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 a 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 $\ell_2$ 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 DDPM 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 relation 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 $\ell_2$ radius 1.5 and 42.2% to 48.2 under $\ell_2$ radius 1.0 on ImageNet; For CIFAR-10, the certified accuracy is improved from 42.8% to 59.2% under $\ell_2$ radius 0.50, and from 39.4% to 43.6% under $\ell_2$ 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
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.

**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).

**3 Background**

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

**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 \( \ell_p \) 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 \( \ell_p \) 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 \textsc{computeTimestep}(\sigma) \cite{10}.
\textbf{Input:} Magnitude of the smoothing noise \(\sigma\).
\textbf{Output:} Start time step for the reverse process and the corresponding \(\alpha_t\).
\begin{enumerate}[1:]
    \item \(t \leftarrow 0\)
    \item \textbf{while} \(1 - \frac{\alpha_t}{\alpha_{t-1}} < \sigma^2\) \textbf{do}
    \item \(t \leftarrow t+1\)
    \item \textbf{end while}
    \item \textbf{return} \(t, \alpha_t\)
\end{enumerate}

Algorithm 2 \textsc{denoise}(x_t, t) \cite{10}. \# One-shot denoising
\textbf{Input:} Intermediate sample \(x_t\) and its associated timestep \(t\).
\textbf{Output:} Predicted original clean image \(\hat{x}_0\).
\begin{enumerate}[1:]
    \item \(\varepsilon \leftarrow \varepsilon_0(x_t, t)\)
    \item \(\hat{x}_0 \leftarrow \sqrt{\frac{\alpha_t}{\sigma^2}} (x_t - \sqrt{1 - \alpha_t} \varepsilon)\)
    \item \textbf{return} \(\hat{x}_0\)
\end{enumerate}

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 \(\delta \sim \mathcal{N}(0, \sigma^2 \mathbf{I})\).
Assume \(p_A\) is the lower bound of prediction probability of the top class \(c_A\), and \(p_B\) is the upper bound of prediction probability for the “runner-up” class, then the smoothed classifier \(g\) is robust around \(x\) within the radius:
\[
R = \frac{1}{\sigma} \left( \Phi^{-1}(p_A) - \Phi^{-1}(p_B) \right),
\]
where \(\Phi^{-1}\) 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 \cite{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 \cite{43} and Consistency \cite{26}.

\textbf{Denoised Smoothing.} To apply the off-the-shelf standard model with randomized smoothing, Salman et al. \cite{44} propose to prepend a custom-trained denoiser \(D_B : \mathbb{R}^d \rightarrow \mathbb{R}^d\), which is trained to remove the Gaussian noise appeared in the instances, to the standard classifier \(f_{\text{std}}\), and treat \(f_{\text{std}} \circ D_B\) as the new base classifier \(f\). In this way, the noisy input \(x + \delta\) will first be purified by \(D_B\), 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 \(f = f_{\text{std}} \circ D_B\) on the noisy instances with Gaussian perturbation is close to the accuracy of \(f_{\text{std}}\) given a clean instances when the denoiser \(D_B\) performs well.

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

where \(\alpha_t = e^{-\frac{1}{2} \gamma(s) ds} \) and \(\mathbf{e} \sim \mathcal{N}(0, \mathbf{I})\). The reverse process can be defined by the stochastic differential equation:
\[
dx = [h(x, t) - g(t)^2 \nabla_x \log p_0(x)] dt + g(t) d\mathbf{W}
\]
\textbf{(reverse-SDE)}
where \(dt\) denotes the infinitesimal reverse time step, and \(\mathbf{W}(t)\) is the reverse-time standard Wiener process. We use \(\{x(t)\}_{t \in [0, T]}\) and \(\{\hat{x}(t)\}_{t \in [0, T]}\) to denote the diffusion process and reverse process respectively. \cite{1} shows that if \(p \in \mathcal{C}^2\) and \(\mathbb{P}_{x \sim p}[||x||^2_2 < \infty, \{x(t)\}_{t \in [0, T]} \text{ and } \{\hat{x}(t)\}_{t \in [0, T]} \text{ have the same distribution.}

\textbf{Denoising Diffusion Probabilistic Models (DDPM)} \cite{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 \(g(x, x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_t \mathbf{I})\). Generally, \(\beta_t\) 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
\[
x_t = \sqrt{\alpha_t} \cdot x_0 + \sqrt{1 - \alpha_t} \cdot \mathbf{e},
\]
where \(\alpha_t := 1 - \beta_t, \beta_t := \prod_{i=1}^t \alpha_i\), and \(\mathbf{e} \sim \mathcal{N}(0, \mathbf{I})\), 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 \(p_T(x_t | x_{t-1})\), which is defined as \(\mathcal{N}(x_t; \mu_0(x_{t-1}, t), \Sigma_0(x_{t-1}, t))\). Here, \(\mu_0(x_{t-1}, t) := \frac{1}{\sqrt{\alpha_t}} (x_{t-1} - \frac{\beta_t}{\sqrt{1 - \alpha_t}} \epsilon_0(x_{t-1}, t))\) in which \(\epsilon_0\) is trained to predict the random noise \(\epsilon\) for \(x_t\), and \(\Sigma_0(x_{t-1}, t)\) is defined as \(\sigma^2 \mathbf{I}\) as in \cite{23} while it can also be learned following \cite{34}. As a result, starting from \(x_T \sim \mathcal{N}(0, \mathbf{I})\), DDPM will generate an instance through the reverse sampling.

An interesting observation from \cite{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 \cite{44} and acted as \(D_B\). Formally, given the corrupted instance \(x_{cor} = x + \delta\) where \(x\) is the clean instance and \(\delta \sim \mathcal{N}(0, \sigma^2 \mathbf{I})\), 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 \(\delta\) to get the original instance \(x\) with the reverse process which
We show that such a mean instance will have the ground-truth label of the corresponding original instance if the points 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 $\epsilon$ in $x_r$ with $\epsilon_0$, and then directly plug it into Equation (2) to obtain $\hat{\epsilon}_0$, for which the corresponding pseudo-code is shown in Algorithm 2.

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 $\epsilon$ in $x_r$ with $\epsilon_0$, and then directly plug it into Equation (2) to obtain $\hat{\epsilon}_0$, 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 $p \in \mathcal{C}^T$ and $\mathbb{E}_{x \sim p}||x||^2_2 < \infty$. Let $p_t$ be the distribution of $x(t)$ generated by SDE and suppose $\nabla_x \log p_t(x) \leq C$ for some constant $C$ and $\forall t \in [0, T]$. Let $\gamma(t)$ be the coefficient defined in SDE and $\bar{\epsilon}_t = e^{-\frac{\gamma^2(t)}{2}} \epsilon_{\sigma_t}$. Then given an adversarial sample $x_{rs} = x_0 + \delta$ with original instance $x_0$ and perturbation $\delta$, solving reverse-
**SDE** starting at time $t^*$ and point $x_{t^*} = \sqrt{\alpha_t} x_{t^*}$ until time 0 will generate a reversed random variable $x_0$ such that with a probability of at least $1 - \eta$, we have

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

(3)

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

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

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

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

$$\Pr( ||\mathbf{E}|| \geq \sqrt{d + 2\sqrt{d \log \frac{1}{\eta} + 2 \log \frac{1}{\eta}}} ) \leq \eta.$$ 

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

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

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

**Remark.** Theorem 1 implies that as long as $x_{t^*}$ is not far away from the corresponding original instance $x_0$, $\alpha_{t^*}$ 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 $\hat{x}_0$ 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 $C_\tau$ 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 $\ell_t(x, t)$ to predict the likely noise added to the loss function:

$$\ell_t(x, t) := \mathbb{E}_{x_0, \mathbf{E}} \left[ \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \alpha_t)} \left( \mathbf{E} - \ell_0 \left( \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t} \mathbf{E}, t \right) \right)^2 \right].$$

where $t \in [0, T]$, $x_0 \sim p$, and $\mathbf{E} \sim \mathcal{N}(0, I)$. For fast sampling in DDPM, the “one-shot” denoising (algorithm 2) was frequently used [10, 23] where we have

$$\hat{x}_0 = \left( \frac{x_{t^*}}{\sqrt{1 - \alpha_t} \ell_0 (x_{t^*}, t^*)} \right) / \sqrt{\alpha_t}.$$ 

(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 $t$:

$$\ell_t(x_t) := \mathbb{E}_{x_0, \mathbf{E}} \left[ \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \alpha_t)} \left( \mathbf{E} - \ell_0 \left( \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t} \mathbf{E}, t \right) \right)^2 \right].$$

where $x_0 \sim p$ and $\mathbf{E} \sim \mathcal{N}(0, I)$.

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

$$||\hat{x}_0 - \mathbb{E} (\hat{x}_0 | x_{t^*} = x_t)\|| \leq \frac{2\sigma_t^2 \alpha_t (1 - \alpha_t)^{3/2} \sqrt{\alpha_t}}{\beta_t^2} \cdot \ell_t(x_t)$$

(5)

where $\hat{x}_0, \hat{x}_t$ are random variables generated by reverse-SDE, $\mathbb{P} (\hat{x}_0 = x_0 | \hat{x}_t = x_t) \propto p(x) \cdot \frac{1}{\sqrt{2\pi \sigma_t^2}} \exp \left( -\frac{||x_t-x_t||^2}{2\sigma_t^2} \right)$ 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 $p \in C^2$ and $\mathbb{E}_{x \sim p}(||x||^2) < \infty$, the diffusion process by SDE and the reverse process by reverse-SDE follow the same distribution ideally. Therefore, $\mathbb{P} (\hat{x}_0 = x_0 | \hat{x}_t = x_t) = \mathbb{P} (x_0 = x | x_t = x_t) = p(x) \cdot \frac{1}{\sqrt{2\pi \sigma_t^2}} \exp \left( -\frac{||x_t-x_t||^2}{2\sigma_t^2} \right)$.

(5)

implies $\sqrt{\alpha_t}\hat{x}_0 + \sqrt{1 - \alpha_t} \mathbf{E} = x_t$, where $\mathbf{E} \sim \mathcal{N}(0, I)$, we have

$$\ell_t(x_t) = \mathbb{E}_{x_0, \mathbf{E}} \left[ \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \alpha_t)} \left( \mathbf{E} - \ell_0 \left( \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t} \mathbf{E}, t \right) \right)^2 \right].$$

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 $\mathbb{P} (\hat{x}_0 = x_0 | \hat{x}_t = x_t)$. In addition, this conditional distribution has a high density on points with high data density in the original distribution $p$ and close to $x_t$. 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 DiffSmoother

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 PURIFY CLASSIFIER($x_{rs}$; $\sigma'$, $m$).

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

Output: The prediction for $x_{rs}$.

1: $t^*, \hat{a}_t, \hat{\delta}_t \leftarrow \text{COMPUTETIMESTEP}(2\sigma)$
2: $\hat{x} \leftarrow \text{DENoise}((\sqrt{\alpha_t}(2x_{rs} - 1), t^*) + 1)/2$
3: $\hat{y} \leftarrow 0$
4: for $i = 1$ to $m$ do
5: \quad $\delta'_i \sim \mathcal{N}(0, \sigma'^2 I)$
6: \quad $\hat{y} \leftarrow \hat{y} + \frac{1}{m} F(\hat{x} + \delta'_i)$
7: end for
8: return $\arg\max_{y \in \mathcal{Y}} \hat{y}$

Algorithm 4 SAMPLE UNDER NOISE($f, x, n, \sigma$) [13].

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

Output: A vector of class counts.

1: $\text{counts} \leftarrow \{0, 0, ..., 0\}$
2: for $i = 1$ to $n$ do
3: \quad $x_{rs} \leftarrow x + \mathcal{N}(0, \sigma^2 I)$
4: \quad $y \leftarrow f(x_{rs})$
5: \quad $\text{counts}[y] + 1$
6: end for
7: return $\text{counts}$

Algorithm 5 Certification Process for DiffSmooth.

Input: The magnitude of the smoothing noise $\sigma$, the magnitude of the local smoothing noise $\sigma'$, 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.

1: $f \leftarrow \text{PURIFY CLASSIFIER}(\cdot; \sigma', m)$
2: $\text{counts}_0 \leftarrow \text{SAMPLE UNDER NOISE}(f, x, n_0, \sigma)$
3: $\hat{c}_A \leftarrow \text{top index in \text{counts}_0}$
4: $\text{counts} \leftarrow \text{SAMPLE UNDER NOISE}(f, x, n, \sigma)$
5: $p_A \leftarrow \text{LOWER CONF BOUND}(\text{counts}[\hat{c}_A], n, 1 - \alpha)$
6: if $p_A > \frac{1}{2}$ then
7: \quad return $\hat{c}_A$ and radius $\sigma \Phi^{-1}(p_A)$
8: else
9: \quad return $\text{ABSTAIN}$
10: end if

smoothed model, in order to further improve its certified robustness by improving its clear 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 $x_0 = \frac{1}{\sqrt{\alpha_t}}(x_t - \sqrt{1 - \alpha_t} e_0(x_t, t))$ 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 $x_{rs} = \sqrt{\alpha_t}(2x_{rs} - 1)$ and perform the one-shot reverse step where $t^*$ is the solution to equation $(2\sigma)^2 = 1/\alpha_{t^*}$.

Second, we perform the local smoothing for each purified instance $\hat{x}$; in other words, the local smoothing prediction is provided as $\arg\max_{y \in \mathcal{Y}} \sum_{i=1}^{m} F(DENoise(x_{rs} + \delta'_i)/m$ where $\delta'_i \sim \mathcal{N}(x, \sigma'^2 I)$ with $\sigma' \leq \sigma$ and $m$ is the number of sampled local smoothing noise. As shown in Figure 1, the local smoothing noise magnitude $\sigma'$ should be smaller than the smoothing noise level $\sigma$; in the meantime, it is also 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 $\sigma'$ to be close to the model smoothing level $\sigma$. Nevertheless, to certify the advance smoothed model trained with SmoothAdv [43], setting $\sigma'$ to be around 1/2 of the $\sigma$ 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 $f(\cdot) = \text{PURIFY CLASSIFIER}(\cdot; \sigma', m)$ which is shown in Algorithm 3. The whole certification process for DiffSmooth is provided in Algorithm 5, where the function SAMPLE UNDER NOISE is shown in Algorithm 4 and the LOWER CONF BOUND($k, n, 1 - \alpha$) is a function which returns a one-sided $(1 - \alpha)$ lower confidence bound $p$ 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 $\ell_2$ 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...
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 $\ell_2$ 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 $224 \times 224$ instead of $512 \times 512$ which is used in [10]; other training hyperparameters of the BEiT are set the same as the finetuning on standard classifiers and ImageNet-1K based on the self-supervised pretrained model.

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

**Baselines.** We consider eight state-of-the-art $\ell_2$ 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) 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 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.

### Table 1: Certified accuracy of ResNet-110 on CIFAR-10 under different $\ell_2$ 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Extra data</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.00</td>
</tr>
<tr>
<td>Gaussian [13]</td>
<td>×</td>
<td>75.0</td>
</tr>
<tr>
<td>SmoothAdv [43]</td>
<td>×</td>
<td>73.6</td>
</tr>
<tr>
<td>SmoothAdv [43]</td>
<td>✓</td>
<td>80.8</td>
</tr>
<tr>
<td>MACER [59]</td>
<td>×</td>
<td>81.0</td>
</tr>
<tr>
<td>Consistency [26]</td>
<td>×</td>
<td>77.8</td>
</tr>
<tr>
<td>SmoothMix [25]</td>
<td>×</td>
<td>77.1</td>
</tr>
<tr>
<td>Boosting [24]</td>
<td>×</td>
<td>83.4</td>
</tr>
<tr>
<td>DDS(Standard) [10]</td>
<td>✓</td>
<td>79.0</td>
</tr>
<tr>
<td>DDS(Smoothed) [10]</td>
<td>✓</td>
<td>79.8</td>
</tr>
<tr>
<td>DiffSmooth(Gaussian)</td>
<td>×</td>
<td>78.2</td>
</tr>
<tr>
<td>DiffSmooth(SmoothAdv)</td>
<td>✓</td>
<td>82.8</td>
</tr>
<tr>
<td>DiffSmooth(SmoothAdv)</td>
<td>✓</td>
<td>85.4</td>
</tr>
</tbody>
</table>

---

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 $\ell_2$ 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.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Method$^1$</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00 0.50 1.00 1.50 2.00 2.50 3.00</td>
<td></td>
</tr>
<tr>
<td>ResNet-50</td>
<td>Gaussian [13]</td>
<td>66.4 48.6 37.0 25.4 18.4 13.8 10.4</td>
</tr>
<tr>
<td></td>
<td>SmoothAdv [43]</td>
<td>66.6 52.6 42.2 34.6 25.2 21.4 18.8</td>
</tr>
<tr>
<td></td>
<td>MACER [59]</td>
<td>68.0 57.0 43.0 37.0 27.0 25.0 20.0</td>
</tr>
<tr>
<td></td>
<td>Consistency [26]</td>
<td>57.0 50.0 44.0 34.0 24.0 21.0 17.0</td>
</tr>
<tr>
<td></td>
<td>SmoothMix [25]</td>
<td>55.0 50.0 43.0 38.0 26.0 24.0 20.0</td>
</tr>
<tr>
<td></td>
<td>Boosting [24]$^2$</td>
<td>68.0 57.0 44.6 38.4 28.6 24.6 21.2</td>
</tr>
<tr>
<td></td>
<td>DDS(Standard) [10]$^3$</td>
<td>67.4 49.0 33.0 22.2 17.4 12.8 8.0</td>
</tr>
<tr>
<td></td>
<td>DDS(Smoothed) [10]$^4$</td>
<td>48.0 40.6 29.6 23.8 18.6 16.0 13.4</td>
</tr>
<tr>
<td></td>
<td>DiffSmooth(Gaussian)</td>
<td>66.2 57.8 44.2 36.8 28.6 25.0 19.8</td>
</tr>
<tr>
<td></td>
<td>DiffSmooth(SmoothAdv)</td>
<td>66.2 59.2 48.2 39.6 31.0 25.4 22.4</td>
</tr>
<tr>
<td>BEiT$^6$</td>
<td>Gaussian [13]</td>
<td>82.0 70.2 51.8 33.0 22.2 17.4 13.1</td>
</tr>
<tr>
<td></td>
<td>DDS(Standard) [10]</td>
<td>82.8 71.1 54.3 38.1 29.5 - 13.1</td>
</tr>
<tr>
<td></td>
<td>DDS(Smoothed) [10]</td>
<td>76.2 60.2 43.8 31.8 22.0 17.8 12.2</td>
</tr>
<tr>
<td></td>
<td>DiffSmooth(Gaussian)</td>
<td>83.8 77.2 63.2 53.0 37.6 31.4 24.8</td>
</tr>
</tbody>
</table>

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 $\sigma'$ is set to "−", it represents the standard randomized smoothing setting, indicating that local smoothing is required only when diffusion-based purification is performed.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$\sigma'$</th>
<th>ACR</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>- 0.534</td>
<td>65.0</td>
<td>54.4 41.4 32.0 23.0 15.2 9.4 5.4</td>
</tr>
<tr>
<td></td>
<td>0.12 0.538</td>
<td>65.0</td>
<td>54.2 42.8 33.0 23.4 15.6 10.0 5.0</td>
</tr>
<tr>
<td></td>
<td>0.25 0.537</td>
<td>64.2</td>
<td>54.2 41.0 33.2 24.2 16.8 10.4 6.2</td>
</tr>
<tr>
<td></td>
<td>0.50 0.433</td>
<td>44.8</td>
<td>39.0 33.8 26.0 20.4 15.0 11.6 6.0</td>
</tr>
<tr>
<td>ImageNet</td>
<td>- 0.640</td>
<td>56.8</td>
<td>51.2 45.2 41.8 37.0 31.4 24.6 0.0</td>
</tr>
<tr>
<td></td>
<td>0.25 0.333</td>
<td>32.0</td>
<td>27.4 23.8 21.6 18.6 15.0 11.8 0.0</td>
</tr>
<tr>
<td></td>
<td>0.50 0.021</td>
<td>2.2</td>
<td>1.8 1.6 1.4 1.2 0.8 0.8 0.0</td>
</tr>
</tbody>
</table>

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 $\sigma$ is selected in $\{0.25, 0.50, 1.00\}$ for all models. For each $\sigma$, 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] and SmoothAdv [43], 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,
Table 4: Certified accuracy of DiffSmooth on CIFAR-10 under different $\ell_2$ radii for smoothed models with noise level $\sigma$ and local smoothing with noise level $\sigma'$. 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.

<table>
<thead>
<tr>
<th>Methods</th>
<th>$\sigma$</th>
<th>$\sigma'$</th>
<th>ACR</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00</td>
<td>0.25</td>
<td>0.50</td>
<td>0.75</td>
</tr>
<tr>
<td>DiffSmooth (Gaussian)</td>
<td>0.25</td>
<td>0.543</td>
<td>0.556</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
<td>0.703</td>
<td>0.745</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>1.00</td>
<td>0.581</td>
<td>0.638</td>
<td>0.0</td>
</tr>
<tr>
<td>DiffSmooth (SmoothAdv)</td>
<td>0.25</td>
<td>0.593</td>
<td>0.572</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
<td>0.733</td>
<td>0.799</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>1.00</td>
<td>0.584</td>
<td>0.724</td>
<td>0.0</td>
</tr>
<tr>
<td>DiffSmooth (SmoothAdv w/ extra data)</td>
<td>0.25</td>
<td>0.624</td>
<td>0.622</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
<td>0.771</td>
<td>0.830</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>1.00</td>
<td>0.623</td>
<td>0.709</td>
<td>0.0</td>
</tr>
</tbody>
</table>

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 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 magnitude $\sigma'$ of 0.03 under all smoothing level $\sigma$; and we add the local smoothing noise with magnitude $\sigma'$ of 0.01 when $\sigma = 0.25$ while adding local smoothing noise with magnitude $\sigma'$ of 0.02 for other $\sigma$ 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 $\ell_2$ 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 $\sigma$. 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 $\ell_2$ radius 0.50, and from 23.0% to 37.4% under larger $\ell_2$ 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 $\ell_2$ 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%
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 $\ell_2$ 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 $\ell_2$ 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 $\sigma'$ 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 $\sigma'$. 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 $\sigma'$, the performance will be further improved. In addition, we find that to certify a small radius, smaller $\sigma'$ is preferred; however, for certifying a large radius, the choice of $\sigma'$ depends on the model resilience to random noise, i.e., for Gaussian smoothed models, $\sigma'$ needs to be close to $\sigma$ while for SmoothAdv $\sigma'$ 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

### Table 5: Certified accuracy of DiffSmooth on ImageNet under different $\ell_2$ radii for smoothed models with noise level $\sigma$ and local smoothing with noise level $\sigma'$. 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.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Methods</th>
<th>$\sigma$</th>
<th>$\sigma'$</th>
<th>ACR</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.25</td>
<td>0.25</td>
<td></td>
<td>0.00</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>DiffSmooth (Gaussian)</td>
<td>0.25</td>
<td>0.25</td>
<td>0.467</td>
<td>66.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.25</td>
<td>0.710</td>
<td>57.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.50</td>
<td>0.741</td>
<td>55.8</td>
</tr>
<tr>
<td></td>
<td>DiffSmooth (SmoothAdv)</td>
<td>1.00</td>
<td>0.25</td>
<td>0.809</td>
<td>43.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>1.00</td>
<td>1.008</td>
<td>48.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.00</td>
<td>1.00</td>
<td>1.013</td>
<td>43.6</td>
</tr>
<tr>
<td>BEiT</td>
<td>DiffSmooth (Gaussian)</td>
<td>0.25</td>
<td>0.25</td>
<td>0.478</td>
<td>66.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.25</td>
<td>0.792</td>
<td>59.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.50</td>
<td>0.741</td>
<td>54.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.00</td>
<td>0.25</td>
<td>1.000</td>
<td>48.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>1.00</td>
<td>1.087</td>
<td>47.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.00</td>
<td>1.00</td>
<td>0.937</td>
<td>37.8</td>
</tr>
<tr>
<td></td>
<td>DiffSmooth (SmoothAdv)</td>
<td>0.25</td>
<td>0.12</td>
<td>0.623</td>
<td>83.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25</td>
<td>0.25</td>
<td>0.618</td>
<td>82.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.12</td>
<td>1.044</td>
<td>79.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.25</td>
<td>1.061</td>
<td>79.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>0.50</td>
<td>1.023</td>
<td>73.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.00</td>
<td>0.12</td>
<td>1.216</td>
<td>62.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25</td>
<td>1.282</td>
<td>62.0</td>
<td>57.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
<td>1.333</td>
<td>61.4</td>
<td>55.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.00</td>
<td>1.214</td>
<td>50.8</td>
<td>47.4</td>
</tr>
</tbody>
</table>
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
We first provide theoretical analysis to show that the recovered words, we will maintain a neighborhood of the corresponding original instance with instances from (adversarial) inputs will be in the bounded to provide improved certified robustness for.

In this work, we aim to leverage diffusion-based purification that 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.

### 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 pipepline, 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.

### Table 6: Certified accuracy of ResNet-110 on CIFAR-10 under different $\ell_2$ radii with the number of predictions as 100,000.

<table>
<thead>
<tr>
<th>Method</th>
<th>Setting</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius r</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.00 0.25 0.50 0.75 1.00 1.50 1.75 2.00</td>
</tr>
<tr>
<td>DDS(Standard)</td>
<td>$N = 100,000$</td>
<td>79.0 62.0 45.8 32.6 25.0 17.6 11.0 6.2</td>
</tr>
<tr>
<td>DDS(Smoothed)</td>
<td>$N = 100,000$</td>
<td>79.8 69.9 55.0 47.6 37.4 32.4 28.6 24.8</td>
</tr>
<tr>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 20,000, m = 5$</td>
<td>77.2 67.4 55.6 44.4 35.0 29.4 21.8 18.4</td>
</tr>
<tr>
<td></td>
<td>$N = 10,000, m = 10$</td>
<td>76.8 66.0 56.6 42.8 36.0 29.0 23.6 18.2</td>
</tr>
<tr>
<td></td>
<td>$N = 5,000, m = 20$</td>
<td>77.2 66.8 58.2 43.8 36.6 29.4 22.0 18.0</td>
</tr>
<tr>
<td>DiffSmooth(SmoothAdv)</td>
<td>$N = 20,000, m = 5$</td>
<td>82.2 71.6 62.8 49.2 39.8 35.2 29.8 24.0</td>
</tr>
<tr>
<td></td>
<td>$N = 10,000, m = 10$</td>
<td>82.8 71.0 62.4 48.4 40.0 35.4 29.6 24.6</td>
</tr>
<tr>
<td></td>
<td>$N = 5,000, m = 20$</td>
<td>82.6 71.8 61.8 47.4 40.4 34.4 27.2 24.2</td>
</tr>
<tr>
<td>DiffSmooth(SmoothAdv) with extra data</td>
<td>$N = 20,000, m = 5$</td>
<td><strong>86.0</strong> 75.8 <strong>65.6</strong> <strong>54.0</strong> <strong>41.8</strong> 35.6 <strong>30.2</strong> 23.8 <strong>22.2</strong></td>
</tr>
<tr>
<td></td>
<td>$N = 10,000, m = 10$</td>
<td>85.2 <strong>76.0</strong> 64.2 53.8 <strong>41.8</strong> <strong>36.0</strong> 28.8 <strong>24.4</strong> 21.4</td>
</tr>
<tr>
<td></td>
<td>$N = 5,000, m = 20$</td>
<td>85.2 <strong>76.0</strong> 64.8 49.2 41.4 <strong>34.4</strong> 26.6 23.0 20.6</td>
</tr>
</tbody>
</table>

### Table 7: Certified accuracy on ImageNet under different $\ell_2$ radii with the number of predictions as 10,000.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Method</th>
<th>Setting</th>
<th>Certified Accuracy (%) under $\ell_2$ Radius r</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>DDS(Standard)</td>
<td>$N = 10,000$</td>
<td><strong>67.4</strong> 49.0 33.0 22.2 17.4 <strong>12.8</strong></td>
</tr>
<tr>
<td>DNS</td>
<td>DDS(Smoothed)</td>
<td>$N = 10,000$</td>
<td>48.0 40.6 29.6 23.8 18.6 <strong>16.0</strong></td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 2,000, m = 5$</td>
<td>65.4 54.8 42.4 30.2 26.8 <strong>21.0</strong></td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 1,000, m = 10$</td>
<td>65.8 55.2 42.4 30.6 <strong>27.6</strong> -</td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 500, m = 20$</td>
<td>65.4 53.8 41.8 30.6 <strong>25.4</strong> -</td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(SmoothAdv)</td>
<td>$N = 20,000, m = 5$</td>
<td>64.0 <strong>57.6</strong> 46.4 <strong>33.8</strong> <strong>28.6</strong> <strong>23.4</strong></td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(SmoothAdv)</td>
<td>$N = 1,000, m = 10$</td>
<td>64.6 57.2 46.0 32.8 <strong>27.8</strong> -</td>
</tr>
<tr>
<td>DNS</td>
<td>DiffSmooth(SmoothAdv)</td>
<td>$N = 500, m = 20$</td>
<td>65.0 56.4 45.2 32.4 <strong>26.6</strong> -</td>
</tr>
<tr>
<td>BEiT</td>
<td>DDS(Standard)</td>
<td>$N = 10,000$</td>
<td>82.8 71.1 54.3 38.1 29.5 -</td>
</tr>
<tr>
<td>BEiT</td>
<td>DDS(Smoothed)</td>
<td>$N = 10,000$</td>
<td>76.2 60.2 43.8 31.8 22.0 <strong>17.8</strong></td>
</tr>
<tr>
<td>BEiT</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 2,000, m = 5$</td>
<td>83.0 75.6 60.0 40.1 <strong>34.9</strong> <strong>25.7</strong></td>
</tr>
<tr>
<td>BEiT</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 1,000, m = 10$</td>
<td><strong>83.2</strong> <strong>76.2</strong> <strong>60.6</strong> <strong>40.3</strong> 34.3 -</td>
</tr>
<tr>
<td>BEiT</td>
<td>DiffSmooth(Gaussian)</td>
<td>$N = 500, m = 20$</td>
<td>83.4 75.0 59.6 <strong>40.3</strong> <strong>31.9</strong> -</td>
</tr>
</tbody>
</table>
References


A Theorems And Proofs

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

$$||\hat{x}_0 - x_0|| \leq ||x_{t^*} - x_0|| + \sqrt{2 \alpha^2(t^*) - 1 C \eta + \gamma(t^*) C}$$

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

Proof. We leverage the proof of [35, Theorem 3.2]. By reverse-SDE, we can bound $||\hat{x}(0) - x_0|| = ||\hat{x}(t^*) + \hat{x}(0) - x(t^*) - x_0||$

$$= \left|\int_{t^*}^0 \frac{1}{2} \gamma(t) \left[ x(t) + 2 \nabla_x \log p_t(x(t)) \right] dt + \int_{t^*}^0 \sqrt{\gamma(t)} \nabla_x \log p_t(x(t)) dt \right|$$

$$\leq \left|\int_{t^*}^0 \frac{1}{2} \gamma(t) \nabla_x \log p_t(x(t)) dt \right| + \left|\int_{t^*}^0 \gamma(t) \nabla_x \log p_t(x(t)) dt \right|$$

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 $\nabla_x \log p_t(x(t))$ 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 $\hat{x}'(0)$ its solution, from [47] we know $\hat{x}'(0)$ follows a Gaussian distribution, where its mean $\mu(0)$ and covariance matrix $\Sigma(0)$ are the solutions of the following two differential equations, respectively:

$$\frac{d\mu}{dt} = -\frac{1}{2} \gamma(t) \mu$$

$$\frac{d\Sigma}{dt} = -\gamma(t) \Sigma + \gamma(t) I_d$$

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

Using the reparameterization trick, let $\epsilon \sim \mathcal{N}(0, I_d)$, we have:

$$\hat{x}'(0) - x_0 = e^{\gamma(t^*)} x(t^*) + \sqrt{e^{2 \gamma(t^*)} - 1} \epsilon - x_0$$

$$= x_{t^*} + \sqrt{e^{2 \gamma(t^*)} - 1} \epsilon - x_0$$

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

$$||\hat{x}_0 - x_0|| \leq ||\sqrt{e^{\gamma(t^*)} - 1} \epsilon + x_{t^*} - x_0|| + \gamma(t^*) C$$

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

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

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

$$||\hat{x}_0 - x_0|| \leq ||x_{t^*} - x_0|| + \sqrt{2 \alpha^2(t^*) - 1 C \eta + \gamma(t^*) C}.$$  

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

$$||\hat{x}_0 - x_0|| \leq \left\| 2 \sigma^2 \alpha \left(1 - \frac{\sigma^2}{2} \right)^{3/2} \right\| \right\| \epsilon_{t^*} - x_{t^*} \right\|$$

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

Proof. Given time $t^*$ and point $x_{t^*} = \sqrt{\alpha \cdot x_{t^*}}$ is equivalent to that in the formula of $\epsilon_{t^*}(x_{t^*})$, given $\alpha \cdot x_0 + \sqrt{1 - \alpha \cdot \epsilon} = x_{t^*}$. Then the conditional distribution of $x_0$ will be $\mathbb{P}(x_0 = x|x_{t^*} = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2 \pi \sigma^2)}} \exp \left( \frac{-||x-x_{t^*}||^2}{2 \sigma^2} \right)$. By the assumptions that $p \in C^2$ and $E_{x \sim p}[||x||_2^2] < \infty$, we know the diffusion process by SDE and the reverse process by reverse-SDE follows the same distribution, thus we also have $\mathbb{P}(\hat{x}_0 = x|\hat{x}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2 \pi \sigma^2)}} \exp \left( \frac{-||x-x_{t^*}||^2}{2 \sigma^2} \right)$.

Further note that

$$\epsilon_{t^*}(x_{t^*}) = E_{\hat{x}_0 \cdot \epsilon} \left[ \frac{\beta^2}{2 \sigma^2 \alpha (1 - \alpha \cdot \epsilon)} \right] ||\epsilon - \epsilon_{t^*}(x_{t^*})||^2$$

$$\sqrt{\alpha \cdot x_0 + 1 - \alpha \cdot \epsilon} \epsilon_{t^*}$$

$$= E_{\hat{x}_0 \cdot \epsilon} \left[ \frac{\beta^2}{2 \sigma^2 \alpha (1 - \alpha \cdot \epsilon)} \right] ||\epsilon - \epsilon_{t^*}(x_{t^*})||^2$$

$$x_{t^*} = x_{t^*}$$

$$\Rightarrow \epsilon_{t^*} = x_{t^*}.$$  

Under the condition $\sqrt{\alpha \cdot x_0 + 1 - \alpha \cdot \epsilon} \epsilon_{t^*} = x_{t^*}$, which is equivalent to that $\sqrt{\alpha \cdot x_0 + 1 - \alpha \cdot \epsilon} \epsilon_{t^*} = x_{t^*}$, where $\epsilon \sim \mathcal{N}(0, I_1)$, there is a one-to-one corresponding between $\hat{x}_0$ and $\epsilon$, and

$$||\epsilon - \epsilon_{t^*}(x_{t^*})||^2 = \left\| x_{t^*} - \sqrt{\alpha \cdot x_0} \right\| x_{t^*} - \sqrt{\alpha \cdot \epsilon_{t^*}(x_{t^*})} \right\| \frac{1}{\sqrt{1 - \alpha \cdot \epsilon} \epsilon_{t^*}}$$

$$= ||\hat{x}_0 - x_0|| \cdot \frac{\alpha \cdot \epsilon_{t^*} x_{t^*}}{\sqrt{1 - \alpha \cdot \epsilon}}.$$
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 $\varepsilon$ 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 $1e - 8$.

### 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.