DiffSmooth: Certifiably Robust Learning via Diffusion Models and Local Smoothing

Jiawei Zhang[1], Zhongzhu Chen[2], Huan Zhang[3], Chaowei Xiao[4], Bo Li[1]

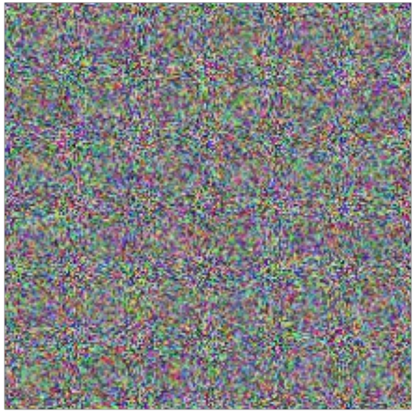
[1]UIUC, [2]University of Michigan, Ann Arbor, [3]Carnegie Mellon University
[4]Arizona State University

Background: Adversarial attack

Background: Adversarial attack

Panda

$+\ 0.005\times$



$=$

Gorilla

An imperceptible perturbation can cause misclassification

Background: Certification

- Given the input x, a base classifier f and radius r :
  report whether there exists a perturbation δ within $||\delta||_2 \leq r$ for which $f(x) \neq f(x+\delta)$

Background: Certification

- Given the input x, a base classifier f and radius r :
  report whether there exists a perturbation δ within $||\delta||_2 \leq r$ for which $f(x) \neq f(x+\delta)$

- Randomized smoothing (Cohen, 2019): construct a new smoothed classifier g

$$g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f(x+\varepsilon) = c) \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$$

Background: Certification

- Given the input x, a base classifier f and radius r :
  report whether there exists a perturbation δ within $||\delta||_2 \leq r$ for which $f(x) \neq f(x+\delta)$

- Randomized smoothing (Cohen, 2019): construct a new smoothed classifier g

$$g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f(x+\varepsilon) = c) \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$$

Theorem (simplified version):

Suppose that when the base classifier f classifies $\mathcal{N}\left(x, \sigma^2 I\right)$, the most probable class c_A is returned with a lower bound probability $\underline{p_A}$, then the smoothed classifier g is robust around x within the radius :

$$R = \sigma \Phi^{-1}\left(\underline{p_A}\right),$$

where Φ^{-1} is the inverse of the standard Gaussian CDF.

Background: Certification

- Randomized smoothing (Cohen, 2019): construct a new smoothed classifier g

$$g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f\boxed{(x + \varepsilon)} = c) \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$$

The base classifier f needs to be trained with Gaussian augmentation
=> f_{robust} (soft version: F_{robust});
We need to train more models, and the clean accuracy will also drop a bit.

Background: Certification

- Randomized smoothing (Cohen, 2019): construct a new smoothed classifier g

$$g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f\boxed{(x+\varepsilon)} = c) \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$$

The base classifier f needs to be trained with Gaussian augmentation
=> f_{robust} ($soft\ version$: F_{robust});
We need to train more models, and the clean accuracy will also drop a bit.

- Denoised smoothing (Salman, 2020):

$$f \coloneqq f_{clf} \circ D_\theta,$$

where f_{clf} is the *off-the-shelf* standard image classifier and D_θ is a custom-trained denoiser

Background: Certification

- Randomized smoothing (Cohen, 2019): construct a new smoothed classifier g

$$g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f\boxed{(x+\varepsilon)} = c) \text{ where } \varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$$

The base classifier f needs to be trained with Gaussian augmentation
=> f_{robust} ($soft\ version$: F_{robust});
We need to train more models, and the clean accuracy will also drop a bit.

- Denoised smoothing (Salman, 2020):

$$f \coloneqq f_{clf} \circ D_\theta,$$

where f_{clf} is the *off-the-shelf* standard image classifier and D_θ is a custom-trained denoiser

- Diffusion Denoised smoothing (Carlini, 2023):

$$f \coloneqq f_{clf} \circ D_\theta,$$

D_θ is a denoising diffusion probabilistic model (DDPM), one-shot denoising

What can we expect from a perfect denoiser?

- Denoised smoothing (Salman, 2020):

$$f := f_{clf} \circ D_\theta,$$

where f_{clf} is the *off-the-shelf* standard image classifier and D_θ is a custom-trained denoiser

- Diffusion Denoised smoothing (Carlini, 2023):

$$f := f_{clf} \circ D_\theta,$$

D_θ is a denoising diffusion probabilistic model (DDPM), one-shot denoising

What can we expect from a perfect denoiser?

- Denoised smoothing (Salman, 2020):

$$f := f_{clf} \circ D_\theta,$$

where f_{clf} is the *off-the-shelf* standard image classifier and D_θ is a custom-trained denoiser

- Diffusion Denoised smoothing (Carlini, 2023):

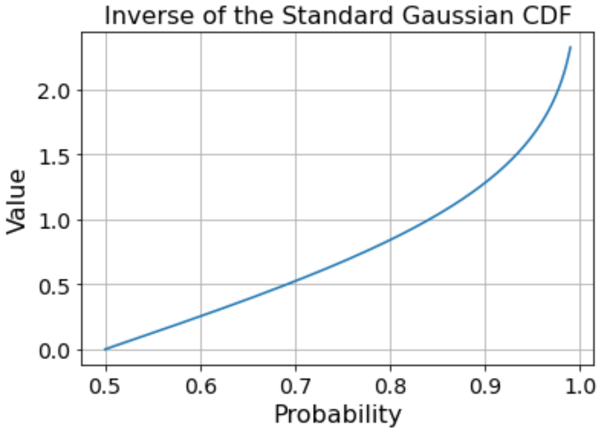
$$f := f_{clf} \circ D_\theta,$$

D_θ is a denoising diffusion probabilistic model (DDPM), one-shot denoising

For every $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$, $D_\theta(x+\varepsilon) \approx x \Rightarrow$

$$f(x+\varepsilon) \approx f_{clf}(x) \Rightarrow \underline{p_A} \approx 1$$

$$\Rightarrow R = \sigma\Phi^{-1}(\underline{p_A})$$



What can we expect from a perfect denoiser?

- Denoised smoothing (Salman, 2020):

$$f := f_{clf} \circ D_\theta,$$

where f_{clf} is the *off-the-shelf* standard image classifier and D_θ is a custom-trained denoiser

- Diffusion Denoised smoothing (Carlini, 2023):

$$f := f_{clf} \circ D_\theta,$$

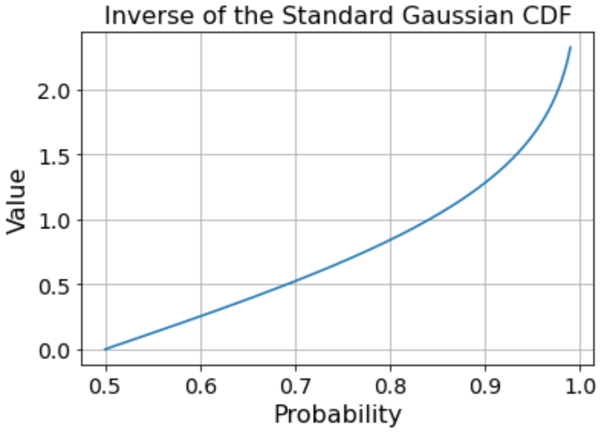
D_θ is a denoising diffusion probabilistic model (DDPM), one-shot denoising

For every $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$, $D_\theta(x + \varepsilon) \approx x \Rightarrow$

$$f(x + \varepsilon) \approx f_{clf}(x) \Rightarrow \underline{p_A} \approx 1$$

$$\Rightarrow R = \sigma\Phi^{-1}(\underline{p_A})$$

However, the certified accuracy under large perturbation radii decreases quickly in practice…



What can we expect from an imperfect denoiser?

What can we expect from an imperfect denoiser?

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

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

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

What can we expect from an imperfect denoiser?

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

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

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

What can we expect from an imperfect denoiser?

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

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

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

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

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

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

What can we expect from an imperfect denoiser?

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

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

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

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

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

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

What can we expect from an imperfect denoiser?

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

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

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

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

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

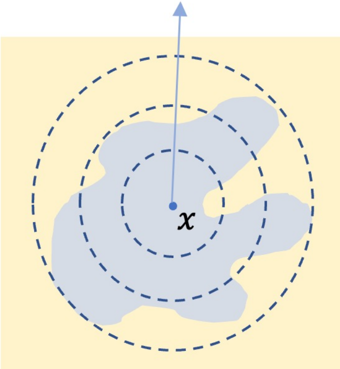
where $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t$ are random variables generated by reverse-SDE, $\mathbb{P}(\hat{\mathbf{x}}_0 = x | \hat{\mathbf{x}}_t = x_{t^*}) \propto p(x) \cdot \frac{1}{\sqrt{(2\pi\sigma_t^2)^n}} \exp\left(\frac{-||x - x_{t^*}||_2^2}{2\sigma_t^2}\right)$ and $\sigma_t^2 = \frac{1-\alpha_t}{\alpha_t}$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

We can expect the one-shot purified adversarial images are within a small bounded neighborhood of the original clean image with high probability

Pipeline:

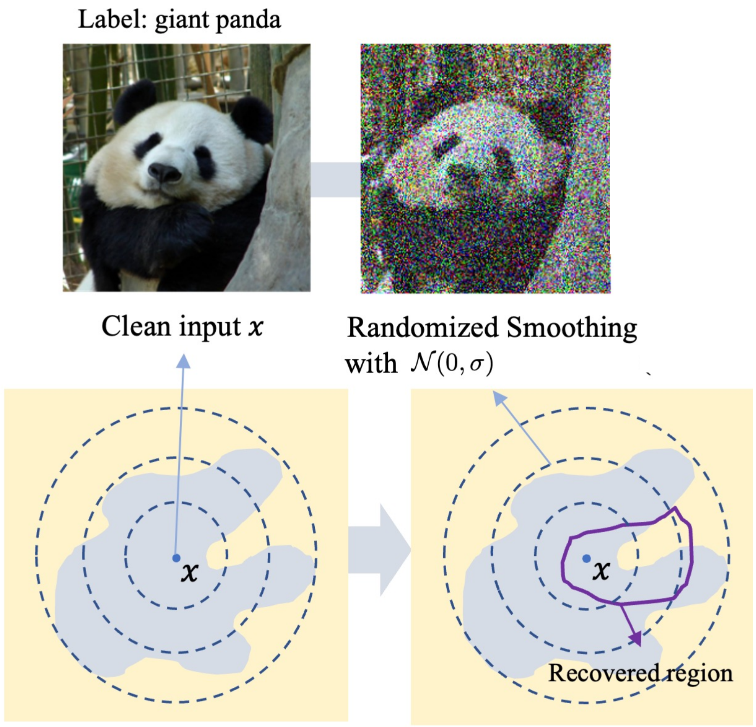
Label: giant panda

Clean input x



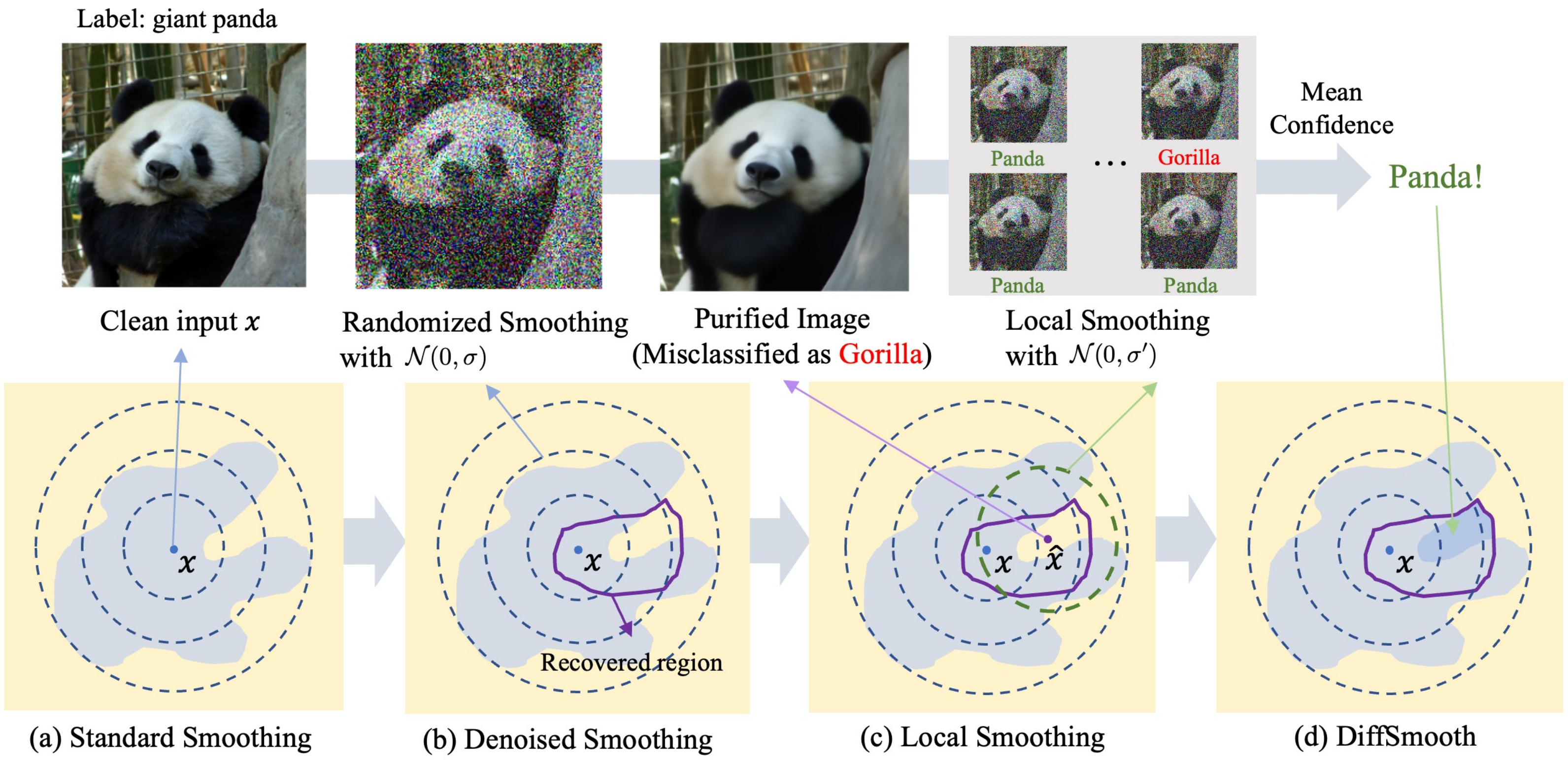
(a) Standard Smoothing

Pipeline:



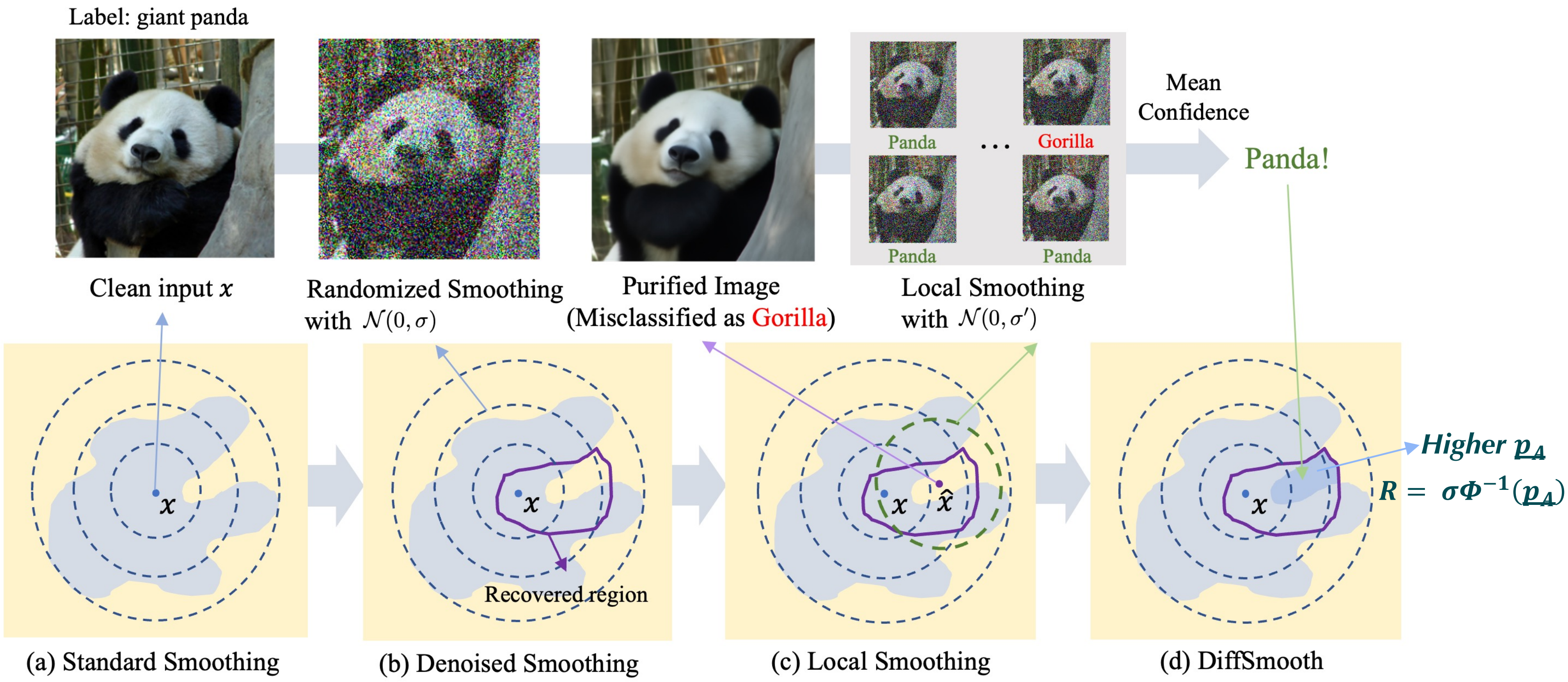
(a) Standard Smoothing (b) Denoised Smoothing

Pipeline:



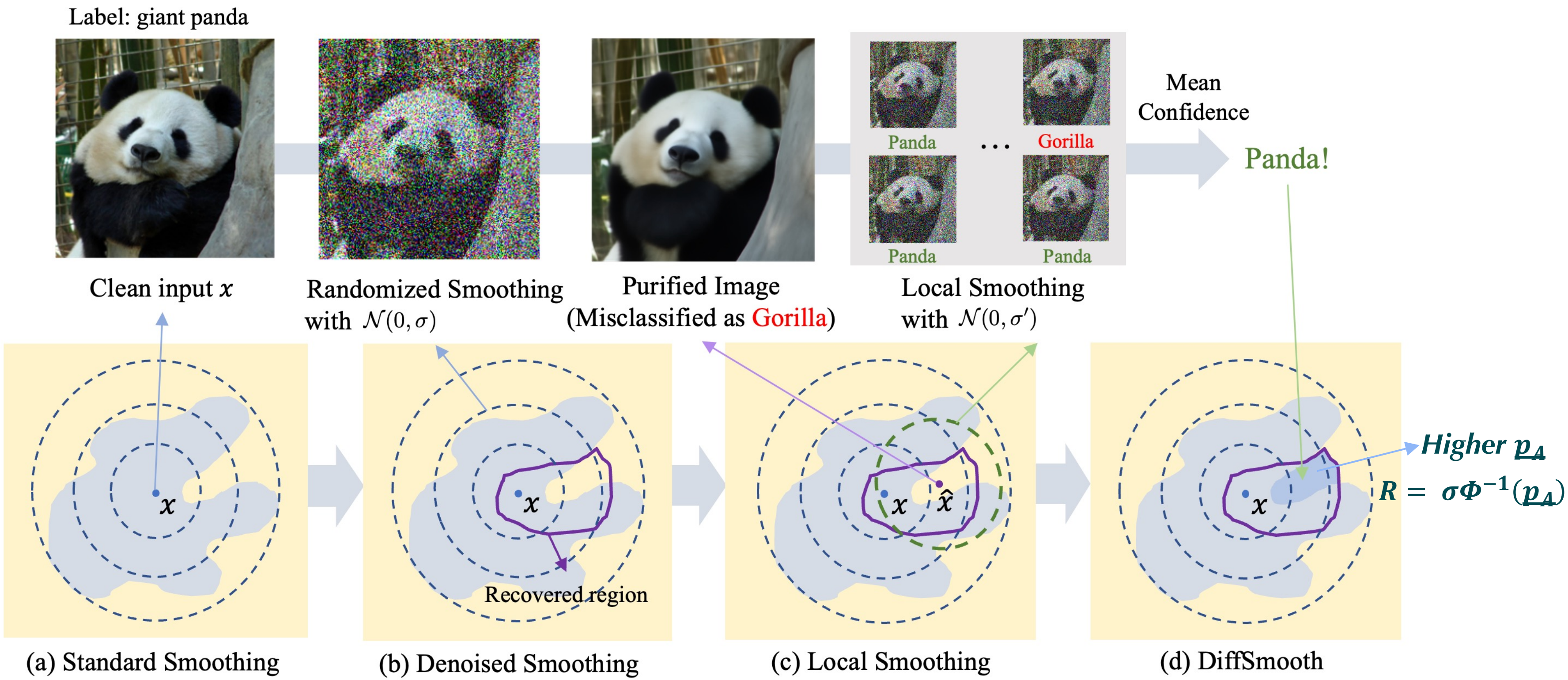
Label: giant panda
Clean input x
Randomized Smoothing with $\mathcal{N}(0,\sigma)$
Purified Image (Misclassified as Gorilla)
Panda
Gorilla
Panda
Panda
Local Smoothing with $\mathcal{N}(0,\sigma')$
Mean Confidence
Panda!
x
Recovered region
$\hat{x}$
(a) Standard Smoothing
(b) Denoised Smoothing
(c) Local Smoothing
(d) DiffSmooth

Pipeline:



Label: giant panda
Clean input x
Randomized Smoothing with $\mathcal{N}(0, \sigma)$
Purified Image (Misclassified as Gorilla)
Panda
Gorilla
Panda
Panda
Local Smoothing with $\mathcal{N}(0, \sigma')$
Mean Confidence
Panda!
Recovered region
Higher p_A
$R = \sigma\Phi^{-1}(p_A)$
(a) Standard Smoothing
(b) Denoised Smoothing
(c) Local Smoothing
(d) DiffSmooth

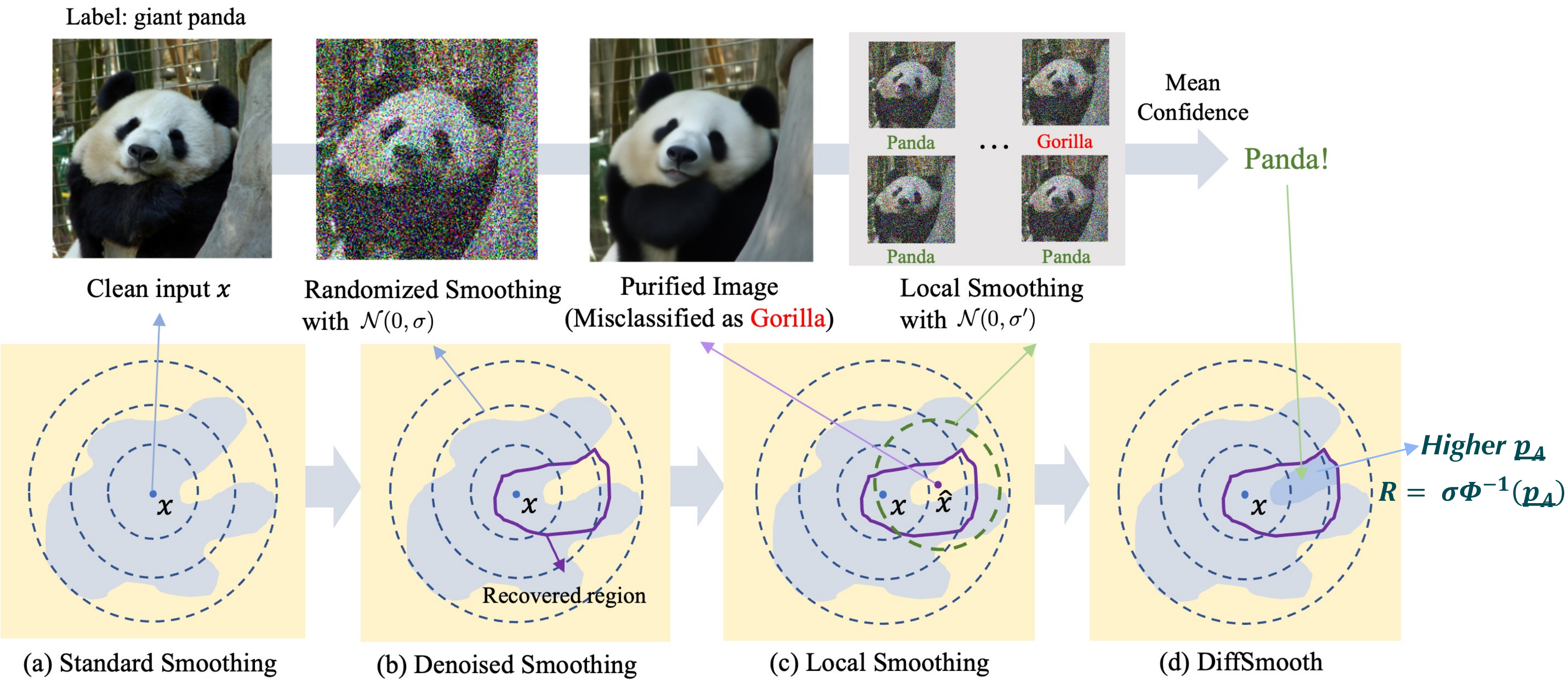
Pipeline:



For a *given* $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$,

$$f(x+\varepsilon) = f_{clf}\big(D_\theta(x+\varepsilon)\big) = f_{clf}(\hat{x})$$

Pipeline:

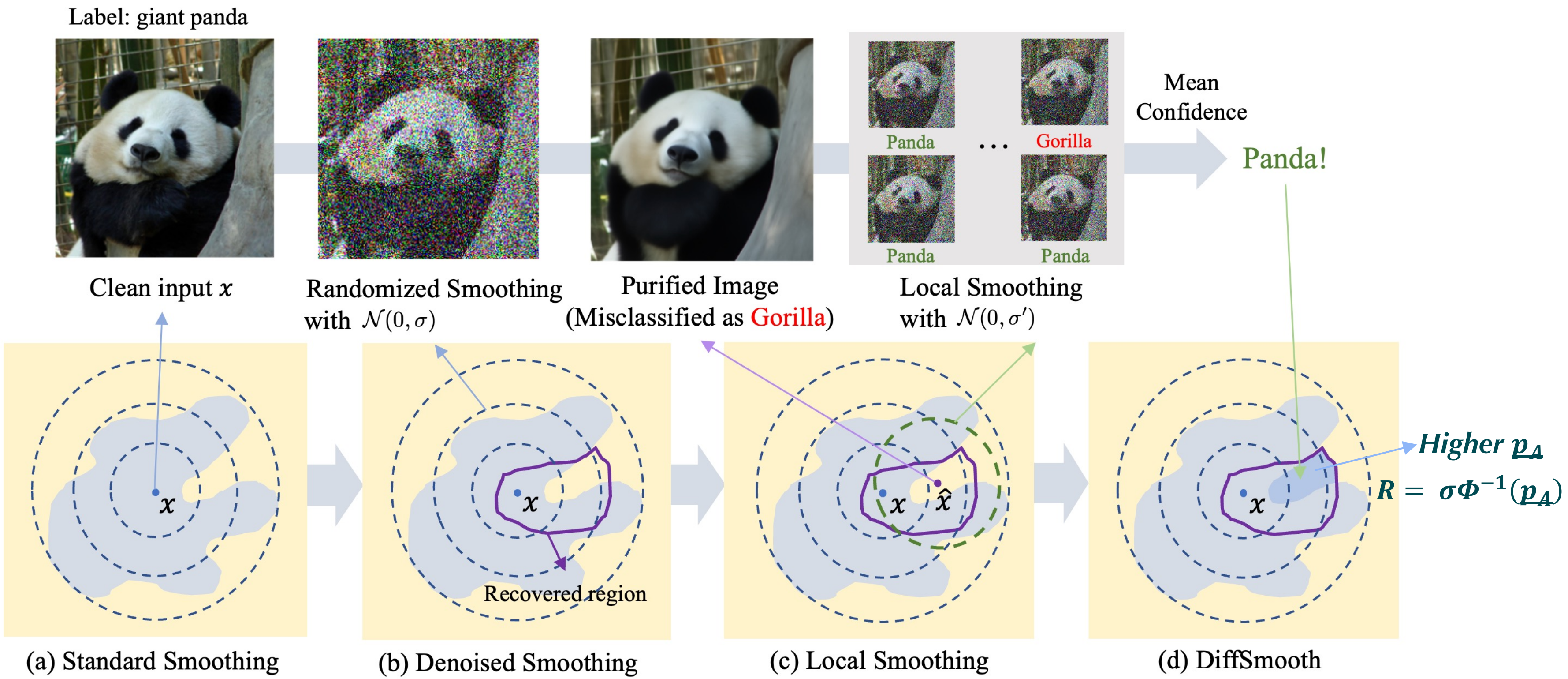


For a *given* $\varepsilon \sim \mathcal{N}\left(0,\sigma^2 I\right)$, $\quad f(x+\varepsilon) = f_{clf}\big(D_\theta(x+\varepsilon)\big) = f_{clf}(\hat{x})$

$$\Rightarrow f(x+\varepsilon) = \operatorname{arg}max\ \mathbb{E}\big(F_{robust}(D_\theta(x+\varepsilon)+\varepsilon')\big) = \operatorname{arg}max\ \mathbb{E}(F_{robust}(\hat{x}+\varepsilon')),\ \ \varepsilon' \sim \mathcal{N}\left(0,\sigma'^2 I\right),\ \sigma' < \sigma$$

(in practice, we just use m fixed sampled ε' for approximating the expectation here.)

Pipeline:

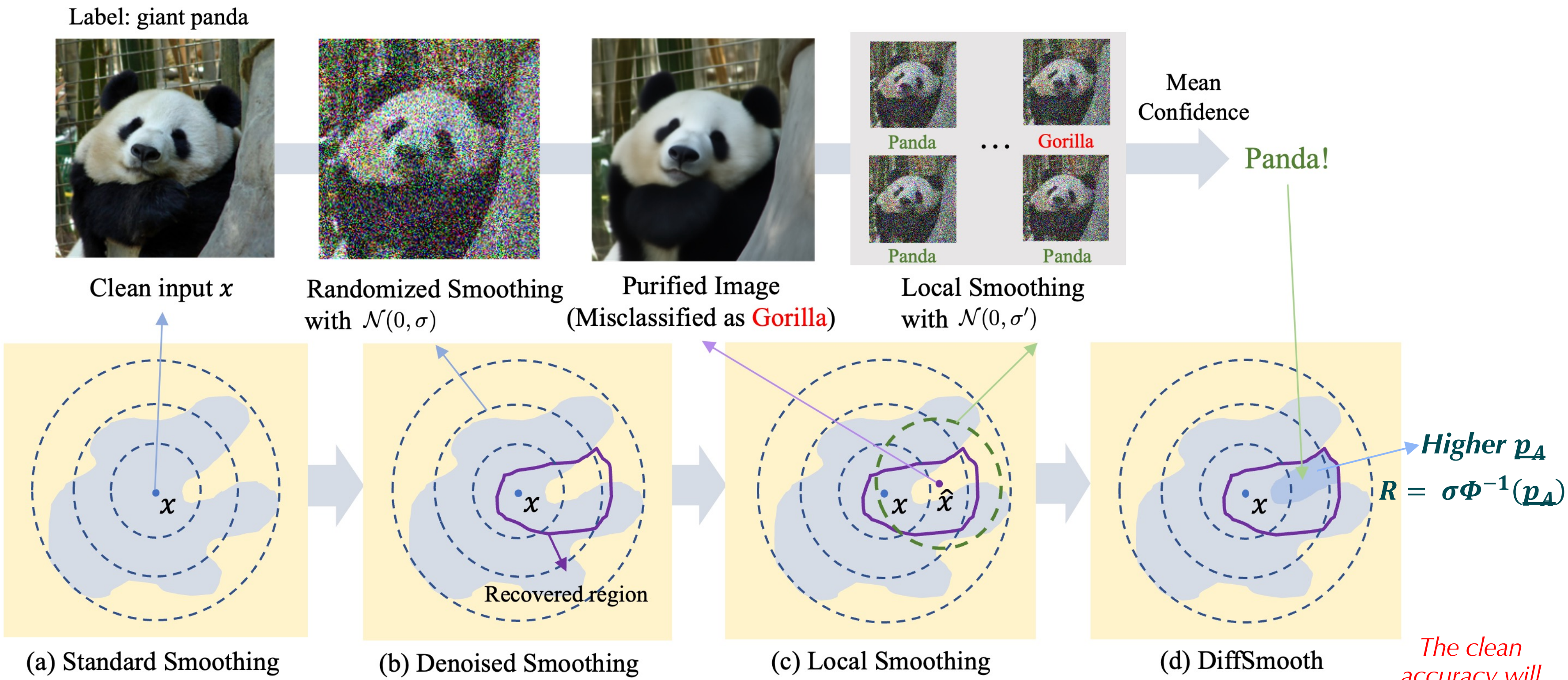


For a *given* $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$, $\quad f(x+\varepsilon) = f_{clf}\big(D_\theta(x+\varepsilon)\big) = f_{clf}(\hat{x})$

$$\Rightarrow f(x+\varepsilon) = \operatorname{argmax} \mathbb{E}\big(\boxed{F_{robust}}\big(D_\theta(x+\varepsilon)+\varepsilon'\big)\big) = \operatorname{argmax} \mathbb{E}\big(\boxed{F_{robust}}(\hat{x}+\varepsilon')\big),\ \ \varepsilon' \sim \mathcal{N}(0, \sigma'^2 I),\ \ \sigma' < \sigma$$

(in practice, we just use m fixed sampled ε' for approximating the expectation here.)

Pipeline:



The clean accuracy will also be higher!

For a *given* $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$, $f(x+\varepsilon) = f_{clf}\left(D_\theta(x+\varepsilon)\right) = f_{clf}(\hat{x})$

$$\Rightarrow f(x+\varepsilon) = \operatorname{argmax}\, \mathbb{E}\left(\boxed{F_{robust}}\left(D_\theta(x+\varepsilon)+\varepsilon'\right)\right) = \operatorname{argmax}\, \mathbb{E}\left(\boxed{F_{robust}}(\hat{x}+\varepsilon')\right),\ \varepsilon' \sim \mathcal{N}\left(0, \sigma'^2 I\right),\ \sigma' < \sigma$$

(in practice, we just use m fixed sampled ε' for approximating the expectation here.)

CIFAR10:

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

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

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

CIFAR10:

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

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

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

CIFAR10:

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

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

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

CIFAR10:

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

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

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

ImageNet:

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

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

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

ImageNet:

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

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

[1] We report the results for Gaussian and SmoothAdv based on pretrained models with the same number of smoothing noise for evaluating DiffSmooth ($N = 10,000$) for a fair comparison.

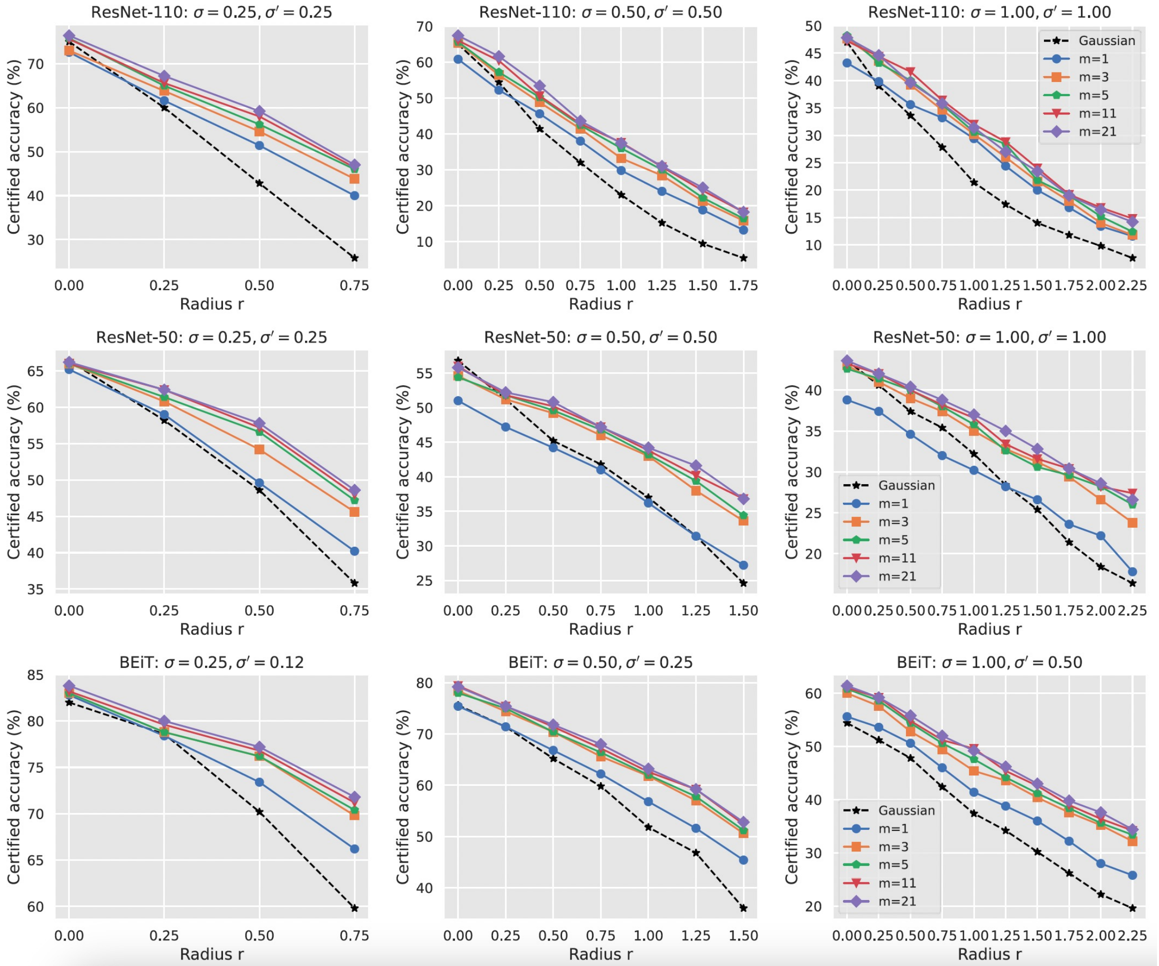
[2] Boosting is an ensemble method with the base models trained under *Gaussian*, *SmoothAdv*, *Consistency* and *MACER*.

[3] The authors use a pretrained BEiT large model [4] in the original paper, and we reimplement DDS on ResNet-50 here and report the results.

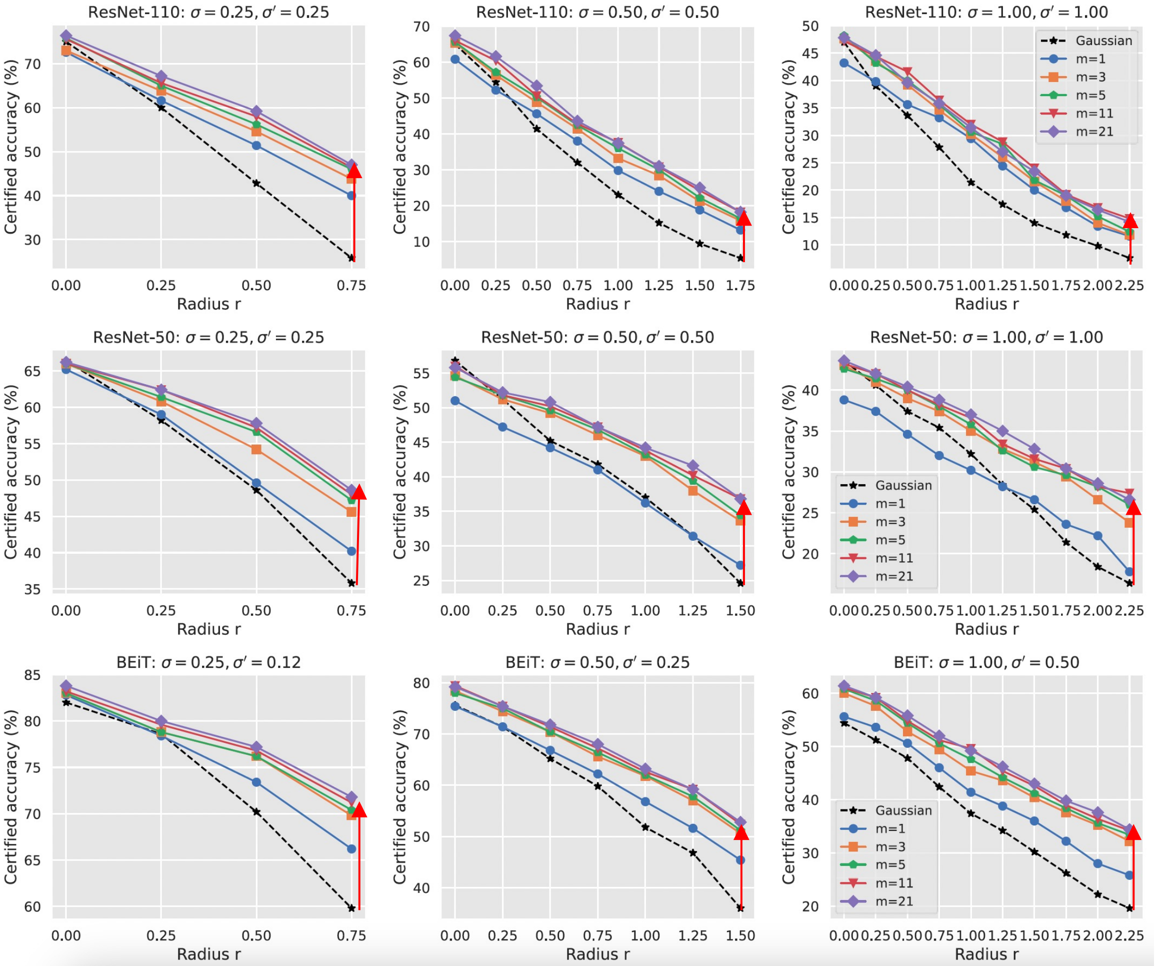
[4] We use the same smoothed models (i.e., Gaussian and SmoothAdv) used in DiffSmooth for DDS and report the best results.

Ablation study (the choice of m: #local smoothing noise):

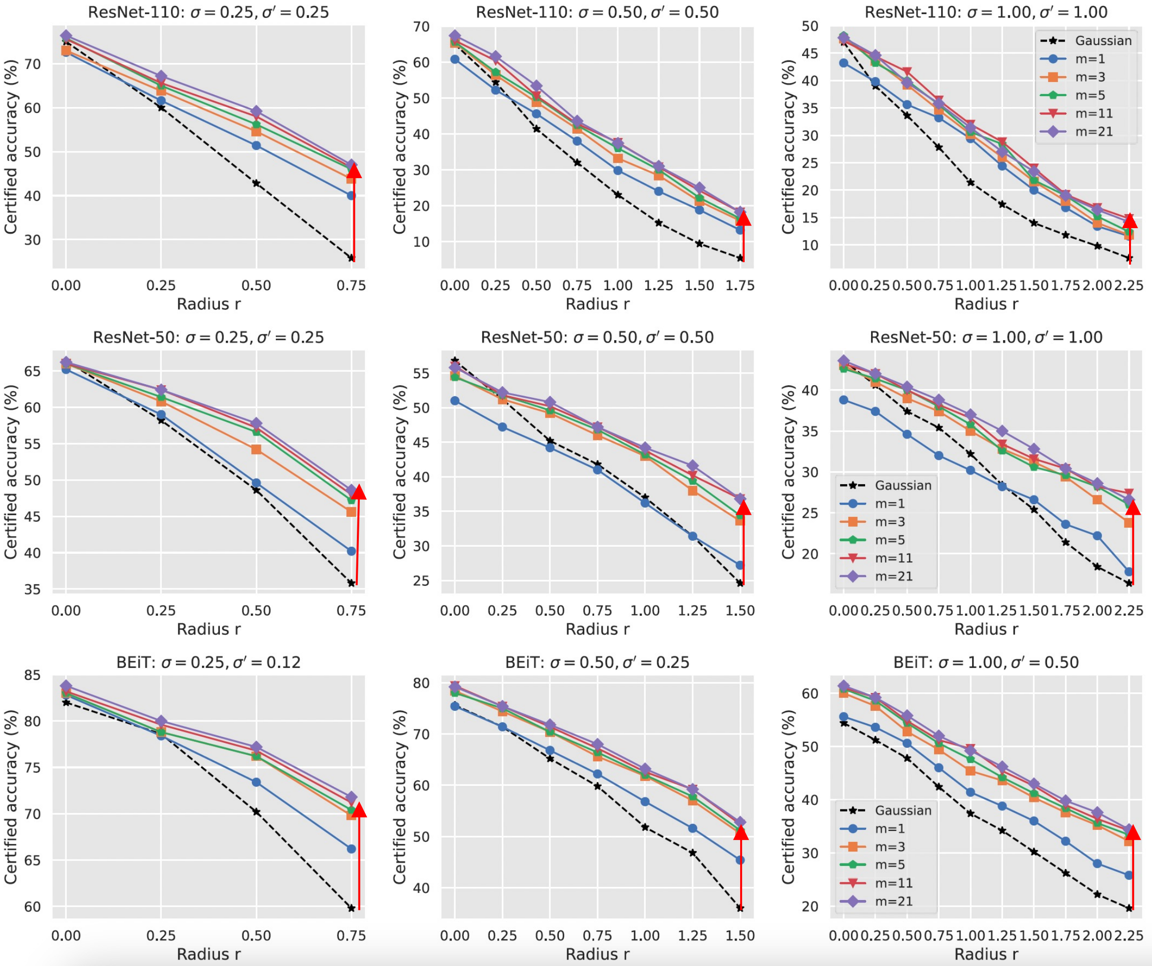
Ablation study (the choice of *m*: #local smoothing noise):



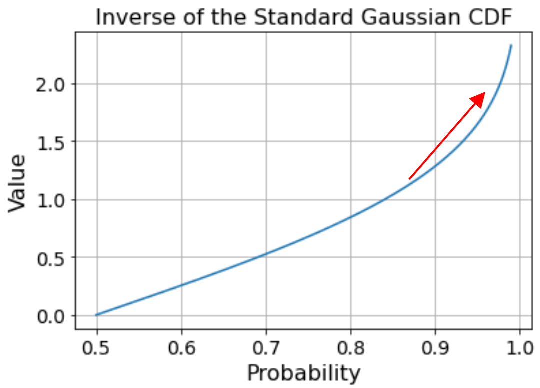
Ablation study (the choice of m: #local smoothing noise):



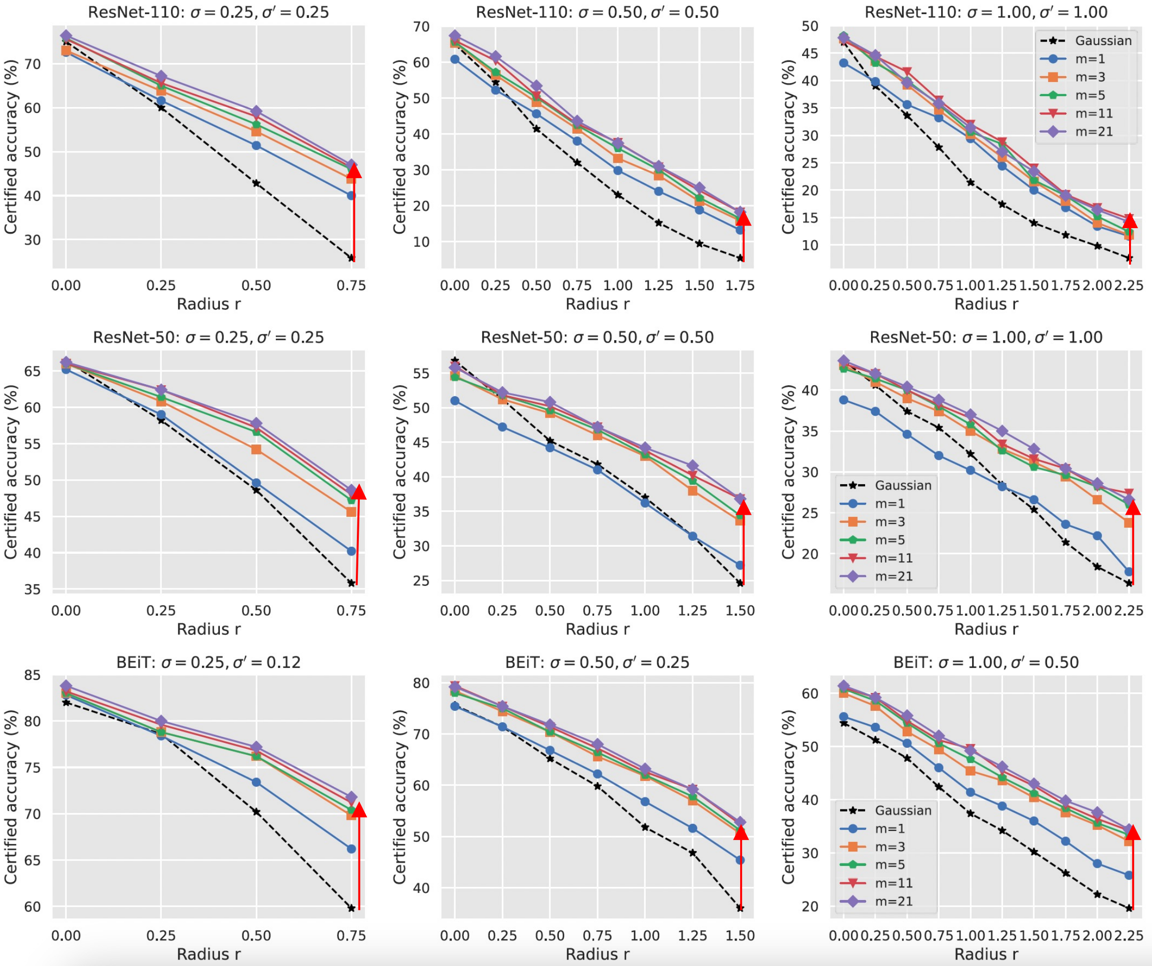
Ablation study (the choice of m: #local smoothing noise):



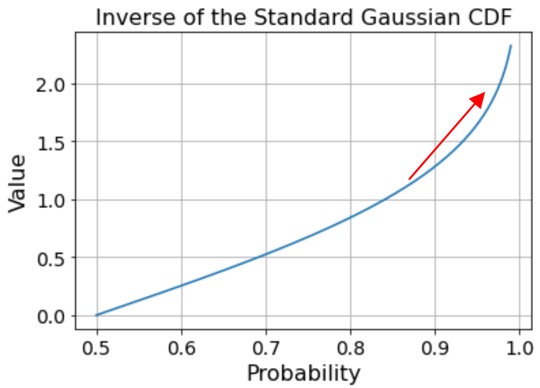
$$R = \sigma\Phi^{-1}(\underline{p_A})$$

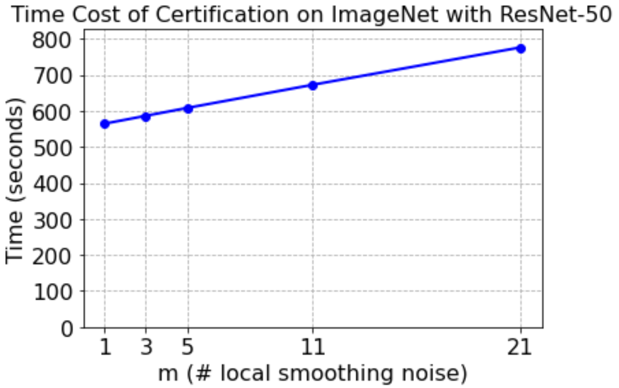


Ablation study (the choice of m: #local smoothing noise):

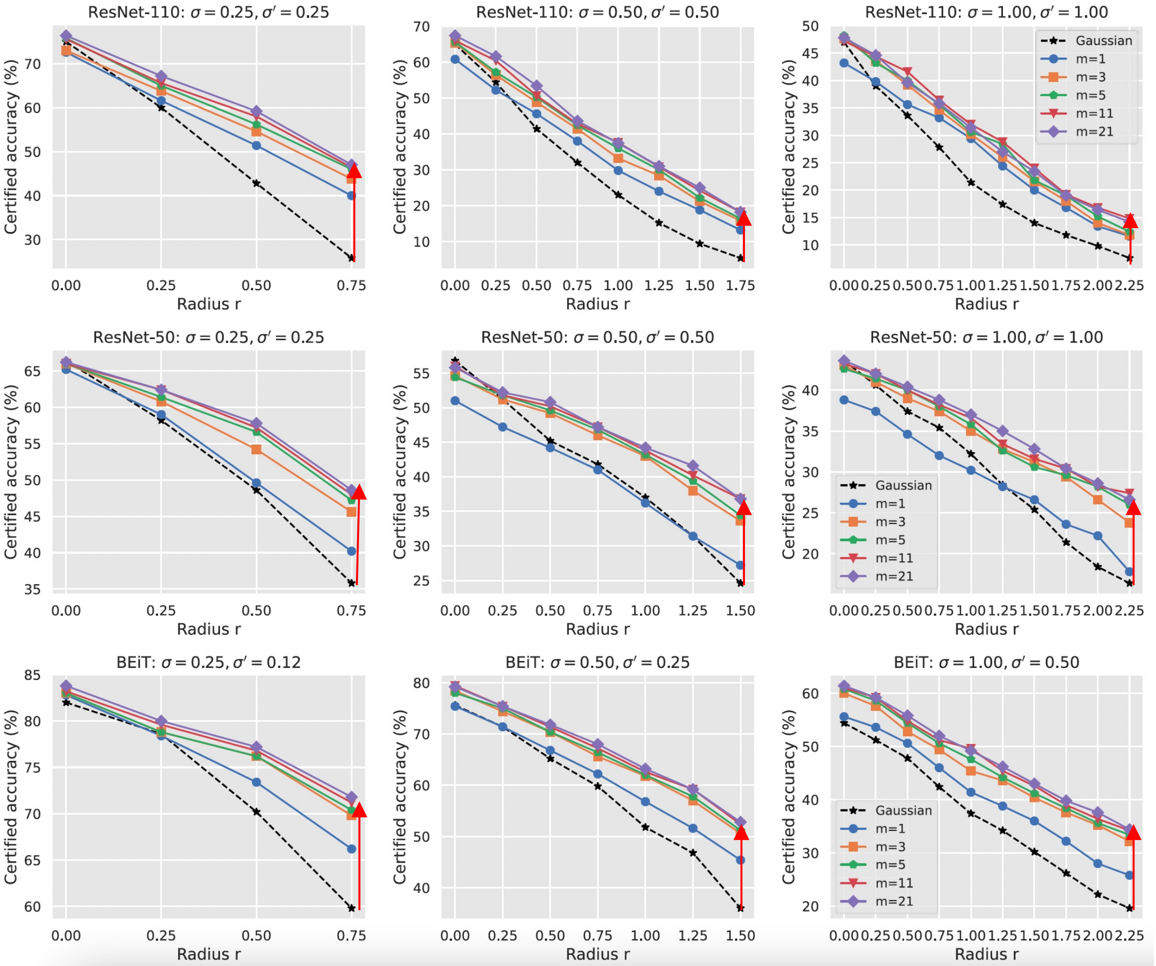


$$R = \sigma\Phi^{-1}(\underline{p_A})$$

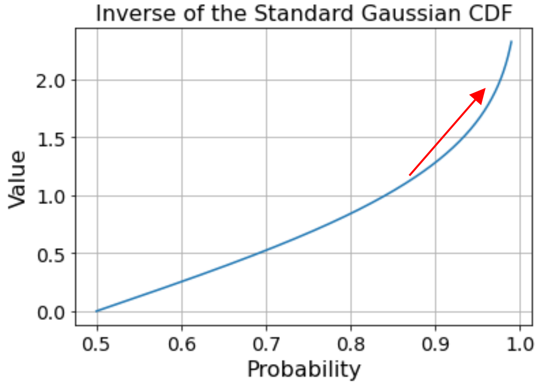


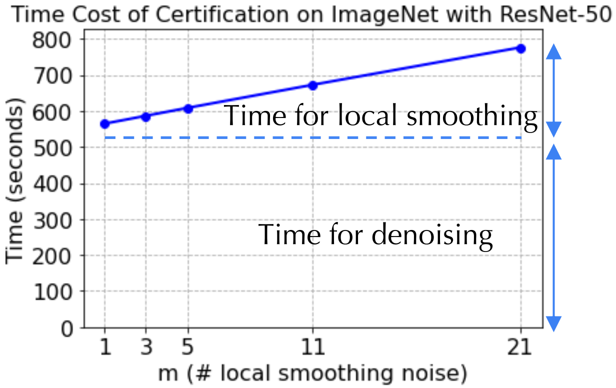


Ablation study (the choice of m: #local smoothing noise):



$$R = \sigma\Phi^{-1}(\underline{p_A})$$





Ablation study (same computation cost)-CIFAR10:

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

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

Ablation study (same computation cost)-CIFAR10:

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

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

We can still achieve much better performance even with the model simply trained with Gaussian Augmentation.

Ablation study (same computation cost)-ImageNet:

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

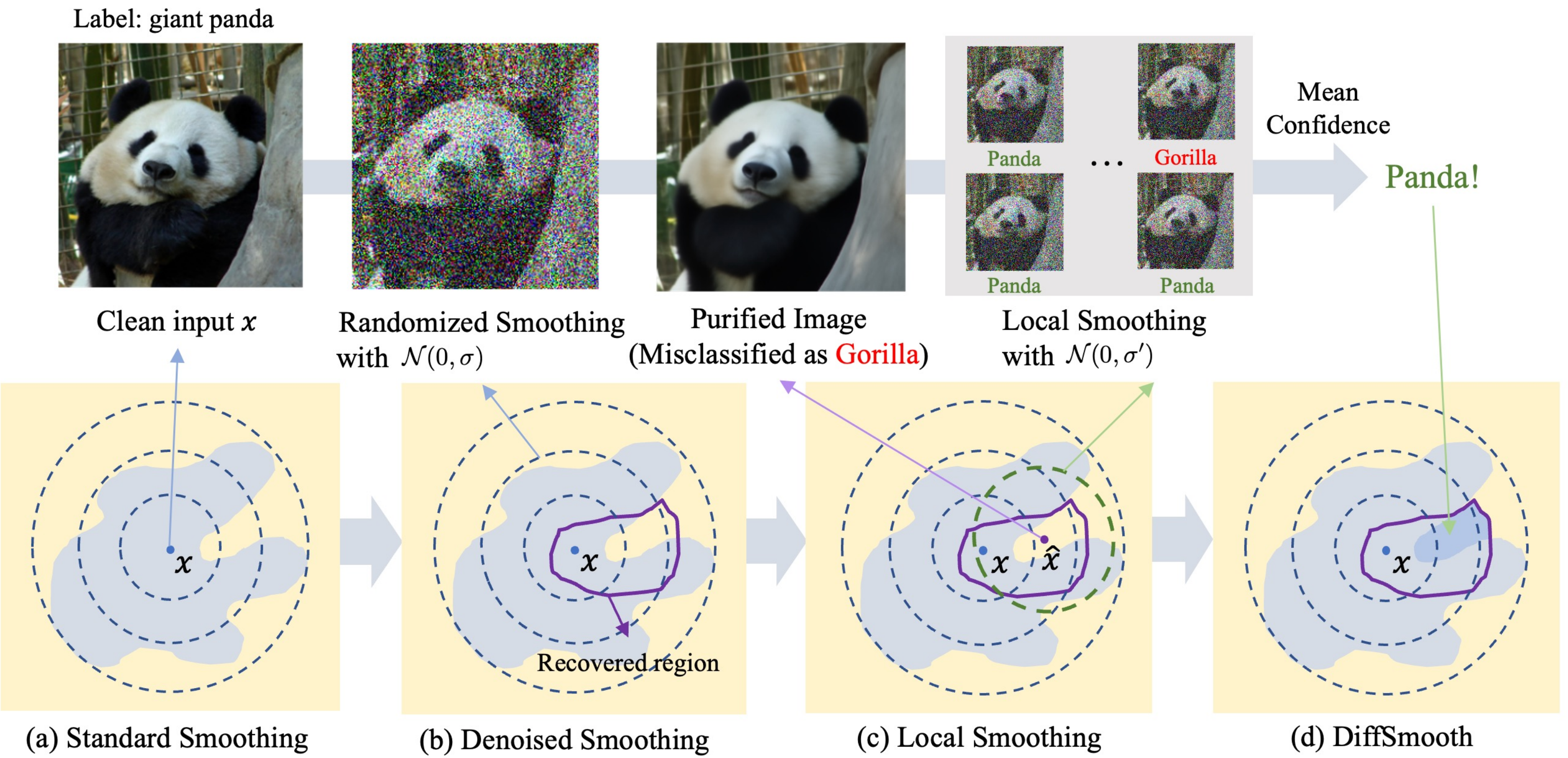
Architecture	Method	Setting	Certified Accuracy (%) under ℓ_2 Radius r					
			0.00	0.50	1.00	1.50	2.00	2.50
ResNet-50	DDS(Standard)	$N = 10,000$	**67.4**	49.0	33.0	22.2	17.4	12.8
	DDS(Smoothed)	$N = 10,000$	48.0	40.6	29.6	23.8	18.6	16.0
	DiffSmooth(Gaussian)	$N = 2,000, m = 5$	65.4	54.8	42.4	30.2	26.8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

We can still achieve much better performance even with the model simply trained with Gaussian Augmentation.

Summary

- We prove that the purified adversarial instances of diffusion models 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 propose an effective and certifiably robust pipeline for smoothed classifiers, DiffSmooth, via denoising and local smoothing.
- We show that naively combining diffusion models with smoothed models cannot effectively improve their certifiable robustness (local smoothing is crucial here).
- We achieve the SOTA certified accuracy on both CIFAR-10 and ImageNet.

Future work:

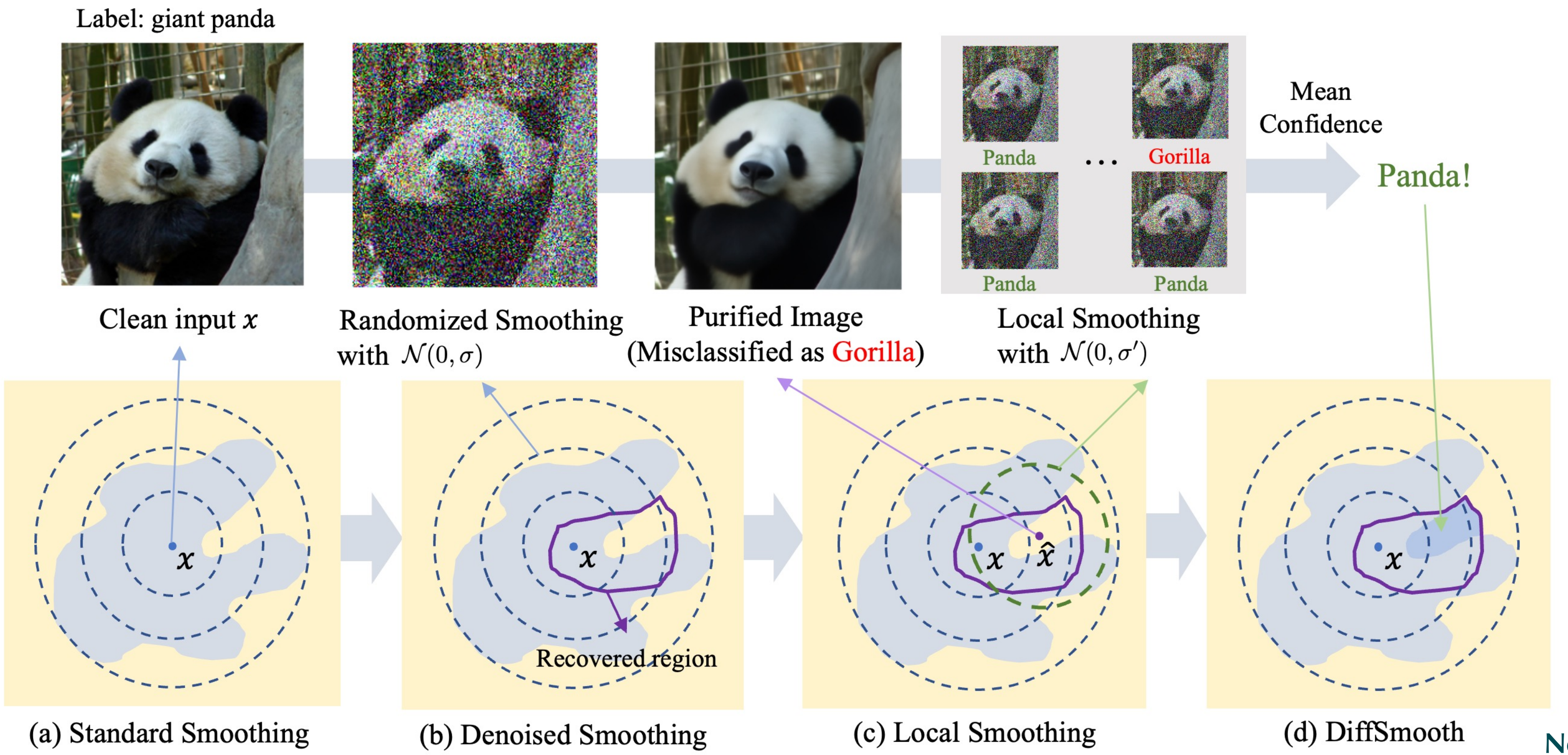


For a *given* $\varepsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$, $\quad f(x+\varepsilon) = f_{clf}\big(D_\theta(x+\varepsilon)\big) = f_{clf}(\hat{x})$

$$\Rightarrow f(x+\varepsilon) = \operatorname{arg}max\ \mathbb{E}\big(F_{robust}(D_\theta(x+\varepsilon)+\varepsilon')\big) = \operatorname{arg}max\ \mathbb{E}(F_{robust}(\hat{x}+\varepsilon')),\ \ \varepsilon' \sim \mathcal{N}\left(0, \sigma'^2 I\right),\ \sigma' < \sigma$$

(in practice, we just use m fixed sampled ε' for approximating the expectation here.)

Future work:



For a *given* $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$, $\quad f(x+\varepsilon) = f_{clf}(D_\theta(x+\varepsilon)) = f_{clf}(\hat{x})$

No need to be Gaussian…

$$\Rightarrow f(x+\varepsilon) = \mathrm{argmax}\ \mathbb{E}(F_{robust}(D_\theta(x+\varepsilon)+\varepsilon')) = \mathrm{argmax}\ \mathbb{E}(F_{robust}(\hat{x}+\varepsilon')),\ \boxed{\varepsilon' \sim \mathcal{N}(0, \sigma'^2 I),\ \sigma' < \sigma}$$

(in practice, we just use m fixed sampled ε′ for approximating the expectation here.)