Consistency Purification: Effective and Efficient Diffusion Purification towards Certified Robustness

Yiquan Li[1*] **Zhongzhu Chen**[2*] **Kun Jin**[2*] **Jiongxiao Wang**[1*] **Jiachen Lei**[3]
Bo Li[4] **Chaowei Xiao**[1]
[1]University of Wisconsin-Madison; [2] University of Michigan-Ann Arbor;
[3]California Institute of Technology ;[4]University of Illinois Urbana-Champaign

Abstract

Diffusion Purification, purifying noised images with diffusion models, has been widely used for enhancing certified robustness via randomized smoothing. However, existing frameworks often grapple with the balance between efficiency and effectiveness. While the Denoising Diffusion Probabilistic Model (DDPM) offers an efficient single-step purification, it falls short in ensuring purified images reside on the data manifold. Conversely, the Stochastic Diffusion Model effectively places purified images on the data manifold but demands solving cumbersome stochastic differential equations, while its derivative, the Probability Flow Ordinary Differential Equation (PF-ODE), though solving simpler ordinary differential equations, still requires multiple computational steps. In this work, we demonstrated that an ideal purification pipeline should generate the purified images on the data manifold that are as much semantically aligned to the original images for effectiveness in one step for efficiency. Therefore, we introduced Consistency Purification, an efficiency-effectiveness Pareto superior purifier compared to the previous work. Consistency Purification employs the consistency model, a one-step generative model distilled from PF-ODE, thus can generate on-manifold purified images with a single network evaluation. However, the consistency model is designed not for purification thus it does not inherently ensure semantic alignment between purified and original images. To resolve this issue, we further refine it through Consistency Fine-tuning with LPIPS loss, which enables more aligned semantic meaning while keeping the purified images on data manifold. Our comprehensive experiments demonstrate that our Consistency Purification framework achieves state-of-the-art certified robustness and efficiency compared to baseline methods.

1 Introduction

Diffusion models were first proposed for high-quality image generation [1, 2, 3, 4, 5] and have been extended to generative tasks across various modalities, including audio [6, 7, 8], video [9, 10], and 3D object [11, 12, 13]. A diffusion model for image generation typically involves two key processes: (1) a forward diffusion process, which transforms the source image into an isotropic Gaussian by gradually adding Gaussian noise, and (2) the reverse diffusion process, which uses a Deep Neural Network (DNN) to perform iterative denoising starting from random Gaussian noise.

Due to the inherent denoising capability of diffusion models, there have been widely applied to improve the robustness of DNNs. This enhancement is achieved by Diffusion Purification [14, 15, 16, 17, 18], which purifies the network inputs to reduce the effects of various types of unforeseen corruptions or adversarial attacks. Among these, one particularly suitable and effective scenario of purification is to improve certified robustness through randomized smoothing [19] for image

*The first four authors contributed equally. Correspondence to: Jiongxiao Wang <jwang2929@wisc.edu>.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3335

classification tasks. This method guarantees a tight robustness in the ℓ_2 norm with a smoothed classifier. However, many previous works [19, 20, 21, 22, 23, 24] have shown that it still requires retraining with Gaussian augmented examples for each noise level to optimize the smoothed classifier. Diffusion models, capable of purifying Gaussian perturbed images before classification, can be seamlessly integrated with any base classifier to produce a smoothed classifier for arbitrary noise levels. This integration has been demonstrated to effectively enhance certified robustness, as supported by numerous studies [18, 25, 26, 27].

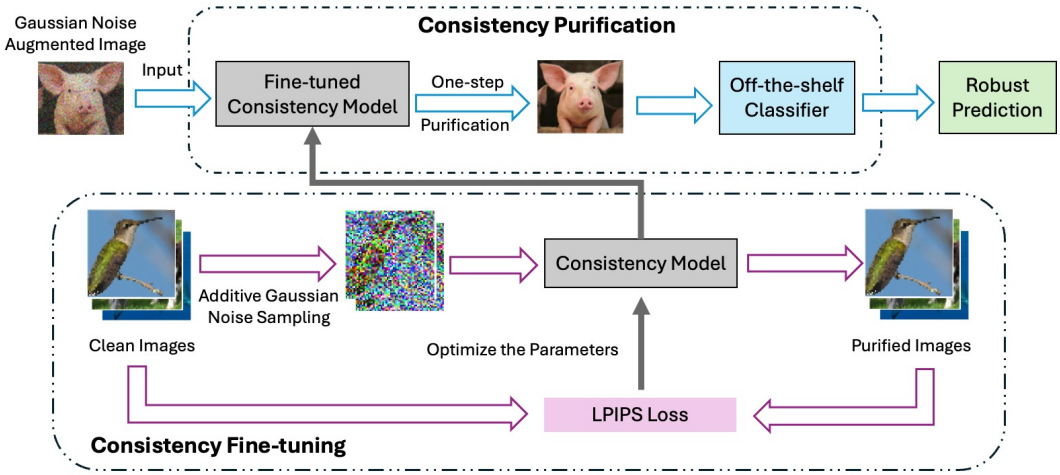


Figure 1: An illustration of Consistency Purification framework.

However, current diffusion purification for certified robustness via randomized smoothing still faces significant trade-offs between *efficiency* and *effectiveness*. Although Denoising Diffusion Probabilistic Model (DDPM) [28] only requires one single network evaluation in the purification process [25], it generates the mean of the posterior data distribution conditioned the noisy sample, which does not necessarily locate on the data manifold and may exhibit ambiguity during classification. To further improve diffusion purification, various methods such as DensePure [26], Local Smoothing [27] and Noised Diffusion Classifiers [29] are applied. However, these methods are considerably less efficient as they require multiple times of the computational costs compared to one-step DDPM. Another promising approach involves using the Probability Flow Ordinary Differential Equation (PF-ODE) [3]. It has offered a method to accelerate the sampling process [4] and achieved a closer distribution to the original data, well balancing efficiency and effectiveness. However, several computational steps are still needed to solve the ODE numerically.

To find a Pareto superior solution in terms of efficiency and effectiveness, we introduce a new framework, **Consistency Purification**, which integrating consistency models into diffusion purification with **Consistency Fine-tuning**. The consistency model is a novel category of diffusion models that learns the trajectory of the PF-ODE that transits the data distribution to the noisy distribution. It is trained to map any point along this trajectory back to its starting point. This property is desirable for diffusion purification, as it allows images with any scale of Gaussian noise to be directly purified to the clean images. Distilled from a pre-trained diffusion model by simulating the PF-ODE trajectory, the consistency model can generate high-quality in-distribution images in a single step, thereby ensuring both efficiency and effectiveness. However, since consistency models are primarily trained for image generation, it may not suffice to guarantee that the purified image that maintains the same semantic meaning as the original image. To address this issue, we propose adding a Consistency Fine-tuning step into the purification framework, which further fine-tunes the consistency model using Learned Perceptual Image Patch Similarity (LPIPS) [30] loss, aiming to minimize the perceptual differences between the purified and original images, thereby ensuring better semantic alignment, while at the same time, ensuring the purified images still lie on the data manifold.

We show that Consistency Purification is Pareto superior compared to baselines from two aspects. First of all, compared with effective methods like DensePure [26], Local Smoothing [27] and Noised Diffusion Classifiers [29], Consistency Purification is much more efficient since it enables single-step purification. Secondly, compared with efficient method like onestep-DDPM [25], we provide both

theoretical analysis and experiment results to support the effectiveness improvement of Consistency Purification. In Example 3.1, we show an one-dimensional example demonstrating that consistency model can generate on-manifold purified samples while onestep-DDPM does not have this property.

In Theorem 3.3, we show an important theoretical result that given a purifier, the lower the transport from the original distribution to the purified distribution (a measure of distance between probability distributions, see [31]), the higher the probability that the purified sample is sufficiently close to the original sample, and thus the better purification outcomes. Our experiment results verify that both the integration of consistency model in Consistency Purification and the further Consistency Fine-tuning decreases such transport and achieves better semantic alignment between purified samples and original samples.

Beyond the validation of our theory, we conduct comprehensive experiments to demonstrate the empirical improvements of Consistency Purification. Compared to various baseline settings, our approach has shown significant improvements, achieving an average 5% gain in performance over the previous onestep-DDPM under the same cost with single-step purification. These observations underscore our success in finding a Pareto superior diffusion purification framework in both efficiency and effectiveness for certified robustness.

2 Backgrounds

Randomized Smoothing [19]. Randomized smoothing is designed to certify the robustness of a given classifier under ℓ_2 norm perturbations. Given a base classifier f and an input $\boldsymbol{x}$, randomized smoothing first defines the smoothed classifier by $g(\boldsymbol{x}) = \arg\max_c \mathbb{P}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})}(f(\boldsymbol{x}+\boldsymbol{\epsilon})=c)$, where σ is the noise level, which controls the trade-off between robustness and accuracy. [19] shows that $g(\boldsymbol{x})$ induces the certifiable robustness for $\boldsymbol{x}$ under the ℓ_2 norm with radius R, where $R = \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right)$, where p_A and p_B are the probability of the most probable class and "runner-up" class respectively; Φ is the inverse of the standard Gaussian CDF. The p_A and p_B can be estimated with arbitrarily high confidence via the Monte Carlo method.

Continuous-Time Diffusion Model [3]. The diffusion model has two components: the *diffusion process* followed by the *reverse process*. Given an input random variable $\boldsymbol{x}_0 \sim p$, the diffusion process adds isotropic Gaussian noises to the data so that the diffused random variable at time t is $\boldsymbol{x}_t = \sqrt{\alpha_t}(\boldsymbol{x}_0 + \boldsymbol{\epsilon}_t)$, s.t., $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma_t^2\boldsymbol{I})$, and $\sigma_t^2 = (1-\alpha_t)/\alpha_t$, and we denote $\boldsymbol{x}_t \sim p_t$. The forward diffusion process can also be defined by the stochastic differential equation

$$\mathrm{d}\boldsymbol{x} = D(\boldsymbol{x}, t)\mathrm{d}t + G(t)\mathrm{d}\boldsymbol{w}, \tag{SDE}$$

where $\boldsymbol{x}_0 \sim p$, $D : \mathbb{R}^d \times \mathbb{R} \mapsto \mathbb{R}^d$ is the drift coefficient and typically has the form $D(\boldsymbol{x}, t) = D(t)\boldsymbol{x}$. $G : \mathbb{R} \mapsto \mathbb{R}$ is the diffusion coefficient, $\mathrm{d}t$ is an infinitesimal time step, and $\boldsymbol{w}(t) \in \mathbb{R}^n$ is the standard Wiener process.

The reverse process exists and removes the added noise by solving the reverse-time SDE [32]

$$\mathrm{d}\boldsymbol{x} = [D(t)\boldsymbol{x} - G(t)^2\nabla_{\hat{\boldsymbol{x}}}\log p_t(\boldsymbol{x})]\mathrm{d}t + G(t)\mathrm{d}\overline{\boldsymbol{w}}, \tag{reverse-SDE}$$

where $p_t(\boldsymbol{x})$ denotes the marginal distribution at time t, and $\overline{\boldsymbol{w}}(t)$ is a reverse-time standard Wiener process. [3] defined the probability flow ODE (PF ODE) which has the same marginal distribution as reverse-SDE but can be solved much faster

$$\mathrm{d}\boldsymbol{x} = \left[D(t)\boldsymbol{x} - \tfrac{1}{2}G(t)^2\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})\right]\mathrm{d}t. \tag{PF-ODE}$$

As shown in [4], the perturbation kernel of SDE has the general form

$$p_{0t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(0)) = \mathcal{N}\left(\boldsymbol{x}(t); s(t)\boldsymbol{x}(0), s(t)^2\sigma(t)^2\mathbf{I}\right) \tag{perturbation-kernel}$$

where $s(t) = \exp\left(\int_0^t f(\xi)\mathrm{d}\xi\right)$ and $\sigma(t) = \sqrt{\int_0^t \frac{g(\xi)^2}{s(\xi)^2}\,\mathrm{d}\xi}$. Under this formulation, PF-ODE can written as

$$\mathrm{d}\boldsymbol{x} = \left[\tfrac{\dot{s}(t)}{s(t)}\boldsymbol{x} - s(t)^2\dot{\sigma}(t)\sigma(t)\nabla_{\boldsymbol{x}}\log p\left(\tfrac{\boldsymbol{x}}{s(t)};\sigma(t)\right)\right]\mathrm{d}t$$

where $\cdot$ denotes the time derivative and $p\left(\frac{\boldsymbol{x}}{s(t)};\sigma(t)\right)$ denotes the marginal distribution at time t. In our context, we use the *EDM* parameter [4] where $s(t) = 1$ and $\sigma(t) = t$ which gives us a probability flow ODE

$$\mathrm{d}\boldsymbol{x} = -t\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})\mathrm{d}t. \tag{EDM-ODE}$$

 https://doi.org/10.52202/079017-3335

We use $\{\boldsymbol{x}_t\}_{t\in[0,1]}$ and $\{\hat{\boldsymbol{x}}_t\}_{t\in[0,1]}$ to denote the diffusion process and the reverse process generated by SDE and reverse-SDE respectively, which follow the same distribution. We also use $\{\tilde{\boldsymbol{x}}_t\}_{t\in[0,1]}$ to denote the reverse process generated by PF-ODE, which has the same marginal distribution as $\{\boldsymbol{x}_t\}_{t\in[0,1]}$ and $\{\hat{\boldsymbol{x}}_t\}_{t\in[0,1]}$ given t.

Discrete-Time Diffusion Model (DDPM [28]). DDPM constructs a discrete Markov chain $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_i, \cdots, \boldsymbol{x}_N\}$ as the forward process for the training data $\boldsymbol{x}_0 \sim p$, such that $\mathbb{P}(\boldsymbol{x}_i|\boldsymbol{x}_{i-1}) = \mathcal{N}(\boldsymbol{x}_i; \sqrt{1-\beta_i}\boldsymbol{x}_{i-1}, \beta_i I)$, where $0 < \beta_1 < \beta_2 < \cdots < \beta_N < 1$ are predefined noise scales such that $\boldsymbol{x}_N$ approximates the Gaussian white noise. Denote $\overline{\alpha}_i = \prod_{i=1}^{N}(1-\beta_i)$, we have $\mathbb{P}(\boldsymbol{x}_i|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_i; \sqrt{\overline{\alpha}_i}\boldsymbol{x}_0, (1-\overline{\alpha}_i)\boldsymbol{I})$, i.e., $\boldsymbol{x}_t(\boldsymbol{x}_0, \epsilon) = \sqrt{\overline{\alpha}_i}\boldsymbol{x}_0 + (1-\overline{\alpha}_i)\boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

The reverse process of DDPM learns a reverse direction variational Markov chain $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1}|\boldsymbol{x}_i) = \mathcal{N}(\boldsymbol{x}_{i-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i), \Sigma_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i))$. [28] defines $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ as a function approximator to predict $\boldsymbol{\epsilon}$ from $\boldsymbol{x}_i$ such that $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i) = \frac{1}{\sqrt{1-\beta_i}}\left(\boldsymbol{x}_i - \frac{\beta_i}{\sqrt{1-\overline{\alpha}_i}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i)\right)$. Then the reverse time samples are generated by $\hat{\boldsymbol{x}}_{i-1} = \frac{1}{\sqrt{1-\beta_i}}\left(\hat{\boldsymbol{x}}_i - \frac{\beta_i}{\sqrt{1-\overline{\alpha}_i}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}(\hat{\boldsymbol{x}}_i, i)\right) + \sqrt{\beta_i}\boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, I)$, and the optimal parameters $\boldsymbol{\theta}^*$ are obtained by solving $\boldsymbol{\theta}^* := \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\sqrt{\overline{\alpha}_i}\boldsymbol{x}_0 + (1-\overline{\alpha}_i), i)\|_2^2\right]$. [28] also provided a one-step approximate reconstruction of $\boldsymbol{x}_0$ from any $\boldsymbol{x}_t$,

$$\boldsymbol{x}_0 \approx \hat{\boldsymbol{x}}_0 = \left(\boldsymbol{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t)\right)/\sqrt{\overline{\alpha}_t}. \tag{onestep-DDPM}$$

Consistency Model [33]. Given a solution trajectory of PF-ODE, the consistency model is defined as $D : (\boldsymbol{x}_t, t) \mapsto \boldsymbol{x}_\epsilon$. The model exhibits the property of self-consistency, ensuring that its outputs are consistent for arbitrary pairs of $(\boldsymbol{x}_t, t)$ from the same PF-ODE trajectory; specifically, $D(\boldsymbol{x}_t, t) = D(\boldsymbol{x}_{t'}, t')$ for all $t, t' \in [\epsilon, T]$. As shown by the definition, consistency models are suitable for one-shot denoising, allowing for the recovery of $\boldsymbol{x}_\epsilon$ from any noisy input $\boldsymbol{x}_t$ in one network evaluation. Two distinct training strategies can be employed for training the consistency models: distillation mode and isolation mode. The primary distinction lies in whether the models distill the knowledge from pre-trained diffusion models or train from initial parameters. According to the experiments reported in [33], consistency models trained in the distillation mode have been shown to outperform those trained in isolation mode for generating high-quality images. Consequently, our paper only considers consistency models trained in the distillation mode.

3 Theoretical Analysis

In this section, we provide theoretical explanations on the advantages of Consistency Purification, with a focus on its purification performance improvement in terms of certified robustness over [25].

As demonstrated in [3], PF-ODE maintains the marginal distribution of reverse-SDE, thereby establishing a deterministic mapping between the noisy distribution $\boldsymbol{x}_t$ and the data distribution $\boldsymbol{x}_0$. In other words, PF-ODE guarantees that the purified sample lies on the data manifold, unlike onestep-DDPM, which lacks this assurance. We present here a simple one dimensional example for illustration.

Example 3.1. *Consider a one-dimensional space with a data set consisting of two samples $\{\boldsymbol{y}_1, \boldsymbol{y}_2\}$, where $\boldsymbol{y}_1 = 1$ and $\boldsymbol{y}_2 = -1$. The distribution can be represented as a mixture of Dirac delta distributions: $p_{data}(\boldsymbol{x}) = \frac{1}{2}(\delta(\boldsymbol{x} - \boldsymbol{y}_1) + \delta(\boldsymbol{x} - \boldsymbol{y}_2))$. By setting $s(t) = 1$ and $\sigma(t) = t$ in perturbation-kernel, the distribution at time t becomes: $p_t(\boldsymbol{x}) = \frac{1}{2t\sqrt{2\pi}}\left(e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t}\right)^2} + e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t}\right)^2}\right)$. Then*

$$\begin{aligned}\frac{\mathrm{d}\log p_t(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} &= \frac{-\left(\frac{\boldsymbol{x}-1}{t}\right)e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t^2}\right)^2} - \left(\frac{\boldsymbol{x}+1}{t}\right)e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t^2}\right)^2}}{2t\sqrt{2\pi}p_t(\boldsymbol{x})} \\ &= -\frac{\boldsymbol{x}}{t^2} + \frac{e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t}\right)^2} - e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t}\right)^2}}{e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t}\right)^2} + e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t}\right)^2}}.\end{aligned}$$

From the derivative formula $\frac{\mathrm{d}\log p_t(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}}$, it's evident that $\boldsymbol{x} = 0$ is an equilibrium point, and the right-hand side expression is Lipschitz continuous around $\boldsymbol{x} = 0$ by L'Hôpital's rule. Thus, according to the Picard-Lindelöf theorem, any trajectory starting on either side of $\boldsymbol{x} = 0$ will not cross this point. As PF-ODE drives $p_t(\boldsymbol{x})$ closer to the Dirac delta distribution $p_{data}(\boldsymbol{x})$ as t approaches zero, any initial point on positive/negative side of $\boldsymbol{x} = 0$ will eventually approach 1 or -1, i.e., the data manifold. Furthermore, in this example, PF-ODE generates not only a purified sample on the data

manifold but also closest to the noisy sample. This property is desirable as it establishes a relatively large "robust" neighborhood around each true data point, which implies high certified robustness and a significant certified radius, which will be further discussed later. With the consistency model, we do not need to solve the ODE but rather directly map the noisy sample to either $1/-1$ *depending on its location relative to* $\boldsymbol{x}=0$.

For comparison, given any $\boldsymbol{x}$ *and* t*, the onestep-DDPM will output a posterior mean that is*

$$\frac{e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t}\right)^2}-e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t}\right)^2}}{e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-1}{t}\right)^2}+e^{-\frac{1}{2}\left(\frac{\boldsymbol{x}+1}{t}\right)^2}}=\frac{e^{\frac{2\boldsymbol{x}}{t^2}}-1}{e^{\frac{2\boldsymbol{x}}{t^2}}+1}.$$

The posterior mean will be near 1 *or* -1 *only when* t *is sufficiently small compared to* $\|\boldsymbol{x}\|$*. Otherwise, it deviates from the data manifold. In the case when* t *is large, the posterior mean will be close to zero, locating in an ambiguous classification region. In adversarial purification [25, 26, 14], we typically select* t *based on the variance of the noise added to the data sample rather than using an very small* t*. This practice helps avoid significant deviations in the posterior mean estimation due to the imperfect estimation of score/noise. With a very small* t*, even a slight bias in score/noise estimation can lead to a substantial deviation, resulting in a denoised sample even farther from the data manifold represented by* $p_{data}(\boldsymbol{x})$.

Additionally, PF-ODE is deterministic, eliminating the overhead of majority voting required when using reverse-SDE as a purifier [26]. The consistency model, which reduces ODE solving to a one-step mapping, further ensures purification has the same efficiency as onestep-DDPM while keeping the in-distribution property.

Though the consistency model enjoys both in-distribution property and one-step efficiency, it does not guarantee that the purified sample has the same semantic meaning as the original sample. This is because the derivation of PF-ODE only guarantees a mapping between noisy distribution and data distribution, which is sufficient for generation, but not enough for denoising purposes.

To address this concern, we first delineate the desired characteristics of the purifier. As evidenced in prior works [14, 25, 26, 34], an ideal purifier should yield a purified output situated within a proximate vicinity of the original input. It is generally presumed that such purified outputs retain the semantic meaning of the original inputs with a high probability. The disparity in semantic consistency between the noisy input and the purified output generated by PF-ODE arises due to the proximity of the purified output to other samples. In this regard, we propose quantifying this disparity through the notion of transport between the data distribution and the purified distribution, derived by introducing Gaussian perturbations to the data distribution and subsequently applying denoising via PF-ODE. Given an original sample $\boldsymbol{x}$, Gaussian noise ϵ, and purifier d, the mapping in the transport process is defined as $T:\boldsymbol{x}\to d(\boldsymbol{x}+\epsilon)$, which is probabilistic. We aim to demonstrate that a diminished transport between the data distribution and the purified distribution is conducive to a higher likelihood of the purified output being situated in proximity to the original sample, thereby preserving its semantic meaning.

We will leverage the following definition.

Definition 3.2. Given the data distribution p, Gaussian noise ϵ, timestep t, and a purifier d, we define $\pi_t:\boldsymbol{x}\to d(\boldsymbol{x}+t\epsilon)$ and the "transport" under g_t between the data distribution and purified distribution as $T_{\pi_t}(p):=\int\|\boldsymbol{x}-\pi_t(\boldsymbol{x})\|\cdot p(\boldsymbol{x})d\boldsymbol{x}$.

Intuitively, transport measures the distance between the original and purified samples, which should be small by an effective purifier. Below, we quantify this intuition and present our main theorem. See the detailed proof in Appendix B.

Theorem 3.3. *Given the transport* $T_{\pi_t}(p)$ *between the data distribution* p *and the corresponding purified distribution under* g_t*, then for any* $r>0$*, the probability that the distance between the original sample* $\boldsymbol{x}$ *and purified sample* $\hat{\boldsymbol{x}}=\pi_t(\boldsymbol{x})$ *is larger than* r *is upper bounded by* $\frac{T_{\pi_t}(p)}{r}$.

Remark 3.4. By Theorem 3.3, the efficacy of the purifier hinges on two crucial factors: the transport $T_{\pi_t}(p)$ and the radius r. A theoretically perfect purifier would yield zero transport; however, this is unattainable due to the inherent randomness of g_t. Typically, we can optimize the parameter t to minimize the transport, denoted as $T^*=\min_t\frac{T_{\pi_t}(p)}{r}$. In the context of classification tasks, the selection of r also depends on the robustness of the classifier; a more robust classifier allows a larger r to be chosen, thereby guarantee better purification efficacy.

 https://doi.org/10.52202/079017-3335

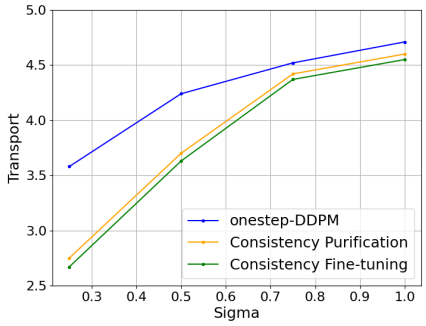


Figure 2: Transport between purified images and clean images with noise level $\sigma \in \{0.25, 0.5, 0.75, 1.0\}$.

	FID at different σ		
Loss	0.25	0.5	1.0
- -	60.3	155.3	350.3
ℓ_1	96.8	205.7	383.6
ℓ_2	102.1	214.8	375.4
LPIPS	**20.5**	**100.9**	**338.1**

Table 1: FID between purified and clean images on CIFAR-10 test set using different types of fine-tuning loss functions with noise level $\sigma \in \{0.25, 0.5, 1.0\}$.

For ensuring consistency in semantic meaning between the original and purified samples, it is insufficient merely to minimize their distance; it is also necessary that the purified sample resides on the data manifold, which is the in-distribution property we previously mentioned. To concurrently achieve both objectives, rather than solely focusing on minimizing the Euclidean distance between the original and purified samples, we opt to minimize the Learned Perceptual Image Patch Similarity (LPIPS) loss between them. This strategy aids in mitigating the risk of the purified sample deviating from the data manifold, thereby preserving semantic meaning. In Table 1, we show that using LPIPS is better than ℓ_1 and ℓ_2 loss for Consistency Fine-tuning when we want to guarantee the generated images are in-distribution, where lower FID scores indicate better in-distribution properties.

Figure 2 validates the effectiveness of Consistency Purification based on our results in Theorem 3.3, it shows that both the integration of consistency model in Consistency Purification and the further Consistency Fine-tuning can decrease the transport from the original distribution to the purified distribution. Specifically, we can see that Consistency Purification achieves a lower average distance from the purified sample to the original sample compared with onestep-DDPM, and Consistency Fine-tuning further decreases this average distance, indicating both components result in a lower transport and thus a better semantic alignment between purified samples and original samples.

4 Method

We propose our framework, Consistency Purification, with a further improved version using Consistency Fine-tuning.

4.1 Consistency Purification

We introduce Consistency Purification, directly applying consistency model as a purifier to integrate with a base classifier into smoothed classifier for randomized smoothing.

Following Diffusion Denoised Smoothing outlined in [25], it is necessary to establish a mapping between Gaussian noise augmented images required by randomized smoothing and the noised image in the ODE trajectory of consistency model. For a given consistency model purifier D_θ, any noisy input $\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}, t^2\boldsymbol{I})$ can be recovered to the trajectory's start $\boldsymbol{x}_\epsilon$ by directly passing it through the model with time t: $\boldsymbol{x}_\epsilon = D_\theta(\boldsymbol{x}_t, t)$.

When comparing this to the image augmented with additive Gaussian noise $\boldsymbol{x}_{rs} \sim \mathcal{N}(\boldsymbol{x}, \sigma^2\boldsymbol{I})$, which is required by randomized smoothing, we observe that $\boldsymbol{x}_{rs}$ and $\boldsymbol{x}_t$ share the same formula when $t = \sigma$. However, since the variances $\sigma \in \{\sigma_i\}_{i=1}^m$ may not be used during the training of the consistency model, we empirically select the nearest time step t from the discrete time steps used in training for each σ.

For the entire time horizon $[\epsilon, T]$ with $N-1$ sub-interval boundaries $t_1 = \epsilon < t_2 < \cdots < t_N = T$, the time steps used in training are computed by: $t_i = (\epsilon^{1/\rho} + {}^{i-1}/_{N-1}(T^{1/\rho} - \epsilon^{1/\rho}))^\rho$, where $\rho = 7$.

Given the variance σ of Gaussian noise used in randomized smoothing, we select the corresponding time step t_σ^* for Consistency Purified Smoothing by $t_\sigma^* = \{t_i | \sigma \in (\frac{t_{i-1}+t_i}{2}, \frac{t_i+t_{i+1}}{2}]\}$.

4.2 Consistency Fine-tuning

To optimize the consistency model for aligning semantic meanings during purification, we fine-tune the purifier D_θ by minimizing the following loss function: $\mathcal{L}_\theta = \mathbb{E}\|\boldsymbol{x} - D_\theta(\boldsymbol{x}_\sigma, t_\sigma^*)\|_{\text{LPIPS}}$, where the expectation is taken with $\boldsymbol{x} \sim p_{data}$, $\sigma \sim \mathcal{U}\{\sigma_i\}_{i=1}^m$, $\boldsymbol{x}_\sigma \sim \mathcal{N}(\boldsymbol{x}, \sigma^2 \boldsymbol{I})$. Here LPIPS denotes the distance computed by the Learned Perceptual Image Patch Similarity [30]. p_{data} represents the distribution of the training data, from which clean images $\boldsymbol{x}$ are sampled. $\mathcal{U}\{\sigma_i\}_{i=1}^m$ denotes the uniform distribution over m different noise scales σ_i used for randomized smoothing. Typically, we select the scale set $\sigma_i \in \{0.25, 0.5, 1.0\}$, which is commonly used to compute the certified radius via randomized smoothing.

After obtaining the fine-tuned consistency model purifier D_{θ^*},it can replace the original model used in Consistency Purified Smoothing to purify any noised image $\boldsymbol{x}_{rs}$ with Gaussian variance σ_i, resulting in the final purified image $\boldsymbol{x}_p$ by $\boldsymbol{x}_p = D_{\theta^*}(\boldsymbol{x}_{rs}, t_{\sigma_i}^*)$.

We present the detailed algorithm of our Consistency Purification in Appendix A.

5 Experiments

In this section, we begin by detailing the experimental settings, followed by our main results. Additionally, we conduct ablation studies to further demonstrate the effectiveness of our framework. All experiments are conducted with 1×NVIDIA RTX A5000 24GB GPU.

5.1 Experimental Settings.

Dataset. We evaluate the Consistency Purification framework on both CIFAR-10 [35] and ImageNet-64 [36]. CIFAR-10 contains 32×32 pixel images across 10 different categories while ImageNet-64 includes 64×64 pixel images across 1000 categories. Due to limited computational resources, we select 500 test images for CIFAR-10 from the 10,000 CIFAR-10 test set, choosing every 20th example in sequence (e.g., the 1st, 21st, 41st, etc.). Similarly, for the ImageNet-64 dataset, we sample 500 test examples from its 50,000 test examples using a fixed interval of 100.

Consistency Purification. For CIFAR-10, to demonstrate the effectiveness of Consistency Purification, we first perform purification with a public unconditional consistency model [37]. After that, to further improve the performance, we fine-tune the model with noise levels σ sampling from $\{0.25, 0.5, 1.0\}$, shown as the (**+ Consistency Fine-tuning**). However, currently there is no publicly available unconditional consistency model checkpoint for the ImageNet dataset that can be used directly for purification purposes. The only available model is the conditional consistency model on ImageNet-64. Thus, here we trained an unconditional consistency model on ImageNet-64, initializing it with the existing conditional consistency model checkpoint. Details of the training process are included in Appendix C. Additionally, we also conduct Consistency Fine-tuning on ImageNet-64 model with noise levels $\sigma \in \{0.05, 0.15, 0.25\}$.

Baselines. For comparative analysis of CIFAR-10, we conduct baseline experiments under various settings. The first baseline involves onestep-DDPM, where we employ the 50-M unconditional improved diffusion models from [2] utilizing the one-shot denoising method [25] for purification. Given that our consistency model is distilled from an EDM model [4], we include EDM as our baselines, applying both one-shot denoising (onestep-EDM) and ODE solver (PF-ODE EDM) for purification. Additionally, we include the recent advancement in diffusion purification methods, Diffusion Calibration, as a baseline following [38], which fine-tunes the diffusion model with the guidance of classifier WideResNet28-10 to improve the purification accuracy under the specific classifier. While for ImageNet-64, due to the lack of public unconditional EDM model, we only include the comparison baseline with onestep-DDPM.

Randomized Smoothing Settings. We set $N = 10000$ for both CIFAR-10 and ImageNet as the number of sampling times used in randomized smoothing. We compute the certified radius for each test example at three different noise levels $\sigma \in \{0.25, 0.5, 1.0\}$ for CIFAR-10 and $\sigma \in \{0.05, 0.15, 0.25\}$ for ImageNet-64. Then we calculate the proportion of test examples whose radius

 https://doi.org/10.52202/079017-3335

exceeds a specific threshold ϵ. The highest accuracy among these noise levels is reported as the certified accuracy at ϵ.

Classifiers. For the classifier used after purification for CIFAR-10, we employ ViT-B/16 model [39], which is pretrained on ImageNet-21k [36] and finetuned on CIFAR-10 dataset. In our ablation studies, we also use ResNet [40] and WideResNet [41] trained on CIFAR-10. For ImageNet-64, we make up-sampling on the 64×64 images and directly apply ViT-B/16 as the classifier.

5.2 Main Results.

We present the certified accuracy of Consistency Purification for both CIFAR-10 and ImageNet-64 dataset, with the results presented in Table 2. We also include the purification steps which decide whether the purifier needs multiple evaluation times through the networks (Multi Steps) other than a single network evaluation (One Step). As observed from Table 2, Consistency Purification significantly outperforms onestep-DDPM for both CIFAR-10 and ImageNet-64 with even higher certified accuracy with Consistency Fine-tuning. Besides, for CIFAR-10, the results also suggest the effectiveness of Consistency Purification with Consistency Fine-tuning when compared with more baseline methods such as onestep-EDM, PF-ODE EDM and Diffusion Calibration. For the detailed certified accuracy evaluation of fine-grained ϵ at different noise levels σ, we present the results in Figure 3 compared with the onestep-DDPM setting. All results have demonstrated that Consistency Purification is able to certify robustness with both efficiency and effectiveness.

Table 2: Certified Accuracy of Consistency Purification for CIFAR-10 and ImageNet-64.

CIFAR-10		Certified Accuracy at ϵ (%)				
Method	Purification Steps	0.0	0.25	0.5	0.75	1.0
onestep-DDPM[25]	One Step	87.6	73.6	55.6	39.2	29.6
onestep-EDM	One Step	87.4	76.2	58.8	40.8	32.4
PF-ODE EDM	Multi Steps	89.6	77.0	60.4	42.6	34.0
Diffusion Calibration[38]	One Step	90.2	76.4	57.2	42.6	32.4
Consistency Purification	One Step	**90.4**	77.2	59.8	42.8	33.2
+ Consistency Fine-tuning	One Step	90.2	**79.4**	**62.4**	**43.8**	**35.4**

ImageNet-64		Certified Accuracy at ϵ (%)				
Method	Purification Steps	0.0	0.05	0.15	0.25	0.35
onestep-DDPM [25]	One Step	55.2	44.8	33.4	15.2	8.8
Consistency Purification	One Step	62.4	54.2	35.2	19.8	13.0
+ Consistency Fine-tuning	One Step	**68.6**	**58.0**	**37.4**	**23.4**	**17.4**

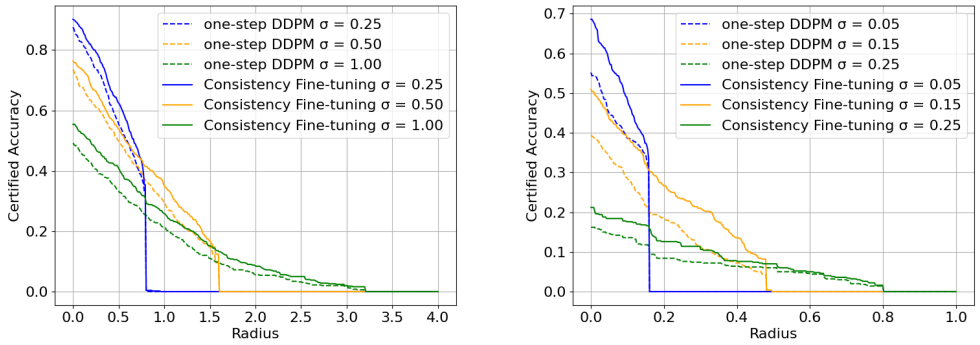
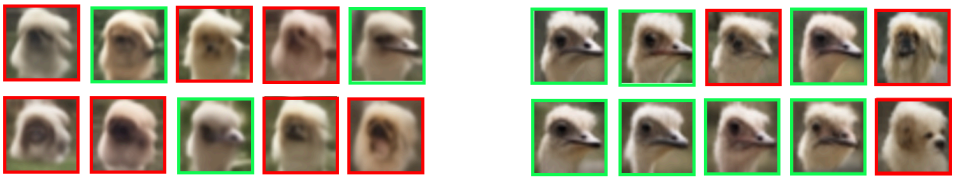


Figure 3: Certified Accuracy of Consistency Purification with fine-grained radius ϵ. The left figure shows results on CIFAR-10, the right figure shows results on ImageNet-64. The lines demonstrate the certified accuracy of various radius ϵ under different Gaussian noise levels σ.

To better illustrate the significant improvement in certified robustness brought by Consistency Purification, we present visualizations of images after diffusion purification in Figure 4 for CIFAR-10 at a noise level of $\sigma = 0.5$, compared with the onestep-DDPM approach. As shown, our method produces significantly higher-quality purified images than onestep-DDPM. Furthermore, these purified images achieve a notably higher classification accuracy when evaluated by the same classifier. Additional visualization examples for ImageNet-64 are included in Appendix D.



(a) Purified images by onestep-DDPM

(b) Purified images by Consistency Purification

Figure 4: Visualization of purified images after the diffusion purification by applying onestep-DDPM and Consistency Purification on CIFAR-10 with $\sigma = 0.5$ noise level. Identical noise patterns are applied to images at corresponding locations. A green border indicates that the purified image is correctly classified, while a red border denotes misclassification by the classifier.

5.3 Ablation Studies.

We conduct various ablation studies to evaluate the effectiveness of our proposed method.

Comparison with Non-Diffusion-based Baselines. To compare Consistency Purification with various non-diffusion-based approaches, we conducted additional experiments to compute the certified accuracy under three non-diffusion-based methods [19, 22, 42]. [19] first proposed training a classifier with noisy images to ensure certified robustness. Subsequent works [22, 42] build on [19]'s methodology, attempting to enhance the smoothed classifier by adding prediction consistency regularization, or incorporating per-sample bias. The experimental results presented in Tabel 3 show that our method surpasses all previous non-diffusion-based methods in achieving higher certified accuracy, particularly with a significantly high clean performance at $\epsilon = 0.0$. Furthermore, in contrast to non-diffusion-based methods, which incur significant costs by requiring additional fine-tuning of robust classifiers for each specific noise level, our method can be applied directly to any off-the-shelf classifiers, significantly broadening its practical applications.

	Certified Accuracy at ϵ%				
Methods	0.0	0.25	0.5	0.75	1.0
Randomized Smoothing [19]	74.8	59.2	42.0	31.8	22.0
Consistency Regularization [22]	74.4	66.0	56.2	41.4	32.8
Aces [42]	74.6	66.4	57.0	43.6	32.8
Consistency Purification	**90.4**	77.2	59.8	42.8	33.2
+ Consistency Fine-tuning	90.2	**79.4**	**62.4**	**43.8**	**35.4**

Table 3: Certified Accuracy of Consistency Purification compared with non-diffusion-based baseline methods.

Fine-tuning Loss Functions. To further demonstrate that LPIPS loss is the best choice considering both on-manifold purification and semantic meaning alignment, we assess the certified accuracy of Consistency Purification using different loss functions during Consistency Fine-tuning. Instead of LPIPS distance between the clean and purified images as the loss function, we experiment with ℓ_1 and ℓ_2 distances. Results in Table 4 indicate that Consistency Purification with LPIPS loss achieves the highest Certified Accuracy. In contrast, fine-tuning with ℓ_1 and ℓ_2 distances compromises the purification performance for certification. This demonstrates that fine-tuning with LPIPS loss function effectively aligns semantic meanings, whereas ℓ_1 or ℓ_2 distances may hurt them.

Noise Levels Sampling Schedules during Consistency Fine-tuning. In our experiments of Consistency Fine-tuning, we simply select the same sampling schedules of noise levels $\sigma \sim \mathcal{U}\{0.25, 0.5, 1.0\}$, uniformly sampling σ used in randomized smoothing. To empirically demonstrate its effectiveness, we compare this approach with continuous sampling schedules where $\sigma \sim \mathcal{U}[0, 1]$. Results presented in Table 5 show that our discrete sampling schedule achieves higher certified accuracy. This indicates that fine-tuning with a discrete scale, aligned with the noise levels used in randomized smoothing, enhances certified robustness.

https://doi.org/10.52202/079017-3335

Table 4: Certified Accuracy of Consistency Purification with different loss functions during fine-tuning for CIFAR-10. "- -" represents the setting without fine-tuning.

	Certified Accuracy at ϵ%				
Distance	0.0	0.25	0.5	0.75	1.0
- -	**90.4**	77.2	59.8	42.8	33.2
ℓ_1	89.4	76.4	59.6	42.4	31.4
ℓ_2	90.0	77.0	59.8	42.4	33.4
LPIPS	90.2	**79.4**	**62.4**	**43.8**	**35.4**

Table 5: Certified Accuracy of Consistency Purification with continuous and discrete sampling schedules during fine-tuning for CIFAR-10. "- -" represents the setting without fine-tuning.

	Certified Accuracy at ϵ%				
Schedules	0.0	0.25	0.5	0.75	1.0
- -	**90.4**	77.2	59.8	42.8	33.2
[0,1]	89.0	76.2	59.8	43.2	33.8
{0.25, 0.5, 1.0}	90.2	**79.4**	**62.4**	**43.8**	**35.4**

Generalizability with Different Classifiers. We compute certified accuracy with various classifiers to test if our framework maintains its effectiveness with arbitrary classifiers. The results, presented in Table 6, compare Consistency Fine-tuning with Diffusion Calibration, an alternative method to fine-tune diffusion models for improving the certified robustness. When evaluated across different classifiers, including ViT-B/16, ResNet56, and WideNet28-10, our method outperforms Diffusion Calibration except certified accuracy at $\epsilon = 0.0$ on WRN28-10 model. It is worth noting that the Diffusion Calibration, which requires a specific classifier for guidance during fine-tuning, exhibits limitations, only achieving comparable performance with the guidance classifier WRN28-10. This demonstrates the advantages of Consistency Fine-tuning in generalizing across different classifiers.

Fine-tuning Classifier vs. Fine-tuning Diffusion Model. A potential concern with Consistency Fine-tuning is the higher certified accuracy and lower training cost associated with Fine-tuning the Classifier (CLS-FT) compared to our approach of Fine-tuning the Diffusion Model (DM-FT). However, our experiments, as shown in Table 7, indicate that DM-FT does not conflict with CLS-FT; rather, combining these two methods achieves even higher certified accuracy. On another hand, although CLS-FT yield slightly higher certified accuracy than DM-FT, its requirement for fine-tuning a specific classifier compromises the natural property of diffusion purification frameworks with arbitrary off-the-shelf classifiers, thus limiting the practical applicability.

Table 6: Certified Accuracy of Consistency Fine-tuning with different classifiers on CIFAR-10. The guidance classifier used in Diffusion Calibration is WideResNet28-10.

		Certified Accuracy at ϵ%				
Method	Classifier	0.0	0.25	0.5	0.75	1.0
Diffusion Calibration [38]	ViT-B/16	90.2	76.4	57.2	42.6	32.4
	WRN28-10	**88.2**	76.4	59.2	42.0	31.8
	ResNet56	86.0	72.8	52.6	35.2	25.8
Consistency Fine-tuning	ViT-B/16	90.2	**79.4**	**62.4**	**43.8**	**35.4**
	WRN28-10	88.0	**76.4**	**59.8**	**42.8**	**33.0**
	ResNet56	**87.2**	**74.8**	**57.6**	**38.2**	**30.2**

Table 7: Certified Accuracy of Fine-tuning the Diffusion Model (DM-FT) compared with Fine-tuning the Classifier (CLS-FT) in diffusion purification frameworks on CIFAR-10.

		Certified Accuracy at ϵ%				
DM-FT	CLS-FT	0.0	0.25	0.5	0.75	1.0
-	-	90.4	77.2	59.8	42.8	33.2
✓	-	90.2	79.4	62.4	43.8	35.4
-	✓	90.4	79.8	63.4	44.2	35.2
✓	✓	**90.8**	**80.0**	**64.8**	**44.6**	**36.8**

6 Conclusion

In this paper, we introduced Consistency Purification, a novel framework proposed to enhance certified robustness via randomized smoothing. By incorporating consistency models into diffusion purification approach and further refining them through Consistency Fine-tuning, our empirical experiments have demonstrate the framework's ability to achieve high certified robustness efficiently with one single network evaluation for purification.

Limitations. A notable limitation of our study is that our empirical results do not include computing certified robustness of high-resolution images such as ImageNet 256×256. This constraint is due to the absence of publicly available checkpoints for the consistency model at this resolution. Additionally, training a consistency model for ImageNet 256×256 would require huge computing resources, which are currently beyond our affordability. However, our framework is designed for adaptability and could be easily extended to ImageNet 256×256 once these checkpoints become available. As a result, our empirical evaluations in this paper are limited to the CIFAR-10 and ImageNet 64×64 datasets.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

[3] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

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

[5] 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.

[6] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2020.

[7] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*, 2020.

[8] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR, 2021.

[9] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[10] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[11] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021.

[12] Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy J Mitra. Holodiffusion: Training a 3d diffusion model using 2d images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18423–18433, 2023.

[13] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[14] 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.

[15] Shutong Wu, Jiongxiao Wang, Wei Ping, Weili Nie, and Chaowei Xiao. Defending against adversarial audio via diffusion model. In *The Eleventh International Conference on Learning Representations*, 2022.

[16] Jiachen Sun, Jiongxiao Wang, Weili Nie, Zhiding Yu, Zhuoqing Mao, and Chaowei Xiao. A critical revisit of adversarial robustness in 3d point cloud recognition with diffusion-driven purification. In *International Conference on Machine Learning*, pages 33100–33114. PMLR, 2023.

 https://doi.org/10.52202/079017-3335

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

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

[19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320. PMLR, 09–15 Jun 2019.

[20] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. *Advances in Neural Information Processing Systems*, 32, 2019.

[21] 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. *arXiv preprint arXiv:2001.02378*, 2020.

[22] Jongheon Jeong and Jinwoo Shin. Consistency regularization for certified robustness of smoothed classifiers. *Advances in Neural Information Processing Systems*, 33:10558–10570, 2020.

[23] Miklós Z Horváth, Mark Niklas Müller, Marc Fischer, and Martin Vechev. Boosting randomized smoothing with variance reduced classifiers. *arXiv preprint arXiv:2106.06946*, 2021.

[24] 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.

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

[26] Chaowei Xiao, Zhongzhu Chen, Kun Jin, Jiongxiao Wang, Weili Nie, Mingyan Liu, Anima Anandkumar, Bo Li, and Dawn Song. Densepure: Understanding diffusion models for adversarial robustness. In *The Eleventh International Conference on Learning Representations*, 2022.

[27] Jiawei Zhang, Zhongzhu Chen, Huan Zhang, Chaowei Xiao, and Bo Li. {DiffSmooth}: Certifiably robust learning via diffusion models and local smoothing. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4787–4804, 2023.

[28] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[29] Huanran Chen, Yinpeng Dong, Shitong Shao, Zhongkai Hao, Xiao Yang, Hang Su, and Jun Zhu. Your diffusion model is secretly a certifiably robust classifier. *arXiv preprint arXiv:2402.02316*, 2024.

[30] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[31] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

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

[33] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[34] 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.

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

[36] 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.

[37] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023.

[38] Jongheon Jeong and Jinwoo Shin. Multi-scale diffusion denoised smoothing. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

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

[40] 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.

[41] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks, 2017.

[42] Miklós Z Horváth, Mark Niklas Müller, Marc Fischer, and Martin Vechev. Robust and accurate–compositional architectures for randomized smoothing. *arXiv preprint arXiv:2204.00487*, 2022.

 https://doi.org/10.52202/079017-3335

A Consistency Purification Algorithm

We provide detailed descriptions of Consistency Purification in the following algorithms. Algorithm 1 presents the function of Consistency Fine-tuning and Consistency Purification respectively. Algorithm 2 shows the randomized smoothing algorithm from [19] with applying Consistency Purification to do prediction and compute the certified radius.

Algorithm 1 Consistency Fine-tuning and Consistency Purification

Input: Consistency model purifier D_θ where θ represents the model parameters. Noise levels used in randomized smoothing $\{\sigma_i\}_{i=1}^m$. Arbitrary classification model f_{clf}. Fine-tuning learning rate η.

function CONSISTENCYFINE-TUNING(D_θ)
  repeat
    sample $x \in$ Training Dataset, $\sigma \in \{\sigma_i\}_{i=1}^m$
    $x_\sigma \leftarrow x + \mathcal{N}(0, \sigma^2 \boldsymbol{I})$
    $t_\sigma^* \leftarrow$ GETTIMESTEP(σ)
    $\mathcal{L} \leftarrow \text{LPIPS}(x, D_\theta(x_\sigma, t^*))$
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$
  until convergence
  return D_θ
end function

function CONSISTENCYPURIFICATION($f_{\text{clf}}, x, \sigma$)
  $t_\sigma^* \leftarrow$ GETTIMESTEP(σ)
  $\boldsymbol{x}_{rs} \leftarrow \boldsymbol{x} + \mathcal{N}(0, \sigma^2 I)$
  $\boldsymbol{x}_p \leftarrow D_{\theta^*}(\boldsymbol{x}_{rs}, t_\sigma^*)$
  $y \leftarrow f_{\text{clf}}(\boldsymbol{x}_p)$
  return y
end function

function GETTIMESTEP(σ)
  $t_i \leftarrow (\epsilon^{1/\rho} + \frac{i-1}{N-1}(T^{1/\rho} - \epsilon^{1/\rho}))^\rho$ for $i \in \{1, \ldots, N\}$
  $t_\sigma^* \leftarrow$ find $\{t_i | \sigma \in \left(\frac{t_{i-1}+t_i}{2}, \frac{t_i+t_{i+1}}{2}\right]\}$
  return t_σ^*
end function

B Proof of Theorem 3.3

Theorem 3.3. *Given the transport $T_{\pi_t}(p)$ between the data distribution p and the corresponding purified distribution under g_t, then for any $r > 0$, the probability that the distance between the original sample $\boldsymbol{x}$ and purified sample $\hat{\boldsymbol{x}} = \pi_t(\boldsymbol{x})$ is larger than r is upper bounded by $\frac{T_{\pi_t}(p)}{r}$.*

Proof. We can leverage the Markov's inequality. Because

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|] &= \int_{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\| \le r} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \cdot p(\boldsymbol{x}) d\boldsymbol{x} + \int_{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\| > r} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \cdot p(\boldsymbol{x}) d\boldsymbol{x} \\
&\ge \int_{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\| > r} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \cdot p(\boldsymbol{x}) d\boldsymbol{x} \\
&\ge \int_{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\| > r} r \cdot p(\boldsymbol{x}) d\boldsymbol{x} \\
&= r \cdot P(\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| > r),
\end{aligned}$$

we have

$$
\begin{aligned}
P(\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|>r) &\leq \frac{\mathbb{E}[\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|]}{r} \\
&= \frac{\mathbb{E}[\|\boldsymbol{x}-\pi_t(\boldsymbol{x})\|]}{r} \\
&= \frac{T_{\pi_t}(p)}{r}.
\end{aligned}
$$

□

Algorithm 2 Randomized Smoothing [19]

Input: Sampling times for prediction n. Sampling times for certification N. Significant confidence level α. Function LOWERCONFBOUND$(k, n, 1-\alpha)$ returns a one-sided (1-α) lower confidence interval for the Binomial parameter p given that $k \sim$ Binomial(n, p).

function PREDICT($f_{\text{clf}}, \boldsymbol{x}, \sigma, n, \alpha$)
  counts $\leftarrow 0$
  for $i \in \{1, 2, \ldots, n\}$ **do**
    $y \leftarrow$ CONSISTENCYPURIFICATION($f_{\text{clf}}, \boldsymbol{x}, \sigma$)
    counts[y] $\leftarrow$ counts[y] + 1
  end for
  $\hat{y}_A, \hat{y}_B \leftarrow$ top two labels in counts
  $n_A, n_B \leftarrow$counts$[\hat{y}_A]$,counts$[\hat{y}_B]$
  if BINOMTEST$(n_A, n_A + n_B, \frac{1}{2}) \leq \alpha$ **then**
    return $\hat{y}_A$
  else
    return Abstain
  end if
end function

function CERTIFY($f_{\text{clf}}, \boldsymbol{x}, \sigma, n, N, \alpha$)
  counts0 $\leftarrow 0$
  for $i \in \{1, 2, \ldots, n\}$ **do**
    $y \leftarrow$ CONSISTENCYPURIFICATION($f_{\text{clf}}, \boldsymbol{x}, \sigma$)
    counts0[y] $\leftarrow$ counts0[y] + 1
  end for
  $\hat{y}_A \leftarrow$ top label in counts0
  counts $\leftarrow 0$
  for $i \in \{1, 2, \ldots, N\}$ **do**
    $y \leftarrow$ CONSISTENCYPURIFICATION($f_{\text{clf}}, \boldsymbol{x}, \sigma$)
    counts[y] $\leftarrow$ counts[y] + 1
  end for
  $\underline{p_A} \leftarrow$ LOWERCONFBOUND(counts$[\hat{y}_A], N, 1-\alpha$)
  if $\underline{p_A} > \frac{1}{2}$ **then**
    return prediction $\hat{y}_A$ and radius $\sigma\Phi^{-1}(\underline{p_A})$
  else
    return Abstain
  end if
end function

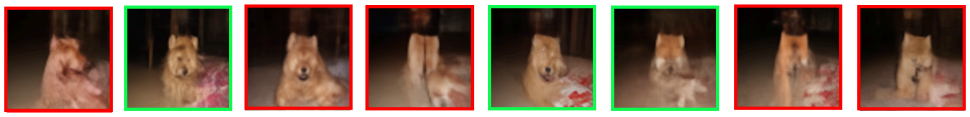
C Training Unconditional Consistency Model for ImageNet-64

We train an unconditional consistency model for ImageNet-64 from the public available conditional version by transiting the class embedding layers to a learnable token, initialization with average class embeddings. For each model forwarding, this token will be combined with the time embeddings for computation. After that, we train the conditional consistency model, initialized with the unconditional model's parameters, on ImageNet-64 training set for 120k steps.

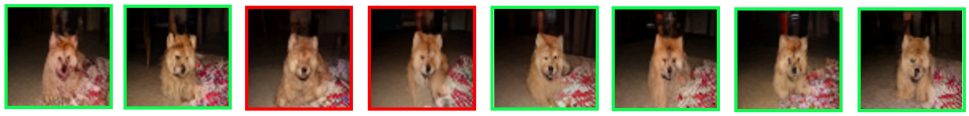
https://doi.org/10.52202/079017-3335

D Purified Images Visualization for ImageNet-64

We present visualization images after diffusion purification by applying onestep-DDPM and Consistency Purification for ImageNet-64 under the noise level $\sigma = 0.25$ in Figure 5.



(a) Purified images by onestep-DDPM



(b) Purified images by Consistency Purification

Figure 5: Visualization of purified images after the diffusion purification by applying onestep-DDPM and Consistency Purification on ImageNet-64 with $\sigma = 0.25$ noise level. Identical noise patterns are applied to images at corresponding locations. A green border indicates that the purified image is correctly classified, while a red border denotes misclassification by the classifier.

NeurIPS Paper Checklist

1. **Claims**
   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: We have included our paper's contributions and scope in the abstract and introduction.
   Guidelines:
   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.
2. **Limitations**
   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: We have discussed the limitaitons of our work after Conclusion Section.
   Guidelines:
   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.
3. **Theory Assumptions and Proofs**
   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?
   Answer: [Yes]
   Justification: We provide detailed assumptions in Section 3 and the proof of theorem in Appendix B.
   Guidelines:
   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.

https://doi.org/10.52202/079017-3335

- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**
   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?
   Answer: [Yes]
   Justification: All experiments in our paper can be reproducible with all disclosed information shown in our experiment section.
   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**
   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?
   Answer: [Yes]
   Justification: We have submitted our code as the Supplementary Material.
   Guidelines:
   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not

including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**
   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?
   Answer: [Yes]
   Justification: We have include the detailed experiment settings in Section 5.1.
   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.
7. **Experiment Statistical Significance**
   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?
   Answer: [No]
   Justification: Error bars are not reported because it would be too computationally expensive for the large language model fine-tuning process.
   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.
8. **Experiments Compute Resources**
   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?
   Answer: [Yes]
   Justification: We include the details about the experiments compute resources in Section 5.

https://doi.org/10.52202/079017-3335

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have checked the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no societal impact of our work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper does not have such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring

that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**
Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?
Answer: [Yes]
Justification: We have included licenses of original owners in our code and made citations in our papers.
Guidelines:
- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**
Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?
Answer: [NA]
Justification: Our paper does not release new assets.
Guidelines:
- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**
Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?
Answer: [NA]
Justification: Our paper does not involve crowdsourcing or research with human subjects.
Guidelines:
- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

https://doi.org/10.52202/079017-3335

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.