DiffSmooth: Certifiably Robust Learning via Diffusion Models and Local Smoothing

Jiawei Zhang
UIUC

Zhongzhu Chen
University of Michigan, Ann Arbor

Huan Zhang
Carnegie Mellon University

Chaowei Xiao
Arizona State University

Bo Li
UIUC

Abstract

Diffusion models have been leveraged to perform adversarial purification and thus provide both empirical and certified robustness for a *standard* model. On the other hand, different robustly trained *smoothed* models have been studied to improve the certified robustness. Thus, it raises a natural question: *Can diffusion model be used to achieve improved certified robustness on those robustly trained smoothed models?* In this work, we first theoretically show that recovered instances by diffusion models are in the bounded neighborhood of the original instance with high probability; and the "one-shot" denoising diffusion probabilistic models (DDPM) can approximate the mean of the generated distribution of a continuous-time diffusion model, which approximates the original instance under mild conditions. Inspired by our analysis, we propose a certifiably robust pipeline DiffSmooth, which first performs adversarial purification via diffusion models and then maps the purified instances to a common region via a simple yet effective *local smoothing* strategy. We conduct extensive experiments on different datasets and show that DiffSmooth achieves SOTA-certified robustness compared with eight baselines. For instance, DiffSmooth improves the SOTA-certified accuracy from 36.0% to 53.0% under ℓ_2 radius 1.5 on ImageNet.

1 Introduction

Despite the fact that the deep neural networks (DNNs) have achieved unprecedented success in different applications, they are still vulnerable to imperceptible adversarial noise, which will mislead the model to predict the perturbed input as an arbitrary adversarial target [5, 53]. Such adversarial perturbations have posed a threat to the real-world application of DNNs on safety-critical scenarios such as fraud detection [19, 37] and automatic driving [9, 18].

Different *empirical* defense approaches have been proposed to prevent such adversarial attacks. For instance, adversarial training [33, 40, 56, 60], which incorporates adversarial instance into the training process, has become the de facto standard method for training robust models. However, empirical defenses may become broken under strong adaptive attacks [2, 11, 14].

Later, *certified* defenses are proposed to provide a lower bound of accuracy for DNNs under constrained perturbations. For instance, the technique of bound propagation [20, 55, 62], which computes the upper and lower bounds of the features layer by layer, is commonly used to provide deterministic certification for small models and low-resolution instances. In addition, randomized smoothing [13, 26, 28, 43, 59] has been proposed as a more scalable technique for providing probabilistic certification on large-scale datasets, such as ImageNet [15], by taking a majority vote over the predictions of Gaussian-smoothed inputs. Such technique typically requires to train a *standard* model with Gaussian augmentation as a robustly trained *smoothed* model.

At the same time, diffusion models [23, 48] recently have demonstrated powerful abilities of generative modeling in different tasks and applications, such as image generation [23, 50, 51], shape generation [8], and image inpainting [52]. In general, the diffusion model contains two processes: (1) the forward diffusion process, which perturbs the input data point with Gaussian noise gradually to populate low data density regions, and (2) the reverse diffusion process, which starts with random Gaussian noise and generates a high-quality instance through a Markov Chain iteratively. The denoising nature of the diffusion models has enabled a line of interesting works to purify adversarial perturbations and therefore improve the robustness of DNNs. For instance, Nie et al. [35] proposed to purify adversarial perturbations with diffusion models.

Given the promising diffusion-based adversarial purification, Lee [29] and Carlini et al. [10] propose a method Diffusion Denoised Smoothing (DDS), to leverage the denoising mechanism of diffusion models to remove the Gaussian noise added during the randomized smoothing process via a *one-shot* reverse diffusion step. Thus, DDS is able to provide certified robustness for any off-the-shelf *standard* model. Nevertheless, this approach can only provide state-of-the-art

certified accuracy under small perturbation radii, and the robustness will decrease quickly for large perturbation radii (e.g., $r \geq 1.0$ in CIFAR-10 and $r \geq 2.0$ in ImageNet). Xiao et al. [57] later attempt to use the *multi-shot* reverse process and propose DensePure to repeat this process several times with different random seeds and take the majority vote over these purified images as the final prediction to further boost the certified robustness. However, as shown in Section 5.3, even with a simple one-shot reverse diffusion step, the certification time for one image in ImageNet with sample size $N = 10,000$ requires 553s for purifying the Gaussian augmented images and only 11s for prediction with a DNN model (i.e., ResNet-50 [22]). Therefore, although DensePure performs much better than DDS, its actual computation cost is much higher than DDS owing to the repetitive multi-shot diffusion step.

Based on existing observations, we aim to answer: *(1) Can we improve the certified robustness of models under large perturbation radii leveraging the diffusion-based purification? (2) Unlike DensePure, can we further boost the certified robustness by executing more prediction steps instead of the reverse diffusion step, which is far more expensive?*

In this paper, we show that it is possible to achieve higher certified robustness with higher benign accuracy leveraging the robustly trained *smoothed* model based on our proposed local smoothing technique (formally introduced in Section 4).

In particular, we first provide theoretical analysis to show that the recovered instances from (adversarial) inputs will be in the bounded neighborhood of the corresponding original instance with high probability. We also prove that the "one-shot" denoising of Denoising Diffusion Probabilistic Models (DDPM) [23] can approximate the mean of the generated posterior distribution by continuous-time diffusion models, which is, in turn, an approximation of the original instance under mild conditions.

Inspired by our theoretical analysis of the properties of reversed instances and the relationship between the smoothed models and the robust regions of reversed instances, we propose a general certifiably robust adversarial purification pipeline DiffSmooth. In particular, as shown in Figure 1, DiffSmooth contains three steps: (1) add a set of random Gaussian noise to the input x for certification purposes; (2) denoise each Gaussian perturbed input with the reverse process of a pre-trained diffusion model to generate a purified sample $\hat{x}$; (3) for each $\hat{x}$, add another set of noise to generate locally smoothed instances and make predictions based on their mean confidence; (4) repeat step (2)-(3) for all Gaussian perturbed inputs and take majority vote as the final *smoothed* prediction. An adversarial instance can be recovered to the neighborhood of the original one with high probability under mild conditions based on our analysis. Thus, adding a set of local smoothing noises to the recovered instance will help map it to a smoothed and robust region.

Finally, we conducted comprehensive evaluations to compare the certified robustness of DiffSmooth and seven SOTA baselines. We specifically evaluate and control the computation cost to compare with baselines in Section 5.3. We make the following technical contributions:

- We theoretically analyze the properties of purified adversarial instances of diffusion models. We prove that they are within the bounded neighborhood of the original clean instance with high probability, and their distances to the original instance depend on the adversarial perturbation magnitude and data density. We also prove that the "one-shot" denoising of DDPM can approximate the mean of the generated posterior distribution by continuous-time diffusion models, which is an approximation of the original instance under mild conditions.
- We show that naively combining diffusion models with smoothed models cannot effectively improve their certifiable robustness. Inspired by our theoretical analysis, we propose an effective and certifiably robust pipeline for smoothed classifiers, DiffSmooth, via *local smoothing*.
- We conduct extensive experimental evaluations on different datasets and show that DiffSmooth achieves significantly higher certified robustness compared with SOTA baselines. For instance, with more inference steps, the certified accuracy is improved from 36.0% to 53.0% under ℓ_2 radius 1.5 and 42.2% to 48.2 under ℓ_2 radius 1.0 on ImageNet; For CIFAR-10, the certified accuracy is improved from 42.8% to 59.2% under ℓ_2 radius 0.50, and from 39.4% to 43.6% under ℓ_2 radius 1.00.
- We also perform a set of ablation studies to show that local smoothing is unique to the diffusion purification process and evaluate the impacts of different parameters, such as the variance of local smoothing noise. We show that DiffSmooth outperforms DDS with the same computation cost.

2 Related work

Certified robustness. Deep neural networks (DNNs) are found vulnerable to adversarial examples [5,53]. To overcome such vulnerabilities, multiple empirical defenses have been proposed [33, 36, 46, 60], most of which have been attacked again by strong adaptive attackers [2, 11, 14]. Thus, certified robustness for DNNs is studied to provide a lower bound of model accuracy under constrained perturbations. While at the same time, the complete certification [38,39,45,55,63], which guarantees to find the adversarial perturbation if it exists is constrained on a small dataset and extremely; and some incomplete certification [20, 38, 39, 45, 62, 63] which may miss some certifiable instances are only applicable for specific model architectures and still can not scale to large datasets like ImageNet [15]. Later, Lecuyer et al. [28] prove robustness guarantee for smoothing with Gaussian and Laplace noise from a differential privacy perspective and first provide non-trivial certification result on ImageNet. The guarantee is later tightened by [13] as the robustness guarantee for randomized

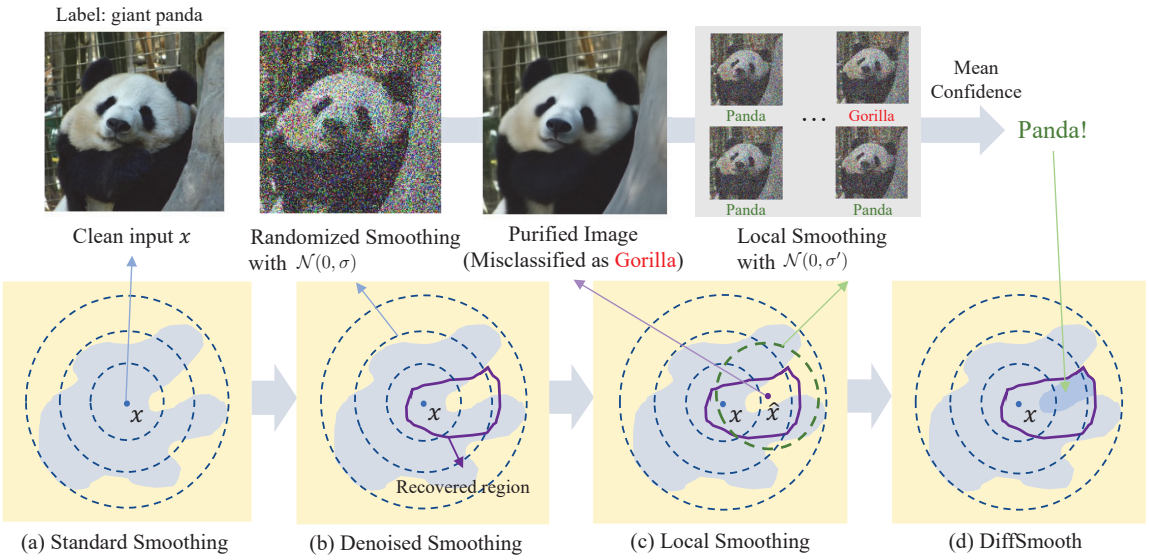

Figure 1: Overview of our pipeline DiffSmooth. The second row shows the decision regions of the base classifier for different smoothing processes at an input x. (a) follows the standard randomized smoothing [13]; (b) represents the denoised smoothing [10, 44], which first attempts to purify the noisy images with a denoiser, and then sends the purified image $\hat{x}$, which is inside the purple recovered region, to a *standard* classifier for prediction. DiffSmooth instead first performs local smoothing on the purified image $\hat{x}$ with a smaller noise level as shown by the green dash line in (c), and then takes a majority vote over the local smoothed predictions as shown in (d).

smoothing, which is *probabilistic*. Based on this, the certification performance is further enhanced with the incorporation of adversarial training [43], consistency regularization [26], or ensemble model [24, 32, 58].

Diffusion models for adversarial purification. Diffusion models [23,34,48,49] have shown impressive performance on generative modeling tasks and have been applied to various tasks such as image inpainting, super-resolution, and even text-to-image synthesis [3, 16, 41, 42]. The connection between the diffusion model and adversarial purification is first explored by [35] to remove adversarial perturbation, and [54] further boosts the adversarial robustness with a guided reverse process. Besides, Carlini et al. [10] leveraged diffusion models to remove added smoothing Gaussian noise during randomized smoothing to provide certified robustness for a standard model, and Xiao et al. [57] propose to repeat the reverse diffusion steps multiple times and take the majority vote of the predictions on these reversed instances as the final prediction. One main limitation of existing work is that the certified robustness under large radii is low, given that the certification is calculated on standard models, which are less robust. In this work, we propose a novel local smoothing strategy to balance the tradeoff between low certified robustness on standard models and high computation cost on performing multi-reverse diffusion processes for a smoothed model. Our local smoothing strategy only requires multiple prediction steps, which are much cheaper than the reverse diffusion steps, while it helps to "smooth" the final predictions to improve robustness. In addition, we theoretically demonstrate that the reversed instances will lie in the vicinity of the original clean instance with a high probability. Finally, we show that our approach achieves state-of-the-art certified robustness compared with different baselines.

3 Background

Notations. We mainly consider the classification problem in this paper. Let Δ^k be a k-dimensional probability simplex, we define the soft classifier $F : \mathbb{R}^d \to \Delta^{|\mathcal{Y}|}$ as the function which maps the input to a *confidence vector*. The associated hard classifier f is defined as $f(x) := \arg\max_{c\in\mathcal{Y}} F(x)$, which maps $\mathbb{R}^d$ to classes $\mathcal{Y}$.

Robustness Certification. Given a radius $r \in \mathbb{R}_+$, the *robustness certificate* provides a lower bound of the classification accuracy given perturbations within r, regardless of the concrete attack algorithms [30,31]. Formally, a certification algorithm takes a clean instance x and the base classifier f as inputs and outputs a robust radius r, such that $f(x) = f(x')$ when the distance between x and x' satisfies $d(x,x') < r$, where $d(\cdot,\cdot)$ denotes a distance metric, e.g., the metric induced by ℓ_p norm. Generally, the implication of certified robustness is that it provides a lower bound of model robustness, given any perturbation whose magnitude is bounded by a ℓ_p norm. In other words, the empirical robustness under the same perturbation radius is always higher than the certified robustness, which is empirically tested in [21].

Randomized Smoothing. In this paper, we mainly adopt *randomized smoothing* [13] for achieving the certification of robustness. Specifically, randomized smoothing leverages the

Algorithm 1 COMPUTETIMESTEP(σ) [10].

Input: Magnitude of the smoothing noise σ.
Output: Start time step for the reverse process and the corresponding $\bar{\alpha}_t$.

$t \leftarrow 0$
while $\frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} < \sigma^2$ **do**
  $t \leftarrow t+1$
end while
return $t, \bar{\alpha}_t$

Algorithm 2 DENOISE(x_t, t) [10]. # One-shot denoising

Input: Intermediate sample x_t and its associated timestep t.
Output: Predicted original clean image $\hat{x}_0$.

$\varepsilon \leftarrow \varepsilon_\theta(x_t, t)$
$\hat{x}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon)$
return $\hat{x}_0$

smoothed base classifier f for prediction, which is defined by $g(x) := \arg\max_{c\in\mathcal{Y}} \mathbb{P}(f(x+\delta)=c)$ with $\delta \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. Assume $\underline{p_A}$ is the lower bound of prediction probability of the top class c_A, and $\overline{p_B}$ is the upper bound of prediction probability for the "runner-up" class, then the smoothed classifier g is robust around x within the radius:

$$R = \frac{\sigma}{2}\left(\Phi^{-1}\left(\underline{p_A}\right) - \Phi^{-1}\left(\overline{p_B}\right)\right), \tag{1}$$

where Φ^{-1} is the inverse of the standard Gaussian CDF. Such certification requires the classifier to be robust under Gaussian noise, so usually, the base classifier will be trained with Gaussian augmentation [13], and such robustly trained models are referred as *smoothed models*. Other robust training algorithms proposed later to further provide robustly trained smoothed models such as *SmoothAdv* [43] and *Consistency* [26].

Denoised Smoothing. To apply the off-the-shelf standard model with randomized smoothing, Salman et al. [44] propose to prepend a custom-trained denoiser $\mathcal{D}_\theta : \mathbb{R}^d \to \mathbb{R}^d$, which is trained to remove the Gaussian noise appeared in the instances, to the standard classifier f_{clf}, and treat $f_{\text{clf}} \circ \mathcal{D}_\theta$ as the new base classifier f. In this way, the noisy input $x+\delta$ will first be purified by $\mathcal{D}_\theta$, and then the purified sample will be directly predicted with the pretrained standard classifier. As a result, it is expected that the prediction accuracy of $f := f_{\text{clf}} \circ \mathcal{D}_\theta$ on the noisy instances with Gaussian perturbation is close to the accuracy of f_{clf} given a clean instances when the denoiser $\mathcal{D}_\theta$ performs well.

Continuous-time Diffusion Models. A continuous-time diffusion model contains two components: (i) a diffusion process that adds random noises to the data gradually and finally researches a noise distribution, e.g., Gaussian distribution, and (2) a reverse process that removes the added noise to recover the original data distribution. The diffusion process can be defined by the stochastic differential equation:

$$d\mathbf{x} = h(\mathbf{x}, t)dt + g(t)d\mathbf{w} \tag{SDE}$$

where $\mathbf{x}(0) \sim p$ (the original data distribution), $t \in [0, T]$, $h(\mathbf{x}, t)$ is the drift coefficient and $g(t)$ is the diffusion coefficient, and $\mathbf{w}(t)$ is the standard Wiener process [1]. Here we took the convention used by VP-SDE in [52] where $h(\mathbf{x}; t) := -\frac{1}{2}\gamma(t)\mathbf{x}$ and $g(t) := \sqrt{\gamma(t)}$ where $\gamma(t)$ is positive and continuous over $[0, T]$ such that

$$\mathbf{x}(t) = \sqrt{\overline{\alpha}_t}\mathbf{x}(0) + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\varepsilon}$$

where $\overline{\alpha}_t = e^{-\int_0^t \gamma(s)ds}$ and $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The reverse process can be defined by the stochastic differential equation:

$$d\hat{\mathbf{x}} = [h(\hat{\mathbf{x}}, t) - g(t)^2 \nabla_{\hat{\mathbf{x}}} \log p_t(\hat{\mathbf{x}})]dt + g(t)d\overline{\mathbf{w}} \tag{reverse-SDE}$$

where dt denotes the infinitesimal reverse time step, and $\overline{\mathbf{w}}(t)$ is the reverse-time standard Wiener process. We use $\{\mathbf{x}(t)\}_{t\in[0,T]}$ and $\{\hat{\mathbf{x}}(t)\}_{t\in[0,T]}$ to denote the diffusion process and reverse process respectively. [1] shows that if $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x}\sim p}[||\mathbf{x}||_2^2] < \infty$, $\{\mathbf{x}(t)\}_{t\in[0,T]}$ and $\{\hat{\mathbf{x}}(t)\}_{t\in[0,T]}$ have the same distribution.

Denoising Diffusion Probabilistic Models (DDPM) [23, 34] which construct the discrete forward diffusion process, has been shown effective to generate high-quality data through learning the reverse of the forward diffusion process. In particular, given the number of the forward steps T and image x_0 sampled from the original data distribution, the forward diffusion acts to gradually adds a small amount of Gaussian noise following a variance schedule $\{\beta_t\}_{t=1}^T$, such that $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right)$. Generally, β_t is designed to increase with the time step t and takes a value between 0 and 1; thus, the forward process will finally transform x_0 into an isotropic Gaussian noise. By leveraging the reparameterization trick, a property of this forward process is that

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1-\bar{\alpha}_t} \cdot \boldsymbol{\varepsilon}, \tag{2}$$

where $\alpha_t := 1-\beta_t, \bar{\alpha}_t := \prod_{i=1}^s \alpha_s$ and $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, which indicates that we can sample the intermediate noisy x_t at any timestep t. Then, the diffusion model is trained to reverse the diffusion process and learn the posteriors with the Markov chain $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, which is defined as $\mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$. Here, $\mu_\theta(x_t, t) := \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t, t)\right)$ in which ε_θ is trained to predict the random noise ε for x_t, and $\Sigma_\theta(x_t, t)$ is defined as $\sigma_t^2\mathbf{I}$ as in [23] while it can also be learned following [34]. As a result, starting from $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, DDPM will generate an instance through the reverse sampling.

An interesting observation from [10, 29] is that this reverse process can be perfectly used to denoise Gaussian-corrupted images; hence it can be applied to the denoised smoothing [44] and acted as $\mathcal{D}_\theta$. Formally, given the corrupted instance $x_{rs} = x + \delta$ where x is the clean instance and $\delta \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, we want to relate it to the noisy image x_t sampled from the forward diffusion process with a specific timestep t, then we can hopefully remove the Gaussian noise δ to get the original instance x with the reverse process which

starts at x_t. To achieve this, notice that the intermediate sample from DDPM is shown in the form of $x_t = \sqrt{\bar{\alpha}_t} \cdot x + \sqrt{1-\bar{\alpha}_t} \cdot \varepsilon$ where $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; thus, if we scale x_{rs} with $\sqrt{\bar{\alpha}_t}$ and equate the variance between the scaled x_{rs} and x_t, we will obtain $\sigma^2 = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}$. The solution of the timestep t^* for this equation is straightforward: notice that the $\hat{\alpha}_t$ decreases monotonically with t and $\hat{\alpha}_0 = 1$, and thus the value of $\frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}$ will increase monotonically with t. Finally, the equation can be simply solved via 1D root-finding as shown in Algorithm 1. As a result, we can simply start at $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$ and perform the left reverse diffusion steps to recover the original x. In other words, we will recursively sample the previous intermediate image $\hat{x}_{t-1}$ based on the pre-defined Markov chain $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ until we get $\hat{x}_0$, which is exactly the purified image we want. However, the information of the x contained in x_{rs} will be destroyed in each iterative reverse diffusion process owing to the addition of the Gaussian noise. In addition, Carlini et al. [10] propose to adopt one-shot denoising instead of running the full reverse diffusion process, where they first predict the likely ε in x_{t^*} with ε_θ, and then directly plug it into Equation (2) to obtain $\hat{x}_0$, for which the corresponding pseudo-code is shown in Algorithm 2.

4 DiffSmooth: Diffusion Based Adversarial Purification with Local Smoothing

In this section, we will first provide the motivation of our method, and then we analyze the diffusion-based adversarial purification in Section 4.2, which leverages reverse-SDE to generate reversed instances, and we prove that such reversed instances will stay in the bounded neighborhood of the original clean instance with high probability. In addition to the reversed instance, to further understand the reversed posterior distribution, we will show that the "one-shot" denoising of DDPM (Algorithm 2) will output the mean (approximation) of the conditional distribution generated by reverse-SDE, based on the adversarial sample input and a given time step. We show that such a mean instance will have the ground-truth label of the corresponding original instance if the points with the ground-truth label have a high enough density in the original distribution.

Inspired by our theoretical analysis, we propose DiffSmooth in Section 4, consisting of diffusion-based adversarial purification and a simple yet effective local smoothing strategy.

4.1 Motivation

As shown in Equation (1), the certified robustness largely depends on the *consistency* of predictions of N sampled points (larger p_A will result in larger certified radius R). Intuitively, the "one-shot" reverse diffusion step in prior work DDS [10] helps to increase the prediction consistency by generating denoised samples near the original sample, for which we will provide the first formal theoretical analysis in Section 4.2. On the other hand, Xiao et al. [57] propose to repeat the reverse diffusion step multiple times and then take the majority vote of the predictions over these purified images as the final prediction. This leads to an approximation of consistent predictions for samples in the high-density region, thereby improving certified robustness.

Nevertheless, as demonstrated in Section 5.3, the reverse diffusion step is actually the primary bottleneck for computation cost during certification. Thus, repeating the multi-shot reverse diffusion steps, as in [57], will significantly increase the computation cost and render it impractical. Thus, we aim to propose a simple yet effective *local smoothing* strategy by taking the majority vote of predictions on Gaussian smoothed samples given a smoothed model. This way, the computation cost for the reverse diffusion step is the same as DDS, while only extra computation cost for predictions is required, which is much lower. In the meantime, the consistency among the predictions will be improved since the distributions from which the Gaussian smoothed samples are drawn are sampled close. In addition, given that a *smoothed* model is more stable and therefore more robust than a *standard* model at the cost of sacrificing benign accuracy, the local smoothing also helps to improve the benign accuracy of smoothed models.

Overall, DiffSmooth first performs the one-shot diffusion-based adversarial purification, and then multiple Gaussian noises are sampled to locally smooth the prediction for each purified sample. Such local smoothing will improve certified robustness for smoothed models and help to maintain or even improve benign accuracy since the smoothing Gaussian noises of the model and the locally smoothed samples are from similar distributions.

4.2 Properties of Diffusion-Based Adversarial Purification

There are several works applying diffusion models to (adversarial) inputs by performing the diffusion and reverse processes on them, aiming to remove potential adversarial perturbations [11, 35]. On the other hand, it is also possible to directly perform the reverse process to given inputs, and here we will analyze the properties and advantages of such reversed samples. In particular, we theoretically prove that the reverse process of the diffusion model generates reversed samples in the bounded neighborhood of the original clean samples with high probability. We will analyze directly based on the stochastic equations SDE and reverse-SDE, as other diffusion models such as DDPM [23] and score-based diffusion models [52] are approximations of the stochastic differential equations, and our results can also be extended to other models. Our main theorems are as follows.

Theorem 1. *Given a data distribution $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x}\sim p}[||\mathbf{x}||_2^2] < \infty$. Let p_t be the distribution of $\mathbf{x}(t)$ generated by SDE and suppose $\nabla_x \log p_t(x) \leq \frac{1}{2}C$ for some constant C and $\forall t \in [0,T]$. Let $\gamma(t)$ be the coefficient defined in SDE and $\bar{\alpha}_t = e^{-\int_0^t \gamma(s)ds}$. Then given an adversarial sample $x_{rs} = x_0 + \delta$ with original instance x_0 and perturbation δ, solving reverse-*

SDE starting at time t^* and point $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$ until time 0 will generate a reversed random variable $\hat{\mathbf{x}}_0$ such that with a probability of at least $1-\eta$, we have

$$||\hat{\mathbf{x}}_0 - x_0|| \leq ||x_{rs} - x_0|| + \sqrt{e^{2\tau(t^*)} - 1} C_\eta + \tau(t^*) C \quad (3)$$

where $\tau(t) := \int_0^t \frac{1}{2}\gamma(s)ds$, $C_\eta := \sqrt{d + 2\sqrt{d \log \frac{1}{\eta}} + 2\log \frac{1}{\eta}}$, and d is the dimension of x_0.

Proof. (sketch) Based on [35, Theorem 3.2], we can obtain that

$$||\hat{\mathbf{x}}_0 - x_0|| \leq ||\sqrt{\left(e^{\tau(t^*)} - 1\right)}\boldsymbol{\varepsilon} + x_{rs} - x_0|| + \tau(t^*) C$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Since $||\boldsymbol{\varepsilon}^2|| \sim \chi^2(n)$, by the concentration inequality [6], we have

$$\Pr\left(||\boldsymbol{\varepsilon}|| \geq \sqrt{d + 2\sqrt{d \log \frac{1}{\eta}} + 2\log \frac{1}{\eta}} \right) \leq \eta.$$

Thus, with probability at least $1-\eta$, we have

$$||\hat{\mathbf{x}}_0 - x_0|| \leq ||x_{rs} - x_0|| + \sqrt{e^{2\tau(t^*)} - 1} C_\eta + \tau(t^*) C.$$

□

For Theorem 1 and 2, please check Appendix A for complete proofs.

Remark. Theorem 1 implies that as long as x_{rs} is not far away from the corresponding original instance x_0, $\bar{\alpha}_{t^*}$ is not very close to zero, and $\nabla_x \log p_t(x)$ is upper bounded by a reasonable value, the right-hand side of (3) will be small. This means that the reverse process of the diffusion model will generate a reversed sample in a small neighborhood of x_0 with high probability based on the scaled adversarial sample. Such examples are highly likely to have the same labels as x_0. One should note that since the diffusion model is a generative model with mode coverage on the whole dataset, it could be possible that $\hat{\mathbf{x}}_0$ lies far away from x_0. Theorem 1, developed based on a deep analysis of the stochastic differential equation reverse-SDE, ensures that this unwanted case rarely happens. Further, Theorem 1 provides a tighter upper bound than [35, Theorem 3.2] in that C_η is only half of the corresponding constant, which is due to the removal of the diffusion process (the reverse process is directly applied to a given input).

In practice, we cannot implement the continuous-time diffusion model directly, and DDPM [23] was proposed as one efficient approximation to the reverse process reverse-SDE. In particular, DDPM learns a neural network $\varepsilon_\theta(\mathbf{x}_t, t)$ to predict the likely noise added to $\mathbf{x}_0$ with the loss function:

$$\mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \left\| \boldsymbol{\varepsilon} - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}, t \right) \right\|^2 \right].$$

where $t \in [0, T]$, $\mathbf{x}_0 \sim p$, and $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. For fast sampling in DDPM, the "one-shot" denoising (algorithm 2) was frequently used [10, 23] where we have

$$\hat{x}_0 = \left(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_\theta(x_t, t) \right) / \sqrt{\bar{\alpha}_t}. \quad (4)$$

In the following theorem, we will show that given t and x_t, the distance of $\hat{x}_0$ in (4) to the mean of a conditional distribution generated by the reverse process reverse-SDE starting at time t and point x_t will be bounded by the loss at time t:

$$\ell_t(x_t) := \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\varepsilon} - \varepsilon_\theta(x_t, t)\|^2 \middle| \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon} = x_t \right]$$

where $\mathbf{x}_0 \sim p$ and $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

Theorem 2. *Given a data distribution $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x} \sim p}[||\mathbf{x}||_2^2] < \infty$, given a time t^* and point $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$, the one-shot denoising for DDPM (algorithm 2) will output an $\hat{x}_0$ such that*

$$\|\hat{x}_0 - \mathbb{E}[\hat{\mathbf{x}}_0 \mid \hat{\mathbf{x}}_{t^*} = x_{t^*}]\| \leq \frac{2\sigma_{t^*}^2 \alpha_{t^*} (1-\bar{\alpha}_{t^*})^{3/2}}{\beta_{t^*}^2 \sqrt{\bar{\alpha}_{t^*}}} \cdot \ell_{t^*}(x_{t^*}) \quad (5)$$

where $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t$ are random variables generated by reverse-SDE, $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2} \right)$ and $\sigma_t^2 = \frac{1-\alpha_t}{\alpha_t}$ is the variance of Gaussian noise added at time t in the diffusion process.

Proof. (sketch) Under the assumptions that $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x} \sim p}[||\mathbf{x}||_2^2] < \infty$, the diffusion process by SDE and the reverse process by reverse-SDE follow the same distribution ideally. Therefore, $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*}) = \mathbb{P}(\mathbf{x}_0 = x | \mathbf{x}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2} \right)$. Since $\sqrt{\bar{\alpha}_{t^*}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}$ implies $\sqrt{\bar{\alpha}_{t^*}}\hat{\mathbf{x}}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}$ where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we have

$$\begin{aligned} \ell_{t^*}(x_{t^*}) &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_{t^*}^2 \sqrt{\bar{\alpha}_{t^*}}}{2\sigma_{t^*}^2 \alpha_{t^*} (1-\bar{\alpha}_{t^*})^{3/2}} \|\hat{\mathbf{x}}_0 - \hat{x}_0\| \middle| \mathbf{x}_{t^*} = x_{t^*} \right] \\ &\geq \frac{\beta_{t^*}^2 \sqrt{\bar{\alpha}_{t^*}}}{2\sigma_{t^*}^2 \alpha_{t^*} (1-\bar{\alpha}_{t^*})^{3/2}} \cdot \|\hat{x}_0 - \mathbb{E}[\hat{\mathbf{x}}_0 \mid \hat{\mathbf{x}}_{t^*} = x_{t^*}]\|, \end{aligned}$$

where the last inequality is by Jensen's inequality [7]. □

Remark. The right-hand side of (5) is the multiplication of a constant depending on t^* and the loss $\ell_{t^*}(x_{t^*})$ at time t^*. This implies that if we have smaller (zero) loss $\ell_{t^*}(x_{t^*})$ at time t^*, $\hat{x}_0$ will approximate the mean of the conditional distribution $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*})$. In addition, this conditional distribution has a high density on points with high data density in the original distribution p and close to x_{rs}. Such points tend to have the same ground-truth label as the original instance and can be recognized well by the classifiers trained on the data manifold. That means, as long as the original clean instance lies in a high enough data density region in the original distribution, the mean of the generated conditional distribution will have a similar property (i.e., prediction label) as such a clean instance.

4.3 Certifying Smoothed Models with DiffSmooth

Based on our theoretical analysis above, it is clear that the reversed samples are in the bounded neighborhood of the corresponding clean instance. Thus, given a robustly trained

Algorithm 3 PURIFYCLASSIFIER$(x_{rs};\sigma',m)$.

Input: The input noisy sample x_{rs}, the magnitude of the local smoothing noise σ', the number of sampled local smoothing noise m.

Output: The prediction for x_{rs}.

$t^*, \bar{\alpha}_{t^*} \leftarrow$ COMPUTETIMESTEP(2σ)
$\hat{x} \leftarrow \left(\text{DENOISE}(\sqrt{\bar{\alpha}_{t^*}}(2x_{rs}-1), t^*)+1\right)/2$
$\hat{y} \leftarrow 0$
for $i = 1$ to m **do**
  $\delta_i' \sim \mathcal{N}\left(0, \sigma'^2\mathbf{I}\right)$
  $\hat{y} \leftarrow \hat{y} + \frac{1}{m} F(\hat{x} + \delta_i')$
end for
return $\arg\max_{c\in\mathcal{Y}} \hat{y}$

Algorithm 4 SAMPLEUNDERNOISE(f,x,n,σ) [13].

Input: Base classifier f, clean input image x, the number of smoothing noise n, smoothing noise magnitude σ.

Output: A vector of class counts.

`counts` $\leftarrow [0,0,...,0]$
for $i = 1$ to n **do**
  $x_{rs} \leftarrow x + \mathcal{N}\left(\mathbf{0}, \sigma^2\mathbf{I}\right)$
  $y \leftarrow f(x_{rs})$
  `counts`$[y] += 1$
end for
return `counts`

smoothed model, in order to further improve its certified robustness by improving its clean accuracy, we propose a simple yet effective *local smoothing* technique for the reversed samples based on diffusion models. In this section, we will describe in detail how DiffSmooth works.

First, for each given (adversarial) input x, we will add standard Gaussian smoothing noise to get a set of x_{rs} for certification purposes following [10]. We then denoise each noisy input x_{rs} with a diffusion model to get a purified sample $\hat{x}$. We only run the reverse diffusion step once with the diffusion model and directly output the optimal estimate $\hat{x}$ for prediction accuracy and efficiency purposes. In specific, this one-shot reverse diffusion process is implemented to output $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t,t))$ given the input x_t and timestep t as shown in Algorithm 2. Usually, the clean instance x is assumed to be in $[0,1]^d$ following prior certification literature; however, the diffusion model expects the input in $[-1,1]^d$, and outputs denoised instance in $[-1,1]^d$. Thus, we start at $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}}(2x_{rs}-1)$ to perform the one-shot reverse step where t^* is the solution to equation $(2\sigma)^2 = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}$.

Second, we perform the *local smoothing* for each purified instance $\hat{x}$; in other words, the local smoothed prediction is provided as $\arg\max_{c\in\mathcal{Y}} \sum_{i=1}^m F(\text{DENOISE}(x_{t^*}) + \delta_i')/m$ where $\delta_i' \sim \mathcal{N}\left(x, \sigma'^2\mathbf{I}\right)$ with $\sigma' \le \sigma$ and m is the number of sampled local smoothing noise. As shown in Figure 1, the local smoothing noise magnitude σ' should be smaller than the smoothing noise level σ; in the meantime, it is also

Algorithm 5 Certification Process for DiffSmooth.

Input: The magnitude of the smoothing noise σ, the magnitude of the local smoothing noise σ', the number of sampled local smoothing noise m, the number of the smoothing noise for selection n_0, the number of the smoothing noise for estimation n, the certification confidence $(1-\alpha)$.

Output: Certified prediction and its robust radius.

$f \leftarrow$ PURIFYCLASSIFIER$(\cdot;\sigma',m)$.
`counts0` $\leftarrow$ SAMPLEUNDERNOISE(f,x,n_0,σ)
$\hat{c}_A \leftarrow$ top index in `counts0`
`counts` $\leftarrow$ SAMPLEUNDERNOISE(f,x,n,σ)
$\underline{p_A} \leftarrow$ LOWERCONFBOUND(`counts`$[\hat{c}_A]$, n, $1-\alpha$)
if $\underline{p_A} > \frac{1}{2}$ **then**
  return $\hat{c}_A$ and radius $\sigma\Phi^{-1}(\underline{p_A})$
else
  return ABSTAIN
end if

related to the robustness of smoothed models to different random noises. For instance, we find that to certify the simple Gaussian augmented model [13], we need to set σ' to be close to the model smoothing level σ. Nevertheless, to certify the advance smoothed model trained with SmoothAdv [43], setting σ' to be around $1/2$ of the σ achieves the best certification. Thus, the local smoothing noise level will also reflect the inherent robustness/stability of the *smoothed* models.

Finally, we will take the majority vote based on these locally smoothed predictions and provide robustness certification for given *smoothed* models following standard randomized smoothing [13].

During the experiment, the first two steps can be wrapped as a single base classifier $f(\cdot) = \text{PURIFYCLASSIFIER}(\cdot;\sigma',m)$ which is shown in Algorithm 3. The whole certification process for DiffSmooth is provided in Algorithm 5, where the function SAMPLEUNDERNOISE is shown in Algorithm 4 and the LOWERCONFBOUND$(k,n,1-\alpha)$ is a function which returns a one-sided $(1-\alpha)$ lower confidence bound $\underline{p}$ for the Binomial parameter p given $k \sim \text{Binomial}(n,p)$.

5 Experiments

In this section, we present the evaluation results for our method DiffSmooth. We first show the effectiveness of DiffSmooth compared with the existing baselines. Then we conduct a set of ablation studies to evaluate the influence of different factors, including (1) the importance of local smoothing and diffusion model purification process, (2) the influence of the magnitude of local smoothing noise, and (3) the influence of the number of noise sampled during local smoothing. Concretely, we show that: (i) DiffSmooth consistently outperforms all the other baselines under any ℓ_2 radius in terms of the certified robustness, and the performance can be further improved with a better model (e.g., ViT models); (ii) the significant performance improvement of our method DiffSmooth

Table 1: Certified accuracy of ResNet-110 on CIFAR-10 under different ℓ_2 radii. The smoothed model used for our method DiffSmooth is indicated inside the brackets, e.g., DiffSmooth(Gaussian) indicates the base smoothed model is trained with *Gaussian*.

Method[1]	Extra data	Certified Accuracy (%) under ℓ_2 Radius r									
		0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25
Gaussian [13]	✗	75.0	60.0	42.8	32.0	23.0	17.4	14.0	11.8	9.8	7.6
SmoothAdv [43]	✗	73.6	66.8	57.2	47.2	37.6	32.8	28.8	23.6	19.4	16.8
SmoothAdv [43]	✓	80.8	71.4	63.2	52.6	39.4	32.2	26.2	22.2	20.2	18.4
MACER [59]	✗	81.0	71.0	59.0	47.0	38.8	33.0	29.0	23.0	19.0	17.0
Consistency [26]	✗	77.8	68.8	58.1	48.5	37.8	33.9	29.9	25.2	19.5	17.3
SmoothMix [25]	✗	77.1	67.9	57.9	47.7	37.2	31.7	25.7	20.2	17.2	14.7
Boosting [24]	✗	83.4	70.6	60.4	52.4	38.8	34.4	30.4	25.0	19.8	16.6
DDS(Standard) [10][2]	✗	79.0	62.0	45.8	32.6	25.0	17.6	11.0	6.2	4.2	2.2
DDS(Smoothed) [10][3]	✓	79.8	69.9	55.0	47.6	37.4	32.4	28.6	24.8	15.4	13.6
DiffSmooth(Gaussian)	✗	78.2	67.2	59.2	47.0	37.4	31.0	25.0	19.0	16.4	14.2
DiffSmooth(SmoothAdv)	✗	82.8	72.0	62.8	51.2	41.2	36.2	**32.0**	**27.0**	**22.0**	19.0
DiffSmooth(SmoothAdv)	✓	**85.4**	**76.2**	**65.6**	**57.0**	**43.6**	**37.2**	31.4	25.2	21.6	**20.0**

[1] We report the performance for Gaussian and SmoothAdv based on pretrained models.
[2] We reimplement and report the results of DDS [17] on ResNet-110.
[3] We use the same smoothed models as tested on DiffSmooth (i.e., Gaussian and SmoothAdv) for DDS and report the best results.

is attributed to the combination of the diffusion model based purification and local smoothing, which verifies the rationale of our method design. We show that without local smoothing or diffusion-based purification, the certified robustness will drop significantly; iii) using a small magnitude of the local smoothing noise will benefit the certification under small ℓ_2 radii, while noises with large magnitude provide higher certified accuracy under large radii; iv) the certified accuracy will also be consistently improved with the increase of the number of sampled local smoothing noise; v) even with a similar computation cost required by DDS, our method still outperforms DDS. The detailed experimental settings and results are shown below.

5.1 Experimental Setup.

Dataset and Base Classifiers. We conduct the experimental evaluation of our method on datasets CIFAR-10 [27] and ImageNet [15]. Following the common setting [13], we use ResNet-110 as the base classifier on CIFAR-10 and use ResNet-50 [22] on ImageNet. To further demonstrate the effectiveness of our method, we also conduct the experiments with a BEiT large model [4] on ImageNet following [10]. Specifically, we only finetune the BEiT large model under Gaussian augmentation with $\sigma \in \{0.25, 0.50, 1.00\}$ using ImageNet-1K based on the self-supervised pretrained model (with intermediate finetuned on ImageNet-22K). For other smoothed models, including ResNet-110 and ResNet-50, we directly use the pretrained ones from [13,43]. The BEiT large model is fine-tuned with 30 epochs, and the resolution for both the training and prediction is 224×224 instead of 512×512 which is used in [10]; other training hyperparameters of the BEiT are set the same as the finetuning on standard classifier[1] and the detailed settings are shown in Appendix B.3.

Diffusion Models. We use the unconditional improved diffusion model[2] from [34], which is trained under L_{hybrid} objective, to denoise images from CIFAR-10; and use the unconditional 256×256 guided diffusion model[3] from [16] to denoise images from ImageNet.

Baselines. We consider eight state-of-the-art ℓ_2 certifiably robust models as baselines: (1) *Gaussian smoothing* [13], which trains a standard model with Gaussian augmentation for training data; (2) *SmoothAdv* [43], which introduces adversarial training into the Gaussian augmented training, and it also provides the semi-supervised trained model with extra unlabelled data used in [12]; (3) *MACER* [59], which tries to maximize the certified radius directly instead of applying an attack-free algorithm: (4) *Consistency* [26], which adds a consistency regularization term into the training loss; (5) *SmoothMix* [25], which combines mixup [61] with adversarial training to boost the certified robustness; (6) *Boosting* [24], which adopts variance-reduced *ensemble* model to generate more consistent prediction; (7) *Diffusion Denoised Smoothing (DDS(Standard))* [10], which leverages diffusion models to remove the added Gaussian smoothing noise and then applies off-the-shelf standard classifiers to predict the purified instances; and (8) *DDS(Smoothed)*, which replaces the standard classifier in DDS(standard) with smoothed classifiers including Gaussian augmented classifier [13] and SmoothAdv [43] respectively, and then selects the maximal certified accuracy among these two.

[1] https://github.com/microsoft/unilm/tree/master/beit
[2] https://github.com/openai/improved-diffusion
[3] https://github.com/openai/guided-diffusion

Table 2: Certified accuracy on ImageNet under different ℓ_2 radii. The smoothed model used for our method DiffSmooth is indicated inside the brackets, e.g., DiffSmooth(Gaussian) indicates the base smoothed model is trained with *Gaussian*.

Architecture	Method[1]	Certified Accuracy (%) under ℓ_2 Radius r						
		0.00	0.50	1.00	1.50	2.00	2.50	3.00
ResNet-50	Gaussian [13]	66.4	48.6	37.0	25.4	18.4	13.8	10.4
	SmoothAdv [43]	66.6	52.6	42.2	34.6	25.2	21.4	18.8
	MACER [59]	**68.0**	57.0	43.0	37.0	27.0	25.0	20.0
	Consistency [26]	57.0	50.0	44.0	34.0	24.0	21.0	17.0
	SmoothMix [25]	55.0	50.0	43.0	38.0	26.0	24.0	20.0
	Boosting [24][2]	**68.0**	57.0	44.6	38.4	28.6	24.6	21.2
	DDS(Standard) [10][3]	67.4	49.0	33.0	22.2	17.4	12.8	8.0
	DDS(Smoothed) [10][4]	48.0	40.6	29.6	23.8	18.6	16.0	13.4
	DiffSmooth(Gaussian)	66.2	57.8	44.2	36.8	28.6	25.0	19.8
	DiffSmooth(SmoothAdv)	66.2	**59.2**	**48.2**	**39.6**	**31.0**	**25.4**	**22.4**
BEiT[6]	Gaussian [13]	82.0	70.2	51.8	38.4	32.0	23.0	17.0
	DDS(Standard) [10]	82.8	71.1	54.3	38.1	29.5	-	13.1
	DDS(Smoothed) [10]	76.2	60.2	43.8	31.8	22.0	17.8	12.2
	DiffSmooth(Gaussian)	**83.8**	**77.2**	**63.2**	**53.0**	**37.6**	**31.4**	**24.8**

[1] We report the results for Gaussian and SmoothAdv based on pretrained models with the same number of smoothing noise for evaluating DiffSmooth ($N = 10,000$) for a fair comparison.
[2] Boosting is an ensemble method with the base models trained under *Gaussian*, *SmoothAdv*, *Consistency* and *MACER*.
[3] The authors use a pretrained BEiT large model [4] in the original paper, and we reimplement DDS on ResNet-50 here and report the results.
[4] We use the same smoothed models (i.e., Gaussian and SmoothAdv) used in DiffSmooth for DDS and report the best results.

Table 3: Certified accuracy of *Gaussian* with $\sigma = 0.50$ under different magnitudes of local smoothing noise *without* the diffusion-based purification process. The base classifier is trained under noise level 0.50, and the number of local smoothing noise m is 21. When σ' is set to "$-$", it represents the standard randomized smoothing setting, indicating that local smoothing is required only when diffusion-based purification is performed.

Dataset	σ'	ACR	Certified Accuracy under ℓ_2 Radius r							
			0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75
CIFAR-10	-	0.534	**65.0**	**54.4**	41.4	32.0	23.0	15.2	9.4	5.4
	0.12	**0.538**	**65.0**	54.2	**42.8**	33.0	23.4	15.6	10.0	5.0
	0.25	0.537	64.2	54.2	41.0	**33.2**	**24.2**	**16.8**	10.4	**6.2**
	0.50	0.433	44.8	39.0	33.8	26.0	20.4	15.0	**11.6**	6.0
ImageNet	-	**0.640**	**56.8**	**51.2**	**45.2**	**41.8**	**37.0**	**31.4**	**24.6**	0.0
	0.25	0.333	32.0	27.4	23.8	21.6	18.6	15.0	11.8	0.0
	0.50	0.021	2.2	1.8	1.6	1.4	1.2	0.8	0.8	0.0

Certification Details. For both CIFAR-10 and ImageNet, we certify a subset of 500 samples from their test set with confidence 99.9%. Besides, each data point is certified with $N = 100,000$ samples of smoothing noise on CIFAR-10 following prior work [13], and we use $N = 10,000$ on ImageNet following [10], the σ is selected in $\{0.25, 0.50, 1.00\}$ for all models. For each σ, we try different $\sigma' \leq \sigma$ to explore the influence of the magnitude of local smoothing noise. The details are in the Appendix B.2.

We evaluate our method on the smoothed model trained with two different methods, *Gaussian* and *SmoothAdv*, denoted DiffSmooth(Gaussian) and DiffSmooth(SmoothAdv), respectively. In specific, for the experiments on CIFAR-10 with ResNet-110 and the experiments on ImageNet with ResNet-50, we directly use the pretrained smoothed models from *Gaussian* [13] [4] and *SmoothAdv* [43] [5], and the detailed information of the selected pretrained models are deferred to Appendix B.1. For the experiments on ImageNet with BEiT large model, we finetuned the BEiT large model that is pretrained on ImageNet-22K with the method *Gaussian*.

Note that our method does not need to be further finetuned on these purified images. All the smoothed models are trained with the clean images x, while the prediction during the certification is conducted on the purified images $\hat{x}$. Nevertheless,

[4] https://github.com/locuslab/smoothing
[5] https://github.com/Hadisalman/smoothing-adversarial

Table 4: Certified accuracy of DiffSmooth on CIFAR-10 under different ℓ_2 radii for smoothed models with noise level σ and local smoothing with noise level σ'. We defer the full certification results w/o diffusion-based purification or local smoothing for smoothed models to **??** for comparison. The number of used local smoothing noise m is 21, and ACR denotes the average certified radius.

Methods	σ	σ'	ACR	Certified Accuracy (%) under ℓ_2 Radius r									
				0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25
DiffSmooth (Gaussian)	0.25	0.12	0.543	**78.2**	67.0	58.0	44.8	0.0	0.0	0.0	0.0	0.0	0.0
		0.25	**0.556**	76.4	**67.2**	**59.2**	**47.0**	0.0	0.0	0.0	0.0	0.0	0.0
	0.50	0.12	0.703	**70.4**	61.2	52.4	41.6	33.0	25.4	19.4	12.2	0.0	0.0
		0.25	0.745	69.6	**62.8**	**55.6**	**45.2**	36.4	28.2	21.6	14.6	0.0	0.0
		0.50	**0.760**	67.4	61.6	53.4	43.6	**37.4**	**31.0**	**25.0**	**18.2**	0.0	0.0
	1.00	0.12	0.581	51.0	43.4	36.2	31.8	25.4	19.6	14.6	11.4	7.6	6.0
		0.25	0.638	**54.0**	**46.2**	39.6	33.2	26.6	21.8	17.4	12.6	9.6	7.6
		0.50	0.699	53.0	46.0	**40.2**	34.8	29.8	23.8	20.2	15.6	12.4	10.2
		1.00	**0.784**	47.8	44.6	39.6	**35.8**	**31.4**	**27.0**	**23.4**	**19.0**	**16.4**	**14.2**
DiffSmooth (SmoothAdv)	0.25	0.12	**0.593**	**82.8**	**72.0**	**62.8**	49.4	0.0	0.0	0.0	0.0	0.0	0.0
		0.25	0.572	73.4	66.2	59.6	**51.2**	0.0	0.0	0.0	0.0	0.0	0.0
	0.50	0.12	0.733	**74.2**	**64.4**	54.0	45.0	33.8	28.0	18.4	13.4	0.0	0.0
		0.25	**0.799**	68.6	61.0	**55.4**	**48.2**	**41.2**	34.8	26.8	18.6	0.0	0.0
		0.50	0.726	49.8	47.2	43.4	40.4	38.2	**36.2**	**32.0**	**27.0**	0.0	0.0
	1.00	0.12	0.584	52.0	45.6	37.6	30.4	25.2	18.8	13.6	10.2	7.6	6.0
		0.25	0.724	**52.6**	**47.2**	**40.6**	36.6	31.4	26.4	21.4	15.8	12.2	10.4
		0.50	0.880	45.0	42.4	39.6	**37.4**	**34.4**	30.2	26.8	23.2	19.6	17.8
		1.00	**0.910**	44.4	41.8	39.0	36.0	33.8	**31.0**	**28.0**	**25.6**	**22.0**	**19.0**
DiffSmooth (SmoothAdv w/ extra data)	0.25	0.12	**0.624**	**85.4**	**76.2**	**65.6**	52.0	0.0	0.0	0.0	0.0	0.0	0.0
		0.25	0.622	79.2	72.8	65.4	**57.0**	0.0	0.0	0.0	0.0	0.0	0.0
	0.50	0.12	0.771	**74.8**	64.8	57.0	47.8	36.4	29.6	21.8	14.2	0.0	0.0
		0.25	**0.830**	72.2	**65.0**	**58.8**	**50.4**	**43.6**	33.6	26.2	18.2	0.0	0.0
		0.50	0.794	60.8	56.0	49.6	45.6	41.4	**37.2**	**31.4**	**25.2**	0.0	0.0
	1.00	0.12	0.623	53.8	46.6	39.0	33.0	26.2	21.4	16.4	10.8	8.2	6.4
		0.25	0.709	**54.0**	**48.2**	**43.6**	37.0	30.0	24.6	20.4	15.0	11.4	8.2
		0.50	0.827	49.6	46.0	42.6	**39.2**	**33.8**	30.0	25.6	20.6	16.8	14.2
		1.00	**0.914**	43.0	41.2	38.6	35.2	32.0	**30.6**	**27.2**	**24.0**	**21.6**	**20.0**

as shown in Figure 2, the purified images are usually blurrier and less fine-grained than the clean images, and thus the classification accuracy on the purified images will drop a bit when compared to the clean accuracy. So [10] propose to further finetune the pretrained smoothed models with the purified images. However, such finetuning is quite expensive for ImageNet due to the high cost of the purification. As a compromise, we find that using the local smoothing noise with a magnitude slightly larger than the nominal magnitude can effectively offset the influence of such a distribution shift. Empirically, on CIFAR-10, we in fact, add the local smoothing noise with the magnitude of $(\sigma' + 0.03)$ under all smoothing level σ; and we add the local smoothing noise with magnitude $(\sigma' + 0.01)$ when $\sigma = 0.25$ while adding local smoothing noise with magnitude $(\sigma' + 0.02)$ for other σ on ImageNet. For reference, we also provide the experiment results without this strategy in Appendix C.

Evaluation Metrics. We report the certified accuracy under different ℓ_2 radius r following the standard certification setting [13]. In addition, we report the average certified radius (ACR) of 500 certified images following [59].

5.2 Main Results

We compare DiffSmooth with existing baselines on different model architectures and σ. The results on CIFAR-10 and ImageNet are shown in Table 1 and Table 2, respectively. As we can see, our method DiffSmooth on smoothed models (i.e., DiffSmooth(Gaussian) and DiffSmooth(SmoothAdv)) consistently outperforms all other baselines on both CIFAR-10 and ImageNet. In specific, on pretrained *Gaussian* model (DiffSmooth(Gaussian)) with CIFAR-10 data, DiffSmooth improves the certified accuracy largely from 42.8% to 59.2% under ℓ_2 radius 0.50, and from 23.0% to 37.4% under larger ℓ_2 radius 1.00. On the pretrained *SmoothAdv* model (DiffSmooth(SmoothAdv)), DiffSmooth improves the certified accuracy compared to DiffSmooth(Gaussian), achieving state-of-the-art certified robustness. On ImageNet with ResNet-50, under ℓ_2 radius 1.5, the certified accuracy of DiffSmooth can be improved from 25.4% to 36.8% on the pretrained *Gaussian* model (i.e., DiffSmooth(Gaussian)), and from 34.6% to 39.6%

Table 5: Certified accuracy of DiffSmooth on ImageNet under different ℓ_2 radii for smoothed models with noise level σ and local smoothing with noise level σ'. We defer the full certification results w/o diffusion-based purification or local smoothing for smoothed models to **??** for comparison. The number of used local smoothing noise m is 21, and ACR denotes the average certified radius.

Architecture	Methods	σ	σ'	ACR	Certified Accuracy (%) under ℓ_2 Radius r						
					0.00	0.50	1.00	1.50	2.00	2.50	3.00
ResNet-50	DiffSmooth (Gaussian)	0.25	0.25	**0.467**	**66.2**	**57.8**	0.0	0.0	0.0	0.0	0.0
		0.50	0.25	0.710	**57.2**	50.4	41.4	31.4	0.0	0.0	0.0
			0.50	**0.741**	55.8	**50.8**	**44.2**	**36.8**	0.0	0.0	0.0
		1.00	0.25	0.809	43.4	38.4	31.8	24.6	19.8	16.6	12.4
			0.50	1.008	**48.4**	**42.4**	36.2	31.6	27.2	24.0	18.4
			1.00	**1.013**	43.6	40.4	**37.0**	**32.8**	**28.6**	**25.0**	**19.8**
	DiffSmooth (SmoothAdv)	0.25	0.25	**0.478**	**66.2**	**59.2**	0.0	0.0	0.0	0.0	0.0
		0.50	0.25	**0.792**	**59.0**	**53.4**	**48.2**	**39.6**	0.0	0.0	0.0
			0.50	0.741	54.0	49.6	44.2	38.2	0.0	0.0	0.0
		1.00	0.25	1.000	**48.4**	43.0	36.0	31.8	27.6	23.2	16.0
			0.50	**1.087**	47.6	**43.8**	**39.8**	**35.0**	**31.0**	**25.4**	**22.4**
			1.00	0.937	37.8	34.8	32.8	30.4	27.0	23.8	21.0
BEiT	DiffSmooth (Gaussian)	0.25	0.12	**0.623**	**83.8**	**77.2**	0.0	0.0	0.0	0.0	0.0
			0.25	0.618	82.0	76.6	0.0	0.0	0.0	0.0	0.0
		0.50	0.12	1.044	**79.2**	**72.6**	61.8	50.2	0.0	0.0	0.0
			0.25	**1.061**	**79.2**	71.8	**63.2**	52.8	0.0	0.0	0.0
			0.50	1.023	73.4	67.6	62.0	**53.0**	0.0	0.0	0.0
		1.00	0.12	1.216	**62.2**	55.0	47.8	38.2	32.4	24.2	18.6
			0.25	1.282	62.0	**57.4**	49.0	40.6	34.0	27.6	20.0
			0.50	**1.333**	61.4	55.8	**49.2**	**43.0**	**37.6**	**31.4**	22.6
			1.00	1.214	50.8	47.4	43.2	38.8	35.4	31.0	**24.8**

on the *SmoothAdv* model (i.e., DiffSmooth(SmoothAdv)). The performance is further improved on the *Gaussian* smoothed BEiT large model, and the certified accuracy is improved from 70.2% to 77.2% under ℓ_2 radius 0.50 and from 36.0% to 53.0% under radius 1.5.

Note that the only difference between DiffSmooth(SmoothAdv) and DDS(Smoothed) is the local smoothing design (DDS(Smoothed) reports the best results of DDS on smoothed models Gaussian and SmoothAdv). When we compare the results of DiffSmooth(SmoothAdv) with DDS(Smooth), we find that the certified accuracy drops from 65.6% (DiffSmooth(SmoothAdv)) to 55.0% (DDS(Smoothed))) on CIFAR-10, and drop from 48.2% (DiffSmooth(SmoothAdv)) to 29.6% (DDS(Smoothed)) on ImageNet under ℓ_2 radius 1.0. Thus, it verifies the importance of local smoothing design in DiffSmooth.

5.3 Ablation studies

Local smoothing w/o diffusion models. To verify the importance of our methodology design (i.e., diffusion model + local smoothing + smoothed classifier), we remove the diffusion model from our system design by directly applying local smoothing noise δ' on *Gaussian* [13] (i.e., local smoothing + smoothed classifier). The results are shown in Table 3. We find that local smoothing can not help to improve the certified accuracy of [13], and the performance even degrades with large σ'. The reason is that $\|(x+\delta)-x\| \gg \|\hat{x}-x\|$, then with further local smoothing, the image will only be corrupted more, and thus the classification performance will drop naturally. This result further verifies the rationale of our methodology design.

Magnitude of the local smoothing noise. To study the influence of the magnitude of local smoothing noise, we conduct experiments among different $\sigma' \in \{0.12, 0.25, 0.50\}$. The full results of our method on CIFAR-10 and ImageNet are shown in Table 4 and Table 5, respectively. We can observe that by carefully selecting the σ', the performance will be further improved. In addition, we find that to certify a small radius, smaller σ' is preferred; however, for certifying a large radius, the choice of σ' depends on the model resilience to random noise, i.e., for Gaussian smoothed models, σ' needs to be close to σ while for SmoothAdv σ' can be chosen to be around $\sigma/2$.

Number of noise samples for local smoothing. To evaluate the influence of the number of noise samples m for local smoothing, we evaluate the certified robustness with $m \in 1, 3, 5, 11, 21$. The certified accuracy under different numbers of smoothing noise samples is shown in Figure 3. We observe that the certified accuracy will increase monotonically with

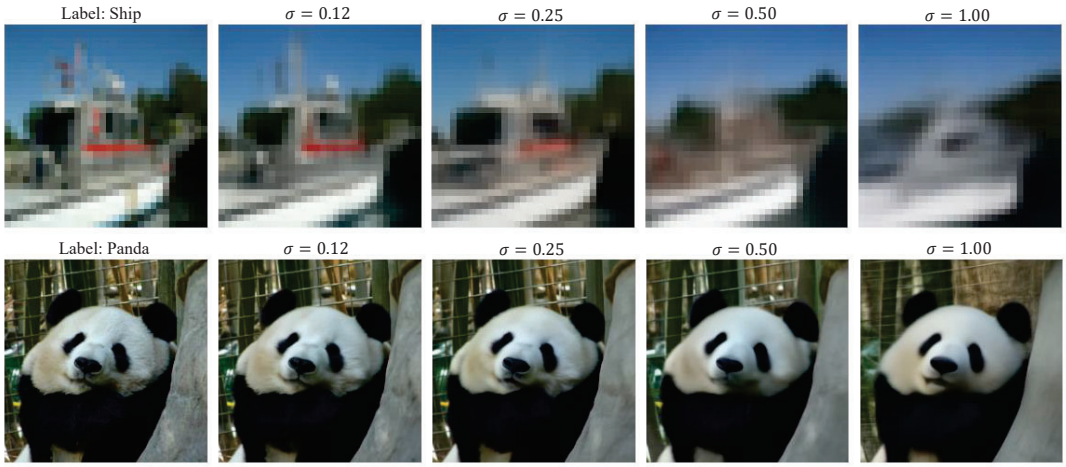

Figure 2: Purified images under different magnitudes of Gaussian smoothing noises. Original clean images are shown in the first column. It shows that with higher smoothing noise, the purified image is more blurred.

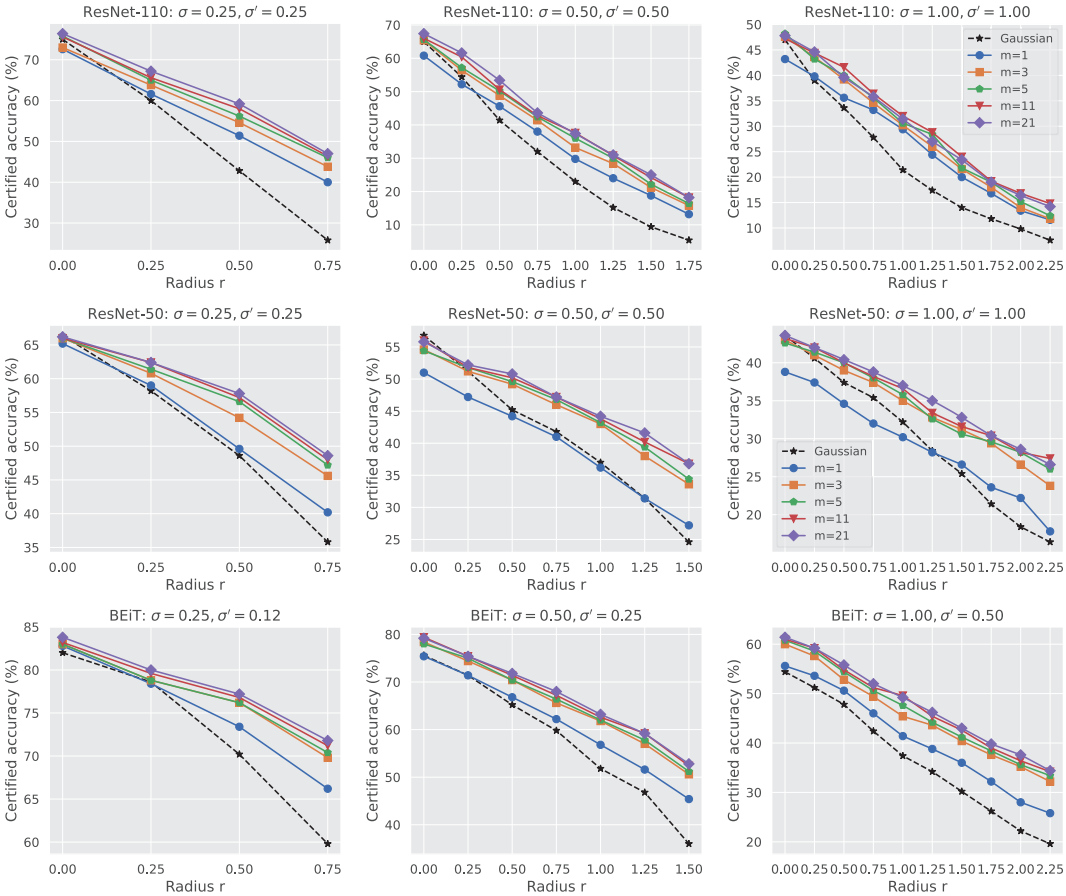

Figure 3: The certified accuracy of DiffSmooth(Gaussian) with different numbers of local smoothing noise on CIFAR-10 and ImageNet. We show the results on CIFAR-10 in the first row, while the results on ImageNet with ResNet-50 and BEiT are shown in the second and third rows, respectively.

the number of smoothing noise samples. In practice, only using five samples ($m = 5$) is enough to achieve non-trivial robustness certification.

Computation cost. Additionally, we calculate the time efficiency for certifying one image with different m on one NVIDIA RTX A6000. Specifically, on CIFAR-10, our certification takes 97s, 111s, 125s, 169s, 240s for $m = 1, 3, 5, 11, 21$, respectively; while on ImageNet with ResNet-50, it takes 564s, 586s, 608s, 672s, 776s for $m = 1, 3, 5, 11, 21$, respectively. The corresponding computation cost of the one-shot reverse diffusion step on certifying one image is 90s on CIFAR-10 and 553s on ImageNet, respectively. Based on the similar

Table 6: Certified accuracy of ResNet-110 on CIFAR-10 under different ℓ_2 radii with the number of predictions as $100,000$.

Method	Setting	Certified Accuracy (%) under ℓ_2 Radius r								
		0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00
DDS(Standard)	$N = 100,000$	79.0	62.0	45.8	32.6	25.0	17.6	11.0	6.2	4.2
DDS(Smoothed)	$N = 100,000$	79.8	69.9	55.0	47.6	37.4	32.4	28.6	24.8	15.4
DiffSmooth(Gaussian)	$N = 20,000, m = 5$	77.2	67.4	55.6	44.4	35.0	29.4	21.8	18.4	15.0
	$N = 10,000, m = 10$	76.8	66.0	56.6	42.8	36.0	29.0	23.6	18.2	16.6
	$N = 5,000, m = 20$	77.2	66.8	58.2	43.8	36.6	29.4	22.0	18.0	15.0
DiffSmooth(SmoothAdv)	$N = 20,000, m = 5$	82.2	71.6	62.8	49.2	39.8	35.2	29.8	24.0	22.4
	$N = 10,000, m = 10$	82.8	71.0	62.4	48.4	40.0	35.4	29.6	24.6	21.0
	$N = 5,000, m = 20$	82.6	71.8	61.8	47.4	40.4	34.4	27.2	24.2	20.6
DiffSmooth(SmoothAdv) with extra data	$N = 20,000, m = 5$	**86.0**	75.8	**65.6**	**54.0**	**41.8**	35.6	**30.2**	23.8	**22.2**
	$N = 10,000, m = 10$	85.2	**76.0**	64.2	53.8	**41.8**	**36.0**	28.8	**24.4**	21.4
	$N = 5,000, m = 20$	85.2	**76.0**	64.8	49.2	41.4	34.4	26.6	23.0	20.6

Table 7: Certified accuracy on ImageNet under different ℓ_2 radii with the number of predictions as $10,000$.

Architecture	Method	Setting	Certified Accuracy (%) under ℓ_2 Radius r					
			0.00	0.50	1.00	1.50	2.00	2.50
ResNet-50	DDS(Standard)	$N = 10,000$	**67.4**	49.0	33.0	22.2	17.4	12.8
	DDS(Smoothed)	$N = 10,000$	48.0	40.6	29.6	23.8	18.6	16.0
	DiffSmooth(Gaussian)	$N = 2,000, m = 5$	65.4	54.8	42.4	30.2	26.8	21.0
		$N = 1,000, m = 10$	65.8	55.2	42.4	30.6	27.6	-
		$N = 500, m = 20$	65.4	53.8	41.8	30.6	25.4	-
	DiffSmooth(SmoothAdv)	$N = 20,000, m = 5$	64.0	**57.6**	**46.4**	**33.8**	**28.6**	**23.4**
		$N = 1,000, m = 10$	64.6	57.2	46.0	32.8	27.8	-
		$N = 500, m = 20$	65.0	56.4	45.2	32.4	26.6	-
BEiT	DDS(Standard)	$N = 10,000$	82.8	71.1	54.3	38.1	29.5	-
	DDS(Smoothed)	$N = 10,000$	76.2	60.2	43.8	31.8	22.0	17.8
	DiffSmooth(Gaussian)	$N = 2,000, m = 5$	83.0	75.6	60.0	40.1	**34.9**	**25.7**
		$N = 1,000, m = 10$	**83.2**	**76.2**	**60.6**	**40.3**	34.3	-
		$N = 500, m = 20$	83.4	75.0	59.6	**40.3**	31.9	-

computation costs of the two methods, we can see that the main bottleneck of the computation cost is the reverse diffusion step instead of the local smoothing.

Number of local smoothing predictions. For a fair comparison, we constrain the number of local smoothing predictions to be the same as the number of predictions in DDS. In other words, we will maintain $100,000$ prediction queries on CIFAR10 and $10,000$ prediction queries on ImageNet, and the corresponding results are shown in Table 6 and Table 7 respectively. As we can see, DiffSmooth performs significantly better than DDS even with the same computation cost, and setting $m = 5$ is already good enough in practice.

6 Conclusion

In this work, we aim to leverage diffusion-based purification to provide improved certified robustness for *smoothed* models. We first provide theoretical analysis to show that the recovered instances from (adversarial) inputs will be in the bounded neighborhood of the corresponding original instance with high probability, and the "one-shot" DDPM can approximate the original instance under mild conditions.

Based on our analysis, we propose a certifiably robust pipeline, DiffSmooth, for *smoothed* models. In particular, DiffSmooth performs diffusion-based adversarial purification, followed by a local smoothing step to provide certified robustness for smoothed models. We conduct extensive experiments on different datasets and show that DiffSmooth can achieve state-of-the-art certified robustness.

One limitation of our method is that it will take more time for certification; however, the main computation cost during certification actually comes from the diffusion step instead of the local smoothing part. We show that under the same computation cost with DDS, our method still achieves higher certified robustness and benign accuracy, which provides interesting and promising directions. Overall, we hope our study sheds light on developing certifiably robust ML models based on diffusion models and smoothed classifiers.

References

[1] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

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

[3] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. *arXiv preprint arXiv:2208.09392*, 2022.

[4] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*, 2021.

[5] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

[6] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[7] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[8] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision*, pages 364–381. Springer, 2020.

[9] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. Chen, M. Liu, and B. Li. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1302–1320, Los Alamitos, CA, USA, may 2021. IEEE Computer Society.

[10] Nicholas Carlini, Florian Tramer, J Zico Kolter, et al. (certified!!) adversarial robustness for free! *arXiv preprint arXiv:2206.10550*, 2022.

[11] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 3–14, 2017.

[12] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. *Advances in neural information processing systems*, 32, 2019.

[13] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

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

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[18] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[19] Ugo Fiore, Alfredo De Santis, Francesca Perla, Paolo Zanetti, and Francesco Palmieri. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, 479:448–455, 2019.

[20] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

[21] Jamie Hayes. Extensions and limitations of randomized smoothing for robustness guarantees. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 786–787, 2020.

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

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

[24] Miklós Z Horváth, Mark Niklas Mueller, Marc Fischer, and Martin Vechev. Boosting randomized smoothing with variance reduced classifiers. In *International Conference on Learning Representations*, 2021.

[25] Jongheon Jeong, Sejun Park, Minkyu Kim, Heung-Chang Lee, Do-Guk Kim, and Jinwoo Shin. Smoothmix: Training confidence-calibrated smoothed classifiers for certified robustness. *Advances in Neural Information Processing Systems*, 34:30153–30168, 2021.

[26] Jongheon Jeong and Jinwoo Shin. Consistency regularization for certified robustness of smoothed classifiers. In *34th Conference on Neural Information Processing Systems (NeurIPS) 2020*. NeurIPS committee, 2020.

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

[28] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

[29] Kyungmin Lee. Provable defense by denoised smoothing with learned score function. In *ICLR Workshop on Security and Safety in Machine Learning Systems*, 2021.

[30] Linyi Li, Tao Xie, and Bo Li. Sok: Certified robustness for deep neural networks. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, 22-26 May 2023*. IEEE, 2023.

[31] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, Mykel J Kochenderfer, et al. Algorithms for verifying deep neural networks. *Foundations and Trends® in Optimization*, 4(3-4):244–404, 2021.

[32] Chizhou Liu, Yunzhen Feng, Ranran Wang, and Bin Dong. Enhancing certified robustness via smoothed weighted ensembling. *arXiv preprint arXiv:2005.09363*, 2020.

[33] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

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

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

[36] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE, 2016.

[37] Apapan Pumsirirat and Yan Liu. Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine. *International Journal of advanced computer science and applications*, 9(1), 2018.

[38] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

[39] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*, 2018.

[40] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[42] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[43] Hadi Salman, Jerry Li, Ilya P Razenshteyn, Pengchuan Zhang, Huan Zhang, Sébastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

[44] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. *Advances in Neural Information Processing Systems*, 33:21945–21957, 2020.

[45] Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robustness verification of neural networks. *Advances in Neural Information Processing Systems*, 32:9835–9846, 2019.

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

[47] Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

[48] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[49] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

[50] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[51] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

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

[53] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

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

[55] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems*, 34:29909–29921, 2021.

[56] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2019.

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

[58] Zhuolin Yang, Linyi Li, Xiaojun Xu, Bhavya Kailkhura, Tao Xie, and Bo Li. On the certified robustness for ensemble models and beyond. In *International Conference on Learning Representations*, 2022.

[59] Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. In *International Conference on Learning Representations*, 2019.

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

[61] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[62] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *International Conference on Learning Representations*, 2019.

[63] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, 2018.

A Theorems And Proofs

Theorem 1. *Given a data distribution $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x}\sim p}[||\mathbf{x}||_2^2] < \infty$. Let p_t be the distribution of $\mathbf{x}(t)$ generated by SDE and suppose $\nabla_x \log p_t(x) \le \frac{1}{2}C, \forall t \in [0,T]$. Let $\gamma(t)$ be the coefficient defined in SDE and $\bar{\alpha}_t = e^{-\int_0^t \gamma(s)ds}$. Then given an adversarial sample $x_{rs} = x_0 + \delta$, solving reverse-SDE starting at time t^* and point $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$ until time 0 will generate a reversed random variable $\hat{\mathbf{x}}_0$ such that with a probability of at least $1-\eta$, we have*

$$||\hat{\mathbf{x}}_0 - x_0|| \le ||x_{rs} - x_0|| + \sqrt{e^{2\tau(t^*)} - 1} C_\eta + \tau(t^*) C$$

where $\tau(t) := \int_0^t \frac{1}{2}\gamma(s)ds$, $C_\eta := \sqrt{d + 2\sqrt{d\log\frac{1}{\eta}} + 2\log\frac{1}{\eta}}$, and d is the dimension of x_0.

Proof. We leverage the proof of [35, Theorem 3.2]. By reverse-SDE, we can bound

$$\begin{aligned}
\|\hat{\mathbf{x}}(0) - x_0\| &= \|x(t^*) + \hat{\mathbf{x}}(0) - x(t^*) - x_0\| \\
&= \left\| x(t^*) + \int_{t^*}^0 -\frac{1}{2}\gamma(t)\left[\mathbf{x}(t) + 2\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))\right] dt + \int_{t^*}^0 \sqrt{\gamma(t)} d\overline{\boldsymbol{w}} - x_0 \right\| \\
&\le \left\| x(t^*) + \int_{t^*}^0 -\frac{1}{2}\gamma(t)\mathbf{x}(t)dt + \int_{t^*}^0 \sqrt{\gamma(t)} d\overline{\boldsymbol{w}} - x_0 \right\| + \left\| \int_{t^*}^0 -\gamma(t)\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t)) dt \right\|
\end{aligned}$$

where the second equation follows from the integration of reverse-SDE, and in the last line we have separated the integration of the linear SDE from non-linear SDE involving $\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))$ by using the triangle inequality.

The above linear SDE is a time-varying Ornstein–Uhlenbeck process with a negative time increment that starts from $t = t^*$ to $t = 0$ with the initial value set to $x(t^*)$. Denote by $\mathbf{x}'(0)$ its solution, from [47] we know $\mathbf{x}'(0)$ follows a Gaussian distribution, where its mean $\boldsymbol{\mu}(0)$ and covariance matrix $\boldsymbol{\Sigma}(0)$ are the solutions of the following two differential equations, respectively:

$$\begin{aligned}
\frac{d\boldsymbol{\mu}}{dt} &= -\frac{1}{2}\gamma(t)\boldsymbol{\mu} \\
\frac{d\boldsymbol{\Sigma}}{dt} &= -\gamma(t)\boldsymbol{\Sigma} + \gamma(t)\boldsymbol{I}_d
\end{aligned}$$

with the initial conditions $\boldsymbol{\mu}(t^*) = x(t^*)$ and $\boldsymbol{\Sigma}(t^*) = \mathbf{0}$. By solving these two differential equations, we have that conditioned on $x(t^*), \mathbf{x}'(0) \sim \mathcal{N}\left(e^{\tau(t^*)} x(t^*), \left(e^{2\tau(t^*)} - 1\right)\boldsymbol{I}_d\right)$, where $\tau(t^*) := \int_0^{t^*} \frac{1}{2}\gamma(s)ds$.

Using the reparameterization trick, let $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, we have:

$$\begin{aligned}
\mathbf{x}'(0) - x_0 &= e^{\tau(t^*)} x(t^*) + \sqrt{e^{2\tau(t^*)} - 1}\boldsymbol{\varepsilon} - x_0 \\
&= x_{rs} + \sqrt{e^{2\tau(t^*)} - 1}\boldsymbol{\varepsilon} - x_0
\end{aligned}$$

together with $\nabla_x \log p_t(x) \le \frac{1}{2}C, \forall t \in [0,T]$, results in that

$$||\hat{\mathbf{x}}_0 - x_0|| \le ||\sqrt{\left(e^{\tau(t^*)} - 1\right)}\boldsymbol{\varepsilon} + x_{rs} - x_0|| + \tau(t^*)C$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Since $||\boldsymbol{\varepsilon}^2|| \sim \chi^2(n)$, by the concentration inequality [6], we have

$$\Pr\left(||\boldsymbol{\varepsilon}|| \ge \sqrt{d + 2\sqrt{d\log\frac{1}{\eta}} + 2\log\frac{1}{\eta}}\right) \le \eta.$$

Thus, with probability at least $1-\eta$, we have

$$||\hat{\mathbf{x}}_0 - x_0|| \le ||x_{t^*} - x_0|| + \sqrt{e^{2\tau(t^*)} - 1} C_\eta + \tau(t^*) C.$$

□

Theorem 2. *Given a data distribution $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x}\sim p}[||\mathbf{x}||_2^2] < \infty$, given a time t^* and point $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$, the one-shot reverse diffusion for DDPM algorithm 2 will output an $\hat{x}_0$ such that*

$$\|\hat{x}_0 - \mathbb{E}[\hat{\mathbf{x}}_0 \mid \hat{\mathbf{x}}_{t^*} = x_{t^*}]\| \le \frac{2\sigma_{t^*}^2 \alpha_{t^*} (1-\bar{\alpha}_{t^*})^{3/2}}{\beta_{t^*}^2 \sqrt{\bar{\alpha}_{t^*}}} \cdot \ell_{t^*}(x_{t^*})$$

where $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t$ are random variables generated by reverse-SDE, $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2}\right)$ and $\sigma_t^2 = \frac{1-\alpha_t}{\alpha_t}$ is the variance of Gaussian noise added at time t in the diffusion process.

Proof. Given time t^* and point $x_{t^*} = \sqrt{\bar{\alpha}_{t^*}} x_{rs}$ is equivalent to that in the formula of $\ell_{t^*}(x_{t^*})$, given $\sqrt{\bar{\alpha}_{t^*}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}$. Then the conditional distribution of $\mathbf{x}_0$ will be $\mathbb{P}(\mathbf{x}_0 = x | \mathbf{x}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2}\right)$. By the assumptions that $p \in \mathcal{C}^2$ and $\mathbb{E}_{\mathbf{x}\sim p}[||\mathbf{x}||_2^2] < \infty$, we know the diffusion process by SDE and the reverse process by reverse-SDE follows the same distribution, thus we also have $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2}\right)$.

Further note that

$$\begin{aligned}
\ell_{t^*}(x_{t^*}) &= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_{t^*}^2}{2\sigma_{t^*}^2 \alpha_{t^*}(1-\bar{\alpha}_{t^*})} \|\boldsymbol{\varepsilon} - \varepsilon_\theta(x_{t^*}, t^*)\|^2 \,\middle|\, \sqrt{\bar{\alpha}_{t^*}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}\right] \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_{t^*}^2}{2\sigma_{t^*}^2 \alpha_{t^*}(1-\bar{\alpha}_{t^*})} \|\boldsymbol{\varepsilon} - \varepsilon_\theta(x_{t^*}, t^*)\|^2 \,\middle|\, \mathbf{x}_{t^*} = x_{t^*}\right] \\
&= \mathbb{E}_{\hat{\mathbf{x}}_0, \boldsymbol{\varepsilon}}\left[\frac{\beta_{t^*}^2}{2\sigma_{t^*}^2 \alpha_{t^*}(1-\bar{\alpha}_{t^*})} \|\boldsymbol{\varepsilon} - \varepsilon_\theta(x_{t^*}, t^*)\|^2 \,\middle|\, \hat{\mathbf{x}}_{t^*} = x_{t^*}\right].
\end{aligned}$$

Under the condition $\sqrt{\bar{\alpha}_{t^*}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}$, which is equivalent to that $\sqrt{\bar{\alpha}_{t^*}}\hat{\mathbf{x}}_0 + \sqrt{1-\bar{\alpha}_{t^*}}\boldsymbol{\varepsilon} = x_{t^*}$ where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, there is a one-to-one corresponding between $\hat{\mathbf{x}}_0$ and $\boldsymbol{\varepsilon}$, and

$$\begin{aligned}
\|\boldsymbol{\varepsilon} - \varepsilon_\theta(x_{t^*}, t^*)\|^2 &= \left\| \frac{x_{t^*} - \sqrt{\bar{\alpha}_{t^*}}\hat{\mathbf{x}}_0}{\sqrt{1-\bar{\alpha}_{t^*}}} - \frac{x_{t^*} - \sqrt{\bar{\alpha}_{t^*}}\varepsilon_\theta(x_{t^*}, t^*)}{\sqrt{1-\bar{\alpha}_{t^*}}} \right\|. \\
&= \|\hat{\mathbf{x}}_0 - \hat{x}_0\| \cdot \frac{\bar{\alpha}_{t^*}}{\sqrt{1-\bar{\alpha}_{t^*}}}.
\end{aligned}$$

Table 8: Certified accuracy of DiffSmooth on ImageNet under different smoothing levels with different magnitudes of the local smoothing noise σ' at various ℓ_2 radius. σ is the smoothing noise magnitude on input. The pretrained base classifier was originally trained under σ'. ACR is the average certified radius. The number of used local smoothing noise m is 21 here, and the magnitude of the local smoothing is not shifted during the experiment.

Architecture	Methods	σ	σ'	ACR	Certified Accuracy (%) under ℓ_2 Radius r						
					0.00	0.50	1.00	1.50	2.00	2.50	3.00
ResNet-50	DiffSmooth (Gaussian)	0.25	0.25	0.467	66.0	57.2	0.0	0.0	0.0	0.0	0.0
		0.50	0.25	0.723	**61.4**	**51.8**	40.0	30.8	0.0	0.0	0.0
			0.50	**0.730**	56.8	49.8	**42.6**	**36.2**	0.0	0.0	0.0
		1.00	0.25	0.827	44.0	36.8	32.0	26.0	21.6	15.8	11.6
			0.50	0.969	**44.6**	**41.8**	**36.2**	30.0	25.2	22.8	18.4
			1.00	**0.989**	42.4	38.4	35.6	**32.2**	**29.6**	**24.4**	**18.8**
	DiffSmooth (SmoothAdv)	0.25	0.25	0.475	66.8	58.2	0.0	0.0	0.0	0.0	0.0
		0.50	0.25	**0.765**	**58.2**	**52.0**	**46.0**	36.6	0.0	0.0	0.0
			0.50	0.723	53.4	48.8	43.4	**37.4**	0.0	0.0	0.0
		1.00	0.25	0.938	**46.4**	41.4	34.8	29.8	25.6	21.2	15.4
			0.50	**1.053**	**46.4**	**42.6**	**38.8**	**33.6**	**28.8**	**25.4**	**22.4**
			1.00	0.931	37.0	34.8	33.2	29.8	26.8	24.6	21.0

Therefore,

$$\ell_{t^*}(x_{t^*}) = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\varepsilon}}\left[\frac{\beta_{t^*}^2 s\sqrt{\bar{\alpha}_{t^*}}}{2\sigma_{t^*}^2\alpha_{t^*}\left(1-\bar{\alpha}_{t^*}\right)^{3/2}}\left\|\hat{\mathbf{x}}_0-\hat{x}_0\right\|\,\middle|\,\mathbf{x}_{t^*}=x_{t^*}\right]$$
$$\geq \frac{\beta_{t^*}^2\sqrt{\bar{\alpha}_{t^*}}}{2\sigma_{t^*}^2\alpha_{t^*}\left(1-\bar{\alpha}_{t^*}\right)^{3/2}}\cdot\left\|\hat{x}_0-\mathbb{E}\left[\hat{\mathbf{x}}_0\mid\hat{\mathbf{x}}_{t^*}=x_{t^*}\right]\right\|,$$

where the last inequality is by Jensen's inequality [7]. □

B Experiment detail

B.1 Pretrained smoothed models

We directly use the smoothed models trained under *Gaussian* from [13]; while for the models trained under *SmoothAdv*, we select the best-performing models from [43], and the detailed specification of the hyperparameters for each picked model is shown in Table 9 and Table 10. Notice, when the smoothing level is with $\sigma = 0.25$, the empirical best magnitude of the local smoothing noise for the SmoothAdv model is $\sigma' = \sigma/2 \approx 0.12$. However, there is no pretrained model provided on ImageNet for SmoothAdv under noise level 0.12; then, as an alternative, we in fact, use the SmoothAdv model trained with smaller ε which is 64 instead of 512 when certifying on the smoothing level $\sigma = 0.25$ for ImageNet.

B.2 The setting of the magnitude of the local smoothing noise

On CIFAR10, we test the $\sigma' \in \{0.12, 0.25\}$ for the smoothing level $\sigma = 0.25$, the $\sigma' \in \{0.12, 0.25, 0.50\}$ for the smoothing level $\sigma = 0.50$, and the $\sigma' \in \{0.12, 0.25, 0.50, 1.00\}$ for the smoothing level $\sigma = 1.00$.

On ImageNet, for ResNet-50, we test the $\sigma' \in \{0.25\}$ for the smoothing level $\sigma = 0.25$, the $\sigma' \in \{0.25, 0.50\}$ for the smoothing level $\sigma = 0.50$, and the $\sigma' \in \{0.25, 0.50, 1.00\}$ for the smoothing level $\sigma = 1.00$. And for BEiT large model, we test the $\sigma' \in \{0.12, 0.25\}$ for the smoothing level $\sigma = 0.25$, the $\sigma' \in \{0.12, 0.25, 0.50\}$ for the smoothing level $\sigma = 0.50$.

B.3 Finetuning details of BEiT model

We finetune the BEiT with the checkpoints that are self-supervised pretrained and then intermediate fine-tuned on ImageNet-22k and train it with Gaussian augmentation with $\sigma \in \{0.25, 0.50, 1.00\}$ in 30 epochs. The batch size is 32, the learning rate is $2e-5$, the update frequency is 2, the number of warmup epochs is 5, the layerwise learning rate decay is 0.9, the drop path is set to 0.4, and the weight decay is set to $1e-8$.

Table 9: Detailed specification of the hyperparameters for the selected SmoothAdv models on CIFAR-10 and ImageNet.

Dataset	σ	Method	# steps	ε	m
CIFAR-10	0.12	PGD	10	64	4
	0.25	PGD	10	255	8
	0.50	PGD	10	512	2
	1.00	PGD	10	512	2
ImageNet	0.25	DNN	2	512	1
	0.50	PGD	1	255	1
	1.00	PGD	1	512	1

Table 10: Detailed specification of the hyperparameters for the selected SmoothAdv models with self-training on CIFAR-10.

Dataset	σ	Method	# steps	ε	weight
CIFAR-10 (Self-training)	0.12	PGD	8	64	1.0
	0.25	PGD	4	127	1.0
	0.50	PGD	2	255	0.5
	1.00	PGD	8	512	0.5

C Influence of the shifting on the magnitude of local smoothing noise

We provide the experiment results without magnitude shifting for ResNet-50 on ImageNet in Table 8 for comparison.