A. Proofs of Theorems 1 and 2

Here we provide the complete proofs for Theorem 1 and Theorem 2. We fist prove the following lemma, which is essentially a restatement of the Neyman-Pearson lemma (Neyman & Pearson, 1933) from statistical hypothesis testing.

Lemma 3 (**Neyman-Pearson**). *Let X and Y be random variables in $\mathbb{R}^d$ with densities μ_X and μ_Y. Let $h : \mathbb{R}^d \to \{0, 1\}$ be a random or deterministic function. Then:*

1. *If $S = \left\{ z \in \mathbb{R}^d : \frac{\mu_Y(z)}{\mu_X(z)} \leq t \right\}$ for some $t > 0$ and $\mathbb{P}(h(X) = 1) \geq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \geq \mathbb{P}(Y \in S)$.*
2. *If $S = \left\{ z \in \mathbb{R}^d : \frac{\mu_Y(z)}{\mu_X(z)} \geq t \right\}$ for some $t > 0$ and $\mathbb{P}(h(X) = 1) \leq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \leq \mathbb{P}(Y \in S)$.*

Proof. Without loss of generality, we assume that h is random and write $h(1|x)$ for the probability that $h(x) = 1$.

First we prove part 1. We denote the complement of S as S^c.

$$\begin{aligned}
\mathbb{P}(h(Y) = 1) - \mathbb{P}(Y \in S) &= \int_{\mathbb{R}^d} h(1|z)\, \mu_Y(z) dz - \int_S \mu_Y(z) dz \\
&= \left[\int_{S^c} h(1|z) \mu_Y(z) dz + \int_S h(1|z) \mu_Y(z) dx \right) - \left(\int_S h(1|z) \mu_Y(z) dz + \int_S h(0|z) \mu_Y(z) dz \right) \\
&= \int_{S^c} h(1|z) \mu_Y(z) dz - \int_S h(0|z) \mu_Y(z) dz \\
&\geq t \left[\int_{S^c} h(1|z) \mu_X(z) dz - \int_S h(0|z) \mu_X(z) \right] \\
&= t \left(\int_{S^c} h(1|z) \mu_X(z) dz + \int_S h(1|z) \mu_X(z) dz - \int_S h(1|z) \mu_X(z) dz - \int_S h(0|z) \mu_X(z) \right] \\
&= t \left[\int_{\mathbb{R}^d} h(1|z) \mu_X(z) dz - \int_S \mu_X(z) dz \right] \\
&= t \left[\mathbb{P}(h(X) = 1) - \mathbb{P}(X \in S) \right] \\
&\geq 0
\end{aligned}$$

The inequality in the middle is due to the fact that $\mu_Y(z) \leq t\, \mu_X(z)\ \forall z \in S$ and $\mu_Y(z) > t\, \mu_X(z)\ \forall z \in S^c$. The inequality at the end is because both terms in the product are non-negative by assumption.

The proof for part 2 is virtually identical, except both "$\geq$" become "$\leq$." □

Remark: connection to statistical hypothesis testing. Part 2 of Lemma 3 is known in the field of statistical hypothesis testing as the Neyman-Pearson Lemma (Neyman & Pearson, 1933). The hypothesis testing problem is this: we are given a sample that comes from one of two distributions over $\mathbb{R}^d$: either the null distribution X or the alternative distribution Y. We would like to identify which distribution the sample came from. It is worse to say "Y" when the true answer is "X" than to say "X" when the true answer is "Y." Therefore we seek a (potentially randomized) procedure $h : \mathbb{R}^d \to \{0, 1\}$ which returns "Y" when the sample really came from X with probability no greater than some failure rate α. In particular, out of all such rules h, we would like the *uniformly most powerful* one h^*, i.e. the rule which is most likely to correctly say "Y" when the sample really came from Y. Neyman & Pearson (1933) showed that h^* is the rule which returns "Y" deterministically on the set $S^* = \{z \in \mathbb{R}^d : \frac{\mu_Y(z)}{\mu_X(z)} \geq t\}$ for whichever t makes $\mathbb{P}(X \in S^*) = \alpha$. In other words, to state this in a form that looks like Part 2 of Lemma 3: if h is a different rule with $\mathbb{P}(h(X) = 1) \leq \alpha$, then h^* is more powerful than h, i.e. $\mathbb{P}(h(Y) = 1) \leq \mathbb{P}(Y \in S^*)$.

Now we state the special case of Lemma 3 for when X and Y are isotropic Gaussians.

Lemma 4. *Let $X \sim \mathcal{N}(x, \sigma^2 I)$ and $Y \sim \mathcal{N}(x + \delta, \sigma^2 I)$. Let $h : \mathbb{R}^d \to \{0, 1\}$ be any deterministic or random function. Then:*

1. If $S = \left\{ z \in \mathbb{R}^d : \delta^T z \leq \beta \right\}$ for some β and $\mathbb{P}(h(X) = 1) \geq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \geq \mathbb{P}(Y \in S)$

2. If $S = \left\{z \in \mathbb{R}^d : \delta^T z \geq \beta\right\}$ for some β and $\mathbb{P}(h(X)=1) \leq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y)=1) \leq \mathbb{P}(Y \in S)$

Proof. This lemma is the special case of Lemma 3 when X and Y are isotropic Gaussians with means x and $x+\delta$.

By Lemma 3 it suffices to simply show that for any β, there is some $t > 0$ for which:

$$\{z : \delta^T z \leq \beta\} = \left\{z : \frac{\mu_Y(z)}{\mu_X(z)} \leq t\right\} \quad \text{and} \quad \{z : \delta^T z \geq \beta\} = \left\{z : \frac{\mu_Y(z)}{\mu_X(z)} \geq t\right\} \tag{5}$$

The likelihood ratio for this choice of X and Y turns out to be:

$$\begin{aligned}\frac{\mu_Y(z)}{\mu_X(z)} &= \frac{\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^d (z_i - (x_i+\delta_i))^2\right)}{\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^d (z_i - x_i)^2\right)} \\ &= \exp\left(\frac{1}{2\sigma^2}\sum_{i=1}^d 2z_i\delta_i - \delta_i^2 - 2x_i\delta_i\right) \\ &= \exp(a\delta^T z + b)\end{aligned}$$

where $a > 0$ and b are constants w.r.t z, specifically $a = \frac{1}{\sigma^2}$ and $b = \frac{-(2\delta^T x + \|\delta\|^2)}{2\sigma^2}$.

Therefore, given any β we may take $t = \exp(a\beta + b)$, noticing that

$$\begin{aligned}\delta^T z \leq \beta &\iff \exp(a\delta^T z + b) \leq t \\ \delta^T z \geq \beta &\iff \exp(a\delta^T z + b) \geq t\end{aligned}$$

□

Finally, we prove Theorem 1 and Theorem 2.

Theorem 1 (restated). *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be any deterministic or random function. Let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. Let $g(x) = \arg\max_c \mathbb{P}(f(x+\varepsilon) = c)$. Suppose that for a specific $x \in \mathbb{R}^d$, there exist $c_A \in \mathcal{Y}$ and $\underline{p_A}, \overline{p_B} \in [0,1]$ such that:*

$$\mathbb{P}(f(x+\varepsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(f(x+\varepsilon) = c) \tag{6}$$

Then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R$, where

$$R = \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \tag{7}$$

Proof. To show that $g(x+\delta) = c_A$, it follows from the definition of g that we need to show that

$$\mathbb{P}(f(x+\delta+\varepsilon) = c_A) > \max_{c_B \neq c_A} \mathbb{P}(f(x+\delta+\varepsilon) = c_B)$$

We will prove that $\mathbb{P}(f(x+\delta+\varepsilon) = c_A) > \mathbb{P}(f(x+\delta+\varepsilon) = c_B)$ for every class $c_B \neq c_A$. Fix one such class c_B without loss of generality.

For brevity, define the random variables

$$\begin{aligned}X &:= x + \varepsilon = \mathcal{N}(x, \sigma^2 I) \\ Y &:= x + \delta + \varepsilon = \mathcal{N}(x+\delta, \sigma^2 I)\end{aligned}$$

In this notation, we know from (6) that

$$\mathbb{P}(f(X) = c_A) \geq \underline{p_A} \quad \text{and} \quad \mathbb{P}(f(X) = c_B) \leq \overline{p_B} \tag{8}$$

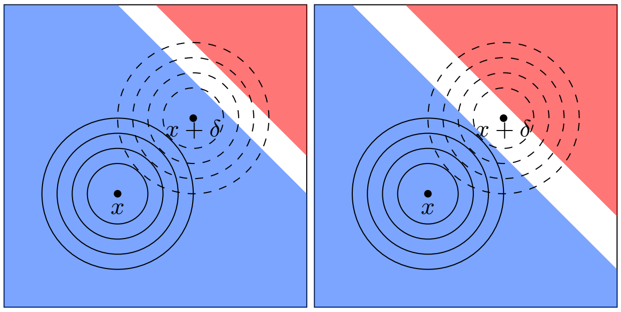


Figure 9. Illustration of the proof of Theorem 1. The solid line concentric circles are the density level sets of $X := x + \varepsilon$; the dashed line concentric circles are the level sets of $Y := x + \delta + \varepsilon$. The set A is in blue and the set B is in red. The figure on the left depicts a situation where $\mathbb{P}(Y \in A) > \mathbb{P}(Y \in B)$, and hence $g(x + \delta)$ may equal c_A. The figure on the right depicts a situation where $\mathbb{P}(Y \in A) < \mathbb{P}(Y \in B)$ and hence $g(x + \delta) \neq c_A$.

and our goal is to show that

$$\mathbb{P}(f(Y) = c_A) > \mathbb{P}(f(Y) = c_B) \tag{9}$$

Define the half-spaces:

$$A := \{z : \delta^T (z - x) \leq \sigma \|\delta\| \Phi^{-1}(\underline{p_A})\}$$
$$B := \{z : \delta^T (z - x) \geq \sigma \|\delta\| \Phi^{-1}(1 - \overline{p_B})\}$$

Algebra (deferred to the end) shows that $\mathbb{P}(X \in A) = \underline{p_A}$. Therefore, by (8) we know that $\mathbb{P}(f(X) = c_A) \geq \mathbb{P}(X \in A)$. Hence we may apply Lemma 4 with $h(z) := \mathbf{1}[f(z) = c_A]$ to conclude:

$$\mathbb{P}(f(Y) = c_A) \geq \mathbb{P}(Y \in A) \tag{10}$$

Similarly, algebra shows that $\mathbb{P}(X \in B) = \overline{p_B}$. Therefore, by (8) we know that $\mathbb{P}(f(X) = c_B) \leq \mathbb{P}(X \in B)$. Hence we may apply Lemma 4 with $h(z) := \mathbf{1}[f(z) = c_B]$ to conclude:

$$\mathbb{P}(f(Y) = c_B) \leq \mathbb{P}(Y \in B) \tag{11}$$

To guarantee (9), we see from (10, 11) that it suffices to show that $\mathbb{P}(Y \in A) > \mathbb{P}(Y \in B)$, as this step completes the chain of inequalities

$$\mathbb{P}(f(Y) = c_A) \geq \mathbb{P}(Y \in A) > \mathbb{P}(Y \in B) \geq \mathbb{P}(f(Y) = c_B) \tag{12}$$

We can compute the following:

$$\mathbb{P}(Y \in A) = \Phi\left(\Phi^{-1}(\underline{p_A}) - \frac{\|\delta\|}{\sigma}\right) \tag{13}$$
$$\mathbb{P}(Y \in B) = \Phi\left(\Phi^{-1}(\overline{p_B}) + \frac{\|\delta\|}{\sigma}\right) \tag{14}$$

Finally, algebra shows that $\mathbb{P}(Y \in A) > \mathbb{P}(Y \in B)$ if and only if:

$$\|\delta\| < \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \tag{15}$$

which recovers the theorem statement. □

We now restate and prove Theorem 2, which shows that the bound in Theorem 1 is tight. The assumption below in Theorem 2 that $\underline{p_A} + \overline{p_B} \leq 1$ is mild: given any $\underline{p_A}$ and $\overline{p_B}$ which do not satisfy this condition, one could have always redefined $\overline{p_B} \leftarrow 1 - \underline{p_A}$ to obtain a Theorem 1 guarantee with a larger certified radius, so there is no reason to invoke Theorem 1 unless $\underline{p_A} + \overline{p_B} \leq 1$.

Theorem 2 (restated). *Assune $\underline{p_A} + \overline{p_B} \leq 1$. For any perturbation $\delta \in \mathbb{R}^d$ with $\|\delta\|_2 > R$, there exists a base classifier f^* consistent with the observed class probabilities (6) such that if f^* is the base classifier for g, then $g(x+\delta) \neq c_A$.*

Proof. We re-use notation from the preceding proof.

Pick any class c_B arbitrarily. Define A and B as above, and consider the function

$$f^*(x) := \begin{cases} c_A & \text{if } x \in A \\ c_B & \text{if } x \in B \\ \text{other classes} & \text{otherwise} \end{cases}$$

This function is well-defined, since $A \cap B = \emptyset$ provided that $\underline{p_A} + \overline{p_B} \leq 1$.

By construction, the function f^* satisfies (6) with equalities, since

$$\mathbb{P}(f^*(x+\varepsilon) = c_A) = \mathbb{P}(X \in A) = \underline{p_A} \qquad \mathbb{P}(f^*(x+\varepsilon) = c_B) = \mathbb{P}(X \in B) = \overline{p_B}$$

It follows from (13) and (14) that

$$\mathbb{P}(Y \in A) < \mathbb{P}(Y \in B) \iff \|\delta\|_2 > R$$

By assumption, $\|\delta\|_2 > R$, so $\mathbb{P}(Y \in A) < \mathbb{P}(Y \in B)$, or equivalently,

$$\mathbb{P}(f^*(x+\delta+\varepsilon) = c_A) < \mathbb{P}(f^*(x+\delta+\varepsilon) = c_B)$$

Therefore, if f^* is the base classifier for g, then $g(x+\delta) \neq c_A$. □

A.0.1. Deferred Algebra

Claim. $\mathbb{P}(X \in A) = \underline{p_A}$

Proof. Recall that $X \sim \mathcal{N}(x, \sigma^2 I)$ and $A = \{z : \delta^T(z-x) \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})\}$.

$$\begin{aligned} \mathbb{P}(X \in A) &= \mathbb{P}(\delta^T(X-x) \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})) \\ &= \mathbb{P}(\delta^T\mathcal{N}(0,\sigma^2 I) \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})) \\ &= \mathbb{P}(\sigma\|\delta\|Z \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})) && (Z \sim \mathcal{N}(0,1)) \\ &= \Phi(\Phi^{-1}(\underline{p_A})) \\ &= \underline{p_A} \end{aligned}$$

□

Claim. $\mathbb{P}(X \in B) = \overline{p_B}$

Proof. Recall that $X \sim \mathcal{N}(x, \sigma^2 I)$ and $B = \{z : \delta^T(z-x) \leq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})\}$.

$$\begin{aligned} \mathbb{P}(X \in A) &= \mathbb{P}(\delta^T(X-x) \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) \\ &= \mathbb{P}(\delta^T\mathcal{N}(0,\sigma^2 I) \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) \\ &= \mathbb{P}(\sigma\|\delta\|Z \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) && (Z \sim \mathcal{N}(0,1)) \\ &= \mathbb{P}(Z \geq \Phi^{-1}(1-\overline{p_B})) \\ &= 1 - \Phi(\Phi^{-1}(1-\overline{p_B})) \\ &= \overline{p_B} \end{aligned}$$

□

Claim. $\mathbb{P}(Y \in A) = \Phi\left(\Phi^{-1}(\underline{p_A}) - \frac{\|\delta\|}{\sigma}\right)$

Proof. Recall that $Y \sim \mathcal{N}(x+\delta, \sigma^2 I)$ and $A = \{z : \delta^T(z-x) \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})\}$.

$$\begin{aligned}
\mathbb{P}(Y \in A) &= \mathbb{P}(\delta^T(Y-x) \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})) \\
&= \mathbb{P}(\delta^T\mathcal{N}(0,\sigma^2 I) + \|\delta\|^2 \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A})) \\
&= \mathbb{P}(\sigma\|\delta\|Z \leq \sigma\|\delta\|\Phi^{-1}(\underline{p_A}) - \|\delta\|^2) && (Z \sim \mathcal{N}(0,1)) \\
&= \mathbb{P}\left(Z \leq \Phi^{-1}(\underline{p_A}) - \frac{\|\delta\|}{\sigma}\right) \\
&= \Phi\left(\Phi^{-1}(\underline{p_A}) - \frac{\|\delta\|}{\sigma}\right)
\end{aligned}$$

□

Claim. $\mathbb{P}(Y \in B) = \Phi\left(\Phi^{-1}(\overline{p_B}) + \frac{\|\delta\|}{\sigma}\right)$

Proof. Recall that $Y \sim \mathcal{N}(x+\delta, \sigma^2 I)$ and $B = \{z : \delta^T(z-x) \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})\}$.

$$\begin{aligned}
\mathbb{P}(Y \in B) &= \mathbb{P}(\delta^T(Y-x) \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) \\
&= \mathbb{P}(\delta^T\mathcal{N}(0,\sigma^2 I) + \|\delta\|^2 \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) \\
&= \mathbb{P}(\sigma\|\delta\|Z + \|\delta\|^2 \geq \sigma\|\delta\|\Phi^{-1}(1-\overline{p_B})) && (Z \sim \mathcal{N}(0,1)) \\
&= \mathbb{P}\left(Z \geq \Phi^{-1}(1-\overline{p_B}) - \frac{\|\delta\|}{\sigma}\right) \\
&= \mathbb{P}\left(Z \leq \Phi^{-1}(\overline{p_B}) + \frac{\|\delta\|}{\sigma}\right) \\
&= \Phi\left(\Phi^{-1}(\overline{p_B}) + \frac{\|\delta\|}{\sigma}\right)
\end{aligned}$$

□

B. Smoothing a two-class linear classifier

In this appendix, we analyze what happens when the base classifier f is a two-class linear classifier $f(x) = \text{sign}(w^T x + b)$. For mathematical convenience, we take $\text{sign}(0)$ to be undefined (to match the definition of g).

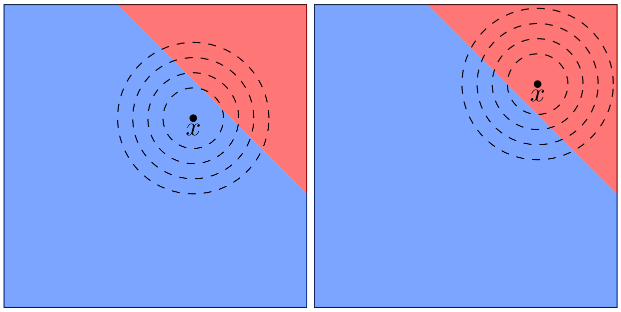


Figure 10. Illustration of Proposition 3. A binary linear classifier $f(x) = \text{sign}(w^T x + b)$ partitions $\mathbb{R}^d$ into two half-spaces, drawn here in blue and red. An isotropic Gaussian $\mathcal{N}(x, \sigma^2 I)$ will put more mass on whichever half-space its center x lies in: in the figure on the left, x is in the blue half-space and $\mathcal{N}(x, \sigma^2 I)$ puts more mass on the blue than on red. In the figure on the right, x is in the red half-space and $\mathcal{N}(x, \sigma^2 I)$ puts more mass on red than on blue. Since the smoothed classifier's prediction $g(x)$ is defined to be whichever half-space $\mathcal{N}(x, \sigma^2 I)$ puts more mass in, and the base classifier's prediction $f(x)$ is defined to be whichever half-space x is in, we have that $g(x) = f(x)$ for all x.

Proposition 3. *If f is a two-class linear classifier $f(x) = \text{sign}(w^T x + b)$, and g is the smoothed version of f with any σ, then $g(x) = f(x)$ for all x.*

Proof. By the definition of g, we know that $g(x) = 1$ if and only if:

$$\begin{aligned}
g(x) = 1 &\iff \mathbb{P}_\varepsilon(f(x+\varepsilon) = 1) \geq \frac{1}{2} && (\varepsilon \sim \mathcal{N}(0, \sigma^2 I)) \\
&\iff \mathbb{P}_\varepsilon\left(\text{sign}(w^T(x+\varepsilon) + b) = 1\right) \geq \frac{1}{2} \\
&\iff \mathbb{P}_\varepsilon\left(w^T x + w^T \varepsilon + b \geq 0\right) \geq \frac{1}{2} \\
&\iff \mathbb{P}\left(\sigma\|w\| Z \geq -w^T x - b\right) \geq \frac{1}{2} && (Z \sim \mathcal{N}(0,1)) \\
&\iff \mathbb{P}\left(Z \leq \frac{w^T x + b}{\sigma\|w\|}\right) \geq \frac{1}{2} \\
&\iff \frac{w^T x + b}{\sigma\|w\|} \geq 0 \\
&\iff w^T x + b \geq 0 \\
&\iff f(x) = 1
\end{aligned}$$

□

Proposition 4. *If f is a two-class linear classifier $f(x) = \text{sign}(w^T x + b)$, and g is the smoothed version of f with any σ, then invoking Theorem 1 at any x with $\underline{p_A} = p_A$ and $\overline{p_B} = p_B$ will yield the certified radius $R = \frac{|w^T x + b|}{\|w\|}$.*

Proof. In binary classification, $p_A = 1 - p_B$, so Theorem 1 returns $R = \sigma \Phi^{-1}(\underline{p_A})$.

We have:

$$\begin{aligned} p_A &= \mathbb{P}_\varepsilon(f(x+\varepsilon) = g(x)) \\ &= \mathbb{P}_\varepsilon(\text{sign}(w^T(x+\varepsilon)+b) = \text{sign}(w^Tx+b)) && \text{(By Proposition 3, } g(x) = f(x)\text{)} \\ &= \mathbb{P}_\varepsilon(\text{sign}(w^Tx + \sigma\|w\|Z + b) = \text{sign}(w^Tx+b)) \end{aligned}$$

There are two cases: if $w^Tx + b \geq 0$, then

$$\begin{aligned} p_A &= \mathbb{P}_\varepsilon(w^Tx + \sigma\|w\|Z + b \geq 0) \\ &= \mathbb{P}_\varepsilon\left(Z \geq \frac{-w^Tx - b}{\sigma\|w\|}\right) \\ &= \mathbb{P}_\varepsilon\left(Z \leq \frac{w^Tx + b}{\sigma\|w\|}\right) \\ &= \Phi\left(\frac{w^Tx + b}{\sigma\|w\|}\right) \end{aligned}$$

On the other hand, if $w^Tx + b < 0$, then

$$\begin{aligned} p_A &= \mathbb{P}_\varepsilon(w^Tx + \sigma\|w\|Z + b < 0) \\ &= \mathbb{P}_\varepsilon\left(Z < \frac{-w^Tx - b}{\sigma\|w\|}\right) \\ &= \Phi\left(\frac{-w^Tx - b}{\sigma\|w\|}\right) \end{aligned}$$

In either case, we have:

$$p_A = \Phi\left(\frac{|w^Tx + b|}{\sigma\|w\|}\right)$$

Therefore, the bound in Theorem 1 returns a radius of

$$\begin{aligned} R &= \sigma\Phi^{-1}(p_A) \\ &= \frac{|w^Tx + b|}{\|w\|} \end{aligned}$$

□

Proposition 5. *Let f be a two-class linear classifier $f(x) = \text{sign}(w^Tx + b)$, let g be the smoothed version of f for some σ, let x be any point, and let R be the radius certified around x by Theorem 1 with $\underline{p_A} = p_A$ and $\overline{p_B} = p_B$. Then there always exists a perturbation δ with $\|\delta\|_2 = R$ for which $g(x+\delta) \neq g(x)$.*

Proof. By Proposition 3 it suffices to show that there exists some perturbation δ with $\|\delta\|_2 = R$ for which $f(x+\delta) \neq f(x)$.

By Proposition 4, we know that $R = \frac{|w^Tx+b|}{\|w\|_2}$.

Consider the perturbation $\delta = -\frac{w^Tx+b}{\|w\|_2^2}w$. This perturbation satisfies $\|\delta\|_2 = R$ and

$$\begin{aligned} w^T(x+\delta) + b &= w^Tx + b + w^T\delta \\ &= w^Tx + b - (w^Tx + b) \\ &= 0 \end{aligned}$$

Therefore, $f(x+\delta) = \text{sign}(w^T(x+\delta)+b)$ is undefined. □

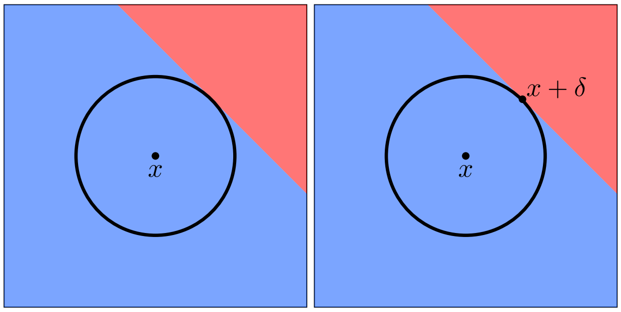


Figure 11. **Left**: Illustration of of Proposition 4. The red/blue half-spaces are the decision regions of both the base classifier f and the smoothed classifier g. (Since the base classifier is binary linear, $g = f$ everywhere.) The black circle is the robustness radius R certified by Theorem 1. **Right**: Illustration of Proposition 5. There exists a perturbation δ with $\|\delta\|_2 = R$ for which $g(x + \delta) \neq g(x)$.

C. Practical algorithms

In this appendix, we elaborate on the prediction and certification algorithms described in Section 3.2. The pseudocode in Section 3.2 makes use of several helper functions:

- SAMPLEUNDERNOISE(f, x, num, σ) works as follows:
  1. Draw num samples of noise, $\varepsilon_1 \dots \varepsilon_{\text{num}} \sim \mathcal{N}(0, \sigma^2 I)$.
  2. Run the noisy images through the base classifier f to obtain the predictions $f(x + \varepsilon_1), \dots, f(x + \varepsilon_{\text{num}})$.
  3. Return the counts for each class, where the count for class c is defined as $\sum_{i=1}^{\text{num}} \mathbf{1}[f(x + \varepsilon_i) = c]$.
- BINOMPVALUE($n_A, n_A + n_B, p$) returns the p-value of the two-sided hypothesis test that $n_A \sim \text{Binomial}(n_A + n_B, p)$. Using `scipy.stats.binom_test`, this can be implemented as: `binom_test(nA, nA + nB, p)`.
- LOWERCONFBOUND(k, n, $1 - \alpha$) returns a one-sided $(1 - \alpha)$ lower confidence interval for the Binomial parameter p given that $k \sim \text{Binomial}(n, p)$. In other words, it returns some number $\underline{p}$ for which $\underline{p} \leq p$ with probability at least $1 - \alpha$ over the sampling of $k \sim \text{Binomial}(n, p)$. Following Lecuyer et al. (2019), we chose to use the Clopper-Pearson confidence interval, which inverts the Binomial CDF (Clopper & Pearson, 1934). Using `statsmodels.stats.proportion.proportion_confint`, this can be implemented as

  ```
  proportion_confint(k, n, alpha=2*alpha, method="beta")[0]
  ```

C.1. Prediction

The randomized algorithm given in pseudocode as PREDICT leverages the hypothesis test given in Hung & Fithian (2019) for identifying the top category of a multinomial distribution. PREDICT has one tunable hyperparameter, α. When α is small, PREDICT abstains frequently but rarely returns the wrong class. When α is large, PREDICT usually makes a prediction, but may often return the wrong class. We now prove that with high probability, PREDICT will either return $g(x)$ or abstain.

Proposition 1 (restated). *With probability at least* $1 - \alpha$ *over the randomness in* PREDICT, PREDICT *will either abstain or return* $g(x)$. *(Equivalently: the probability that* PREDICT *returns a class other than* $g(x)$ *is at most* α.*)*

Proof. For notational convenience, define $p_c = \mathbb{P}(f(x + \varepsilon) = c)$. Let $c_A = \max_c p_c$. Notice that by definition, $g(x) = c_A$.

We can describe the randomized procedure PREDICT as follows:

1. Sample a vector of class counts $\{n_c\}_{c \in \mathcal{Y}}$ from Multinomial$(\{p_c\}_{c \in \mathcal{Y}}, n)$.
2. Let $\hat{c}_A = \arg\max_c n_c$ be the class whose count is largest. Let n_A and n_B be the largest count and the second-largest count, respectively.

3. If the p-value of the two-sided hypothesis test that n_A is drawn from Binom $\left(n_A + n_B, \frac{1}{2}\right)$ is less than α, then return $\hat{c}_A$. Else, abstain.

The quantities c_A and the p_c's are fixed but unknown, while the quantities $\hat{c}_A$, the n_c's, n_A, and n_B are random.

We'd like to prove that the probability that PREDICT returns a class other than c_A is at most α. PREDICT returns a class other than c_A if and only if (1) $\hat{c}_A \neq c_A$ and (2) PREDICT does not abstain.

We have:

$$\begin{aligned}\mathbb{P}(\text{PREDICT returns class } \neq c_A) &= \mathbb{P}(\hat{c}_A \neq c_A, \text{PREDICT does not abstain}) \\ &= \mathbb{P}(\hat{c}_A \neq c_A)\; \mathbb{P}(\text{PREDICT does not abstain} | \hat{c}_A \neq c_A) \\ &\leq \mathbb{P}(\text{PREDICT does not abstain} | \hat{c}_A \neq c_A)\end{aligned}$$

Recall that PREDICT does not abstain if and only if the p-value of the two-sided hypothesis test that n_A is drawn from Binom$(n_A + n_B, \frac{1}{2})$ is less than α. Theorem 1 in Hung & Fithian (2019) proves that the conditional probability that this event occurs given that $\hat{c}_A \neq c_A$ is exactly α. That is,

$$\mathbb{P}(\text{PREDICT does not abstain} | \hat{c}_A \neq c_A) = \alpha$$

Therefore, we have:

$$\mathbb{P}(\text{PREDICT returns class } \neq c_A) \leq \alpha$$

□

C.2. Certification

The certification task is: given some input x and a randomized smoothing classifier described by (f, σ), return both (1) the prediction $g(x)$ and (2) a radius R in which this prediction is certified robust. This task requires identifying the class c_A with maximal weight in $f(x + \varepsilon)$, estimating a lower bound $\underline{p_A}$ on $p_A := \mathbb{P}(f(x + \varepsilon) = c_A)$ and estimating an upper bound $\overline{p_B}$ on $p_B := \max_{c \neq c_A} \mathbb{P}(f(x + \varepsilon) = c)$ (Figure 1).

Suppose for simplicity that we already knew c_A and needed to obtain $\underline{p_A}$. We could collect n samples of $f(x + \varepsilon)$, count how many times $f(x + \varepsilon) = c_A$, and use a Binomial confidence interval to obtain a lower bound on p_A that holds with probability at least $1 - \alpha$ over the n samples.

However, estimating $\underline{p_A}$ and $\overline{p_B}$ while simultaneously identifying the top class c_A is a little bit tricky, statistically speaking. We propose a simple two-step procedure. First, use n_0 samples from $f(x + \varepsilon)$ to take a guess $\hat{c}_A$ at the identity of the top class c_A. In practice we observed that $f(x + \varepsilon)$ tends to put most of its weight on the top class, so n_0 can be set very small. Second, use n samples from $f(x + \varepsilon)$ to obtain some $\underline{p_A}$ and $\overline{p_B}$ for which $\underline{p_A} \leq p_A$ and $\overline{p_B} \geq p_B$ with probability at least $1 - \alpha$. We observed that it is much more typical for the mass of $f(x + \varepsilon)$ not allocated to c_A to be allocated entirely to one runner-up class than to be allocated uniformly over all remaining classes. Therefore, the quantity $1 - \underline{p_A}$ is a reasonably tight upper bound on p_B. Hence, we simply set $\overline{p_B} = 1 - \underline{p_A}$, so our bound becomes

$$\begin{aligned}R &= \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(1 - \underline{p_A})) \\ &= \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) + \Phi^{-1}(\underline{p_A})) \\ &= \sigma \Phi^{-1}(\underline{p_A})\end{aligned}$$

The full procedure is described in pseudocode as CERTIFY. If $\underline{p_A} < \frac{1}{2}$, we abstain from making a certification; this can occur especially if $\hat{c}_A \neq g(x)$, i.e. if we misidentify the top class using the first n_0 samples of $f(x + \varepsilon)$.

Proposition 2 (restated). *With probability at least $1 - \alpha$ over the randomness in* CERTIFY, *if* CERTIFY *returns a class $\hat{c}_A$ and a radius R (i.e. does not abstain), then we have the robustness guarantee*

$$g(x + \delta) = \hat{c}_A \quad \textit{whenever} \quad \|\delta\|_2 < R$$

Proof. From the contract of LOWERCONFBOUND, we know that with probability at least $1 - \alpha$ over the sampling of $\varepsilon_1 \dots \varepsilon_n$, we have $\underline{p_A} \leq \mathbb{P}[f(x + \varepsilon) = \hat{c}_A]$. Notice that CERTIFY returns a class and radius only if $\underline{p_A} > \frac{1}{2}$ (otherwise it abstains). If $\underline{p_A} \leq \mathbb{P}[f(x + \varepsilon) = \hat{c}_A]$ and $\frac{1}{2} < \underline{p_A}$, then we can invoke Theorem 1 with $\overline{p_B} = 1 - \underline{p_A}$ to obtain the desired guarantee. □

D. Estimating the certified test-set accuracy

In this appendix, we show how to convert the "approximate certified test accuracy" considered in the main paper into a lower bound on the true certified test accuracy that holds with high probability over the randomness in CERTIFY.

Consider a classifier g, a test set $S = \{(x_1, c_1) \dots (x_m, c_m)\}$, and a radius r. For each example $i \in [m]$, let z_i indicate whether g's prediction at x_i is both correct and robust at radius r, i.e.

$$z_i = \mathbf{1}[g(x_i + \delta) = c_i \;\; \forall \|\delta\|_2 < r]$$

The certified test set accuracy of g at radius r is defined as $\frac{1}{m}\sum_{i=1}^m z_i$. If g is a randomized smoothing classifier, we cannot compute this quantity exactly, but we can estimate a lower bound that holds with arbitrarily high probability over the randomness in CERTIFY. In particular, suppose that we run CERTIFY with failure rate α on each example x_i in the test set. Let the Bernoulli random variable Y_i denote the event that on example i, CERTIFY returns the correct label $c_A = c_i$ and a certified radius R which is greater than r. Let $Y = \sum_{i=1}^m Y_i$. In the main paper, we referred to Y/m as the "approximate certified accuracy." It is "approximate" because $Y_i = 1$ does not mean that $z_i = 1$. Rather, from Proposition 2, we know the following: if $z_i = 0$, then $\mathbb{P}(Y_i = 1) \leq \alpha$. We now show how to exploit this fact to construct a one-sided confidence interval for the unobserved quantity $\frac{1}{m}\sum_{i=1}^m z_i$ using the observed quantities Y and m.

Theorem 5. *For any $\rho > 0$, with probability at least $1 - \rho$ over the randomness in* CERTIFY*,*

$$\frac{1}{m}\sum_{i=1}^m z_i \geq \frac{1}{1-\alpha}\left(\frac{Y}{m} - \alpha - \sqrt{\frac{2\alpha(1-\alpha)\log(1/\rho)}{m}} - \frac{\log(1/\rho)}{3m}\right) \tag{16}$$

Proof. Let $m_{\text{good}} = \sum_{i=1}^m z_i$ and $m_{\text{bad}} = \sum_{i=1}^m (1 - z_i)$ be the number of test examples on which $z_i = 1$ or $z_i = 0$, respectively. We model $Y_i \sim \text{Bernoulli}(p_i)$, where p_i is in general unknown. Let $Y_{\text{good}} = \sum_{i: z_i = 1} Y_i$ and $Y_{\text{bad}} = \sum_{i: z_i = 0} Y_i$. The quantity of interest, the certified accuracy $\frac{1}{m}\sum_{i=1}^m z_i$, is equal to m_{good}/m. However, we only observe $Y = Y_{\text{good}} + Y_{\text{bad}}$.

Note that if $z_i = 0$, then $p_i \leq \alpha$, so we have $\mathbb{E}[Y_i] = p_i \leq \alpha$ and assuming $\alpha \leq \frac{1}{2}$, we have $\text{Var}[Y_i] = p_i(1 - p_i) \leq \alpha(1 - \alpha)$.

Since Y_{bad} is a sum of m_{bad} independent random variables each bounded between zero and one, with $\mathbb{E}[Y_{\text{bad}}] \leq \alpha m_{\text{bad}}$ and $\text{Var}(Y_{\text{bad}}) \leq m_{\text{bad}}\alpha(1-\alpha)$, Bernstein's inequality (Blanchard, 2007) guarantees that with probability at least $1 - \rho$ over the randomness in CERTIFY,

$$Y_{\text{bad}} \leq \alpha m_{\text{bad}} + \sqrt{2 m_{\text{bad}}\alpha(1-\alpha)\log(1/\rho)} + \frac{\log(1/\rho)}{3}$$

From now on, we manipulate this inequality — remember that it holds with probability at least $1 - \rho$.

Since $Y = Y_{\text{good}} + Y_{\text{bad}}$, may write

$$Y_{\text{good}} \geq Y - \alpha m_{\text{bad}} - \sqrt{2 m_{\text{bad}}\alpha(1-\alpha)\log(1/\rho)} - \frac{\log(1/\rho)}{3}$$

Since $m_{\text{good}} \geq Y_{\text{good}}$, we may write

$$m_{\text{good}} \geq Y - \alpha m_{\text{bad}} - \sqrt{2 m_{\text{bad}}\alpha(1-\alpha)\log(1/\rho)} - \frac{\log(1/\rho)}{3}$$

Since $m_{\text{good}} + m_{\text{bad}} = m$, we may write

$$m_{\text{good}} \geq \frac{1}{1-\alpha}\left(Y - \alpha\, m - \sqrt{2 m_{\text{bad}}\alpha(1-\alpha)\log(1/\rho)} - \frac{\log(1/\rho)}{3}\right)$$

Finally, in order to make this confidence interval depend only on observables, we use $m_{\text{bad}} \leq m$ to write

$$m_{\text{good}} \geq \frac{1}{1-\alpha}\left(Y - \alpha\, m - \sqrt{2 m\alpha(1-\alpha)\log(1/\rho)} - \frac{\log(1/\rho)}{3}\right)$$

Dividing both sides of the inequality by m recovers the theorem statement.

□

E. ImageNet and CIFAR-10 Results

E.1. Certification

Tables 2 and 3 show the approximate certified top-1 test set accuracy of randomized smoothing on ImageNet and CIFAR-10 with various noise levels σ. By "approximate certified accuracy," we mean that we ran CERTIFY on a subsample of the test set, and for each r we report the fraction of examples on which CERTIFY (a) did not abstain, (b) returned the correct class, and (c) returned a radius R greater than r. There is some probability (at most α) that any example's certification is inaccurate. We used $\alpha = 0.001$ and $n = 100000$. On CIFAR-10 our base classifier was a 110-layer residual network and we certified the full test set; on ImageNet our base classifier was a ResNet-50 and we certified a subsample of 500 points. Note that the certified accuracy at $r = 0$ is just the standard accuracy of the smoothed classifier. See Appendix J for more experimental details.

	$r = 0.0$	$r = 0.5$	$r = 1.0$	$r = 1.5$	$r = 2.0$	$r = 2.5$	$r = 3.0$
$\sigma = 0.25$	**0.67**	**0.49**	0.00	0.00	0.00	0.00	0.00
$\sigma = 0.50$	0.57	0.46	**0.37**	**0.29**	0.00	0.00	0.00
$\sigma = 1.00$	0.44	0.38	0.33	0.26	**0.19**	**0.15**	**0.12**

Table 2. Approximate certified test accuracy on ImageNet. Each row is a setting of the hyperparameter σ, each column is an ℓ_2 radius. The entry of the best σ for each radius is bolded. For comparison, random guessing would attain 0.001 accuracy.

	$r = 0.0$	$r = 0.25$	$r = 0.5$	$r = 0.75$	$r = 1.0$	$r = 1.25$	$r = 1.5$
$\sigma = 0.12$	**0.83**	0.60	0.00	0.00	0.00	0.00	0.00
$\sigma = 0.25$	0.77	**0.61**	0.42	0.25	0.00	0.00	0.00
$\sigma = 0.50$	0.66	0.55	**0.43**	**0.32**	0.22	0.14	0.08
$\sigma = 1.00$	0.47	0.41	0.34	0.28	**0.22**	**0.17**	**0.14**

Table 3. Approximate certified test accuracy on CIFAR-10. Each row is a setting of the hyperparameter σ, each column is an ℓ_2 radius. The entry of the best σ for each radius is bolded. For comparison, random guessing would attain 0.1 accuracy.

E.2. Prediction

Table 4 shows the performance of PREDICT as the number of Monte Carlo samples n is varied between 100 and 10,000. Suppose that for some test example (x, c), PREDICT returns the label $\hat{c}_A$. We say that this prediction was *correct* if $\hat{c}_A = c$ and we say that this prediction was *accurate* if $\hat{c}_A = g(x)$. For example, a prediction could be correct but inaccurate if g is wrong at x, yet PREDICT accidentally returns the correct class. Ideally, we'd like PREDICT to be both correct and accurate.

With $n = 100$ Monte Carlo samples and a failure rate of $\alpha = 0.001$, PREDICT is cheap to evaluate (0.15 seconds on our hardware) yet it attains relatively high top-1 accuracy of 65% on the ImageNet test set, and only abstains 12% of the time. When we use $n =$ 10,000 Monte Carlo samples, PREDICT takes longer to evaluate (15 seconds), yet only abstains 4% of the time. Interestingly, we observe from Table 4 that most of the abstentions when $n = 100$ were for examples on which g was wrong, so in practice we would lose little accuracy by taking n to be as small as 100.

N	CORRECT, ACCURATE	CORRECT, INACCURATE	INCORRECT, ACCURATE	INCORRECT, INACCURATE	ABSTAIN
100	0.65	0.00	0.23	0.00	0.12
1000	0.68	0.00	0.28	0.00	0.04
10000	0.69	0.00	0.30	0.00	0.01

Table 4. Performance of PRECICT as n is varied. The dataset was ImageNet and $\sigma = 0.25$, $\alpha = 0.001$. Each column shows the fraction of test examples which ended up in one of five categories; the prediction at x is "correct" if PREDICT returned the true label, while the prediction is "accurate" if PREDICT returned $g(x)$. Computing $g(x)$ exactly is not possible, so in order to determine whether PREDICT was accurate, we took the gold standard to be the top class over $n =$100,000 Monte Carlo samples.

F. Training with Noise

As mentioned in section 3.3, in the experiments for this paper, we followed Lecuyer et al. (2019) and trained the base classifier by minimizing the cross-entropy loss with Gaussian data augmentation. We now provide some justification for this idea.

Let $\{(x_1, c_1), \dots, (x_n, c_n)\}$ be a training dataset. We assume that the base classifier takes the form $f(x) = \arg\max_{c \in \mathcal{Y}} f_c(x)$, where each f_c is the scoring function for class c.

Suppose that our goal is to maximize the sum of of the log-probabilities that f will classify each $x_i + \varepsilon$ as c_i:

$$\sum_{i=1}^{n} \log \mathbb{P}_\varepsilon(f(x_i + \varepsilon) = c_i) = \sum_{i=1}^{n} \log \mathbb{E}_\varepsilon \; \mathbf{1}\left[\arg\max_c f_c(x_i + \varepsilon) = c_i\right] \tag{17}$$

Recall that the softmax function can be interpreted as a continuous, differentiable approximation to $\arg\max$:

$$\mathbf{1}\left[\arg\max_c f_c(x_i + \varepsilon) = c_i\right] \approx \frac{\exp(f_{c_i}(x_i + \varepsilon))}{\sum_{c \in \mathcal{Y}} \exp(f_c(x_i + \varepsilon))}$$

Therefore, our objective is approximately equal to:

$$\sum_{i=1}^{n} \log \mathbb{E}_\varepsilon \left[\frac{\exp(f_{c_i}(x_i + \varepsilon))}{\sum_{c \in \mathcal{Y}} \exp(f_c(x_i + \varepsilon))}\right] \tag{18}$$

By Jensen's inequality and the concavity of $\log$, this quantity is lower-bounded by:

$$\sum_{i=1}^{n} \mathbb{E}_\varepsilon \left[\log \frac{\exp(f_{c_i}(x_i + \varepsilon))}{\sum_{c \in \mathcal{Y}} \exp(f_c(x_i + \varepsilon))}\right]$$

which is the negative of the cross-entropy loss under Gaussian data augmentation.

Therefore, minimizing the cross-entropy loss under Gaussian data augmentation will maximize (18), which will approximately maximize (17).

G. Noise Level can Scale with Input Resolution

Since our robustness guarantee (3) in Theorem 1 does not explicitly depend on the data dimension d, one might worry that randomized smoothing is less effective for images in high resolution — certifying a fixed ℓ_2 radius is "less impressive" for, say, 224×224 image than for a 56×56 image. However, it turns out that in high resolution, images can be corrupted with larger levels of isotropic Gaussian noise while still preserving their content. This fact is made clear by Figure 12, which shows an image at high and low resolution corrupted by Gaussian noise with the same variance. The class ("hummingbird") is easy to discern from the high-resolution noisy image, but not from the low-resolution noisy image. As a consequence, in high resolution one can take σ to be larger while still being able to obtain a base classifier that classifies noisy images accurately. Since our Theorem 1 robustness guarantee scales linearly with σ, this means that in high resolution one can certify larger radii.

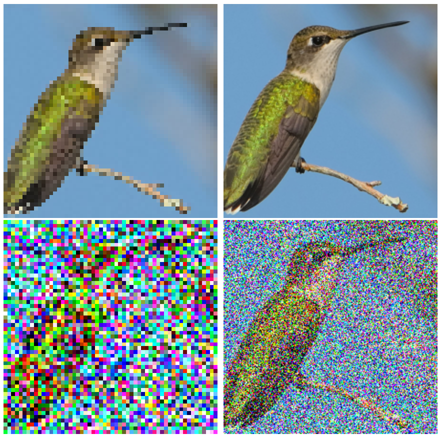


Figure 12. **Top**: An ImageNet image from class "hummingbird" in resolutions 56x56 (left) and 224x224 (right). **Bottom**: the same images corrupted by isotropic Gaussian noise at $\sigma = 0.5$. On noiseless images the class is easy to distinguish no matter the resolution, but on noisy data the class is much easier to distinguish when the resolution is high.

The argument above can be made rigorous, though we first need to decide what it means for two images to be high- and low-resolution versions of each other. Here we present one solution:

Let $\mathcal{X}$ denote the space of "high-resolution" images in dimension $2k \times 2k \times 3$, and let $\mathcal{X}'$ denote the space of "low-resolution" images in dimension $k \times k \times 3$. Let $\textsc{AvgPool} : \mathcal{X} \to \mathcal{X}'$ be the function which takes as input an image x in dimension $2k \times 2k \times 3$, averages together every 2x2 square of pixels, and outputs an image in dimension $k \times k \times 3$.

Equipped with these definitions, we can say that $(x, x') \in \mathcal{X} \times \mathcal{X}'$ are a high/low resolution image pair if $x' = \textsc{AvgPool}(x)$.

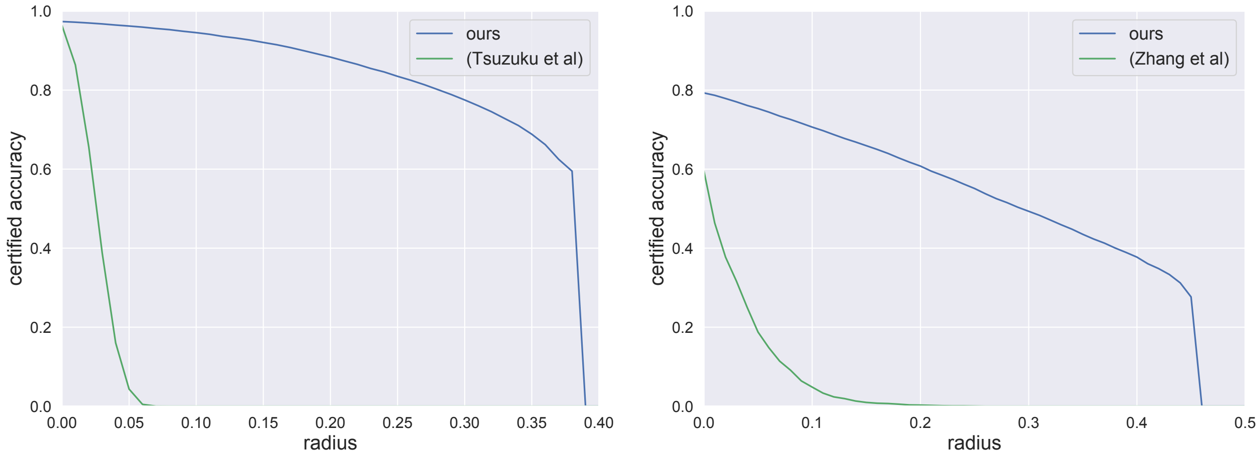
Proposition 6. *Given any smoothing classifier $g' : \mathcal{X}' \to \mathcal{Y}$, one can construct a smoothing classifier $g : \mathcal{X} \to \mathcal{Y}$ with the following property: for any $x \in \mathcal{X}$ and $x' = \textsc{AvgPool}(x)$, g predicts the same class at x that g' predicts at x', but is certifiably robust at twice the radius.*

Proof. Given some smoothing classifier $g' = (f', \sigma')$ from $\mathcal{X}'$ to $\mathcal{Y}$, define g to be the smoothing classifier (f, σ) from $\mathcal{X}$ to $\mathcal{Y}$ with noise level $\sigma = 2\sigma'$ and base classifier $f(x) = f'(\textsc{AvgPool}(x))$. Note that the average of four independent copies of $\mathcal{N}(0, (2\sigma)^2)$ is distributed as $\mathcal{N}(0, \sigma^2)$. Therefore, for any high/low-resolution image pair $x' = \textsc{AvgPool}(x)$, the random variable $\textsc{AvgPool}(x + \varepsilon)$, where $\varepsilon \sim \mathcal{N}(0, (2\sigma)^2 I_{2k \times 2k \times 3})$, is equal in distribution to the random variable $x' + \varepsilon'$, where $\varepsilon' \sim \mathcal{N}(0, \sigma^2 I_{k \times k \times 3})$. Hence, $f(x + \varepsilon) = f'(\textsc{AvgPool}(x + \varepsilon))$ has the same distribution as $f'(x' + \varepsilon')$. By the definition of smoothing, this means that $g(x) = g'(x')$, Additionally, by Theorem 1, since $\sigma = 2\sigma'$, this means that g's prediction at x is certifiably robust at twice the radius as g''s prediction at x'. □

H. Additional Experiments

H.1. Comparisons to baselines

Figure 13 compares the certified accuracy of a smoothed 20-layer resnet to that of the released models from two recent works on certified ℓ_2 robustness: the Lipschitz approach from Tsuzuku et al. (2018) and the approach from Zhang et al. (2018). Note that in these experiments, the base classifier for smoothing was larger than the networks of competing approaches. The comparison to Zhang et al. (2018) is on CIFAR-10, while the comparison to Tsuzuku et al. (2018) is on SVHN. Note that for each comparison, we preprocessed the dataset to follow the preprocessing used when the baseline was trained; therefore, the radii reported for CIFAR-10 here are not comparable to the radii reported elsewhere in this paper. Full experimental details are in Appendix J.



(a) Tsuzuku et al. (2018)

(b) Zhang et al. (2018)

Figure 13. Randomized smoothing with a 20-layer resnet base classifier attains higher certified accuracy than the released models from two recent works on certified ℓ_2 robustness.

H.2. High-probability guarantees

Appendix D details how to use CERTIFY to obtain a lower bound on the certified test accuracy at radius r of a randomized smoothing classifier that holds with high probability over the randomness in CERTIFY. In the main paper, we declined to do this and simply reported the approximate certified test accuracy, defined as the fraction of test examples for which CERTIFY gives the correct prediction and certifies it at radius r. Of course, with some probability (guaranteed to be less than α), each of these certifications is wrong.

However, we now demonstrate empirically that there is a negligible difference between a proper high-probability lower bound on the certified accuracy and the approximate version that we reported in the paper. We created a randomized smoothing classifier g on ImageNet with a ResNet-50 base classifier and noise level $\sigma = 0.25$. We used CERTIFY with $\alpha = 0.001$ to certify a subsample of 500 examples from the ImageNet test set. From this we computed the approximate certified test accuracy at each radius r. Then we used the correction from Appendix D with $\rho = 0.001$ to obtain a lower bound on the certified test accuracy at r that holds pointwise with probability at least $1 - \rho$ over the randomness in CERTIFY. Figure 14 plots both quantities as a function of r. Observe that the difference is so negligible that the lines almost overlap.

H.3. How much noise to use when training the base classifier?

In the main paper, whenever we created a randomized smoothing classifier g at noise level σ, we always trained the corresponding base classifier f with Gaussian data augmentation at noise level σ. In Figure 15, we show the effects of training the base classifier with a different level of Gaussian noise. Observe that g has a lower certified accuracy if f was trained using a different noise level. It seems to be worse to train with noise $< \sigma$ than to train with noise $> \sigma$.

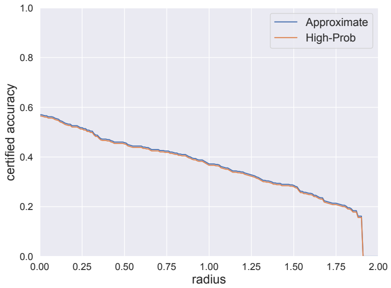
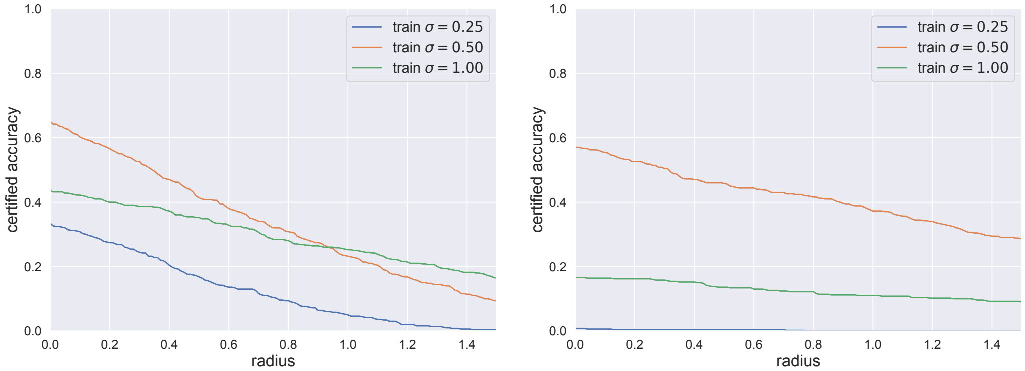


Figure 14. The difference between the approximate certified accuracy, and a high-probability lower bound on the certified accuracy, is negligible.



(a) CIFAR-10

(b) ImageNet

Figure 15. Vary training noise while holding prediction noise fixed at $\sigma = 0.50$.

I. Derivation of Prior Randomized Smoothing Guarantees

In this appendix, we derive the randomized smoothing guarantees of Lecuyer et al. (2019) and Li et al. (2018) using the notation of our paper. Both guarantees take same general form as ours, except with a different expression for R:

Theorem (generic guarantee): *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be any deterministic or random function, and let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. Let g be defined as in (1). Suppose $c_A \in \mathcal{Y}$ and $\underline{p_A}, \overline{p_B} \in [0, 1]$ satisfy:*

$$\mathbb{P}(f(x+\varepsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(f(x+\varepsilon) = c) \tag{19}$$

Then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R$.

For convenience, define the notation $X \sim \mathcal{N}(x, \sigma^2 I)$ and $Y \sim \mathcal{N}(x+\delta, \sigma^2 I)$.

I.1. Lecuyer et al. (2019)

Lecuyer et al. (2019) proved a version of the generic robustness guarantee in which

$$R = \sup_{0 < \beta \leq \min\left(1, \frac{1}{2} \log \frac{\underline{p_A}}{\overline{p_B}}\right)} \frac{\sigma \beta}{\sqrt{2 \log \left(\frac{1.25(1+\exp(\beta))}{\underline{p_A} - \exp(2\beta)\overline{p_B}}\right)}}$$

Proof. In order to avoid notation that conflicts with the rest of this paper, we use β and γ where Lecuyer et al. (2019) used ϵ and δ.

Suppose that we have some $0 < \beta \leq 1$ and $\gamma > 0$ such that

$$\sigma^2 = \frac{\|\delta\|^2}{\beta^2} 2 \log \frac{1.25}{\gamma} \tag{20}$$

The "Gaussian mechanism" from differential privacy guarantees that:

$$\mathbb{P}(f(X) = c_A) \leq \exp(\beta)\mathbb{P}(f(Y) = c_A) + \gamma \tag{21}$$

and, symmetrically,

$$\mathbb{P}(f(Y) = c_B) \leq \exp(\beta)\mathbb{P}(f(X) = c_B) + \gamma \tag{22}$$

See Lecuyer et al. (2019), Lemma 2 for how to obtain this form from the standard form of the (β, γ) DP definition.

Fix a perturbation δ. To guarantee that $g(x + \delta) = c_A$, we need to show that $\mathbb{P}(f(Y) = c_A) > \mathbb{P}(f(Y) = c_B)$ for each $c_B \neq c_A$.

Together, (21) and (22) imply that to guarantee $\mathbb{P}(f(Y) = c_A) > \mathbb{P}(f(Y) = c_B)$, it suffices to show that:

$$\mathbb{P}(f(X) = c_A) > \exp(2\beta)\mathbb{P}(f(X) = c_B) + \gamma(1 + \exp(\beta)) \tag{23}$$

Therefore, by (19), in order to guarantee that $g(x + \delta) = c_A$ it suffices to show:

$$\underline{p_A} > \exp(2\beta)\overline{p_B} + \gamma(1 + \exp(\beta)) \tag{24}$$

Now, inverting (20), we obtain:

$$\gamma = 1.25 \exp\left(-\frac{\sigma^2\beta^2}{2\|\delta\|^2}\right) \tag{25}$$

Plugging (25) into (24), we see that to guarantee $g(x + \delta) = c_A$ it suffices to show that:

$$\underline{p_A} > \exp(2\beta)\overline{p_B} + 1.25 \exp\left(-\frac{\sigma^2\beta^2}{2\|\delta\|^2}\right)(1 + \exp(\beta)) \tag{26}$$

which rearranges to:

$$\frac{\underline{p_A} - \exp(2\beta)\overline{p_B}}{1.25(1 + \exp(\beta))} > \exp\left(-\frac{\sigma^2\beta^2}{2\|\delta\|^2}\right) \tag{27}$$

Since the RHS is always positive, and the denominator on the LHS is always positive, this condition can only possibly hold if the numerator on the LHS is positive. Therefore, we need to restrict β to

$$0 < \beta \leq \min\left(1, \frac{1}{2} \log \frac{\underline{p_A}}{\overline{p_B}}\right) \tag{28}$$

The condition (27) is equivalent to:

$$\|\delta\|^2 \log \frac{1.25(1 + \exp(\beta))}{\underline{p_A} - \exp(2\beta)\overline{p_B}} < \frac{\sigma^2\beta^2}{2} \tag{29}$$

Since $\underline{p_A} \leq 1$ and $\overline{p_B} \geq 0$, the denominator in the LHS is ≤ 1 which is in turn $\leq$ the numerator on the LHS. Therefore, the term inside the log in the LHS is greater than 1, so the log term on the LHS is greater than zero. Therefore, we may divide both sides of the inequality by the log term on the LHS to obtain:

$$\|\delta\|^2 < \frac{\sigma^2\beta^2}{2 \log\left(\frac{1.25(1+\exp(\beta))}{\underline{p_A} - \exp(2\beta)\overline{p_B}}\right)} \tag{30}$$

Finally, we take the square root and maximize the bound over all valid β (28) to yield:

$$\|\delta\| < \sup_{0 < \beta \leq \min\left(1, \frac{1}{2}\log\frac{\underline{p_A}}{\overline{p_B}}\right)} \frac{\sigma\beta}{\sqrt{2 \log\left(\frac{1.25(1+\exp(\beta))}{\underline{p_A} - \exp(2\beta)\overline{p_B}}\right)}} \tag{31}$$

□

Figure 16a plots this bound at varying settings of the tuning parameter β, while Figure 16c plots how the bound varies with β for a fixed $\underline{p_A}$ and $\overline{p_B}$.

I.2. Li et al. (2018)

Li et al. (2018) proved a version of the generic robustness guarantee in which

$$R = \sup_{\alpha>0} \sigma \sqrt{-\frac{2}{\alpha} \log\left(1 - \underline{p_A} - \overline{p_B} + 2\left(\frac{1}{2}(\underline{p_A}^{1-\alpha} + \overline{p_B}^{1-\alpha})^{1-\alpha}\right)\right)}$$

Proof. A generalization of KL divergence, the α-Renyi divergence is an information theoretic measure of distance between two distributions. It is parameterized by some $\alpha > 0$. The α-Renyi divergence between two discrete distributions P and Q is defined as:

$$D_\alpha(P||Q) := \frac{1}{\alpha - 1} \log\left(\sum_{i=1}^{k} \frac{p_i^\alpha}{q_i^{\alpha-1}}\right) \tag{32}$$

In the continuous case, this sum is replaced with an integral. The divergence is undefined when $\alpha = 1$ since a division by zero occurs, but the limit of $D_\alpha(P||Q)$ as $\alpha \to 1$ is the KL divergence between P and Q.

Li et al. (2018) prove that if P is a discrete distribution for which the highest probability class has probability $\geq \underline{p_A}$ and all other classes have probability $\leq \overline{p_B}$, then for any other discrete distribution Q for which

$$D_\alpha(P||Q) < -\log\left(1 - \underline{p_A} - \overline{p_B} + 2\left(\frac{1}{2}(\underline{p_A}^{1-\alpha} + \overline{p_B}^{1-\alpha})^{1-\alpha}\right)\right) \tag{33}$$

the highest-probability class in Q is guaranteed to be the same as the highest-probability class in P.

We now apply this result to the discrete distributions $P = f(X)$ and $Q = f(Y)$. If $D_\alpha(f(X)||f(Y))$ satisfies (33), then it is guaranteed that $g(x) = g(x + \delta)$.

The data processing inequality states that applying a function to two random variables can only decrease the α-Renyi divergence between them. In particular,

$$D_\alpha(f(X)||f(Y)) \leq D_\alpha(X||Y) \tag{34}$$

There is a closed-form expression for the α-Renyi divergence between two Gaussians:

$$D_\alpha(X||Y) = \frac{\alpha\|\delta\|^2}{2\sigma^2} \tag{35}$$

Therefore, we can guarantee that $g(x + \delta) = c_A$ so long as

$$\frac{\alpha\|\delta\|^2}{2\sigma^2} < -\log\left(1 - \underline{p_A} - \overline{p_B} + 2\left(\frac{1}{2}(\underline{p_A}^{1-\alpha} + \overline{p_B}^{1-\alpha})^{1-\alpha}\right)\right) \tag{36}$$

which simplifies to

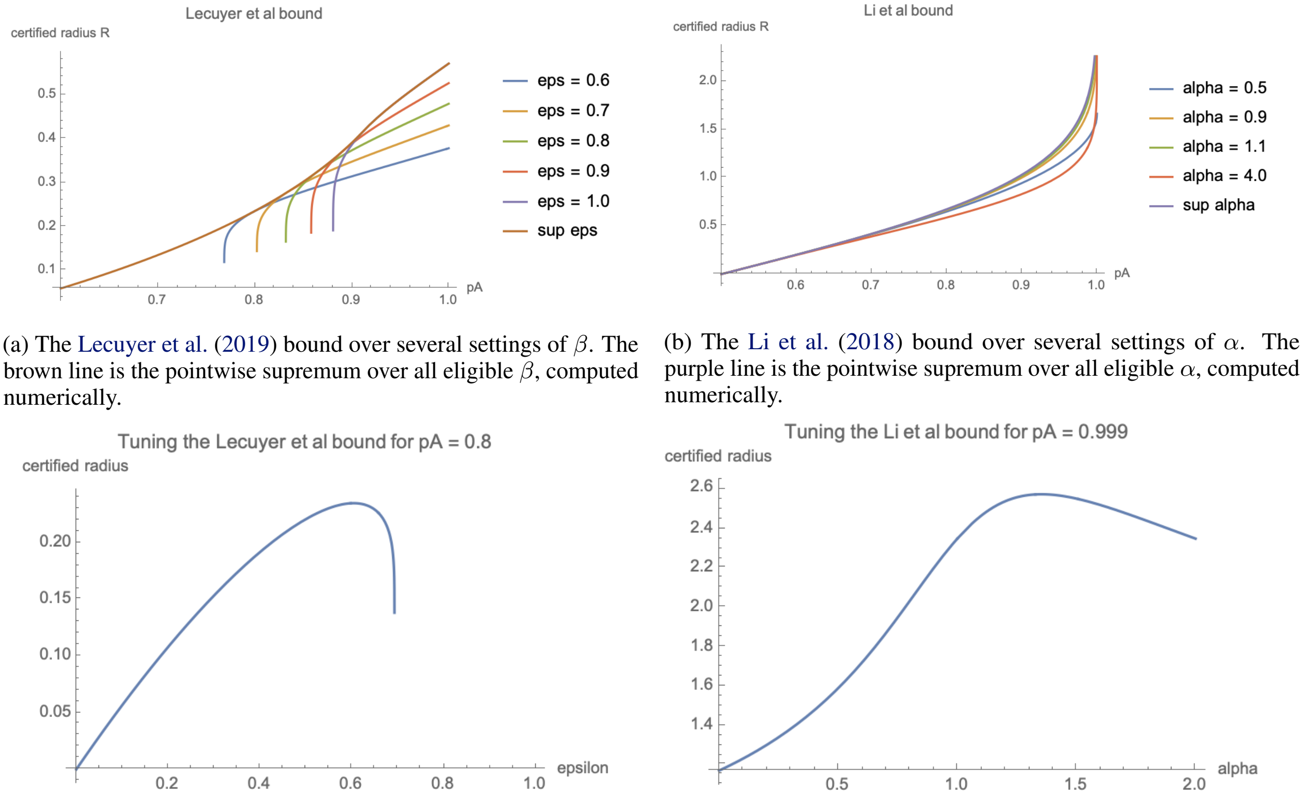
$$\|\delta\| < \sigma \sqrt{-\frac{2}{\alpha} \log\left(1 - \underline{p_A} - \overline{p_B} + 2\left(\frac{1}{2}(\underline{p_A}^{1-\alpha} + \overline{p_B}^{1-\alpha})^{1-\alpha}\right)\right)} \tag{37}$$

Finally, since this result holds for any $\alpha > 0$, we may maximize over α to obtain the largest possible certified radius:

$$\|\delta\| < \sup_{\alpha>0} \sigma \sqrt{-\frac{2}{\alpha} \log\left(1 - \underline{p_A} - \overline{p_B} + 2\left(\frac{1}{2}(\underline{p_A}^{1-\alpha} + \overline{p_B}^{1-\alpha})^{1-\alpha}\right)\right)} \tag{38}$$

□

Figure 16b plots this bound at varying settings of the tuning parameter α, while figure 16d plots how the bound varies with α for a fixed $\underline{p_A}$ and $\overline{p_B}$.



(a) The Lecuyer et al. (2019) bound over several settings of β. The brown line is the pointwise supremum over all eligible β, computed numerically.

(b) The Li et al. (2018) bound over several settings of α. The purple line is the pointwise supremum over all eligible α, computed numerically.

(c) Tuning the Lecuyer et al. (2019) bound wrt β when $\underline{p_A} = 0.8, \overline{p_B} = 0.2$

(d) Tuning the Li et al. (2018) bound wrt α when $\underline{p_A} = 0.999, \overline{p_B} = 0.0001$

J. Experiment Details

J.1. Comparison to baselines

We compared randomized smoothing against three recent approaches for ℓ_2-robust classification (Tsuzuku et al., 2018; Wong et al., 2018; Zhang et al., 2018). Tsuzuku et al. (2018) and Wong et al. (2018) propose both a robust training method and a complementary certification mechanism, while Zhang et al. (2018) propose a method to certify generically trained networks. In all cases we compared against networks provided by the authors. We compared against Wong et al. (2018) and Zhang et al. (2018) on CIFAR-10, and we compared against Tsuzuku et al. (2018) on SVHN.

In image classification it is common practice to preprocess a dataset by subtracting from each channel the mean over the dataset, and dividing each channel by the standard deviation over the dataset. However, we wanted to report certified radii in the original image coordinates rather than in the standardized coordinates. Therefore, throughout most of this work we *first* added the Gaussian noise, and *then* standardized the channels, before feeding the image to the base classifier. (In the practical PyTorch implementation, the first layer of the base classifier was a layer that standardized the input.) However, all of the baselines we compared against provided pre-trained networks which assumed that the dataset was first preprocessed in a specific way. Therefore, when comparing against the baselines we also preprocessed the datasets first, so that we could report certified radii that were directly comparable to the radii reported by the baseline methods.

Comparison to Wong et al. (2018) Following Wong et al. (2018), the CIFAR-10 dataset was preprocessed by subtracting $(0.485, 0.456, 0.406)$ and dividing by $(0.225, 0.225, 0.225)$.

While the body of the Wong et al. (2018) paper focuses on ℓ_∞ certified robustness, their algorithm naturally extends to ℓ_2 certified robustness, as developed in the appendix of the paper. We used three ℓ_2-trained residual networks publicly released by the authors, each trained with a different setting of their hyperparameter $\epsilon \in \{0.157, 0.628, 2.51\}$. We used code publicly released by the authors at `https://github.com/locuslab/convex_adversarial/blob/master/`

examples/cifar_evaluate.py to compute the robustness radius of test images. The code accepts a radius and returns TRUE (robust) or FALSE (not robust); we incorporated this subroutine into a binary search procedure to find the largest radius for which the code returned TRUE.

For randomized smoothing we used $\sigma = 0.6$ and a 20-layer residual network base classifier. We ran CERTIFY with $n_0 = 100$, $n =$ 100,000 and $\alpha = 0.001$.

For both methods, we certified the full CIFAR-10 test set.

Comparison to Tsuzuku et al. (2018) Following Tsuzuku et al. (2018), the SVHN dataset was not preprocessed except that pixels were divided by 255 so as to lie within [0, 1].

We compared against a pretrained network provided to us by the authors in which the hyperparameter of their method was set to $c = 0.1$. The network was a wide residual network with 16 layers and a width factor of 4. We used the authors' code at https://github.com/ytsmiling/lmt to compute the robustness radius of test images.

For randomized smoothing we used $\sigma = 0.1$ and a 20-layer residual network base classifier. We ran CERTIFY with $n_0 = 100$, $n =$ 100,000 and $\alpha = 0.001$.

For both methods, we certified the whole SVHN test set.

Comparison to Zhang et al. (2018) Following Zhang et al. (2018), the CIFAR-10 dataset was preprocessed by subtracting 0.5 from each pixel.

We compared against the cifar_7_1024_vanilla network released by the authors, which is a 7-layer MLP. We used the authors' code at https://github.com/IBM/CROWN-Robustness-Certification to compute the robustness radius of test images.

For randomized smoothing we used $\sigma = 1.2$ and a 20-layer residual network base classifier. We ran CERTIFY with $n_0 = 100$, $n =$ 100,000 and $\alpha = 0.001$.

For randomized smoothing, we certified the whole CIFAR-10 test set. For Zhang et al. (2018), we certified every fourth image in the CIFAR-10 test set.

J.2. ImageNet and CIFAR-10 Experiments

Our code is available at http://github.com/locuslab/smoothing.

In order to report certified radii in the original coordinates, we *first* added Gaussian noise, and *then* standardized the data. Specifically, in our PyTorch implementation, the first layer of the base classifier was a normalization layer that performed a channel-wise standardization of its input. For CIFAR-10 we subtracted the dataset mean $(0.4914, 0.4822, 0.4465)$ and divided by the dataset standard deviation $(0.2023, 0.1994, 0.2010)$. For ImageNet we subtracted the dataset mean $(0.485, 0.456, 0.406)$ and divided by the standard deviation $(0.229, 0.224, 0.225)$.

For both ImageNet and CIFAR-10, we trained the base classifier with random horizontal flips and random crops (in addition to the Gaussian data augmentation discussed explicitly in the paper). On ImageNet we trained with synchronous SGD on four NVIDIA RTX 2080 Ti GPUs; training took approximately three days.

On ImageNet our base classifier used the ResNet-50 architecture provided in torchvision. On CIFAR-10 we used a 110-layer residual network from https://github.com/bearpaw/pytorch-classification.

On ImageNet we certified every 100-th image in the validation set, for 500 images total. On CIFAR-10 we certified the whole test set.

In Figure 8 (**middle**) we fixed $\sigma = 0.25$ and $\alpha = 0.001$ while varying the number of samples n. We did not actually vary the number of samples n that we simulated: we kept this number fixed at 100,000 but varied the number that we fed the Clopper-Pearson confidence interval.

In Figure 8 (**right**), we fixed $\sigma = 0.25$ and $n =$100,000 while varying α.

J.3. Adversarial Attacks

As discussed in Section 4, we subjected smoothed classifiers to a projected gradient descent-style adversarial attack. We now describe the details of this attack.

Let f be the base classifier and let σ be the noise level. Following Li et al. (2018), given an example $(x, c) \in \mathbb{R}^d \times \mathcal{Y}$ and a radius r, we used a projected gradient descent style adversarial attack to optimize the objective:

$$\underset{\delta : \|\delta\|_2 < r}{\arg\max} \; \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I)} \left[\ell(f(x + \delta + \varepsilon), c) \right] \tag{39}$$

where ℓ is the softmax loss function. (Breaking notation with the rest of the paper in which f returns a class, the function f here refers to the function that maps an image in $\mathbb{R}^d$ to a vector of classwise scores.)

At each iteration of the attack, we drew k samples of noise, $\varepsilon_1 \dots \varepsilon_k \sim \mathcal{N}(0, \sigma^2 I)$, and followed the stochastic gradient $g_t = \sum_{i=1}^k \nabla_{\delta_t} \ell(f(x + \delta_t + \varepsilon_k), c)$.

As is typical (Kolter & Madry, 2018), we used a "steepest ascent" update rule, which, for the ℓ_2 norm, means that we normalized the gradient before applying the update. The overall PGD update is: $\delta_{t+1} = \text{proj}_r \left(\delta_t + \eta \frac{g_t}{\|g_t\|} \right)$ where the function proj_r that projects its input onto the ball $\{z : \|z\|_2 \leq r\}$ is given by $\text{proj}_r(z) = \frac{rz}{\max(r, \|z\|_2)}$. We used a constant step size η and a fixed number T of PGD iterations.

In practice, our step size was $\eta = 0.1$, we used $T = 20$ steps of PGD, and we computed the stochastic gradient using $k = 1000$ Monte Carlo samples.

Unfortunately, the objective we optimize (39) is not actually the objective of interest. The real goal of an attacker is to find some perturbation δ with $\|\delta\|_2 < r$ and some class c_B for which

$$\mathbb{P}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I)}(f(x + \delta + \varepsilon) = c_B) \geq \mathbb{P}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I)}(f(x + \delta + \varepsilon) = c)$$

Effective adversarial attacks against randomized smoothing are outside the scope of this paper.

K. Examples of Noisy Images

We now show examples of CIFAR-10 and ImageNet images corrupted with varying levels of noise.

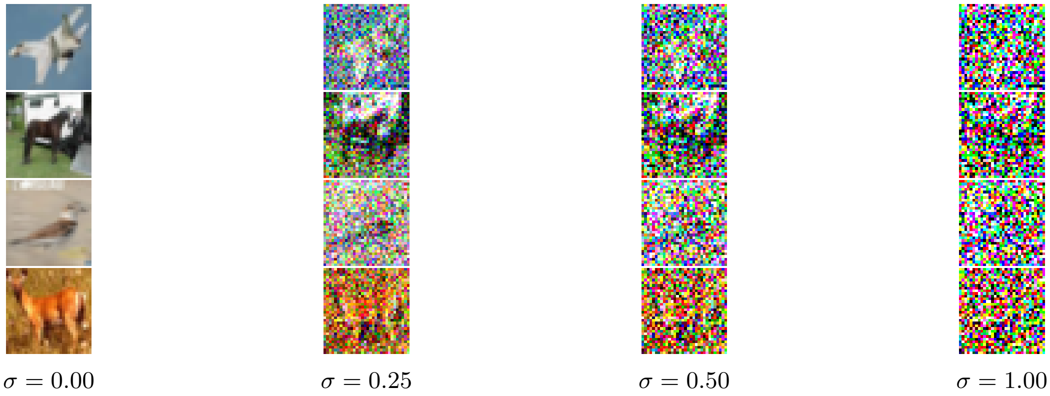


Figure 17. CIFAR-10 images additively corrupted by varying levels of Gaussian noise $\mathcal{N}(0, \sigma^2 I)$. Pixel values greater than 1.0 (=255) or less than 0.0 (=0) were clipped to 1.0 or 0.0.

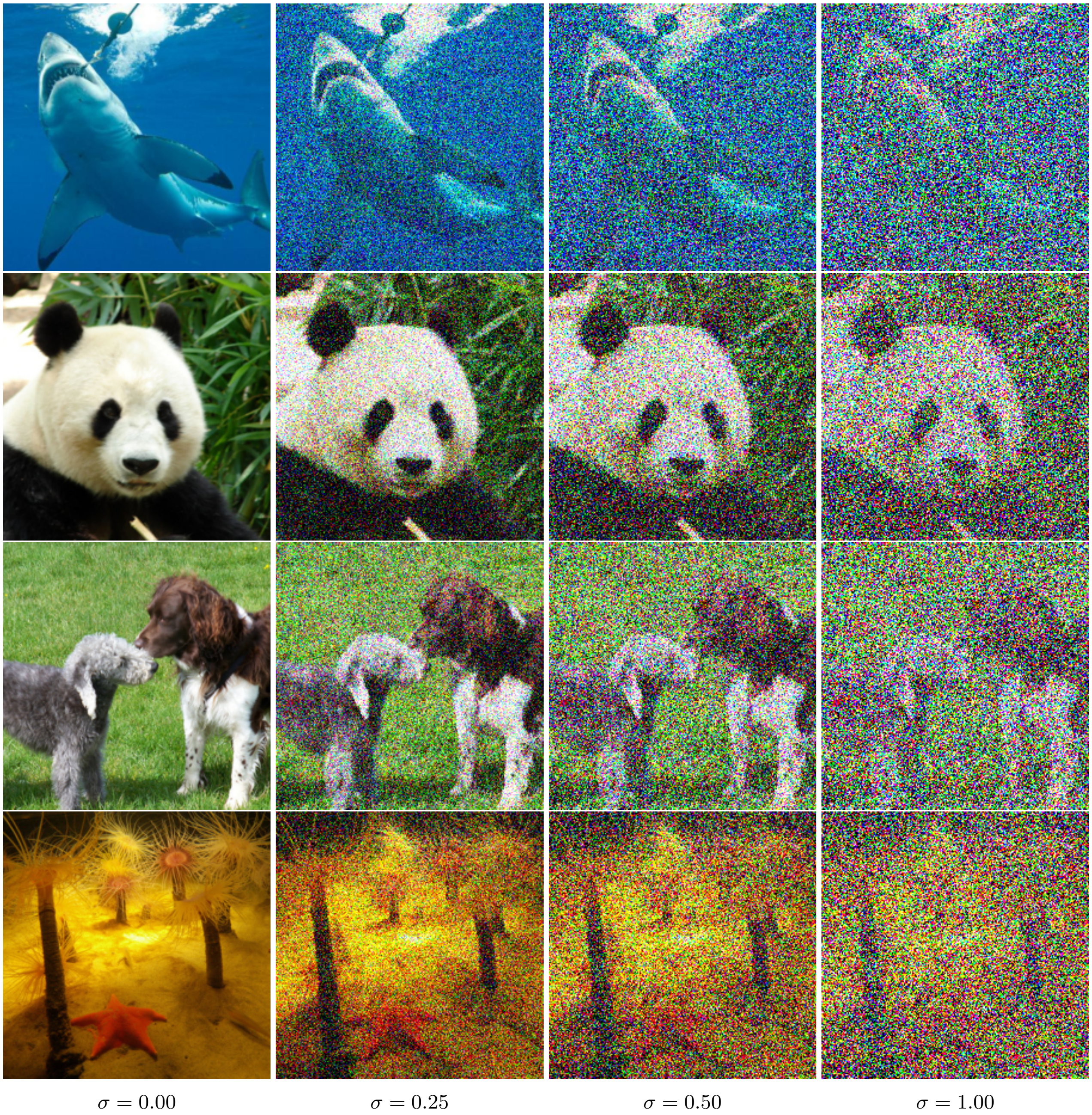


Figure 18. ImageNet images additively corrupted by varying levels of Gaussian noise $\mathcal{N}(0, \sigma^2 I)$. Pixel values greater than 1.0 (=255) or less than 0.0 (=0) were clipped to 1.0 or 0.0.