TU WIEN

Probably Approximately Global Robustness Certification

Peter Blohm, Patrick Indri, Thomas Gärtner, Sagar Malhotra, RuML @ TU Wien

December 19, 2024

Problem Setting: Certification of Neural Network Robustness

Black-Box ML in Critical Applications: Why not?

Black-Box ML in Critical Applications: Why not?

Goodfellow et al (2015)

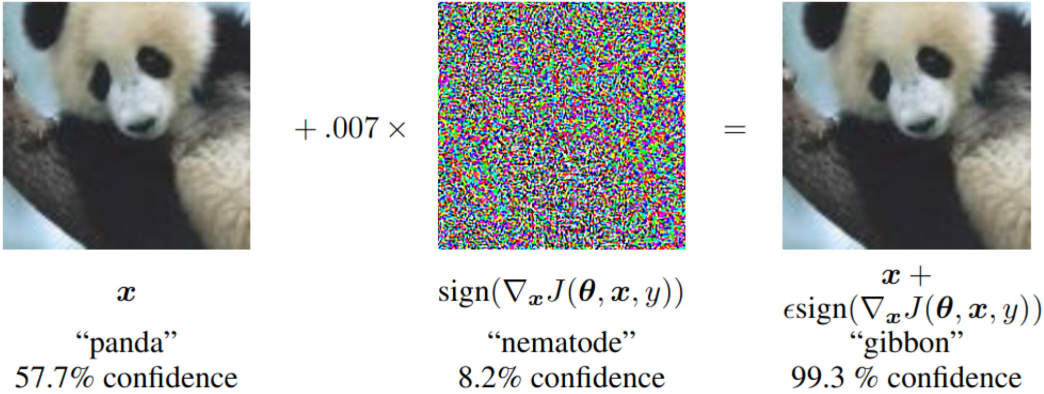


Image Source: Goodfellow et al (2015)

Black-Box ML in Critical Applications: Why not?

Athalye et al (2018)

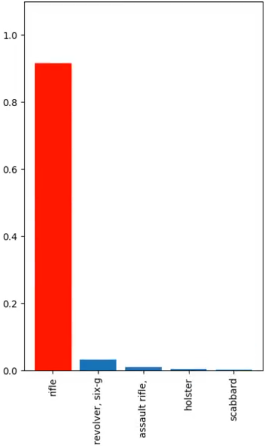


Image Source: Youtube Video

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: “similar” inputs $\rightarrow$ “similar” outputs

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: “similar” inputs $\rightarrow$ “similar” outputs
  Adv. examples are indicator of bad generalization

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: “similar” inputs $\rightarrow$ “similar” outputs
  Adv. examples are indicator of bad generalization
- Measurements are often noisy

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: “similar” inputs $\rightarrow$ “similar” outputs
  Adv. examples are indicator of bad generalization
- Measurements are often noisy
  Classification might be unstable

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: “similar” inputs $\rightarrow$ “similar” outputs
  Adv. examples are indicator of bad generalization
- Measurements are often noisy
  Classification might be unstable
- (Intentional) misclassification might have dangerous consequences

Black-Box ML in Critical Applications: Why not?

Naive Question: Why are adversarial examples an issue?

- Intuitively: "similar" inputs $\rightarrow$ "similar" outputs
  Adv. examples are indicator of bad generalization
- Measurements are often noisy
  Classification might be unstable
- (Intentional) misclassification might have dangerous consequences

Adversarial examples are a security risk

Black-Box ML in Critical Applications: How?

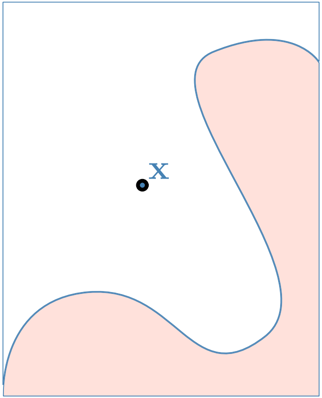
We formalize: “similar” inputs $\rightarrow$ “similar” outputs

Black-Box ML in Critical Applications: How?

We formalize: "similar" inputs $\rightarrow$ "similar" outputs

Definition (Robust Classifier)

We call a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^n$ *robust* around a point $\mathbf{x} \in \mathcal{X}$ iff $\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \textbf{class}(f(\mathbf{x}')) = \textbf{class}(f(\mathbf{x}))$

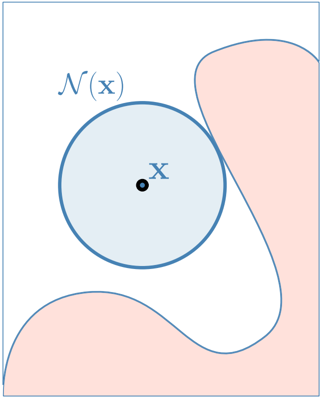


Black-Box ML in Critical Applications: How?

We formalize: “similar” inputs $\rightarrow$ “similar” outputs

Definition (Robust Classifier)

We call a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^n$ *robust* around a point $\mathbf{x} \in \mathcal{X}$ iff $\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \textbf{class}(f(\mathbf{x}')) = \textbf{class}(f(\mathbf{x}))$



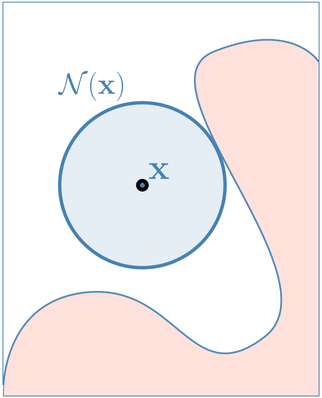
Black-Box ML in Critical Applications: How?

We formalize: "similar" inputs $\rightarrow$ "similar" outputs

Definition (Robust Classifier)

We call a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^n$ *robust* around a point $\mathbf{x} \in \mathcal{X}$ iff $\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{class}(f(\mathbf{x}')) = \mathbf{class}(f(\mathbf{x}))$

We focus on *certification* of robustness



Adversarial Robustness: Projected Gradient Descent (PGD) Madry et al (2018)

One of *many* adversarial attacks

Idea: use gradient descent to optimize input towards a given class y

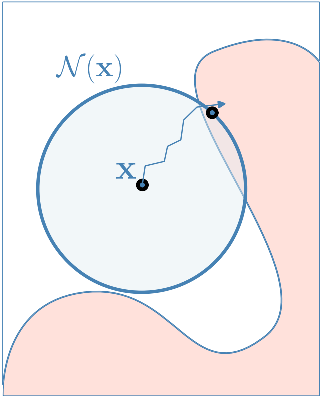
Adversarial Robustness: Projected Gradient Descent (PGD)

Madry et al (2018)

One of *many* adversarial attacks

Idea: use gradient descent to optimize input towards a given class y

Start with input $\mathbf{x}$, classifier f and neighborhood $\mathcal{N}(\mathbf{x})$ and then iteratively



Adversarial Robustness: Projected Gradient Descent (PGD) (Madry et al (2018))

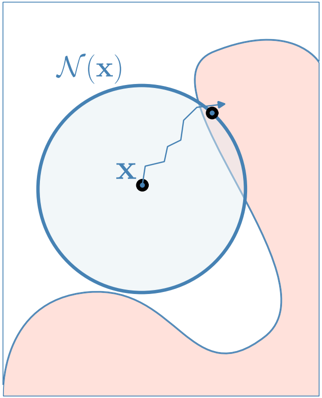
One of *many* adversarial attacks

Idea: use gradient descent to optimize input towards a given class y

Start with input $\mathbf{x}$, classifier f and neighborhood $\mathcal{N}(\mathbf{x})$ and then iteratively

$$\mathbf{x}^{(t+1)} = \Pi_{\mathcal{N}(\mathbf{x})}(\mathbf{x}^{(t)} + \alpha \operatorname{sgn}(\nabla_{\mathbf{x}} L(\mathbf{x}, y))) \tag{1}$$

Where $\Pi(.)$ projects its argument back into $\mathcal{N}(x)$



Adversarial Robustness: Projected Gradient Descent (PGD) (Madry et al (2018))

One of *many* adversarial attacks

Idea: use gradient descent to optimize input towards a given class y

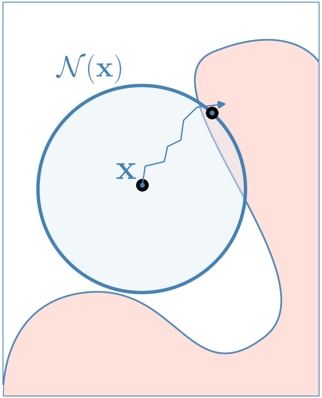
Start with input $\mathbf{x}$, classifier f and neighborhood $\mathcal{N}(\mathbf{x})$ and then iteratively

$$\mathbf{x}^{(t+1)} = \Pi_{\mathcal{N}(\mathbf{x})}(\mathbf{x}^{(t)} + \alpha \operatorname{sgn}(\nabla_{\mathbf{x}} L(\mathbf{x}, y))) \quad (1)$$

Where $\Pi(.)$ projects its argument back into $\mathcal{N}(x)$

Good at finding adversaries... *but not exhaustive!*

(Finding adversarial examples is *hard* (Carlini and Wagner (2017)))



Adversarial Robustness ctd.

Question: How do we test if a classifier is robust *for all inputs*?

Adversarial Robustness ctd.

Question: How do we test if a classifier is robust *for all inputs*?

Adversarial Approach: Just test a bunch of inputs!

Adversarial Robustness ctd.

Question: How do we test if a classifier is robust *for all inputs*?

Adversarial Approach: Just test a bunch of inputs!
But: How to interpret results?

Method	Architecture	PGD10	AutoAttack	Remark
AT	ResNet18	52.73	48.67	
MART	ResNet18	54.73	47.51	
TRADES	ResNet18	53.47	49.45	
AT	ResNet18	55.52	50.80	
MART	ResNet18	57.64	50.03	
TRADES	ResNet18	55.91	**51.62**	👑

Image Source: MAIR Framework Github

Adversarial Robustness ctd.

Question: How do we test if a classifier is robust *for all inputs*?

Adversarial Approach: Just test a bunch of inputs!
But: How to interpret results?

- Results depend attack parameters
- Information gain about f is limited

Method	Architecture	PGD10	AutoAttack	Remark
AT	ResNet18	52.73	48.67	
MART	ResNet18	54.73	47.51	
TRADES	ResNet18	53.47	49.45	
AT	ResNet18	55.52	50.80	
MART	ResNet18	57.64	50.03	
TRADES	ResNet18	55.91	**51.62**	👑

Image Source: MAIR Framework Github

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

- Requires encoding of f as constraint model

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

- Requires encoding of f as constraint model
- f is robust around $\mathbf{x}$ iff following formula holds

$$\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \qquad (2)$$

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

- Requires encoding of f as constraint model
- f is robust around $\mathbf{x}$ iff following formula holds

$$\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \qquad (2)$$

- How to show f is globally robust?

Formal Verification of NN Robustness

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

- Requires encoding of f as constraint model
- f is robust around $\mathbf{x}$ iff following formula holds

$$\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \qquad (2)$$

- How to show f is globally robust?

$$\forall \mathbf{x} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \qquad (3)$$

Is too strict!

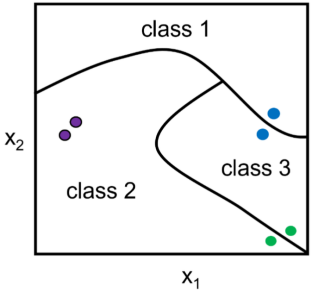


Image Source: Athavale et al (2024)

Formal Verification of NN Robustness ctd.

Leino et al (2021); Athavale et al (2024)

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

Formal Verification of NN Robustness ctd.

Leino et al (2021); Athavale et al (2024)

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

We give the network the option to abstain and only consider *confident* predictions

$$\forall \mathbf{x} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \Rightarrow \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \tag{4}$$

Formal Verification of NN Robustness ctd.

Leino et al (2021); Athavale et al (2024)

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

We give the network the option to abstain and only consider *confident* predictions

$$\forall \mathbf{x} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \Rightarrow \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \quad (4)$$

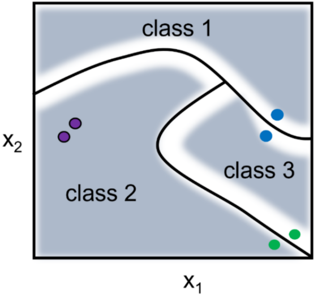


Image Source: Athavale et al (2024)

Formal Verification of NN Robustness ctd.

Leino et al (2021); Athavale et al (2024)

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

We give the network the option to abstain and only consider *confident* predictions

$$\forall \mathbf{x} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \Rightarrow \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \quad (4)$$

$\mathbf{conf}_f(\mathbf{x})$ can be e.g. the *Softmax confidence*

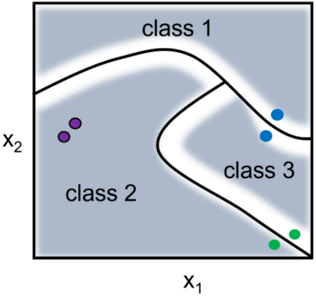


Image Source: Athavale et al (2024)

Formal Verification of NN Robustness ctd.

Leino et al (2021); Athavale et al (2024)

Question: How do we test if a classifier is robust *for all inputs*?

Formal Verification Approach: Prove there exists no counter example (MIP, SMT)!

We give the network the option to abstain and only consider *confident* predictions

$$\forall \mathbf{x} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}) : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \Rightarrow \mathbf{class}(f(\mathbf{x})) = \mathbf{class}(f(\mathbf{x}')) \quad (4)$$

$\mathbf{conf}_f(\mathbf{x})$ can be e.g. the *Softmax confidence*

Very expensive, infeasible above 100s of neurons

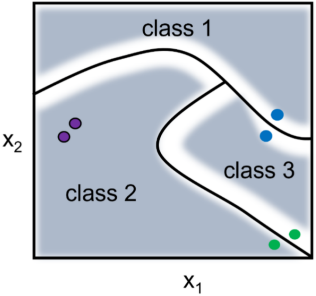


Image Source: Athavale et al (2024)

Global Robustness: Adversarial vs. Formal

Adversarial Robustness Techniques

- sample based
- fast
- (often) no bounds
- limited information required
- How to choose parameters?

Global Robustness: Adversarial vs. Formal

Adversarial Robustness Techniques

- sample based
- fast
- (often) no bounds
- limited information required
- How to choose parameters?

Formal Verification

- expensive locally
- intractable globally
- Proof or Counterexample
- Model needs to be encoded
- Where to verify robustness?

Global Robustness: Adversarial vs. Formal

Adversarial Robustness Techniques

- sample based
- fast
- (often) no bounds
- limited information required
- How to choose parameters?

Formal Verification

- expensive locally
- intractable globally
- Proof or Counterexample
- Model needs to be encoded
- Where to verify robustness?

Our Objective:

- give sample based guarantees about global robustness
- Stay model-agnostic
- *Give specific robustness bounds for each prediction*

Background: Probabilistic Coverage Guarantees with Epsilon-nets

ϵ-Nets

For a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^n$, we want to define a notion of *coverage* of a space under a data distribution $\mathcal{D}$

ϵ-Nets

For a classifier $f : \mathcal{X} \to \mathbb{R}^n$, we want to define a notion of *coverage* of a space under a data distribution $\mathcal{D}$

Definition (Range-Space)

Let $\mathcal{X}$ be a set and $\mathcal{R}$ a set of ranges, where $R \in \mathcal{R} : R \subset \mathcal{X}$ Then $(\mathcal{X}, \mathcal{R})$ is a *range space*

ϵ-Nets

For a classifier $f : \mathcal{X} \to \mathbb{R}^n$, we want to define a notion of *coverage* of a space under a data distribution $\mathcal{D}$

Definition (Range-Space)

Let $\mathcal{X}$ be a set and $\mathcal{R}$ a set of ranges, where $R \in \mathcal{R} : R \subset \mathcal{X}$ Then $(\mathcal{X}, \mathcal{R})$ is a *range space*

Definition (ϵ-Nets)

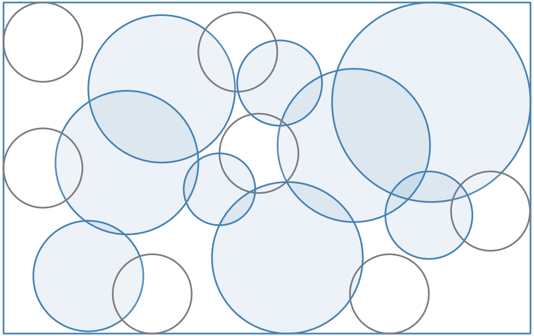
Given a range space $(\mathcal{X}, \mathcal{R})$ and a probability distribution $\mathcal{D}$, a finite set $N \subset \mathcal{X}$ is called an ϵ-*net*, iff N intersects each ϵ-probable $R \in \mathcal{R}$, i.e.,

$$\forall R \in \mathcal{R} : \Pr(R) \geq \epsilon \Rightarrow N \cap R \neq \emptyset \quad \Leftrightarrow \tag{5}$$

$$\forall R \in \mathcal{R} : N \cap R = \emptyset \Rightarrow \Pr(R) < \epsilon \tag{6}$$

ϵ-Nets: Example

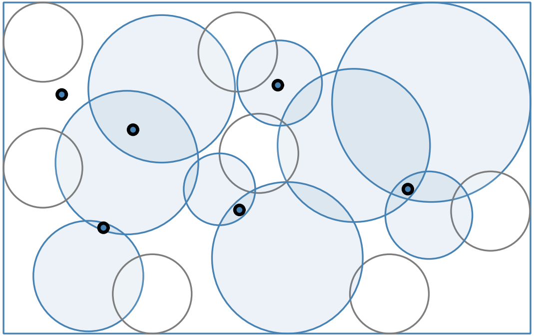
We consider the range space $(\mathbb{R}^2, \mathcal{B})$, with $\mathcal{B}$ is some set of circles



ϵ-Nets: Example

We consider the range space $(\mathbb{R}^2, \mathcal{B})$, with $\mathcal{B}$ is some set of circles

An ϵ-net intersects all likely enough circles



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension Vapnik and Chervonenkis (2015))

Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$

Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

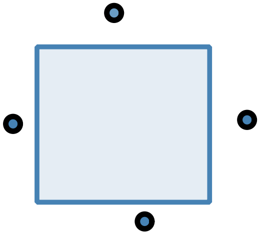
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

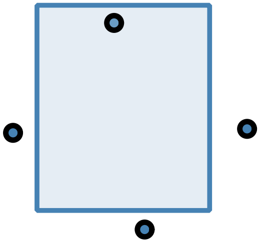
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

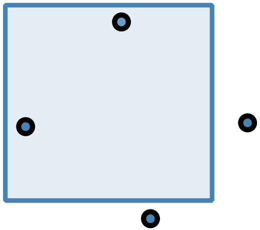
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

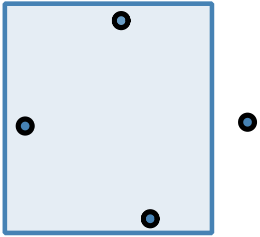
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

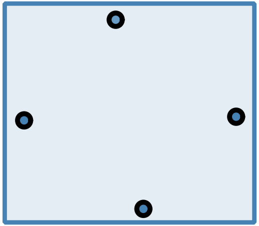
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension Vapnik and Chervonenkis (2015))

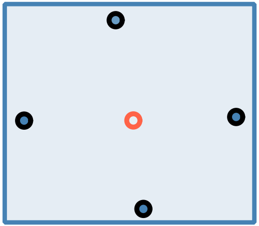
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$
- $\forall S : |S| \geq 5$, no shattering $\Rightarrow d \leq 4$



Vapnik-Chervonenkis-Dimension

Definition (VC-Dimension (Vapnik and Chervonenkis (2015)))

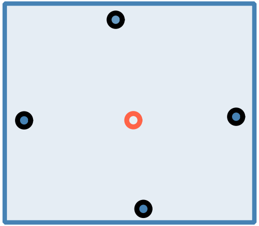
Let $(\mathcal{X}, \mathcal{R})$ be a range space. The Vapnik-Chervonenkis (VC) dimension d of $(\mathcal{X}, \mathcal{R})$ is the size of the largest set $S \subseteq \mathcal{X}$, such that

$$\forall S' \subseteq S : \exists R \in \mathcal{R} : R \cap S = S' \tag{7}$$

where we say S is *shattered* by $\mathcal{R}$

Example (Rectangles in $\mathbb{R}^2$)

- $\exists S : |S| = 4$ with shattering $\Rightarrow d \geq 4$
- $\forall S : |S| \geq 5$, no shattering $\Rightarrow d \leq 4$



Well studied for common hypothesis spaces

ϵ-Nets from iid Samples

Theorem (ϵ-nets from iid samples (Mitzenmacher and Upfal (2017)) **)**

Let $(\mathcal{X}, \mathcal{R})$ be a range-space with VC-dimension d and $\mathcal{D}$ be a probability distribution. For any $0 < \delta, \epsilon \leq \frac{1}{2}$, an iid sample N will be an ϵ-net with probability at least $1 - \delta$ iff

$$|N| = \mathcal{O}\left(\frac{d}{\epsilon} \ln \frac{d}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\delta}\right) \quad (8)$$

ϵ-Nets from iid Samples

Theorem (ϵ-nets from iid samples (Mitzenmacher and Upfal (2017)))

Let $(\mathcal{X}, \mathcal{R})$ be a range-space with VC-dimension d and $\mathcal{D}$ be a probability distribution. For any $0 < \delta, \epsilon \leq \frac{1}{2}$, an iid sample N will be an ϵ-net with probability at least $1 - \delta$ iff

$$|N| = \mathcal{O}\left(\frac{d}{\epsilon} \ln \frac{d}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\delta}\right) \tag{8}$$

We are interested in obtaining minimal samples of sufficient size, so we find $|N| = s$ with

$$s(\epsilon, \delta, d) = \min_{s \in \mathbb{N}} \left\{ s : s \geq \frac{2}{\epsilon}\left(\log\left(\frac{2}{\delta}\right) + d \log(2s)\right)\right\} \tag{9}$$

Distillation with probably approximately global coverage

Problem Setting (Indri et al (2024))

We have formal tools that can prove global robustness for only very small NNs

Problem Setting (Indri et al (2024))

We have formal tools that can prove global robustness for *only very small* NNs

Problem Setting

Indri et al (2024)

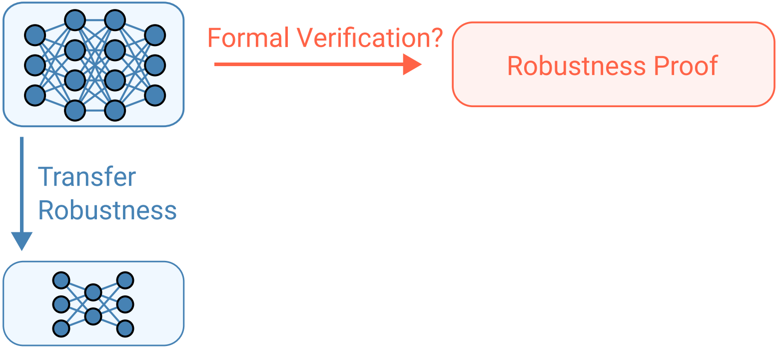
We have formal tools that can prove global robustness for *only very small* NNs

Question: How can we use these tools for larger networks?

Problem Setting (Indri et al (2024))

We have formal tools that can prove global robustness for *only very small* NNs

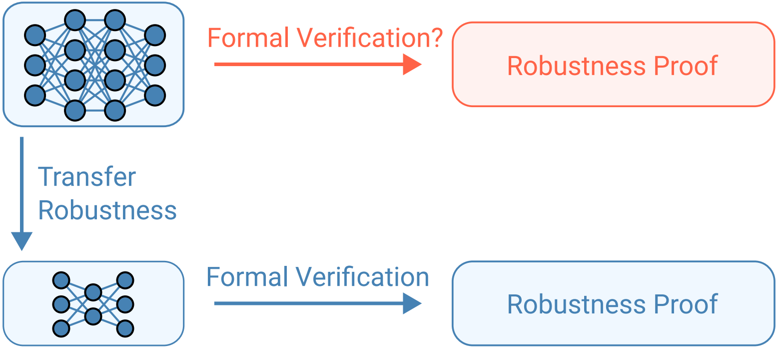
Question: How can we use these tools for larger networks?



Problem Setting (Indri et al (2024))

We have formal tools that can prove global robustness for *only very small* NNs

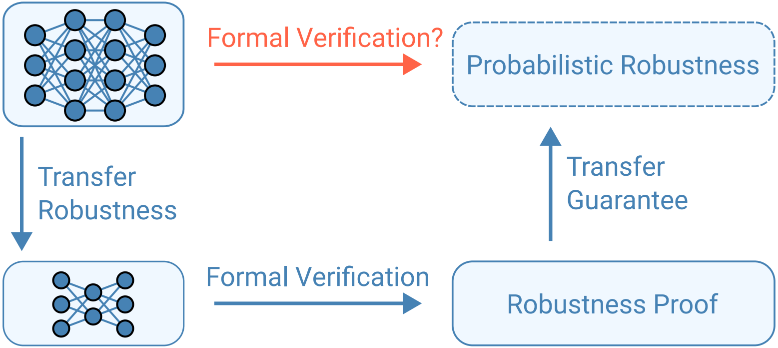
Question: How can we use these tools for larger networks?



Problem Setting (Indri et al (2024))

We have formal tools that can prove global robustness for *only very small* NNs

Question: How can we use these tools for larger networks?



Gradient-Aligned Distillation (Shao et al (2021))

We train a small NN f_S to simulate a given NN f_T
The training tries to minimize the difference in *assigned* labels, logits and gradients

Gradient-Aligned Distillation

Shao et al (2021)

We train a small NN f_S to simulate a given NN f_T

The training tries to minimize the difference in *assigned* labels, logits and gradients

We optimize for

$$\begin{aligned} \mathcal{L}_{CE}(f_s(\mathbf{x}), y) + \mathcal{L}_{KL}(f_s(\mathbf{x}), f_t(\mathbf{x})) + \\ \|\nabla_{\mathbf{x}} \mathcal{L}_{CE}(f_t(\mathbf{x}), y) - \nabla_{\mathbf{x}} \mathcal{L}_{CE}(f_s(\mathbf{x}), y)\| \end{aligned} \quad (10)$$

Where we will use $y = \mathbf{class}(f_T(\mathbf{x}))$

Gradient-Aligned Distillation (Shao et al (2021))

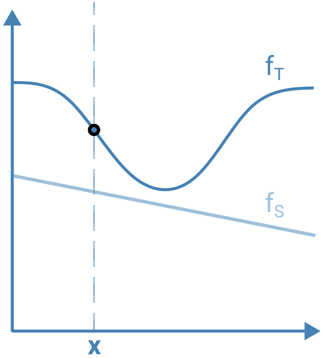
We train a small NN f_S to simulate a given NN f_T

The training tries to minimize the difference in *assigned* labels, logits and gradients

We optimize for

$$\mathcal{L}_{CE}(f_s(\mathbf{x}), y) + \mathcal{L}_{KL}(f_s(\mathbf{x}), f_t(\mathbf{x}))+ \\ \|\nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_t(\mathbf{x}), y) - \nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_s(\mathbf{x}), y)\| \tag{10}$$

Where we will use y = $\textbf{class}(f_T(\mathbf{x}))$

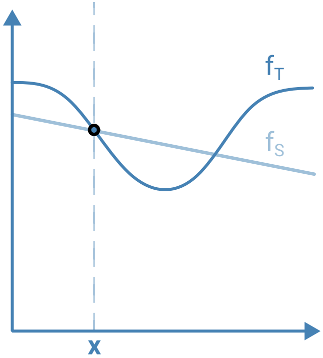


Gradient-Aligned Distillation (Shao et al (2021))

- We train a small NN f_S to simulate a given NN f_T
- The training tries to minimize the difference in *assigned* labels, logits and gradients

We optimize for

$$\mathcal{L}_{CE}(f_s(\mathbf{x}), y) + \mathcal{L}_{KL}(f_s(\mathbf{x}), f_t(\mathbf{x})) + \\ \|\nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_t(\mathbf{x}), y) - \nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_s(\mathbf{x}), y)\| \quad (10)$$



Where we will use $y = \textbf{class}(f_T(\mathbf{x}))$

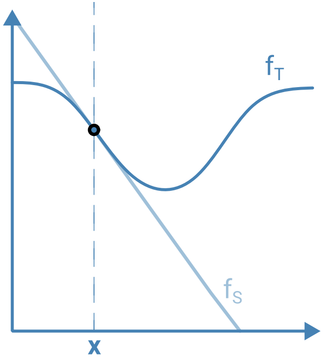
Gradient-Aligned Distillation

Shao et al (2021)

- We train a small NN f_S to simulate a given NN f_T
- The training tries to minimize the difference in *assigned* labels, logits and gradients

We optimize for

$$\mathcal{L}_{CE}(f_s(\mathbf{x}), y) + \mathcal{L}_{KL}(f_s(\mathbf{x}), f_t(\mathbf{x})) + \\ \|\nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_t(\mathbf{x}), y) - \nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_s(\mathbf{x}), y)\| \tag{10}$$



Where we will use $y = \mathbf{class}(f_T(\mathbf{x}))$

Under *perfect conditions*, f_S is as robust as f_T

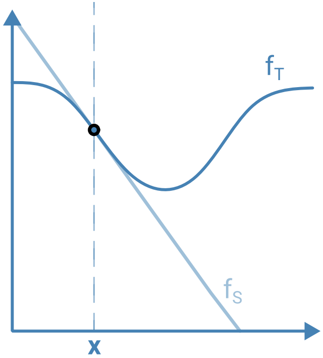
Gradient-Aligned Distillation

Shao et al (2021)

We train a small NN f_S to simulate a given NN f_T
The training tries to minimize the difference in *assigned* labels, logits and gradients

We optimize for

$$\mathcal{L}_{CE}(f_s(\mathbf{x}), y) + \mathcal{L}_{KL}(f_s(\mathbf{x}), f_t(\mathbf{x})) + \|\nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_t(\mathbf{x}), y) - \nabla_{\mathbf{x}}\mathcal{L}_{CE}(f_s(\mathbf{x}), y)\| \quad (10)$$



Where we will use $y = \textbf{class}(f_T(\mathbf{x}))$
Under *perfect conditions*, f_S is as robust as f_T
Assumes both functions are linear in a metric ball $B_r(\mathbf{x})$!

Transferring Robustness Guarantees

Question: How can we transfer guarantees back from f_S to f_T?

Transferring Robustness Guarantees

Question: How can we transfer guarantees back from f_S to f_T?

Distill on an ϵ-net N over metric balls!

Transferring Robustness Guarantees

Question: How can we transfer guarantees back from f_S to f_T?

Distill on an ϵ-net N over metric balls! Informally:

1. We will intersect all ϵ-likely metric balls under $\mathcal{D}$
2. For $\mathbf{x} \in N$, f_S and f_T have same robustness around $\mathbf{x}$ in $B_r(\mathbf{x})$

Transferring Robustness Guarantees

Question: How can we transfer guarantees back from f_S to f_T?

Distill on an ϵ-net N over metric balls! Informally:

1. We will intersect all ϵ-likely metric balls under $\mathcal{D}$
2. For $\mathbf{x} \in N$, f_S and f_T have same robustness around $\mathbf{x}$ in $B_r(\mathbf{x})$

$\Rightarrow$ If f_S is *globally* robust, f_T is robust in all ϵ-likely metric balls

Transferring Robustness Guarantees

Question: How can we transfer guarantees back from f_S to f_T?

Distill on an ϵ-net N over metric balls! Informally:

1. We will intersect all ϵ-likely metric balls under $\mathcal{D}$
2. For $\mathbf{x} \in N$, f_S and f_T have same robustness around $\mathbf{x}$ in $B_r(\mathbf{x})$

$\Rightarrow$ If f_S is *globally* robust, f_T is robust in all ϵ-likely metric balls

We sample sufficiently N iid from some dataset with additive noise

Does It Work? Experimental Results

We constructed f_T with known robustness properties and checked if robustness transferred through distillation

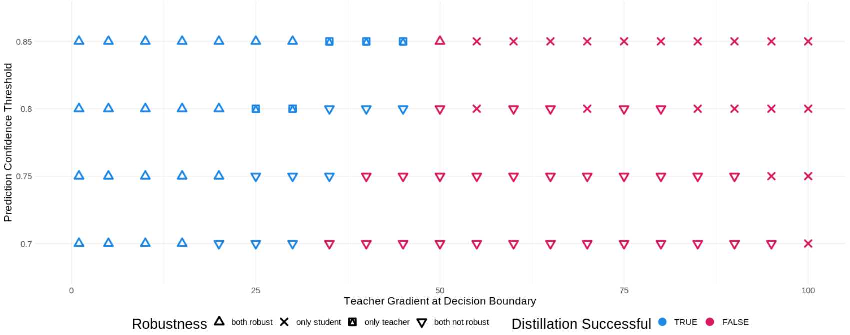


Image Source: Indri et al (2024)

Does It Work? Issues With This Approach

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**
  This trivializes checking robustness! Why not use tangent planes directly?

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space

Does It Work? Issues With This Approach

- Robustness around $\mathbf{x}$ transfers if f_S,f_T are linear around $\mathbf{x}$
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*

Does It Work? Issues With This Approach

- Robustness around $\mathbf{x}$ transfers if f_S,f_T are linear around $\mathbf{x}$
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*
- We cover all ϵ-likely metric balls

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*
- We cover all ϵ-likely metric balls
  But what does this mean?

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*
- We cover all ϵ-likely metric balls
  But what does this mean?
  - How can we detect ϵ-likely balls?

Does It Work? Issues With This Approach

- Robustness around **x** transfers if f_S,f_T are linear around **x**
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*
- We cover all ϵ-likely metric balls
  But what does this mean?
  - How can we detect ϵ-likely balls?
  - If we consider balls of any size: we require local linearity at arbitrary scale

Does It Work? Issues With This Approach

- Robustness around $\mathbf{x}$ transfers if f_S,f_T are linear around $\mathbf{x}$
  This trivializes checking robustness! Why not use tangent planes directly?
- We construct an ϵ-net over metric balls in the input space
  N *scales linearly with the input dimension, expensive for high dimensional data*
- We cover all ϵ-likely metric balls
  But what does this mean?
  - How can we detect ϵ-likely balls?
  - If we consider balls of any size: we require local linearity at arbitrary scale
  - If we consider only small balls: maybe none are ϵ-likely (high dimensional data)

Property-Based Robustness Guarantees

Addressing issues: Local Robustness Checks

Question: *Why not check local robustness directly for any* f*?*

Addressing issues: Local Robustness Checks

Question: *Why not check local robustness directly for any* f*?*

- Local Robustness verification is tractable
- We can use different assumptions about f to use different tools

Addressing issues: Local Robustness Checks

Question: *Why not check local robustness directly for any f?*

- Local Robustness verification is tractable
- We can use different assumptions about f to use different tools

Definition ((Local) Robustness Oracle)

For a classifier $f : \mathcal{X} \to \mathbb{R}^n$, a *robustness oracle* is defined as

$$\mathbf{rob}_f(\mathbf{x}) = \min_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} \{\|\mathbf{x} - \mathbf{x}'\| : \mathbf{class}(\mathbf{x}) \neq \mathbf{class}(\mathbf{x}')\} \tag{11}$$

and returns the *robustness radius* ρ

Addressing issues: Local Robustness Checks

Question: *Why not check local robustness directly for any f?*

- Local Robustness verification is tractable
- We can use different assumptions about f to use different tools

Definition ((Local) Robustness Oracle)

For a classifier $f : \mathcal{X} \to \mathbb{R}^n$, a *robustness oracle* is defined as

$$\mathbf{rob}_f(\mathbf{x}) = \min_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} \{\|\mathbf{x} - \mathbf{x}'\| : \mathbf{class}(\mathbf{x}) \neq \mathbf{class}(\mathbf{x}')\} \tag{11}$$

and returns the *robustness radius* ρ

We can also use (non-exact) oracles that find a counterexample with attacks (e.g. PGD)

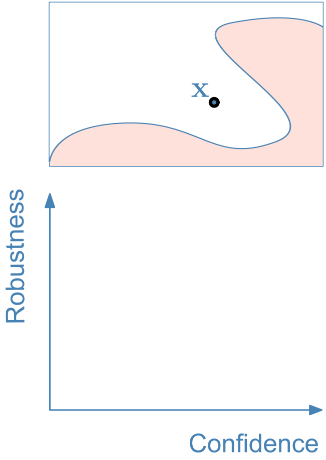
Quality Space

Question: Do we really need to cover the *input* space?

Quality Space

Question: Do we really need to cover the *input* space?
We recall when f is (ρ, κ)-robust:

$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{rob}_f(\mathbf{x}) < \rho \wedge \mathbf{conf}_f(\mathbf{x}) \geq \kappa \tag{12}$$



Quality Space

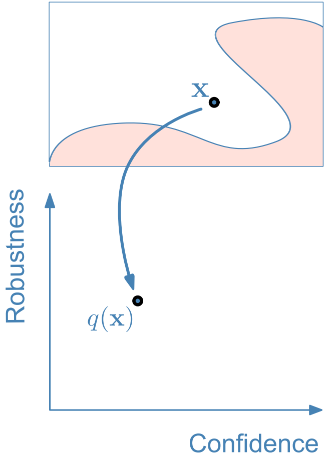
Question: Do we really need to cover the *input* space?
We recall when f is (ρ, κ)-robust:

$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{rob}_f(\mathbf{x}) < \rho \wedge \mathbf{conf}_f(\mathbf{x}) \geq \kappa \tag{12}$$

We can assume an explicit map into $\mathbb{R}^2$:

$$q(\mathbf{x}) \mapsto (\mathbf{rob}_f(\mathbf{x}), \mathbf{conf}_f(\mathbf{x})) \tag{13}$$

We call this space the *quality space* $\mathcal{Q} = \mathbb{R}^2$



Quality Space

Question: Do we really need to cover the *input* space?
We recall when f is (ρ, κ)-robust:

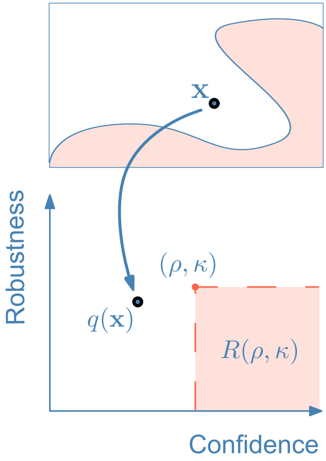
$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{rob}_f(\mathbf{x}) < \rho \wedge \mathbf{conf}_f(\mathbf{x}) \geq \kappa \tag{12}$$

We can assume an explicit map into $\mathbb{R}^2$:

$$q(\mathbf{x}) \mapsto (\mathbf{rob}_f(\mathbf{x}), \mathbf{conf}_f(\mathbf{x})) \tag{13}$$

We call this space the *quality space* $\mathcal{Q} = \mathbb{R}^2$ and define all counterexamples to robustness

$$R(\rho, \kappa) = \{(\rho', \kappa') \in \mathbb{R}^2 : \rho' < \rho, \kappa' \geq \kappa\} \tag{14}$$



Quality Space ctd.

Question: How does the representation in $\mathcal{Q}$ help us here?

Quality Space ctd.

Question: How does the representation in $\mathcal{Q}$ help us here?

We want to prove a relaxed version of global robustness

$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \wedge \mathbf{rob}_f(\mathbf{x}) < \rho \tag{15}$$

Quality Space ctd.

Question: How does the representation in $\mathcal{Q}$ help us here?

We want to prove a relaxed version of global robustness

$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \wedge \mathbf{rob}_f(\mathbf{x}) < \rho \tag{15}$$

We can bound the probability of a random point X being a counterexample:

$$\Pr(\mathbf{rob}_f(X) < \rho \wedge \mathbf{conf}_f(X) \geq \kappa) = \Pr(R(\rho, \kappa)) < \epsilon \tag{16}$$

Quality Space ctd.

Question: How does the representation in $\mathcal{Q}$ help us here?

We want to prove a relaxed version of global robustness

$$\nexists \mathbf{x} \in \mathcal{X} : \mathbf{conf}_f(\mathbf{x}) \geq \kappa \wedge \mathbf{rob}_f(\mathbf{x}) < \rho \tag{15}$$

We can bound the probability of a random point X being a counterexample:

$$\Pr(\mathbf{rob}_f(X) < \rho \wedge \mathbf{conf}_f(X) \geq \kappa) = \Pr(R(\rho, \kappa)) < \epsilon \tag{16}$$

Answer: *We can sample ϵ-nets in $\mathcal{Q}$!*

Quality Space ctd.

Let the family of ranges be

$$\mathcal{R} = \{R(\rho, \kappa) : (\rho, \kappa) \in \mathbb{R}^2\} \tag{17}$$

Quality Space ctd.

Let the family of ranges be

$$\mathcal{R} = \{R(\rho, \kappa) : (\rho, \kappa) \in \mathbb{R}^2\} \tag{17}$$

If N is an ϵ-net of $(\mathcal{Q}, \mathcal{R})$ then

$$\forall \rho, \kappa \in \mathbb{R} : N \cap R(\rho, \kappa) = \emptyset \Rightarrow \Pr(R(\rho, \kappa)) < \epsilon \tag{18}$$

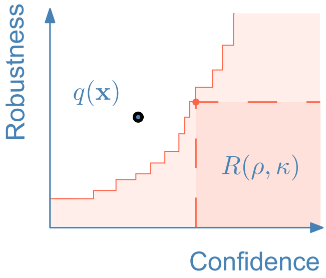
Quality Space ctd.

Let the family of ranges be

$$\mathcal{R} = \{R(\rho, \kappa) : (\rho, \kappa) \in \mathbb{R}^2\} \tag{17}$$

If N is an ϵ-net of $(\mathcal{Q}, \mathcal{R})$ then

$$\forall \rho, \kappa \in \mathbb{R} : N \cap R(\rho, \kappa) = \emptyset \Rightarrow \Pr(R(\rho, \kappa)) < \epsilon \tag{18}$$



Quality Space ctd.

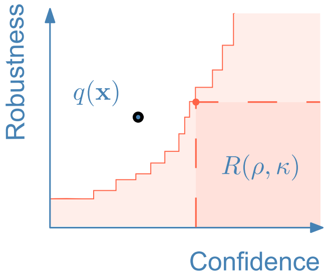
Let the family of ranges be

$$\mathcal{R} = \{R(\rho, \kappa) : (\rho, \kappa) \in \mathbb{R}^2\} \tag{17}$$

If N is an ϵ-net of $(\mathcal{Q}, \mathcal{R})$ then

$$\forall \rho, \kappa \in \mathbb{R} : N \cap R(\rho, \kappa) = \emptyset \Rightarrow \Pr(R(\rho, \kappa)) < \epsilon \tag{18}$$

$|N| = s(\epsilon, \delta, d)$ depends *only on the VC-dimension d of $\mathcal{R}$, not on $\mathcal{X}$*



Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Question: But what information do we gain?

Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Question: But what information do we gain?

Example (Abstract)

We choose (ρ_1,κ_1) and *N* tells us *f* is (ρ_1,κ_1) robust with probability at least $1 - \epsilon$

Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Question: But what information do we gain?

Example (Abstract)

We choose (ρ_1, κ_1) and *N* tells us *f* is (ρ_1, κ_1) robust with probability at least $1 - \epsilon$
Now we use *f* and obtain 100 points with confidence exactly κ_1

Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Question: But what information do we gain?

Example (Abstract)

We choose (ρ_1,κ_1) and N tells us f is (ρ_1,κ_1) robust with probability at least $1-\epsilon$
Now we use f and obtain 100 points with confidence exactly κ_1
We measure their robustness: None of them are ρ robust!
How can this happen?

Counterexample Robustness

We can now efficiently decide counterexample robustness $\forall(\rho, \kappa)$.

Question: But what information do we gain?

Example (Abstract)

We choose (ρ_1,κ_1) and N tells us f is (ρ_1,κ_1) robust with probability at least $1 - \epsilon$
Now we use f and obtain 100 points with confidence exactly κ_1
We measure their robustness: None of them are ρ robust!
How can this happen?

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \wedge \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \epsilon \tag{19}$$

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \epsilon \tag{20}$$

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \frac{\epsilon}{\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa)} \tag{21}$$

Approximately Global Robustness

Question: Why is a conditional probability bound more useful?

Approximately Global Robustness

Question: Why is a conditional probability bound more useful?

- $\mathbf{conf}_f(\mathbf{x})$ is known at inference time
- $\mathbf{rob}_f(\mathbf{x})$ needs to invoke the robustness oracle

Approximately Global Robustness

Question: Why is a conditional probability bound more useful?

- $\mathbf{conf}_f(\mathbf{x})$ is known at inference time
- $\mathbf{rob}_f(\mathbf{x})$ needs to invoke the robustness oracle

If we can give a conditional statement $\forall(\rho, \kappa)$ we can obtain a robustness radius from the confidence:

$$M(\kappa) = \max_{\rho \in \mathbb{R}} : \Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \epsilon \quad (22)$$

Approximately Global Robustness

Question: Why is a conditional probability bound more useful?

- $\mathbf{conf}_f(\mathbf{x})$ is known at inference time
- $\mathbf{rob}_f(\mathbf{x})$ needs to invoke the robustness oracle

If we can give a conditional statement $\forall(\rho, \kappa)$ we can obtain a robustness radius from the confidence:

$$M(\kappa) = \max_{\rho \in \mathbb{R}} : \Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \epsilon \tag{22}$$

We can use conditional guarantees to give "*customized*" robustness lower bounds for each prediction!

Obtaining Constant Bounds

With ϵ-nets we can only get the bound

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \frac{\epsilon}{\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa)} \tag{23}$$

Obtaining Constant Bounds

With ϵ-nets we can only get the bound

For the case $\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa) > 1 - p_{\max}$

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \frac{\epsilon}{\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa)} < \frac{\epsilon}{1 - p_{\max}} \tag{23}$$

Obtaining Constant Bounds

With ϵ-nets we can only get the bound

For the case $\Pr(\textbf{conf}_f(\textbf{x}) \geq \kappa) > 1 - p_{\max}$

$$\Pr(\textbf{rob}_f(\textbf{x}) < \rho \mid \textbf{conf}_f(\textbf{x}) \geq \kappa) < \frac{\epsilon}{\Pr(\textbf{conf}_f(\textbf{x}) \geq \kappa)} < \frac{\epsilon}{1 - p_{\max}} \tag{23}$$

Question: How do we know for which κ: $\Pr(\textbf{conf}_f(\textbf{x}) < \kappa) \leq p_{\max}$

Obtaining Constant Bounds

With ϵ-nets we can only get the bound

For the case $\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa) > 1 - p_{\text{max}}$

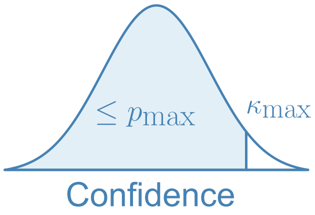
$$\Pr(\mathbf{rob}_f(\mathbf{x}) < \rho \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < \frac{\epsilon}{\Pr(\mathbf{conf}_f(\mathbf{x}) \geq \kappa)} < \frac{\epsilon}{1 - p_{\text{max}}} \tag{23}$$

Question: How do we know for which κ: $\Pr(\mathbf{conf}_f(\mathbf{x}) < \kappa) \leq p_{\text{max}}$

We use *rank statistics* to estimate a bound from the sample!

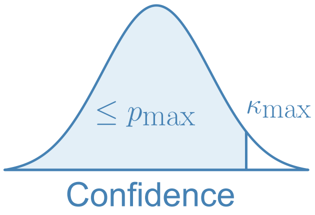
Obtaining Constant Bounds with Rank Statistics

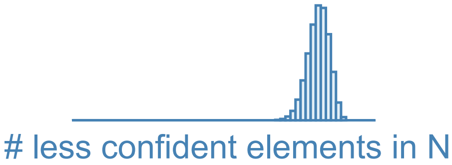
Given a sample N, for which κ:
$\Pr(\mathbf{conf}_f(\mathbf{x}) < \kappa) \leq p_{\text{max}}$?



Obtaining Constant Bounds with Rank Statistics

Given a sample N, for which κ:
$\Pr(\mathbf{conf}_f(\mathbf{x}) < \kappa) \leq p_{\max}$?
We estimate *the rank* of the $p_{\max}$-quantile $\kappa_{\max}$ of κ...

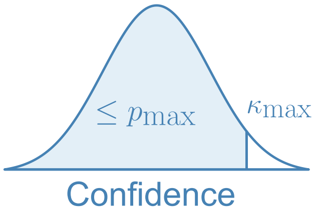


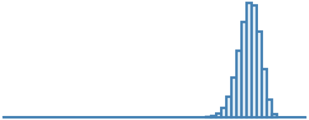


Obtaining Constant Bounds with Rank Statistics

Given a sample N, for which κ:
$\Pr(\mathbf{conf}_f(\mathbf{x}) < \kappa) \leq p_{\max}$?
We estimate *the rank* of the $p_{\max}$-quantile $\kappa_{\max}$ of κ...
Let $N_{(i)}$ be the element in N with ith-biggest confidence

$$\kappa_{p_{\max}} \approxeq N_{(i)} : i = \lfloor |N| p_{\max} \rfloor \tag{24}$$

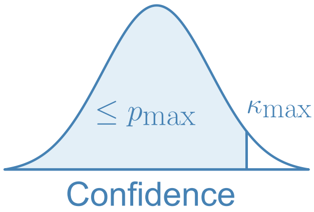




Obtaining Constant Bounds with Rank Statistics

Given a sample N, for which κ:
$\Pr(\mathbf{conf}_f(\mathbf{x}) < \kappa) \leq p_{\max}$?
We estimate *the rank* of the $p_{\max}$-quantile $\kappa_{\max}$ of κ...
Let $N_{(i)}$ be the element in N with ith-biggest confidence

$$\kappa_{p_{\max}} \approxeq N_{(i)} : i = \lfloor |N| p_{\max} \rfloor \tag{24}$$



...and use *Chernoff bounds*

$$\Pr(\kappa_{\max} < N_{(i)}) < \delta \text{ s.t.} \tag{25}$$

$$i < c(|N|, p_{\max}, \delta) = \left\lfloor |N| p_{\max} - \sqrt{2|N| p_{\max} \ln\left(\frac{1}{\delta}\right)} \right\rfloor \tag{26}$$

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \to \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$

For parameters $\epsilon, \delta, p_{\max}$

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \to \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$
For parameters $\epsilon, \delta, p_{\text{max}}$ *we take an iid sample* $N \subset \mathcal{X}$ *with* $|N| \geq s\left(\epsilon, \frac{\delta}{2}, 2\right)$ *and let*
$\kappa_{\text{max}} = c\left(|N|, p_{\text{max}}, \frac{\delta}{2}\right)$

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \to \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$
For parameters $\epsilon, \delta, p_{\max}$ *we take an iid sample* $N \subset \mathcal{X}$ *with* $|N| \geq s\left(\epsilon, \frac{\delta}{2}, 2\right)$ *and let*
$\kappa_{\max} = c\left(|N|, p_{\max}, \frac{\delta}{2}\right)$
Then it holds with probability at least $1 - \delta$ *that*

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \to \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$
For parameters $\epsilon, \delta, p_{\max}$ *we take an iid sample* $N \subset \mathcal{X}$ *with* $|N| \geq s\left(\epsilon, \frac{\delta}{2}, 2\right)$ *and let* $\kappa_{\max} = c\left(|N|, p_{\max}, \frac{\delta}{2}\right)$
Then it holds with probability at least $1 - \delta$ *that*

$$\forall \rho \forall \kappa \leq \kappa_{\max} : \{q(\mathbf{x}) : \mathbf{x} \in N\} \cap R(\rho, \kappa) = \emptyset \tag{27}$$

If we have no counterexample in N,

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \to \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$
For parameters $\epsilon, \delta, p_{\max}$ *we take an iid sample* $N \subset \mathcal{X}$ *with* $|N| \geq s\left(\epsilon, \frac{\delta}{2}, 2\right)$ *and let* $\kappa_{\max} = c\left(|N|, p_{\max}, \frac{\delta}{2}\right)$
Then it holds with probability at least $1 - \delta$ *that*

$$\forall \rho \forall \kappa \leq \kappa_{\max} : \{q(\mathbf{x}) : \mathbf{x} \in N\} \cap R(\rho, \kappa) = \emptyset \tag{27}$$

If we have no counterexample in N, f *is probably*

Probably Approximately Global Robustness Certification

Theorem (PAG Robustness)

Given a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^n$, *a robustness oracle* $\mathbf{rob}_f$ *and a data distribution* $\mathcal{D}$ *over* $\mathcal{X}$
For parameters $\epsilon, \delta, p_{\max}$ *we take an iid sample* $N \subset \mathcal{X}$ *with* $|N| \geq s\left(\epsilon, \frac{\delta}{2}, 2\right)$ *and let*
$\kappa_{\max} = c\left(|N|, p_{\max}, \frac{\delta}{2}\right)$
Then it holds with probability at least $1 - \delta$ *that*

$$\forall \rho \forall \kappa \leq \kappa_{\max} : \{q(\mathbf{x}) : \mathbf{x} \in N\} \cap R(\rho, \kappa) = \emptyset \Rightarrow \Pr\left(\mathbf{rob}_f(X) < \rho \,|\, \mathbf{conf}_f(X) \geq \kappa\right) < \frac{\epsilon}{1 - p_{\max}} \tag{27}$$

If we have no counterexample in N, f *is probably approximately globally* (ρ, κ)*-robust*

Experiments: Setup

Question 1: Do our guarantees hold for *unseen data* in real problems?

Question 2: Do our guarantees provide *constructive* information about classifiers?

Experiments: Setup

Question 1: Do our guarantees hold for *unseen data* in real problems?

Question 2: Do our guarantees provide *constructive* information about classifiers?

We tested MNIST and CIFAR10 classifiers for adversarial robustness against PGD

Experiments: Setup

Question 1: Do our guarantees hold for *unseen data* in real problems?

Question 2: Do our guarantees provide *constructive* information about classifiers?

We tested MNIST and CIFAR10 classifiers for adversarial robustness against PGD

We chose $\epsilon = 10^{-4}$, $p_{\max} = 0.99$, $\delta = 0.02$, with $s(10^{-4}, 0.01, 2) = 670312$ samples

$\mathcal{D}$ is estimated by Gaussian noise around a validation split of the dataset

Experiments: Setup

Question 1: Do our guarantees hold for *unseen data* in real problems?

Question 2: Do our guarantees provide *constructive* information about classifiers?

We tested MNIST and CIFAR10 classifiers for adversarial robustness against PGD

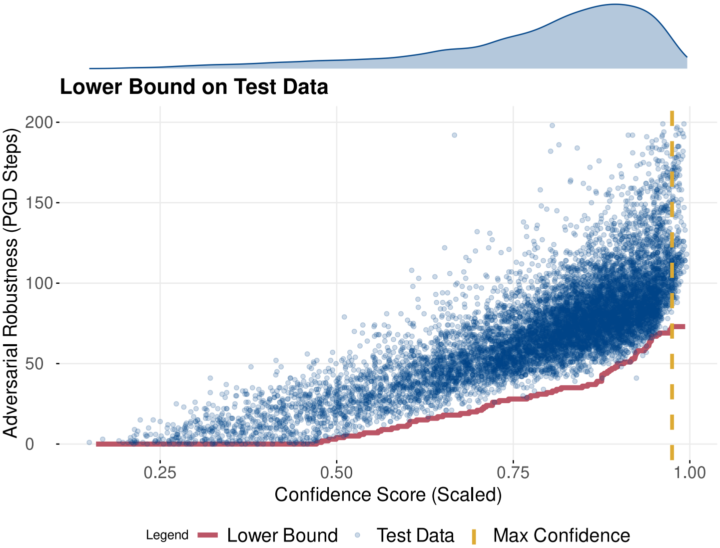
We chose $\epsilon = 10^{-4}$, $p_{\text{max}} = 0.99$, $\delta = 0.02$, with $s(10^{-4}, 0.01, 2) = 670312$ samples

$\mathcal{D}$ is estimated by Gaussian noise around a validation split of the dataset

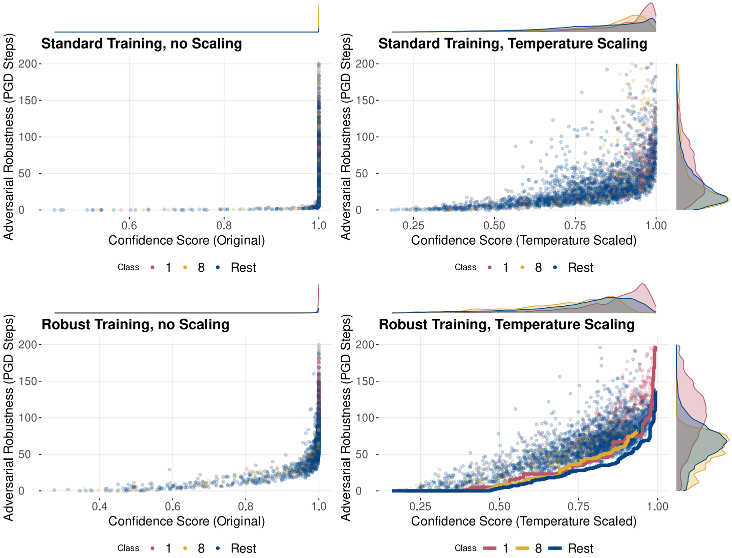
We expect for a given κ

$$\Pr(\mathbf{rob}_f(\mathbf{x}) < M(\kappa) \mid \mathbf{conf}_f(\mathbf{x}) \geq \kappa) < 0.01 \tag{28}$$

Experimental Results



Experimental Results



Summary: Strength of PAG Robustness

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges
  But what does this mean?

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges
  But what does this mean?
  - We can upper-bound probability of empty ranges (no counterexamples found in N)

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges
  But what does this mean?
  - We can upper-bound probability of empty ranges (no counterexamples found in N)
  - We can obtain robustness lower-bounds for predictions with a given confidence

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\textbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges
  But what does this mean?
  - We can upper-bound probability of empty ranges (no counterexamples found in N)
  - We can obtain robustness lower-bounds for predictions with a given confidence
  - We need only one sample N and with probability at least $1 - \delta$ our guarantee hold for all $\kappa < \kappa_{\text{max}}$

Summary: Strength of PAG Robustness

- We perform local robustness checks with the oracle $\mathbf{rob}_f$
  We abstract away from the type of robustness we check we use
- We construct an ϵ-net over metric balls in the quality space $\mathcal{Q}$
  N *is constant with respect to the data-space and the properties of* f
- We cover all ϵ-likely counterexample-ranges
  But what does this mean?
  - We can upper-bound probability of empty ranges (no counterexamples found in N)
  - We can obtain robustness lower-bounds for predictions with a given confidence
  - We need only one sample N and with probability at least $1 - \delta$ our guarantee hold for all $\kappa < \kappa_{\text{max}}$

This method also generalizes to learning other rules in black-box ML

References

Anish Athalye, Logan Engstrom, Andrew Ilyas, Kevin Kwok (2018) Synthesizing robust adversarial examples. In: International conference on machine learning, PMLR, pp 284–293

Anagha Athavale, Ezio Bartocci, Maria Christakis, Matteo Maffei, Dejan Nickovic, Georg Weissenbacher (2024) Verifying global two-safety properties in neural networks with confidence. In: International Conference on Computer Aided Verification, Springer, pp 329–351

Nicholas Carlini, David Wagner (2017) Adversarial examples are not easily detected: Bypassing ten detection methods. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Association for Computing Machinery, New York, NY, USA, AISec '17, p 3–14, DOI 10.1145/3128572.3140444, URL https://doi.org/10.1145/3128572.3140444

Ian J Goodfellow, Jonathon Shlens, Christian Szegedy (2015) Explaining and harnessing adversarial examples. In: Yoshua Bengio, Yann LeCun (eds) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, URL http://arxiv.org/abs/1412.6572

Patrick Indri, Peter Blohm, Anagha Athavale, Ezio Bartocci, Georg Weissenbacher, Matteo Maffei, Dejan Nickovic, Thomas Gärtner, Sagar Malhotra (2024) Distillation based robustness verification with pac guarantees. In: ICML 2024 Next Generation of AI Safety Workshop

Klas Leino, Zifan Wang, Matt Fredrikson (2021) Globally-robust neural networks. In: Marina Meila, Tong Zhang (eds) Proceedings of the 38th International Conference on Machine Learning, PMLR, Proceedings of Machine Learning Research, vol 139, pp 6212–6222, URL https://proceedings.mlr.press/v139/leino21a.html

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, Adrian Vladu (2018) Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations, URL https://openreview.net/forum?id=rJzIBfZAb

Michael Mitzenmacher, Eli Upfal (2017) Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis. Cambridge university press

Rulin Shao, Jinfeng Yi, Pin-Yu Chen, Cho-Jui Hsieh (2021) How and when adversarial robustness transfers in knowledge distillation? URL https://arxiv.org/abs/2110.12072, 2110.12072

V N Vapnik, A Ya Chervonenkis (2015) On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities, Springer International Publishing, Cham, pp 11–30, DOI 10.1007/978-3-319-21852-6_3