# Robustness -1:

## Theory and rethinking

Yunzhen Feng, Haocheng Ju, Zehao Wang, Haotong Yang, Pu Yang

School of Mathematical Science, Peking University

# The Landscape of Robust Models

# The Landscape of Robust Models

## Adversarial Robustness through Local Linearization

# Table of contents

1. The Landscape of Robust Models

    Adversarial Robustness through Local Linearization

2. Certified Robustness

3. The Generalization of Robust Model

4. Rethinking Different Robustness

# Gradient Obfuscation and Robustness [1]

- Landscape for failed defense and clean training:
- Highly non-linear in the vicinity.
- Adversarial training on stronger attack make surface more linear.
- How to train with weak adversary and prevent gradient obfuscation?

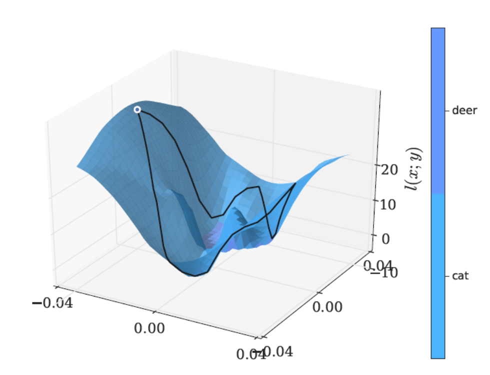



**Figure 1:** The landscape surface

[1] Qin, Chongli, et al. "Adversarial robustness through local linearization." Advances in Neural Information Processing Systems. 2019.

# Local Linearity and Robustness

- Local Linearity Measure:

$$\gamma(\epsilon, x) = \max_{\delta \in B(\epsilon)} \left| \ell(x+\delta) - \ell(x) - \delta^T \nabla_x \ell(x) \right|.$$

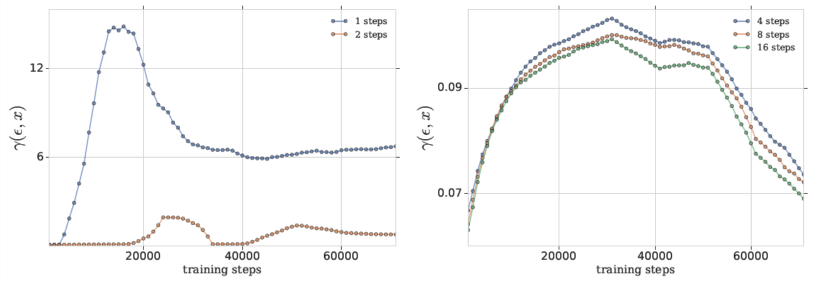



**Figure 2:** The change of $\gamma$ w.r.t. different training settings.

# Induced 'Local Linearity Regularizer'

- By definition of $\gamma(\epsilon, x)$:

$$\ell(x+\delta) \leq \ell(x) + \left|\delta^T \nabla_x \ell(x)\right| + \gamma(\epsilon, x).$$

- For small $\delta$, the middle part is small
- Specifically, it can be controlled by $\gamma(\epsilon, x)$ for quadratic loss or cross-entropy.
- Objective function:

$$L(\mathcal{D}) = \mathbb{E}_{\mathcal{D}}[\ell(x) + \underbrace{\lambda\gamma(\epsilon, \boldsymbol{x}) + \mu\left|\delta_{LLR}^T \nabla_x \ell(x)\right|}_{\text{LLR}}],$$

where $\delta_{LLR} = \text{argmax}_{\delta \in B(\epsilon)} \left|\ell(x+\delta) - \ell(x) - \delta^T \nabla_x \ell(x)\right|$ and $\gamma(\epsilon, x) = \left|\ell\left(x + \delta_{LLR}\right) - \ell(x) - \delta_{LLR}^T \nabla_x \ell(x)\right|.$

# Algorithm

**Algorithm 1** Local Linearization of Network

**Require:** Training data X = $\{(x_1, y_1), \cdots, (x_N, y_N)\}$. Learning rate $lr$ and batch size for training $b$ and number of iterations $N$. Number of iterations for inner optimization $M$ and step size $s$ and network architecture parameterized by $\theta$.

```
Initialize variables θ.
for all i ∈ {0, 1, ..., N} do
    Get mini-batch B = {(x_{i_1}, y_{i_1}), ···, (x_{i_b}, y_{i_b})}.
    Calculte loss wrt to minibatch L_B = (1/b) Σ_{j=1}^{b} ℓ(x_{i_j}; y_{i_j}).
    Initialize initial perturbation δ uniformly in the interval [−ε, ε].
    for all j ∈ {0, 1, ..., M} do
        Calculate g = (1/b) Σ_{t=1}^{b} ∇_δ g(δ; x_{i_t}, y_{i_t}) at δ.
        Update δ ← Proj(δ − s × Optimizer(g))
    end for
    Compute objective L = L_B + 1/b Σ_{j=1}^{b} (λ g(δ; x_{i_j}, y_{i_j}) + μ |δ^T ∇_x l(x)|)
    θ ← θ − lr × Optimizer(∇_θ L)
end for
```

Note $g(\delta; x, y) = \ell(x + \delta; y) - \ell(x; y) - \delta^T \nabla_x \ell(x; y)$.

# Results

| Methods | Nominal | FGSM-20 | Untargeted | Multi-Targeted |
|---|---|---|---|---|
| | **CIFAR-10: Wide-ResNet-28-8 (8/255)** | | | |
| Attack Strength | | Weak | Strong | Very Strong |
| ADV[16] | 87.25% | 48.89% | 45.92% | 44.54% |
| CURE[19] | 80.76% | 39.76% | 38.87% | 37.57% |
| ADV(S) | 85.11% | **56.76%** | **53.96%** | 48.79% |
| CURE(S) | 84.31% | 48.56% | 47.28% | 45.43% |
| TRADES(S) | **87.40%** | 51.63 | 50.46% | 49.48% |
| LLR (S) | 86.83% | 54.24% | 52.99% | **51.13%** |
| | **CIFAR-10: Wide-ResNet-40-8 (8/255)** | | | |
| ADV(R) | 85.58% | 56.32% | 52.34% | 46.89% |
| TRADES(R) | 86.25% | 53.38% | 51.76% | 50.84% |
| ADV(S) | 85.27% | **57.94%** | **55.26%** | 49.79% |
| CURE(S) | 84.45% | 49.41% | 47.69% | 45.51% |
| TRADES(S) | **88.11%** | 53.03% | 51.65% | 50.53% |
| LLR (S) | 86.28% | 56.44% | 54.95% | **52.81%** |

**Figure 3:** Results on Cifar-10

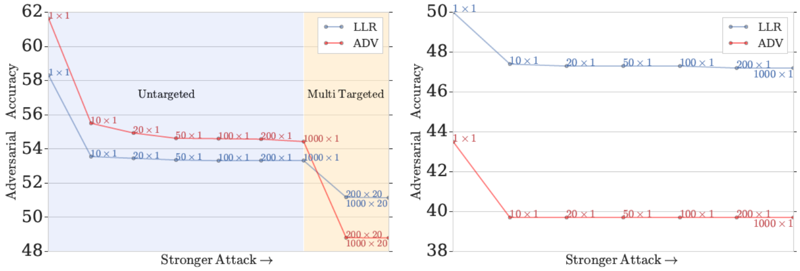
# Results

| | ImageNet: ResNet-152 (4/255) | | | |
|---|---|---|---|---|
| Methods | PGD steps | Nominal | Untargeted | Random-Targeted |
| | | Accuracy | | Success Rate |
| ADV | 30 | 69.20% | 39.70% | 0.50% |
| DENOISE | 30 | 69.70% | 38.90% | **0.40%** |
| LLR | **2** | **72.70%** | **47.00%** | **0.40%** |
| | ImageNet: ResNet-152 (16/255) | | | |
| ADV [28] | 30 | 64.10% | 6.30% | 40.00% |
| DENOISE [28] | 30 | **66.80%** | **7.50%** | **38.00%** |
| LLR | **10** | 51.20% | 6.10% | 43.80% |

**Figure 4:** Results on ImageNet

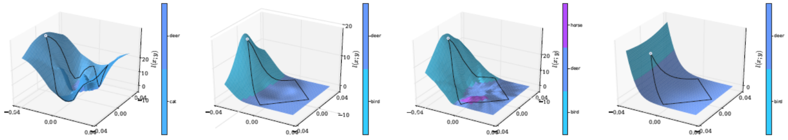
- Trained on128 TPUv3 cores for radius 4/255.
- ADV networks takes 36 hours for 110 epochs
- LLR networks takes 7 hours for 110 epochs

# Results




(a) CIFAR-10 (8/255) (b) ImageNet (4/255)

**Figure 5:** Robustness Drop is smaller




(a) ADV-1 (b) LLR-1 (c) ADV-2 (d) LLR-2

# Certified Robustness

# Certified Robustness

## Adversarial Training and Provable Defenses: Bridging the Gap

# Table of contents

1. The Landscape of Robust Models

2. Certified Robustness

   Adversarial Training and Provable Defenses: Bridging the Gap

3. The Generalization of Robust Model

4. Rethinking Different Robustness

# The most secure defense: certified robustness [5]

- Attacks vs Defense: ongoing war
- Can we certify the vicinity of the data?
- Linear Relaxation [2]
- Interval Bound Propagation [3]
- Randomized Smoothing (certified with probability) [4]

---

[2] Zhang, Huan, et al. "Efficient Neural Network Robustness Certification with General Activation Functions." Advances in Neural Information Processing Systems (2018): 4939-4948.

[3] Gowal, Sven, et al. "On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models.." arXiv: Learning (2018).

[4] Cohen, Jeremy, Elan Rosenfeld, and J. Zico Kolter. "Certified Adversarial Robustness via Randomized Smoothing." International Conference on Machine Learning (2019): 1310-1320.

[5] Balunovic, Mislav, and Martin Vechev. "Adversarial training and provable defenses: Bridging the gap." International Conference on Learning Representations. 2020.

# Certified with Convex Relaxation

- Neural networks in layerwise:

$$h_\theta = h_\theta^k \circ h_\theta^{k-1} \cdots \circ h_\theta^1 \text{ and } h_\theta^i : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i}.$$

- Robustness (even for other properties):

$$c^T h_\theta(\boldsymbol{x}') + d < 0, \forall \boldsymbol{x}' \in \mathbb{S}_0(\boldsymbol{x}),$$

  where $\mathbb{S}_0(\boldsymbol{x}) = \left\{\boldsymbol{x}' \in \mathbb{R}^{d_0}, \|\boldsymbol{x} - \boldsymbol{x}'\|_\infty < \epsilon\right\}$, $c, d$ are specific vectors.
- Perturbation set: $\mathbb{S}_i(\boldsymbol{x}) = h_\theta^i(\mathbb{S}_{i-1}(\boldsymbol{x}))$.
- Certification via convex relaxation:
- Find convex set $\mathbb{C}_i(\boldsymbol{x})$: $\mathbb{S}_i(\boldsymbol{x}) \subseteq \mathbb{C}_i(\boldsymbol{x}), \forall i \in [k]$.
- Directly prove the robustness property for $\mathbb{C}_k(\boldsymbol{x})$.

# Specifically

- $\mathbb{S}_0(\boldsymbol{x}) = \mathbb{C}_0(\boldsymbol{x})$
- All convex set can be represented as

$$\mathbb{C}_l(\boldsymbol{x}) = \{\boldsymbol{a}_l + \boldsymbol{A}_l e | e \in [-1, 1]^{m_l}\},$$

  where $\boldsymbol{a}_l$ is the center.
- For $\mathbb{C}_0(\boldsymbol{x})$, $\boldsymbol{a}_0 = \boldsymbol{x}$ and $\boldsymbol{A}_0 = \epsilon \boldsymbol{I}_{d_0}$.
- How to propagate small enough convex set $\mathbb{C}_l(\boldsymbol{x})$?
- For a point in convex set $\boldsymbol{x}'_i = \boldsymbol{a}_i + \boldsymbol{A}_i \boldsymbol{e}$
- Convolutional and fully connected layer: Already Linear:

$$h_\theta^{i+1}(\boldsymbol{x}'_i) = \boldsymbol{W}_{i+1}\boldsymbol{a}_i + \boldsymbol{b}_{i+1} + \boldsymbol{W}_{i+1}\boldsymbol{A}_i\boldsymbol{e}.$$

- For ReLU activation: upper bound and lower bound

$$l_{i,j} = a_{i,j} - \sum_{k=1}^{m_i} |A_{i,j,k}|, \quad u_{i,j} = a_{i,j} + \sum_{k=1}^{m_i} |A_{i,j,k}|.$$

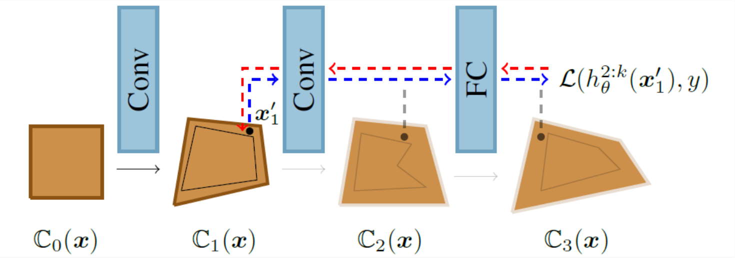
# How to Train the Model?

- Like randomized smoothing, certification only provides a way to verify.
- How to train a model that has good certified robustness?
- Certified Robustness vs Adversarial Training

  $$\min_{\theta^{l+1:k}} \mathbb{E}_{(\mathrm{x},\mathrm{y})\sim D} \max_{\boldsymbol{x}'_l \in \mathbb{C}_l(\boldsymbol{x})} \mathcal{L}\left(h_\theta^{l+1:k}\left(\boldsymbol{x}'_l\right), y, \theta\right),$$

  Certified Robustness find upper bound for the inner maximum while adversarial training find large enough lower bound.
- When the certification tends to fail?
- There exists $\boldsymbol{x}'_1 \in \mathbb{C}_1(\boldsymbol{x}) \backslash \mathbb{S}_1(\boldsymbol{x})$ that is an adversarial example.
- Add these adversarial examples into the training as adv train!

# Hierarchical Adversarial Training




**Algorithm 1:** Convex layerwise adversarial training via convex relaxations

**Data:** $k$-layer network $h_\theta$, training set $(\mathcal{X}, \mathcal{Y})$, learning rate $\eta$, step size $\alpha$, inner steps $n$
**Result:** Certifiably robust neural network $h_\theta$

**for** $l \leq k$ **do**
  **for** $i \leq n_{epochs}$ **do**
    Sample mini-batch $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_b, y_b)\} \sim (\mathcal{X}, \mathcal{Y})$;
    Compute convex relaxations $\mathbb{C}_l(\boldsymbol{x}_1), \mathbb{C}_l(\boldsymbol{x}_2), ..., \mathbb{C}_l(\boldsymbol{x}_b)$;
    Initialize $\boldsymbol{x}'_1 \sim \mathbb{C}_l(\boldsymbol{x}_1), \boldsymbol{x}'_2 \sim \mathbb{C}_l(\boldsymbol{x}_2), ..., \boldsymbol{x}'_b \sim \mathbb{C}_l(\boldsymbol{x}_b)$;
    **for** $j \leq b$ **do**
      Update in parallel $n$ times: $\boldsymbol{x}'_j \leftarrow \Pi_{\mathbb{C}_l(\boldsymbol{x}_j)}(\boldsymbol{x}'_j + \alpha \nabla_{\boldsymbol{x}'_j} \mathcal{L}(h_\theta^{l+1:k}(\boldsymbol{x}'_j), y_j))$;
    **end**
    Update parameters $\theta \leftarrow \theta - \eta \cdot \frac{1}{b} \sum_{j=1}^{b} \nabla_\theta \mathcal{L}(h_\theta^{l+1:k}(\boldsymbol{x}'_j), y_j)$;
  **end**
  Freeze parameters $\theta_{l+1}$ of layer function $h_\theta^{l+1}$;
**end**

# Results

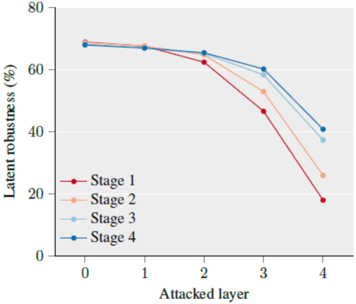
Table 1: Evaluation on CIFAR-10 dataset with $L_\infty$ perturbation 2/255

| Method | Accuracy(%) | Certified Robustness(%) |
|---|---|---|
| Our work | **78.4** | **60.5** |
| Zhang et al. (2020) | 71.5 | 54.0 |
| Wong et al. (2018) | 68.3 | 53.9 |
| Gowal et al. (2018) | 70.2 | 50.0 |
| Xiao et al. (2019) | 61.1 | 45.9 |
| Mirman et al. (2019) | 62.3 | 45.5 |

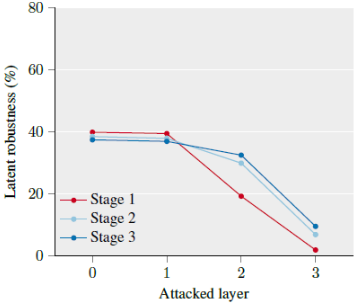
Table 2: Evaluation on CIFAR-10 dataset with $L_\infty$ perturbation 8/255

| Method | Accuracy(%) | Certified Robustness(%) |
|---|---|---|
| Our work | 51.7 | 27.5 |
| Zhang et al. (2020) | **54.5** | **30.5** |
| Mirman et al. (2019) | 46.2 | 27.2 |
| Wong et al. (2018) | 28.7 | 21.8 |
| Xiao et al. (2019) | 40.5 | 20.3 |

# Results




(a) CIFAR-10, 2/255




(b) CIFAR-10, 8/255

# The Generalization of Robust Model

# The Generalization of Robust Model

## Rademacher Complexity for Adversarially Robust Generalization[a]

[a]Yin, Dong, Ramchandran Kannan, and Peter Bartlett. "Rademacher Complexity for Adversarially Robust Generalization." International Conference on Machine Learning. 2019.

# Table of contents


1. The Landscape of Robust Models

2. Certified Robustness

3. The Generalization of Robust Model

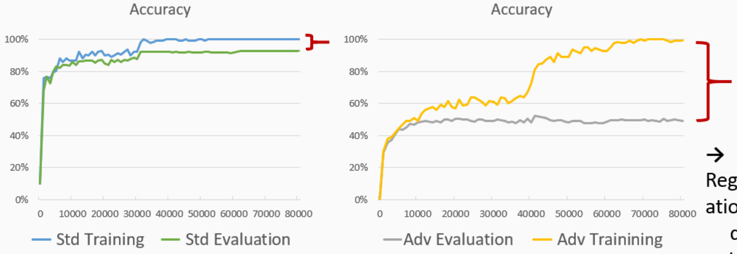
   Rademacher Complexity for Adversarially Robust Generalization[6]

4. Rethinking Different Robustness

# Adversarially robust generalization

- Two different generalization:

$$R(f) := \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}[\ell(f(\mathbf{x}),y)], \qquad \widetilde{R}(f) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\max_{\mathbf{x}'\in\mathbb{B}^{\mathbf{a}}_{\mathbf{x}}(\epsilon)} \ell\left(f\left(\mathbf{x}'\right),y\right)\right].$$




→ Regularization does not help

(small) generalization gap

(large) generalization gap

**Figure 7:** Two generalization gap for standard training and adversarial training

# Generalization theory in two situations

- Empirical Risk

$$R_n(f) := \frac{1}{n}\sum_{i=1}^{n} \ell\left(f\left(\mathbf{x}_i\right), y_i\right), \quad \widetilde{R}_n(f) := \frac{1}{n}\sum_{i=1}^{n} \max_{\mathbf{x}_i' \in \mathbb{B}_{\mathbf{x}_i}^{\infty}(\epsilon)} \ell\left(f\left(\mathbf{x}_i'\right), y_i\right)$$

- Population Risk

$$R(f) := \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}[\ell(f(\mathbf{x}), y)], \quad \widetilde{R}(f) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\mathbf{a}}(\epsilon)} \ell\left(f\left(\mathbf{x}'\right), y\right)\right].$$

- Rademacher Complexity

$$\mathfrak{R}_{\mathcal{D}}(\ell_{\mathcal{F}}) := \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\boldsymbol{h}\in l_{\mathcal{F}}} \sum_{i=1}^{n} \sigma_i h\left(\mathbf{z}_i\right)\right], \qquad \mathfrak{R}_{\mathcal{D}}(\widetilde{\ell}_{\mathcal{F}})$$

- Generalization Bound

$$R(f) \leq R_n(f) + 2B\mathfrak{R}_{\mathcal{D}}\left(\ell_{\mathcal{F}}\right) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}$$

$$\widetilde{R}(f) \leq \widetilde{R}_n(f) + 2B\mathfrak{R}_{\mathcal{D}}\left(\widetilde{\ell}_{\mathcal{F}}\right) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}$$

# Linear Classifier for Binary Classification

- $\mathcal{Y} = \{-1, +1\}$, $\mathcal{F} = \{f_{\mathbf{w}}(\mathbf{x}) : \|\mathbf{w}\|_p \leq W\}$
- Adversarial Function Class

$$\tilde{\mathcal{F}} = \left\{ \min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}^0}(\epsilon)} y \langle \mathbf{w}, \mathbf{x}' \rangle : \|\mathbf{w}\|_p \leq W \right\}.$$

- Theorem: Suppose $\frac{1}{p} + \frac{1}{q} = 1$, there exists a universal constant $c \in (0, 1)$ such that

$$\max \left\{ \mathfrak{R}_{\mathcal{S}}(\mathcal{F}), c\epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}} \right\} \leq \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \leq \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

- Specifically, unavoidable dimension dependence:

$$\frac{c}{2} \left( \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}} \right) \leq \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \leq \mathfrak{P}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

## Multi-class Task

- Ramp loss: $\ell(f(\mathbf{x}), y) = \phi_\gamma(f(x)_y - \max_{y' \neq y} f(x)_{y'})$

$$\phi_\gamma(t) = \begin{cases} 1 & t \leq 0 \\ 1 - \frac{t}{\gamma} & 0 < t < \gamma \\ 0 & t \geq \gamma \end{cases}.$$

- If satisfy

$$\mathbf{1}\left(y \neq \arg\max_{y' \in [K]} [f(\mathrm{x})]_{y'}\right) \leq \ell(f(\mathrm{x}), y) \leq \mathbf{1}\left([f(\mathrm{x})]_y \leq \gamma + \max_{y' \neq y} [f(\mathrm{x})]_{y'}\right)$$

.

- Clean generalization with margin $\gamma$:

$$\begin{aligned} &\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \neq \arg\max_{y' \in [K]} [f(\mathbf{x})]_{y'}\right\} \\ &\leq \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\left(\left[f\left(\mathbf{x}_i\right)\right]_{y_i} \leq \gamma + \max_{y' \neq y}\left[f\left(\mathbf{x}_i\right)\right]_{y'}\right) \\ &+ 2\Re_{\mathcal{S}}\left(\ell_{\mathcal{F}}\right) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}. \end{aligned}$$

# Linear Classifier for Multi-class

- Linear classifier : $\mathbf{W} \in \mathbb{R}^{K \times d}$, i.e., $f(\mathbf{x}) \equiv f_{\mathbf{W}}(\mathbf{x}) = \mathbf{W}\mathbf{x}$, $\mathcal{F} = \left\{ f_{\mathbf{W}}(\mathbf{x}) : \left\| \mathbf{W}^\top \right\|_{p,\infty} \leq W \right\}$.
- Standard Rademacher:

$$\Re_{\mathcal{S}}\left(\ell_{\mathcal{F}}\right) = \frac{2KW}{\gamma n} \mathbb{E}_{\boldsymbol{\sigma}} \left[ \left\| \sum_{i=1}^{n} \sigma_i \mathbf{x}_i \right\|_q \right].$$

- Robust Rademacher:

$$\Re_{\mathcal{S}}\left(\widetilde{\ell}_{\mathcal{F}}\right) \leq \frac{WK}{\gamma} \left[ \frac{\epsilon \sqrt{K} d^{\frac{1}{q}}}{\sqrt{n}} + \frac{1}{n} \sum_{y=1}^{K} \mathbb{E}_{\boldsymbol{\sigma}} \left[ \left\| \sum_{i=1}^{n} \sigma_i \mathbf{x}_i \mathbf{1}\left(y_i = y\right) \right\|_q \right] \right]$$

# Binary Deep Neural Networks

- Deep Neural Networks: $f_{\mathbf{W}}(\mathbf{x}) = \mathbf{W}_L \rho\left(\mathbf{W}_{L-1} \rho\left(\cdots \rho\left(\mathbf{W}_1 \mathbf{x}\right) \cdots\right)\right)$, where $\rho(x)$ is the ReLU activation function.
- Standard Rademacher for function class $\mathcal{F} = \Big\{ f_{\mathbf{W}}(\mathbf{x}) : \mathbf{W} = \left(\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L\right), \|\mathbf{W}_h\|_\sigma \leq s_h, \left\|\mathbf{W}_h^\top\right\|_{2,1} \leq b_h, h \in [L] \Big\} \subseteq \mathbb{R}^{\mathcal{X}}$,

$$\mathfrak{P}_{\mathcal{S}}(\mathcal{F}) \leq \frac{4}{n^{3/2}} + \frac{26 \log(n) \log\left(2 d_{\max}\right)}{n} \|\mathbf{X}\|_F \left(\prod_{h=1}^L s_h\right) \left(\sum_{j=1}^L \left(\frac{b_j}{s_j}\right)^{2/3}\right)^{3/2}.$$

- Robust Rademacher $\widetilde{\mathcal{F}} = \Big\{ (\mathrm{x}, y) \mapsto \min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} y f_{\mathbf{W}}\left(\mathrm{x}'\right) : \mathbf{W} = \left(\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L\right), \prod_{h=1}^L \|\mathbf{W}_h\|_\sigma \leq r \Big\} \subseteq \mathbb{R}^{\mathcal{X} \times \{-1,+1\}}$,

$$\mathfrak{P}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \geq \mathrm{c}r \left(\frac{1}{n} \|\mathbf{X}\|_F + \epsilon \sqrt{\frac{d}{n}}\right).$$

- For two layer neural networks, the problem could be solved using a surrogate loss for function class with constraints on $\ell_1$ norm.

# Experiments

$$\min_{\mathbf{W}} \frac{1}{n} \sum_{i=1}^{n} \max_{\mathbf{x}_i' \in \mathbb{B}_{\mathbf{x}_i}^{\mathbf{a}}(\epsilon)} \ell\left(f_{\mathbf{W}}\left(\mathbf{x}_i'\right), y_i\right) + \lambda \|\mathbf{W}\|_1 .$$

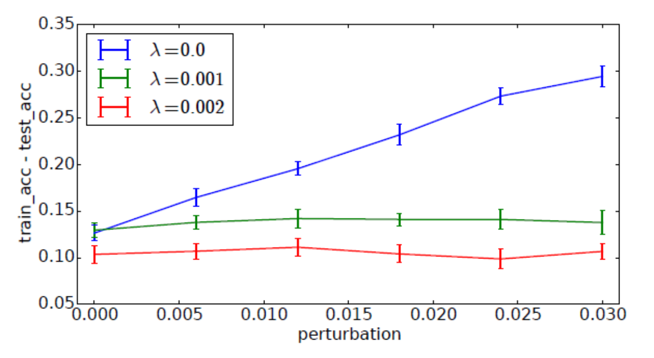



**Figure 8:** Linear Classifier for MNIST.

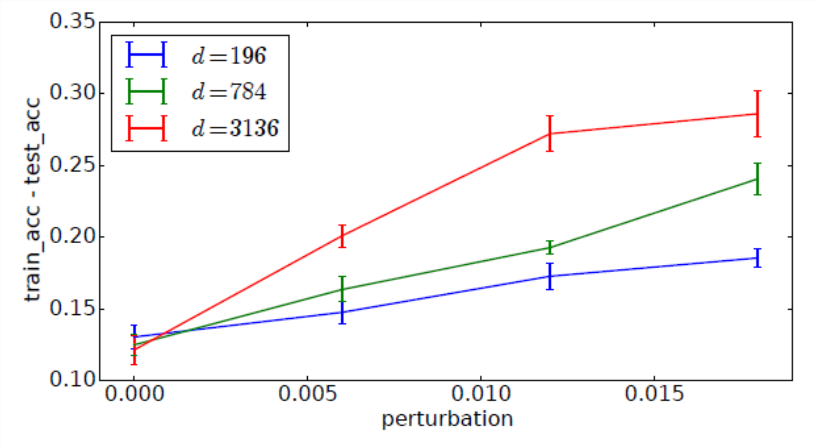



**Figure 9:** Linear Classifier for MNIST: the impact of input dimension on generalization gap

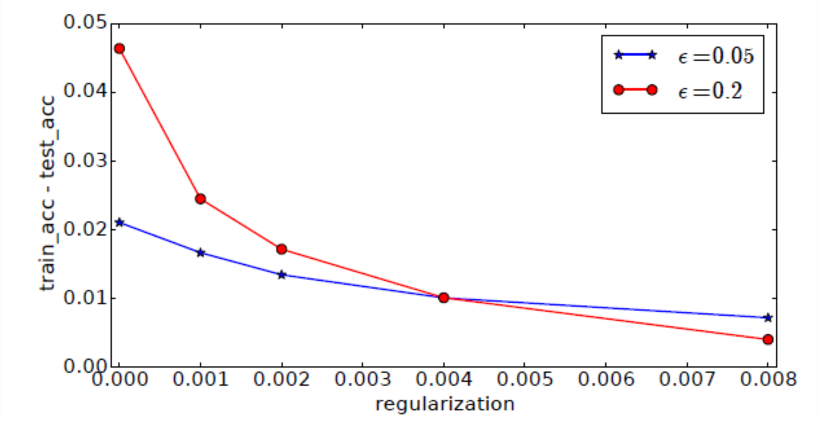



**Figure 10:** Four layer ReLU networks for MNIST: two convolutional and two fully connected layers.

# Rethinking Different Robustness

# Rethinking Different Robustness

## Exploring the Landscape of Spatial Robustness

# Table of contents

1. The Landscape of Robust Models

2. Certified Robustness

3. The Generalization of Robust Model

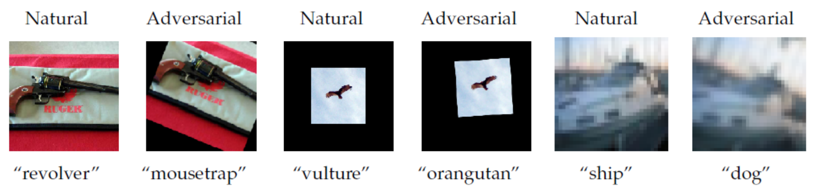
4. Rethinking Different Robustness

    Exploring the Landscape of Spatial Robustness

# Spatial Robustness [7]

- Spatial Robustness: rotation, translation
- The neural networks are not robust to spatial attacks
- Attack formation: move a pixel at position $(u, v)$ to

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix}$$




[7] Engstrom, Logan, et al. "Exploring the Landscape of Spatial Robustness." International Conference on Machine Learning. 2019.

# Attack Methods and Defenses

Attacks:

- First Order Methods: the arguments are differentiable
- Grid Search: low dimensional
- Worst of k randomly samples

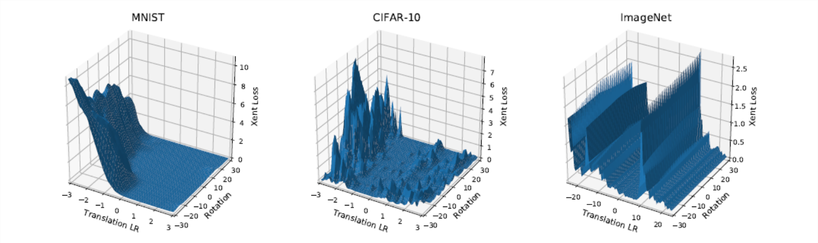
Defenses:

- Standard Training
- $\ell_\infty$-bounded adversarial training
- No random cropping
- Random rotations and translations ($30^o$, 10%pixels)
- Random rotations and translations for larger intervals ($40^o$, 13%pixels)

# Different attacks

| | MNIST | | CIFAR-10 | | ImageNet | |
|---|---|---|---|---|---|---|
| | Standard | Aug. | Standard | Aug. | Standard | Aug. |
| Natural | 99.31% | 99.53% | 92.62% | 90.02% | 75.96% | 65.96% |
| Worst-of-10 | 73.32% | 98.33% | 20.13% | 79.92% | 47.83% | 50.62% |
| First-Order | 79.84% | 98.78% | 62.69% | 85.92% | 63.12% | 66.05% |
| Grid | **26.02%** | **95.79%** | **2.80%** | **58.92%** | **31.42%** | **32.90%** |

**Figure 11:** Standard vs same level of augmentations



# Results

| | Model | Nat. | Rand. | Grid | Rand. T. | Grid T. | Rand. R. | Grid R. |
|---|---|---|---|---|---|---|---|---|
| MNIST | Standard | 99.31% | 94.23% | **26.02%** | 98.61% | 89.80% | 95.68% | 70.98% |
| | $\ell_\infty$-Adv | 98.65% | 88.02% | **1.20%** | 93.72% | 34.13% | 95.27% | 72.03% |
| | Aug. 30 | 99.53% | 99.35% | **95.79%** | 99.47% | 98.66% | 99.34% | 98.23% |
| | Aug. 40 | 99.34% | 99.31% | **96.95%** | 99.39% | 98.65% | 99.40% | 98.49% |
| | W-10 (30) | 99.48% | 99.37% | **97.32%** | 99.50% | 99.01% | 99.39% | 98.62% |
| | W-10 (40) | 99.42% | 99.39% | **97.88%** | 99.45% | 98.89% | 99.36% | 98.85% |
| CIFAR10 | Standard | 92.62% | 60.93% | **2.80%** | 88.54% | 66.17% | 75.36% | 24.71% |
| | No Crop | 90.34% | 54.64% | **1.86%** | 81.95% | 46.07% | 69.23% | 18.34% |
| | $\ell_\infty$-Adv | 80.21% | 58.33% | **6.02%** | 78.15% | 59.02% | 62.85% | 20.98% |
| | Aug. 30 | 90.02% | 90.92% | **58.90%** | 91.76% | 79.01% | 91.14% | 76.33% |
| | Aug. 40 | 88.83% | 91.18% | **61.69%** | 91.53% | 77.42% | 91.10% | 76.80% |
| | W-10 (30) | 91.34% | 92.35% | **69.17%** | 92.43% | 83.01% | 92.33% | 81.82% |
| | W-10 (40) | 91.00% | 92.11% | **71.15%** | 92.28% | 82.15% | 92.53% | 82.25% |
| ImageNet | Standard | 75.96% | 63.39% | **31.42%** | 73.24% | 60.42% | 67.90% | 44.98% |
| | No Crop | 70.81% | 59.09% | **16.52%** | 66.75% | 45.17% | 62.78% | 34.17% |
| | Aug. 30 | 65.96% | 68.60% | **32.90%** | 70.27% | 45.72% | 69.28% | 47.25% |
| | Aug. 40 | 66.19% | 67.58% | **33.86%** | 69.50% | 44.60% | 68.88% | 48.72% |
| | W-10 (30) | 76.14% | 73.19% | **52.76%** | 74.42% | 61.18% | 73.74% | 61.06% |
| | W-10 (40) | 74.64% | 71.36% | **50.23%** | 72.86% | 59.34% | 71.95% | 59.23% |

# Results

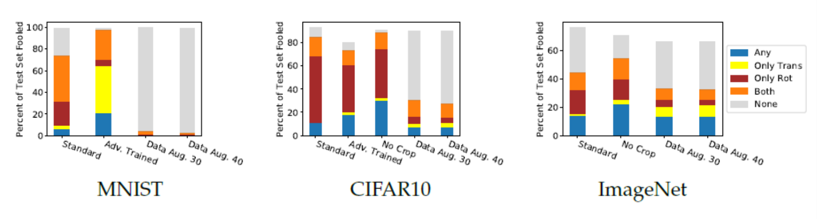



**Figure 12:** Fine-grained analysis

# Future

- How to empirically defense? Different ways to obfuscate gradients?
- Certified Robustness: requires good training methods, take long to certify.
- What robustness is reliable and how to achieve is not clear.
- Need more understandings and theories
- Computer Vision vs Natural Language Processing