PGD+EOT ℓ_2 ($\epsilon = 0.5$)		
	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	91.80±0.84	**82.81±0.97**
	+ *SSNI-L*	**93.82±0.37**	81.12±0.80
	Wang et al. (2022)	92.45±0.64	**82.29±0.82**
	+ *SSNI-L*	**93.62±0.49**	80.66±1.31
	Lee & Kim (2023)	90.10±0.18	83.66±0.46
	+ *SSNI-L*	**93.49±0.33**	**85.29±0.24**
WRN-70-16	Nie et al. (2022)	92.90±0.40	82.94±1.13
	+ *SSNI-L*	**94.99±0.24**	**84.44±0.56**
	Wang et al. (2022)	93.10±0.51	**85.03±0.49**
	+ *SSNI-L*	**93.88±0.49**	82.88±0.79
	Lee & Kim (2023)	89.39±1.12	84.51±0.37
	+ *SSNI-L*	**92.64±0.40**	**84.90±0.09**

Table 8. Standard and robust accuracy (%) against adaptive white-box BPDA+EOT $\ell_\infty(\epsilon = 8/255)$ attack on *CIFAR-10*. We compare the result of DBP methods with and without *SSNI-L*. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

	BPDA+EOT ℓ_∞ ($\epsilon = 8/255$)		
	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	89.71±0.72	**81.90±0.49**
	+ *SSNI-L*	**92.97±0.42**	80.08±0.96
	Wang et al. (2022)	92.45±0.64	79.88±0.89
	+ *SSNI-L*	**93.62±0.49**	**79.95±1.12**
	Lee & Kim (2023)	90.10±0.18	88.40±0.88
	+ *SSNI-L*	**93.49±0.33**	**88.41±0.09**

H. Ablation Study on Score Norm

We further provide experiments with the single score norm instead of the EPS norm. Yoon et al. (2021) shows score norm $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is a valid measurement for adversarial detection. Incorporating single score norm with our SSNI-N framework, it still achieves notable improvement on standard accuracy, but the robustness drops. The single score norm is highly sensitive to noise levels, which makes it insufficient to completely distinguish between natural and adversarial examples.

Table 9. Standard and robust accuracy (%) against adaptive white-box PGD+EOT $\ell_\infty(\epsilon = 8/255)$ attack on *CIFAR-10*. We use *single score norms* (i.e., $\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\|$). We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

	PGD+EOT ℓ_∞ ($\epsilon = 8/255$)		
	DBP Method	Standard	Robust
WRN-28-10	Nie et al. (2022)	89.71±0.72	**47.98±0.64**
	+ *SSNI-L*	91.31±0.24	46.92±0.52
	+ *SSNI-N*	**92.84±0.18**	47.20±1.22
	Wang et al. (2022)	92.45±0.64	**36.72±1.05**
	+ *SSNI-L*	93.15±0.92	35.72±1.33
	+ *SSNI-N*	**93.42±0.60**	34.24±1.45
	Lee & Kim (2023)	90.10±0.18	**56.05±1.11**
	+ *SSNI-L*	93.40±0.49	54.52±0.46
	+ *SSNI-N*	**93.55±0.42**	55.47±1.15

I. Ablation Study on Bias Term

Table 10. Ablation study on the bias term b. We report the standard and robust accuracy of DBP methods against adaptive white-box PGD+EOT on *CIFAR-10*. WideResNet-28-10 and WideResNet-70-16 are used as classifiers. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

PGD+EOT ℓ_∞ ($\epsilon = 8/255$)								
	Bias	0	5	10	15	20	25	30
SSNI-N	WRN-28-10							
	Standard	**94.34±1.43**	93.95±1.17	93.17±1.05	92.38±1.29	92.38±1.29	92.10±1.03	92.97±0.37
	Robust	55.27±0.75	57.23±1.62	57.03±0.54	57.64±0.89	57.64±0.89	58.24±1.22	**59.18±1.65**
	WRN-70-16							
	Standard	94.34±0.45	93.82±1.56	**94.73±1.34**	92.79±0.89	92.79±0.89	92.58±0.67	92.77±0.94
	Robust	56.45±1.22	57.03±0.78	58.10±0.24	58.79±1.48	**59.57±1.19**	58.59±0.52	58.59±0.52
SSNI-L	WRN-28-10							
	Standard	92.91±0.55	92.97±0.42	**93.01±0.78**	92.64±0.28	92.53±0.84	91.69±1.02	91.45±0.88
	Robust	46.29±0.46	46.35±0.72	46.28±0.35	46.75±0.63	47.11±0.92	47.05±0.50	**47.23±0.76**
	WRN-70-16							
	Standard	93.79±0.53	**93.82±0.49**	93.77±0.64	92.85±0.82	92.32±0.90	92.08±0.40	91.89±0.52
	Robust	49.34±0.85	49.94±0.33	48.83±0.92	50.72±0.77	50.80±0.94	**51.25±0.66**	51.05±1.11

Table 11. Ablation study on the bias term b. We report standard and robust accuracy of DBP methods against adaptive white-box PGD+EOT on *ImageNet-1K*. ResNet-50 is used as the classifier. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

PGD+EOT ℓ_∞ ($\epsilon = 4/255$)								
	Bias	0	25	50	75	100	125	150
SSNI-N	RN-50							
	Standard	71.68±1.12	71.73±1.49	**71.96±0.13**	68.80±0.74	68.41±0.59	67.63±1.08	66.45±1.53
	Robust	39.33±0.34	40.28±0.28	**43.88±0.22**	41.45±0.38	43.02±0.41	40.87±0.92	40.05±0.67

J. Ablation Study on Model Architecture

Table 12. Standard and robust accuracy (%) against adaptive white-box PGD+EOT $\ell_\infty(\epsilon = 8/255)$ attack on *CIFAR-10*. The target classifier is a Swin-Transformer. We report mean and standard deviation over three runs. We show the most successful defense in **bold**.

PGD+EOT ℓ_∞ ($\epsilon = 8/255$)			
	DBP Method	Standard	Robust
Swin-T	Nie et al. (2022)	88.48±0.49	37.11±0.83
	+ *SSNI-N*	**88.67±0.55**	**39.65±0.47**
	Wang et al. (2022)	88.09±0.72	27.34±0.18
	+ *SSNI-N*	**89.45±0.34**	**28.71±0.27**
	Lee & Kim (2023)	88.67±0.20	52.54±0.14
	+ *SSNI-N*	**91.21±0.11**	**54.10±0.22**

K. Additional Justification of the Relationship Between Score Norms and Noise Levels

Extending the empirical findings from Section 3, we further justify how score norms of different samples correlate with the noise level t^* in diffusion models. With Proposition K.5, we find that the score norm varies with t, and the variation exceeds a threshold $\epsilon > 0$ when the time difference $|t_2 - t_1|$ is sufficiently large. Thus, for two noisy samples $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ with *different score norms*, the monotonic increase of $\{\beta_t\}_{t\in[0,T]}$ with t implies that different score norms correspond to different noise levels, i.e., $t_1 \neq t_2$. It implies that, for samples with different score norms, it is likely that they have undergone *different noise injection steps*. [1].

Definition K.1 (Marginal Probability Density)**.** Let $\mathcal{X}$ be the image sample space, $p(\mathbf{x}_0)$ be the natural data distribution over $\mathcal{X}$, and the diffusion process kernel defined in Eq. (1). The marginal probability density $p_t : \mathcal{X} \to \mathbb{R}_+$ at time $0 \leq t \leq T$ can be expressed as:

$$p_t(\mathbf{x}) = \int_{\mathcal{X}} q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0, \tag{10}$$

where $q(\mathbf{x}|\mathbf{x}_0)$ describes how a natural sample $\mathbf{x}_0$ evolves under the forward process to $\mathbf{x}$ at time t.

Before proceeding with the proof of Lemma K.4 and Proposition K.5, we start by presenting two lemmas to facilitate the proof.

Lemma K.2. *Let $p_t(\mathbf{x})$ denote the marginal probability density of $\mathbf{x}$ at time t. For any $\mathbf{x} \in \mathcal{X}$ and $0 \leq t \leq T$, the score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ at time t can be expressed as:*

$$\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) = \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}_0)\right],$$

where $p(\mathbf{x}_0|\mathbf{x})$ is the posterior given by Bayes' Rule:

$$p(\mathbf{x}_0|\mathbf{x}) = \frac{q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0}.$$

Proof.

$$\begin{aligned}
\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) &= \nabla_{\mathbf{x}} \log \int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0 && \text{(by Def. K.1)} \\
&= \frac{\nabla_{\mathbf{x}} \int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0} && \text{(chain rule)} \\
&= \frac{\int \nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0} && \text{(Leibniz integral rule)} \\
&= \frac{\int \frac{\nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}_0)}{q(\mathbf{x}|\mathbf{x}_0)} q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0} && \text{(manipulate } q(\mathbf{x}|\mathbf{x}_0)) \\
&= \frac{\int (\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}_0)) q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0} && \text{(chain rule)} \\
&= \int (\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}_0)) p(\mathbf{x}_0|\mathbf{x})d\mathbf{x}_0 && (p(\mathbf{x}_0|\mathbf{x}) \triangleq \frac{q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)}{\int q(\mathbf{x}|\mathbf{x}_0)p(\mathbf{x}_0)d\mathbf{x}_0})
\end{aligned} \tag{11}$$

This ends up with the expectation over the posterior $p(\mathbf{x}_0|\mathbf{x})$ as $\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}_0)\right]$. □

Lemma K.2 establishes a relationship between the marginal density and the forward process, where we can express the behavior of $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ in terms of the well-defined $q(\mathbf{x}|\mathbf{x}_0)$.

Lemma K.3. *Consider the forward process $q(\mathbf{x}_t|\mathbf{x}_0)$ is defined with noise schedule $\{\beta_t\}_{t\in[0,T]}$, where $\beta_t \in (0, 1)$ for all $0 \leq t \leq T$. Then, for any $\mathbf{x} \in \mathcal{X}$ and $0 \leq t \leq T$, the score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ can be decomposed as:*

$$\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) = -g(t)\mathbf{x} + \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right],$$

where $g(t) = 1/(1 - \bar{\alpha}_t)$ and $\bar{\alpha}_t = \prod_{i=1}^{t}(1 - \beta_i)$.

[1]However, we note that this analysis only focuses on the property of diffusion models, disregarding the potential interactions between Gaussian and adversarial noise (which is often difficult to analyze directly). In this sense, it does *not* formally motivate the proposal of SSNI and is *not* considered a core contribution of this study. We include this part of the content for completeness.

Proof. Denote the score function $s_t(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ for convenience. By applying the results from Lemma K.2, we write:

$$s_t(\mathbf{x}) = \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[q(\mathbf{x}|\mathbf{x}_0)\right].$$

Provided that $q(\mathbf{x}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$, we compute the gradient of the log-probability of the forward process as:

$$\begin{aligned}\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}_0) &= \nabla_{\mathbf{x}}\mathcal{N}\left(\mathbf{x}; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right) \\ &= \nabla_{\mathbf{x}}\left[-\frac{1}{2(1-\bar{\alpha}_t)}(\mathbf{x}-\sqrt{\bar{\alpha}_t}\mathbf{x}_0)^\top(\mathbf{x}-\sqrt{\bar{\alpha}_t}\mathbf{x}_0) + C\right] \\ &= -\frac{1}{1-\bar{\alpha}_t}(\mathbf{x}-\sqrt{\bar{\alpha}_t}\mathbf{x}_0),\end{aligned}$$

where C is a constant term independent of $\mathbf{x}$. Substitute the result back into $s_t(\mathbf{x})$, we have:

$$\begin{aligned}s_t(\mathbf{x}) &= \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[q(\mathbf{x}|\mathbf{x}_0)\right] \\ &= -\frac{1}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\mathbf{x}-\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right] \\ &= -\frac{1}{1-\bar{\alpha}_t}\mathbf{x} + \frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0] \\ &= -g(t)\mathbf{x} + \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right].\end{aligned}$$

The Lemma is completed by substituting $g(t) = \frac{1}{1-\bar{\alpha}_t}$ into the last equation. □

Lemma K.3 expresses the score function by the noisy data $\mathbf{x}$ with noise level t and the clean data $\mathbf{x}_0$. We next examine how the score norm evolves to further understand its behavior as time increases.

Lemma K.4. *Suppose there exists a constant $K > 0$ such that for all $t \geq 0$ and all $\mathbf{x}_t \in \mathcal{X}$, the expected norm of the clean data given $\mathbf{x}_t$ satisfies: $\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0|\mathbf{x})}[\|\mathbf{x}_0\|] \leq K\|\mathbf{x}_t\|$. Then, there exist constants $0 < C < 1$ and $T_0 > 0$ such that for all $t \geq T_0$:*

$$\frac{\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\|}{\|\mathbf{x}\|} > C.$$

Proof. Again, denote $s_t(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$. From Lemma K.3, we have:

$$s_t(\mathbf{x}) = -g(t)\mathbf{x} + \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right].$$

Applying the triangle inequality to it, we have:

$$\begin{aligned}\|s_t(\mathbf{x})\| &\geq g(t)\|\mathbf{x}\| - \left\|\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right]\right\| \\ &\geq g(t)\|\mathbf{x}\| - g(t)\sqrt{\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\|\mathbf{x}_0\|\right],\end{aligned}$$

where the second inequality comes from applying Jensen's inequality: $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$ for convex function f and random variable X, which leads to:

$$\begin{aligned}\left\|\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right]\right\| &\leq \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\left\|g(t)\sqrt{\bar{\alpha}_t}\mathbf{x}_0\right\|\right] \\ &= g(t)\sqrt{\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\|\mathbf{x}_0\|\right].\end{aligned}$$

As we have assumed the existence of $K > 0$ that makes $\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0|\mathbf{x})}[\|\mathbf{x}_0\|] \leq K\|\mathbf{x}_t\|$ holds for all $\mathbf{x} \in \mathcal{X}$, we have:

$$\begin{aligned}\|s_t(\mathbf{x})\| &\geq g(t)\|\mathbf{x}\| - g(t)\sqrt{\bar{\alpha}_t}K\|\mathbf{x}\| \\ &= g(t)\|\mathbf{x}\|(1 - K\sqrt{\bar{\alpha}_t}).\end{aligned}$$

We then turn to look into the asymptotic behavior of $g(t)$. We have $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. Since $0 < 1-\beta_s < 1$ for all s, and the sequence is decreasing (as β_s is increasing), it is easy to check that

$$\lim_{t\to\infty} \bar{\alpha}_t = 0$$

and

$$\lim_{t\to\infty} g(t) = \lim_{t\to\infty} \frac{1}{1-\bar{\alpha}_t} = 1.$$

This can then be formalized as, for any $\epsilon > 0$, there exists a T_ϵ such that for all $t > T_\epsilon$:

$$|g(t) - 1| < \epsilon, \text{ and } \sqrt{\bar{\alpha}_t} < \epsilon.$$

Then, for $t > T_\epsilon$, we have

$$\begin{aligned}
\|s_t(\mathbf{x})\| &\geq g(t)\,\|\mathbf{x}\|\,(1 - K\sqrt{\bar{\alpha}_t}) \\
&> (1-\epsilon)\,\|\mathbf{x}\|\,(1-K\epsilon), \\
&= (1 - \epsilon - K\epsilon + K\epsilon^2)\,\|\mathbf{x}\| \\
&= (1 - (K+1)\epsilon + K\epsilon^2)\,\|\mathbf{x}\|\,.
\end{aligned}$$

To establish the desired inequality $\|s_t(\mathbf{x})\| \geq C\,\|\mathbf{x}\|$ for constants $C > 0$, we next investigate whether this quatratic inequality $K\epsilon^2 + (K+1)\epsilon + (1-C) > 0$ can be solved. Denote the discriminant $D = (K+1)^2 - 4K(1-C)$, we have

$$\begin{aligned}
D &= (K+1)^2 - 4K(1-C) \\
&= K^2 - 2K + 1 + 4KC \\
&= (K-1)^2 + 4KC \\
&> 0 \qquad \text{(since } (K-1)^2 \geq 0 \text{ and } K > 0\text{)}.
\end{aligned}$$

Then, let $\epsilon_1 < \epsilon_2$ be two real roots of $K\epsilon^2 + (K+1)\epsilon + (1-C) = 0$, as the parabola opens upwards, i.e., $K > 0$, note that $\epsilon > 0$, we further need to ensure the smaller root $\epsilon = \frac{K+1-\sqrt{D}}{2K} > 0$, such that

$$\begin{aligned}
K + 1 &> \sqrt{(K+1)^2 - 4K(1-C)} \\
(K+1)^2 &> (K-1)^2 + 4KC \\
4K &> 4KC \\
1 &> C.
\end{aligned}$$

Putting them together, we have $\epsilon \in (0, \epsilon_1) \cup (\epsilon_2, \infty)$ that makes $\|s_t(\mathbf{x})\| \geq C\,\|\mathbf{x}\|$ hold, given a constant $0 < C < 1$ in relation to K and ϵ. Setting $T_0 = T_\epsilon$ completes the proof. ☐

Proposition K.5. *Consider the diffusion model satisfying all conditions as specified in Lemma K.4. Assume that there exist constants $K > 0$, such that $\beta_t \leq K$ for all $t \geq 0$. Additionally, suppose $\|\mathbf{x}\| \leq M$ for any $\mathbf{x} \in \mathcal{X}$, for some $M > 0$. Then, for any ϵ, there exists a constant $\Delta = 2\epsilon/(CK)$ such that for $t_1, t_2 \geq 0$, we have:*

$$|\,\|\nabla_{\mathbf{x}} \log p_{t_1}(\mathbf{x})\| - \|\nabla_{\mathbf{x}} \log p_{t_2}(\mathbf{x})\|\,| > \epsilon, \quad \textit{with } |t_1 - t_2| \geq \Delta.$$

Proof. Denote $s_t(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$. Recall that we can express the score function as

$$\begin{aligned}
s_t(\mathbf{x}) &= \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[q(\mathbf{x}|\mathbf{x}_0)\right] \\
&= -\frac{1}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}\left[\mathbf{x} - \sqrt{\bar{\alpha}_t}\mathbf{x}_0\right] \\
&= -\frac{1}{1-\bar{\alpha}_t}\mathbf{x} + \frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0]
\end{aligned}$$

Let $g(t) = \|s_t(\mathbf{x})\|^2$. We compute $\frac{\partial g(t)}{\partial t}$ using the chain rule and the product rule:

$$\begin{aligned}
\frac{\partial g(t)}{\partial t} &= 2\left\langle s_t(\mathbf{x}), \frac{\partial s_t(\mathbf{x})}{\partial t}\right\rangle \\
&= 2\left\langle s_t(\mathbf{x}), \frac{\partial}{\partial t}\left(-\frac{1}{1-\bar{\alpha}_t}\mathbf{x} + \frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0]\right)\right\rangle
\end{aligned}$$

Recall that $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$, we next derive the derivative of $\bar{\alpha}_t$. We consider the continuous approximation of t, where the product is approximated as an exponential of the integral

$$\bar{\alpha}_t = \exp\left(\int_0^t \log(1-\beta_i)\mathrm{d}i\right).$$

As β_t is typically assumed to be small (which is an implicit common practice (Ho et al., 2020)), we further simplify $\log(1-\beta_i)$ with its first-order Taylor approximation, i.e., $\log(1-\beta_i) \approx -\beta_i$, which thus leads to the approximated $\bar{a}_t \approx \exp\left(-\int_0^t \beta_i \mathrm{d}i\right)$. This way we compute the derivative of $\bar{a}_t$ as

$$\begin{aligned}\frac{\partial}{\partial t}\bar{a}_t &= \frac{\partial}{\partial t}\exp\left(-\int_0^t \beta_i \mathrm{d}i\right)\\ &= \exp\left(-\int_0^t \beta_i \mathrm{d}i\right)\frac{\partial}{\partial t}\left(-\int_0^t \beta_i \mathrm{d}i\right)\\ &= \exp\left(-\int_0^t \beta_i \mathrm{d}i\right)(-\beta_t)\\ &= -\beta_t\bar{\alpha}_t\end{aligned}$$

We can then compute the derivatives of the coefficients:

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{1}{1-\bar{\alpha}_t}\right) &= \frac{-1}{(1-\bar{a}_t)^2}\frac{\partial}{\partial t}(1-\bar{a}_t)\\ &= \frac{\beta_t\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}\end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\right) &= \frac{\frac{\partial}{\partial t}\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t) - \sqrt{\bar{\alpha}_t}\frac{\partial}{\partial t}(1-\bar{\alpha}_t)}{(1-\bar{\alpha}_t)^2}\\ &= \frac{\left(\frac{1}{2}\bar{\alpha}_t^{-1/2}\frac{\partial\bar{\alpha}_t}{\partial t}\right)(1-\bar{\alpha}_t) + \sqrt{\bar{\alpha}_t}\frac{\partial\bar{\alpha}_t}{\partial t}}{(1-\bar{\alpha}_t)^2}\\ &= \frac{\left(\frac{1}{2}\bar{\alpha}_t^{-1/2}(-\beta_t\bar{\alpha}_t)\right)(1-\bar{\alpha}_t) + \sqrt{\bar{\alpha}_t}(-\beta_t\bar{\alpha}_t)}{(1-\bar{\alpha}_t)^2}\\ &= \frac{-\frac{1}{2}\beta_t\bar{\alpha}_t^{1/2}(1-\bar{\alpha}_t) - \beta_t\bar{\alpha}_t^{3/2}}{(1-\bar{\alpha}_t)^2}\\ &= -\beta_t\bar{\alpha}_t^{1/2}\frac{\frac{1}{2}(1-\bar{\alpha}_t)+\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}\\ &= -\beta_t\sqrt{\bar{\alpha}_t}\frac{1+\bar{\alpha}_t}{2(1-\bar{\alpha}_t)^2}\end{aligned}$$

Using the computed derivatives:

$$\frac{\partial s_t(\mathbf{x})}{\partial t} = \frac{\beta_t\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}\mathbf{x} - \beta_t\sqrt{\bar{\alpha}_t}\frac{1+\bar{\alpha}_t}{2(1-\bar{\alpha}_t)^2}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0].$$

Substituting these back, we get:

$$\begin{aligned}\frac{\partial g}{\partial t} &= 2\left\langle -\frac{1}{1-\bar{\alpha}_t}\mathbf{x} + \frac{\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0], \frac{\beta_t\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}\mathbf{x} - \beta_t\sqrt{\bar{\alpha}_t}\frac{1+\bar{\alpha}_t}{2(1-\bar{\alpha}_t)^2}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0]\right\rangle\\ &= \frac{2\beta_t\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}\left\langle s_t(\mathbf{x}), \mathbf{x} - \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0]\right\rangle.\end{aligned}$$

Next, under Cauchy-Schwarz inequality: $|\langle \mathbf{a}, \mathbf{b} \rangle| \leq \|\mathbf{a}\| \|\mathbf{b}\|$, we have

$$|\frac{\partial g}{\partial t}| \leq \frac{2\beta_t \bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \|s_t(\mathbf{x})\| \cdot \left\| \mathbf{x} - \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}} \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0] \right\|$$

Then, by the triangle inequality for vectors $\mathbf{a}$ and $\mathbf{b}$:

$$\|\mathbf{a} - \mathbf{b}\| = \|\mathbf{a} + (-\mathbf{b})\| \leq \|\mathbf{a}\| + \|-\mathbf{b}\| = \|\mathbf{a}\| + \|\mathbf{b}\|,$$

and the assumption $\|\mathbf{x}\| \leq M$, we know that $\left\|\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0]\right\| \leq M$ almost surely. Thus,

$$\begin{aligned}
|\frac{\partial g}{\partial t}| &\leq \frac{2\beta_t \bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \|s_t(\mathbf{x})\| \cdot \left\| \mathbf{x} - \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}} \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0] \right\| \\
&\leq \frac{2\beta_t \bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \|s_t(\mathbf{x})\| \cdot \left(\|\mathbf{x}\| + \left\| \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}} \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x})}[\mathbf{x}_0] \right\| \right) \\
&\leq \frac{2\beta_t \bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \|s_t(\mathbf{x})\| \left(M + \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}} M \right).
\end{aligned}$$

Let $C_1 = \sup_{t\geq 0} \frac{2\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}$ and $C_2 = \sup_{t\geq 0} \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}}$. As $0 < \bar{\alpha}_t < 1$, it is easy to check the maximum value of $\frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}}$, i.e., C_2 is achieved when $\bar{\alpha}_t$ gets close to 1. This also aligns with the condition where $\sup \frac{2\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2}$ is reached. Thus, $\left(1 + \frac{1+\bar{\alpha}_t}{2\sqrt{\bar{\alpha}_t}}\right) M \leq (1 + C_2)M = C_3 M$ for some constants $C_3 < 2$, since $C_2 \to 1$ when $\bar{\alpha}_t \to 1$. As a result, we have a constant $C > 0$ leading to the upper bound as

$$|\frac{\partial g}{\partial t}| < C\beta_t \|s_t(\mathbf{x})\|, \quad \text{with } C = C_1 \cdot (C_3 M).$$

Mean Value Theorem states that: for any t_1, t_2, there exists a ξ between t_1 and t_2 such that: $|g(t_2) - g(t_1)| = |\frac{\partial g}{\partial t}(\xi)||t_2 - t_1| \leq C\beta_\xi \|s_\xi(\mathbf{x})\||t_2 - t_1|$.

Applying this to $g(t)$, for any t_1, t_2, there exists a ξ between t_1 and t_2, such that:

$$\begin{aligned}
|g(t_2) - g(t_1)| &= |\frac{\partial g}{\partial t}(\xi)||t_2 - t_1| \\
&\leq C\beta_\xi \|s_\xi(\mathbf{x})\| |t_2 - t_1| \\
&\leq CK \|s_\xi(\mathbf{x})\| |t_2 - t_1| \qquad \text{(by assumption } \beta_t \leq K)
\end{aligned}$$

Then, we can express

$$\begin{aligned}
|\, \|s_{t_2}(\mathbf{x})\| - \|s_{t_1}(\mathbf{x})\| \,| &= \frac{|\, \|s_{t_2}(\mathbf{x})\|^2 - \|s_{t_1}(\mathbf{x})\|^2 \,|}{(\|s_{t_2}(\mathbf{x})\| + \|s_{t_1}(\mathbf{x})\|)} \\
&\geq \frac{CK \|s_\xi(\mathbf{x})\| |t_2 - t_1|}{\|s_{t_2}(\mathbf{x})\| + \|s_{t_1}(\mathbf{x})\|}
\end{aligned}$$

From Lemma K.4, we have: there exists $C' > 0$ and $T_0 > 0$ such that for all $t \geq T_0$: $||s_t(\mathbf{x})|| > C'||\mathbf{x}||$. Applying this to t_1, t_2, and ξ with $\delta > 0$, we find that there exists a C' such that:

$$\begin{aligned}
\|s_{t_1}(\mathbf{x})\| &> C' \|\mathbf{x}\| \\
\|s_{t_2}(\mathbf{x})\| &> C' \|\mathbf{x}\| \\
\|s_\xi(\mathbf{x})\| &> C' \|\mathbf{x}\|
\end{aligned}$$

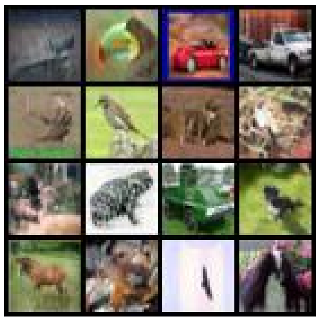
Substituting these back, we have:

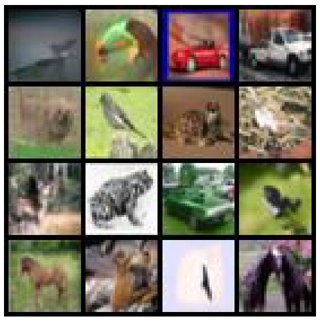
$$\begin{aligned}
|\, \|s_{t_2}(\mathbf{x})\| - \|s_{t_1}(\mathbf{x})\| \,| &\geq \frac{CK_2 C' \|\mathbf{x}\| |t_2 - t_1|}{2C' \|\mathbf{x}\|} \\
&= \frac{CK_2}{2} |t_2 - t_1|.
\end{aligned}$$

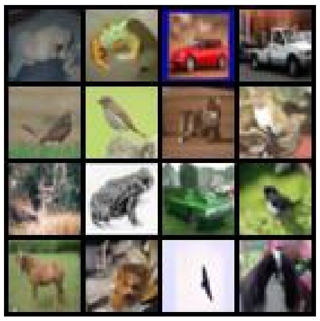
For any $\epsilon > 0$, let $\Delta = \frac{2\epsilon}{CK_2}$. Then for $|t_2 - t_1| \geq \Delta$, we have: $|\, \|s_{t_2}(\mathbf{x})\| - \|s_{t_1}(\mathbf{x})\| \,| > \epsilon$. This completes the proof. □

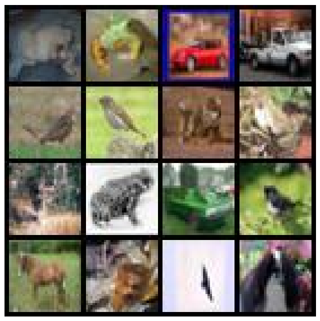
Remark K.6 (**Relationship between score norms and** t^*). According to Proposition K.5, for two distinct noise levels t_1 and t_2, we would obtain different score norms for the same sample $\mathbf{x} \in \mathcal{X}$. Hence, we consider that different t^* would change the score norm of the same sample $\mathbf{x}$. In addition, we also imply that the score norms of the same sample $\mathbf{x}$ differ if and only if the noise levels t^* input to the score network s_θ are different. This is because the score network only takes two input arguments $s_\theta(\mathbf{x}, t^*)$, which is sample $\mathbf{x}$ and noise level t^*. For a natural example $\mathbf{x}$, we treat it as a sample with no adversarial perturbation, such that $\epsilon = 0$. From Figure 2, cleaner samples typically exhibit smaller score norms, which means the data points are close to the high data density region (Song & Ermon, 2019). Then, we should not diffuse the natural data further with a large t^* to retain the original semantic information. Samples subjected to stronger perturbations exhibit larger score norms, which means the adversarial examples are far from the high data density area. We then inject certain amounts of Gaussian noise to the sample to cover its adversarial perturbation during forward diffusion. We identify the score norm as a metric to assess the sample's proximity to the original data distribution $p(\mathbf{x})$.

L. Visualizations for Purification Results


(a) Adversarial Examples


(b) DiffPure

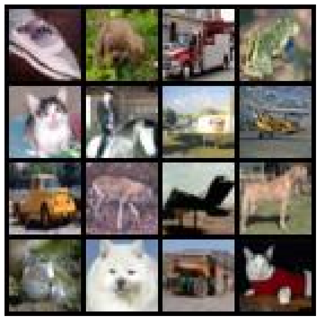

(c) DiffPure + *SSNI-L*

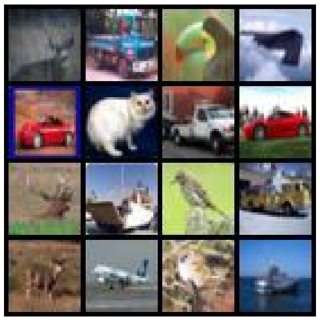

(d) DiffPure + *SSNI-N*

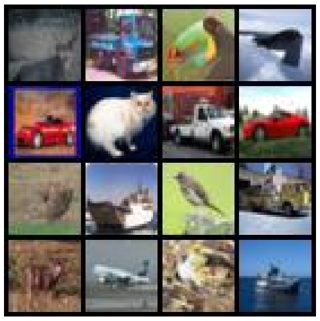
(a) Adversarial Examples

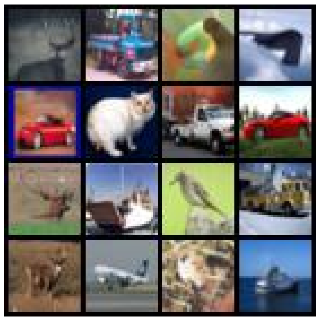
(b) DiffPure

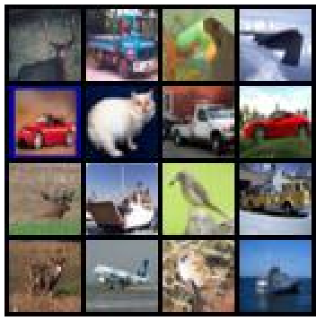
(c) DiffPure + *SSNI-L*


(d) DiffPure + *SSNI-N*


(a) Natural Examples


(b) DiffPure


(c) DiffPure + *SSNI-L*


(d) DiffPure + *SSNI-N*