Roadmap

1. Introduction
2. Construction of adversarial examples
3. Improving robustness
4. Certifiable robustness
   – Exact certification
   – Convex relaxations
   – Lipschitz-continuity
   – **Randomized smoothing**

Introduction

- The robustness certificates studied so far have in common that they try to prove that the predicted class for a given input does not change for some type of perturbations.

- The idea behind randomized smoothing is simple: we transform any given base classifier into a smoothed classifier by **randomly** adding noise (e.g. Gaussian) to the input and predicting the majority class given many samples.

- We certify that the predictions of the resulting **smoothed** classifier do not change when the input is (adversarially) perturbed.

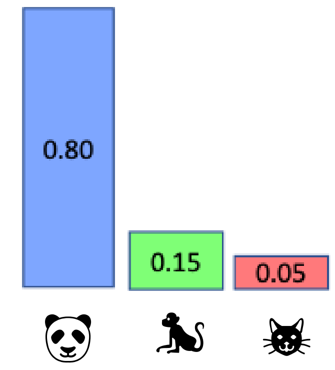
Graphical Overview

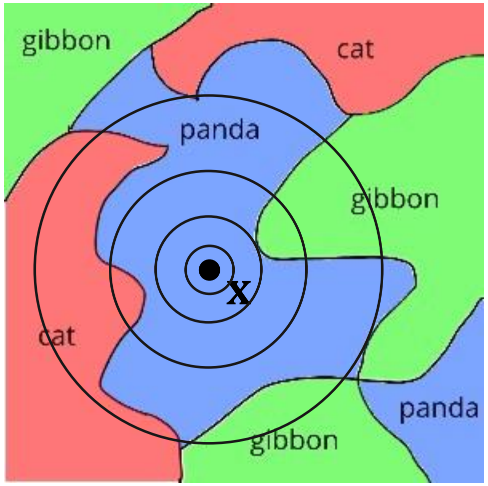
- The image shows the predictions of some base classifier f (e.g. a Neural Network) in different colors for any given input.

- Given a test input $\mathbf{x}$ we sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$ and observe the output of the base classifier $f(\mathbf{x} + \boldsymbol{\epsilon})$.

- We notice that the **majority** of instances are still classified correctly, i.e. $f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐼 even though sometimes $f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐒 or $f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐱.

- By slightly moving the center point $\mathbf{x}$ (i.e. perturbing the original image), the probabilities of observing the different classes will only **change slowly**.





Example and figures from: [Cohen+ 2019b]

General Idea

- We smooth any classifier $f(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} F(\mathbf{x})_c$ into a smooth classifier g
  - e.g. $F(\mathbf{x})$ is the vector of logits returned by a Neural Network, $f(\mathbf{x})$ returns the class.

- $g(\mathbf{x})$ = the **most probable** prediction of f under random Gaussian noise.

- Example: consider input $\mathbf{x} =$  and noisy input $\mathbf{x} + \boldsymbol{\epsilon}$ = 

- Suppose when we sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$ and evaluate $f\ (\mathbf{x} + \boldsymbol{\epsilon})$ it holds:
  - $\mathbb{P}_\epsilon(f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐼$) = 0.80$
  - $\mathbb{P}_\epsilon(f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐒$) = 0.15$
  - $\mathbb{P}_\epsilon(f(\mathbf{x} + \boldsymbol{\epsilon}) =$ 🐱$) = 0.05$

- Then $g(\mathbf{x}) =$ 🐼 i.e. g predicts the majority class when randomly sampling.

Example and figures from: [Cohen+ 2019b]

Formal Definition

- Denote with $g(\mathbf{x})_c$ the probability that f classifies a sample from $\mathcal{N}(\mathbf{x}, \sigma^2 \boldsymbol{I})$ (or equivalently $\mathbf{x} + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right)$) as class c:

$$\mathbb{P}_{\boldsymbol{\epsilon}}(\mathrm{f}(\mathbf{x} + \boldsymbol{\epsilon}) = c) = \mathbb{E}_{\boldsymbol{\epsilon}}(\mathbb{I}[\mathrm{f}(\mathbf{x} + \boldsymbol{\epsilon}) = c]) = \int \mathbb{P}(\mathbf{x} = \boldsymbol{z})\mathbb{I}[f(\boldsymbol{z}) = c]d\boldsymbol{z} = g(\mathbf{x})_c$$

- In other words, the output of the smooth classifier $g(\mathbf{x})$ is a vector with entries $g(\mathbf{x})_c = \mathbb{P}_{\boldsymbol{\epsilon}}(\mathrm{f}(\mathbf{x} + \boldsymbol{\epsilon}) = c)$

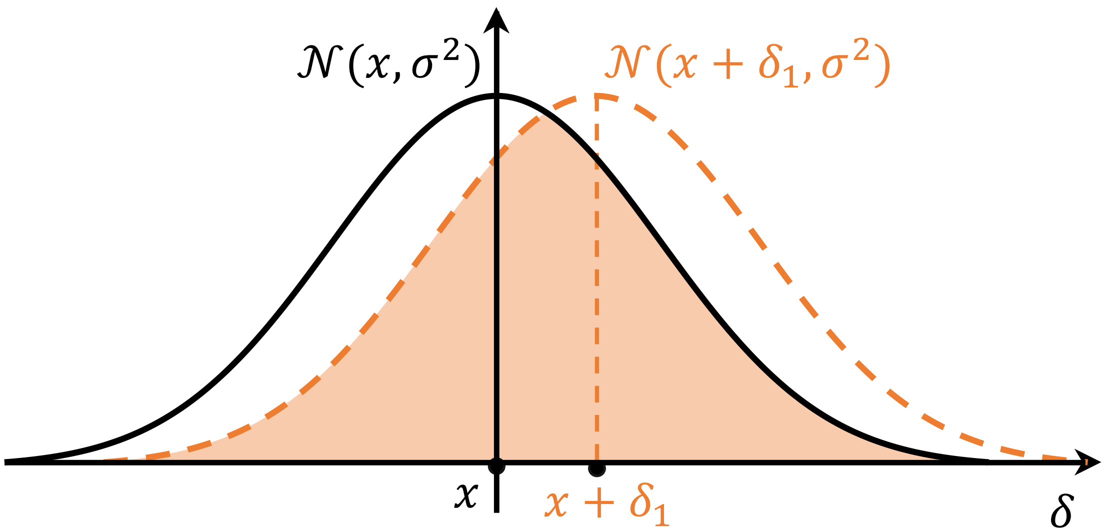
- Now denote with $c^* = \arg\max_c g(\mathbf{x})_c$ the most likely class and denote with $p_{\mathbf{x}}^* = g(\mathbf{x})_{c^*}$ the probability of observing c^*.

- Goal: We want to certify that for any admissible perturbation $\boldsymbol{\delta}$ it holds

$$\arg\max_c g(\mathbf{x} + \boldsymbol{\delta})_c = c^* \quad \text{for all } ||\boldsymbol{\delta}||_2 \leq r$$

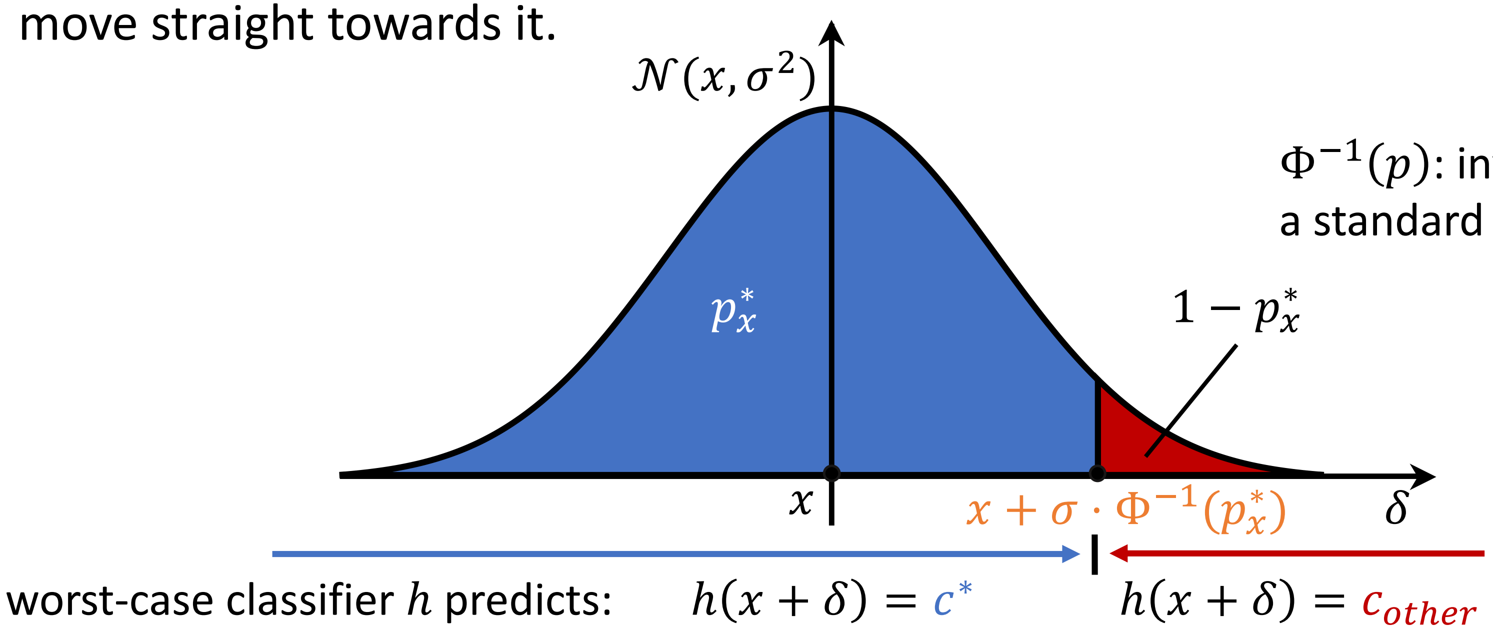
Perturbation of the Input Sample – 1D Case

- Without loss of generality we consider a **1-dimensional** example.
- When we **slightly perturb** x the samples from $\mathcal{N}(x, \sigma^2)$ and $\mathcal{N}((x+\delta), \sigma^2)$ have a large overlap ➔ $g(x+\delta)$ and $g(x)$ produce similar output.
- How large can we make δ (how much can we perturb x) and still **guarantee** that $\arg\max g(x+\delta) = \arg\max g(x)$?



Worst-Case View on Randomized Smoothing

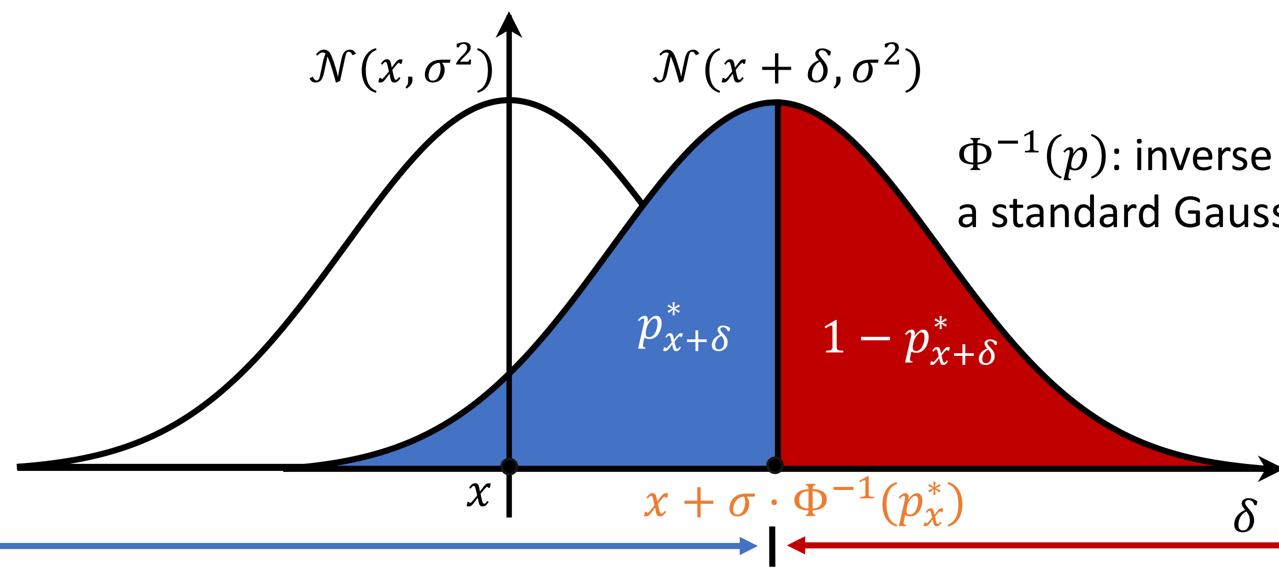
- Suppose that with probability p_x^* the classifier predicts class c^* for a sample from $\mathcal{N}(x, \sigma^2)$.
- In the **worst** case, all the samples from c^* are **concentrated on one side**, and all samples from $c_{other} \neq c^*$ are **concentrated on the other side**.
- In the worst case, the perturbation is orthogonal to this boundary, i.e. we move straight towards it.



$\Phi^{-1}(p)$: inverse CDF of a standard Gaussian

Largest Certifiable Radius

- If $\|\delta\|_2 = \sigma \cdot \Phi^{-1}(p_x^*)$, then (in the worst case), $p_{x+\delta}^* = 1 - p_{x+\delta}^* = 0.5$.
- Thus, for any perturbation $\|\delta\|_2 < \sigma \cdot \Phi^{-1}(p_x^*)$, the smoothed classifier will not change its prediction, i.e. $\arg\max g(x+\delta) = \arg\max g(x)$.
- Larger variance σ^2 could lead to larger radii $\|\delta\|_2$, however it could also lead to reduced p_x^* by introducing too much noise.



$\Phi^{-1}(p)$: inverse CDF of a standard Gaussian

worst-case classifier h predicts: $h(x+\delta) = c^*$ $h(x+\delta) = c_{other}$

The Higher Dimensional Case

- See annotation in class.

How to determine $p^*_{\mathbf{x}}$?

- In the previous section we have assumed we know the true $p^*_{\mathbf{x}}$, i.e. the proportion of the samples classified as c^* under the Gaussian noise.
- However, for neural networks we can, in general, not exactly compute these class probabilities.
- However, we can simply perform a Monte Carlo estimation: sample a large number of samples from the Gaussian to estimate $p^*_{\mathbf{x}}$.
- As the number of samples goes to infinity, we are guaranteed to converge to the true proportion $p^*_{\mathbf{x}}$.

How to determine $p^*_{\mathbf{x}}$?

- We need to be very certain not to overestimate $p^*_{\mathbf{x}}$, since this would lead to invalid certificates!
- Let $\mathrm{A} = \mathbb{I}[f(\mathbf{x} + \boldsymbol{\epsilon}) = c^*] \in \{0, 1\}$ for $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ denote the random variable corresponding to the event of observing the class c^*.
- A is a Bernoulli random variable with probability of observing 1 equal to $p^*_{\mathbf{x}}$.
  - Think of a (biased) coin flip with probability $p^*_{\mathbf{x}}$.

- We can compute a one-sided $(1 - \alpha)$ lower confidence interval for $p^*_{\mathbf{x}}$ given the outcome for many samples (where α is small, e.g. 5%)
- If we sample n times and $\mathrm{A} = 1$ in m/n then $P(A = m) = Binomial(n, p^*_x)$
- We can get a **lower bound** $\underline{p^*_{\mathbf{x}}}$ from the confidence interval for the Binomial.

Practical Considerations

- For certification, we have to determine
  - The most likely class c^*
  - Lower bound for $\underline{p_{\mathbf{x}}^*}$

- We need to take care from a statistical point of view.
- Using a small set of samples we first take a guess at c^*
- Then, using a large set of samples we compute $\underline{p_{\mathbf{x}}^*}$ based on the confidence interval

Training for Randomized Smoothing

- If we train the base classifier f normally the predictions for the noisy inputs might not be accurate for large variance σ^2.
- Idea: **data augmentation** during training.
- Instead of training on the original data we randomly perturb it during training:

$$\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right)$$

- and train on $\tilde{\mathbf{x}}$ rather then $\mathbf{x}$ to make sure f can predict the noisy inputs well.
- Note: we re-sample a new $\boldsymbol{\epsilon}$ every time.

- We can also (approximately) train the smoothed classifier g or even perform adversarial training (e.g. with FGSM) on g but this is outside of the scope.

Randomized Smoothing – Summary

- Randomized smoothing is an easy to implement certification method for perturbations which are L_2 bounded.
- It does not make any assumptions about the model, i.e. we can certify any deep neural network with this strategy.
- Since we need many samples for high-quality certificates, evaluating the model should be relatively cheap.
- Randomized smoothing also lends itself to robust training.

Questions – Rob3

1. Suppose we define a **local** variant of the Lipschitz constant around a given point $\mathbf{x}_0$ as follows
$$\mathcal{D}_{\mathcal{Y}}\big(f(\mathbf{x}_0), f(\mathbf{x})\big) \leq k_{\mathbf{x}_0} \cdot \mathcal{D}_{\mathcal{X}}(\mathbf{x}_0, \mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{X}$$
How does the local constant $k_{\mathbf{x}_0}$ relate to the global constant k of f? Which one would provide better guarantees and why?

2. For which class of classifiers does the certificate for the smoothed classifier g equal the certificate for the underlying base classifier f and why?

3. Given two classifiers f_1 and f_2 with Lipschitz constants k_1 and k_2 with $k_1 < k_2$ which one would provide better guarantees? What if we form smoothed classifiers g_1 and g_2?

Robustness of Machine Learning Models: Summary

- Robustness of machine learning models is a crucial requirement to enable their application in the real world.
- As we have seen, there are **multiple strategies** for **certifying** the robustness of machine learning models and for **robust training**.
- The certification strategies we covered were for L_p-bounded perturbations.
- Of course there are many **other relevant perturbations**, e.g. rotations, translations, illumination changes, etc.
- While adversarial training is often easy to implement even for those perturbations, robustness certification for these scenarios is an active research field.
- **Our Chair** is active in robustness certification for images, vector data, and non-i.i.d. data, e.g. graphs (will also be covered in this course).

References

- Cohen, Jeremy M., Elan Rosenfeld, and J. Zico Kolter. "Certified adversarial robustness via randomized smoothing." Proceedings of the 36th International Conference on Machine Learning, PMLR 97:1310-1320 (2019a).
- Cohen, Jeremy M., Elan Rosenfeld, and J. Zico Kolter. "Certified adversarial robustness via randomized smoothing." Presentation slides at ICML (2019b). Available at https://icml.cc/media/Slides/icml/2019/grandball(12-11-00)-12-11-30-4744-certified_adver.pdf