ML Robustness

Krishnkant Swarnkar, Mike Andrews, Koushik Reddy

COMPSCI 692S
21 Oct 2020

University of Massachusetts Amherst BE REVOLUTIONARY™

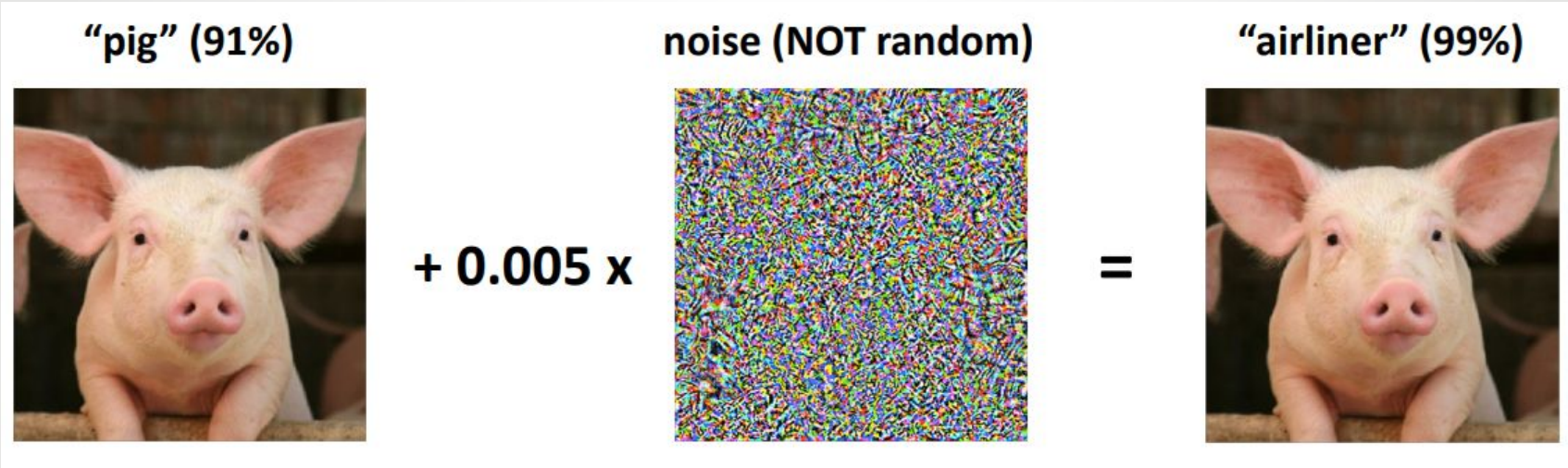
Adversarial Training and Provable Defenses: Bridging the Gap

Mislav Balunovic, Martin Vechev
[ICLR 2020]

Presenter: Krishnkant Swarnkar

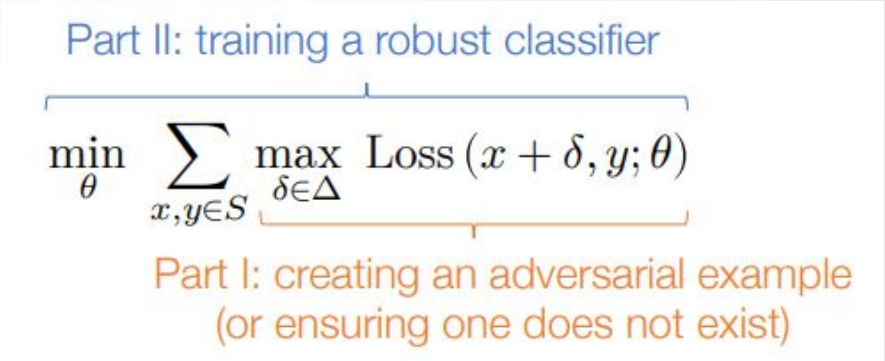
Adversarial Examples

ML Predictions are mostly accurate but brittle



Adversarial Robustness !

The Big Picture:



Part I:

- **Local Search (standard adv. attacks),**
- **Combinatorial Optimization (exactly solve the objective),**
- **Convex Relaxation (upper Bound)**

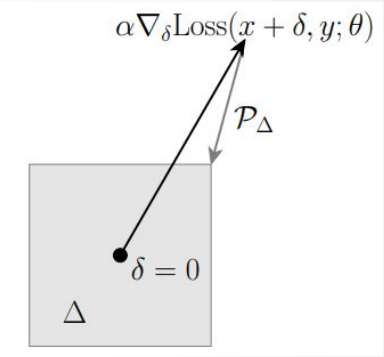
Part II:

- **Standard DL optimization on the loss from part I**

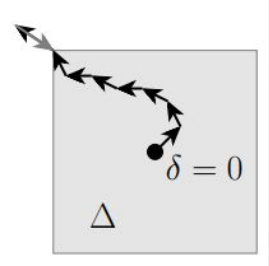
Popular Adversarial Attacks (Part I) !

Various ways to attack ML models:

1. **FGSM (Fast Gradient Sign Method)**



2. **PGD (Projected Gradient Descent)**



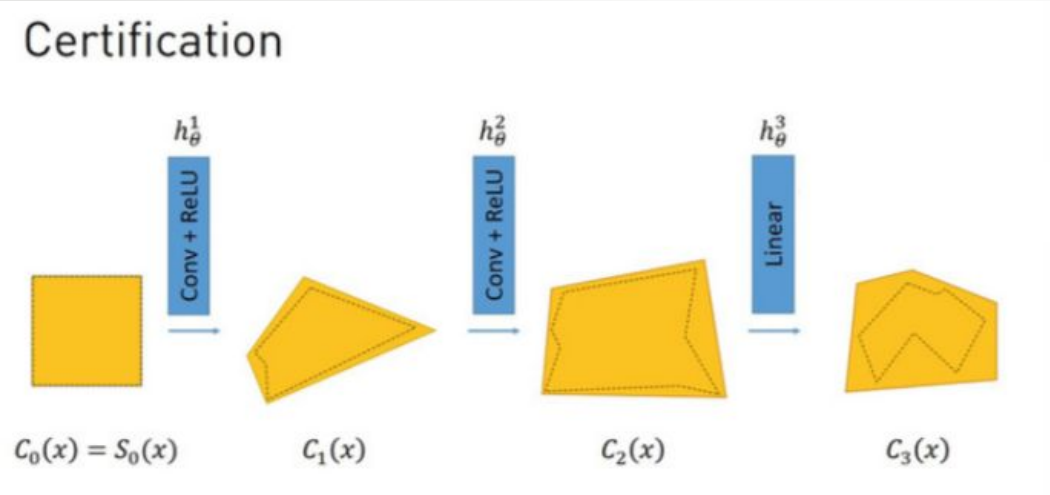
Adversarial Defense (so far...)

1. **Adversarial Training:**
   - When Part-I is “Local Search” (an approximation)
   - Empirically robust models (but no guarantees)

2. **Provably Robust Training (Adversarial Certification):**
   - When Part-I is “Convex Relaxation”
   - Guaranteed robustness (but poor standard accuracy)
   - Two major disadvantages of current methods:
     - Loose upper bound.
     - Relationship between standard loss and parameters more complex

3. **Randomized Classifiers**
   - Training ML models with Probabilistic Guarantees.
   - Bounds come from the relationship between L2 robustness and Gaussian distribution.

Certification



Adversarial Certification if:

$$c^T x_3^{'} + d < 0 \; \forall x \in S_0(x)$$

$$S_0(x) = \{x^{'} \in R^{d_0}, ||x^{'} - x||_\infty < \epsilon\}$$

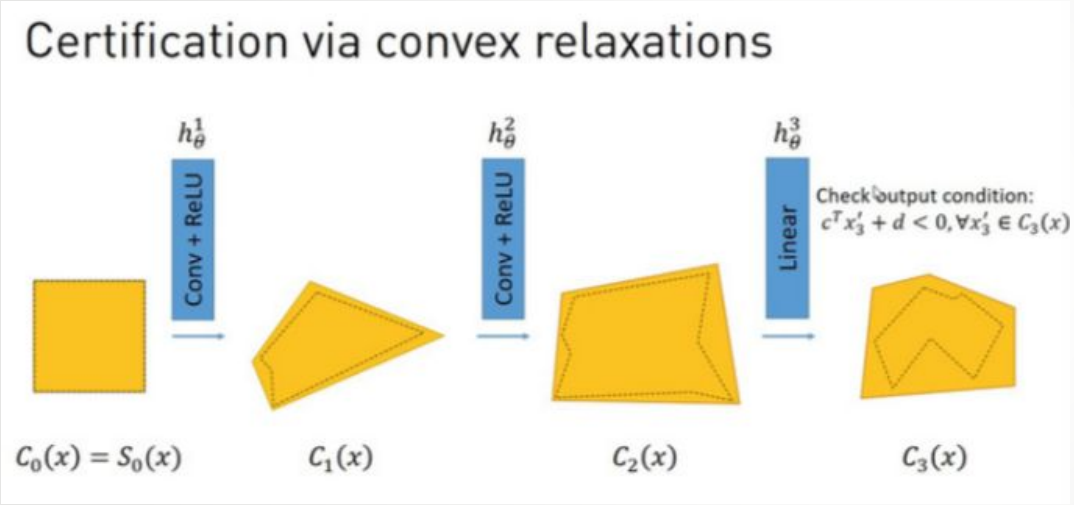
In simpler terms, if:

$$x_{3,target}^{'} - x_{3,orig}^{'} > 0$$

then no adversary possible.

Otherwise, may or may not have an adversary.

Certification via Convex Relaxation



Adversarial Certification via Convex Relaxation, if:

$$c^T x_3' + d < 0 \; \forall x_3' \in C_3(x)$$

Adversarial Defense

So, Far...

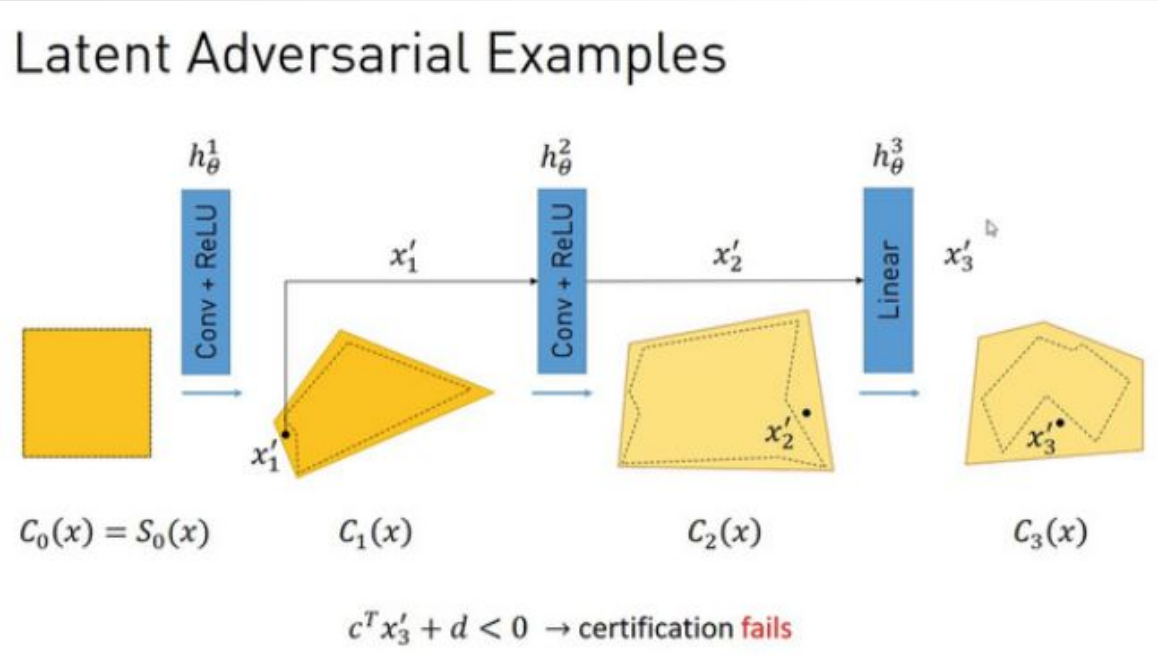
1. **Adversarial Training:**
   - Empirically robust models (but no guarantees)
2. **Provably Robust Training (Adversarial Certification):**
   - Guaranteed robustness (but poor standard accuracy)

This work:

Combines adversarial and Provable Defenses [COLT]

- Best of both worlds
- Layerwise training
- Training on Latent Adversarial Examples to reduce the precision error between polytope generated by relaxations.

Latent Adversarial Examples



- Points in Intermediate layers
- Certified as per convex relaxation
- But not certified actually as per convex bounds on the input.

Layerwise Provable Optimization via Convex Relaxations

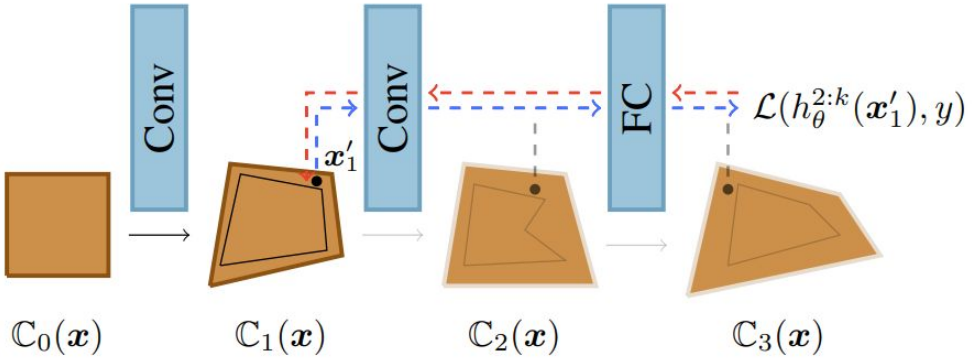


Figure 1: An iteration of convex layerwise adversarial training. Latent adversarial example $\boldsymbol{x}_1'$ is found in the convex region $\mathbb{C}_1(\boldsymbol{x})$ and propagated through the rest of the layers in a forward pass, shown with the blue line. During backward pass, gradients are propagated through the same layers, shown with the red line. Note that the first convolutional layer does not receive any gradients.

Layerwise Provable Optimization via Convex Relaxations

Algorithm 1: Convex layerwise adversarial training via convex relaxations

Data: k-layer network h_θ, training set $(\mathcal{X}, \mathcal{Y})$, learning rate η, step size α, inner steps n
Result: Certifiably robust neural network h_θ

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 θ_{l+1} of layer function h_θ^{l+1};
end

Convex Layerwise Adversarial Training Using Linear Relaxations

Important TRICKS and speedups !!

1. **Projections to Linear Convex Regions**
   - Change of Variables (Zonotope Abstraction)
   - Instead of projecting x on C(x), Project e on the hyperrectangle $\boldsymbol{e} \in [-1, 1]^{m_l}$

$$\boldsymbol{e}_j \leftarrow clip(\boldsymbol{e}_j + \alpha \boldsymbol{A}_l^T \nabla_{\boldsymbol{x}'_j} \mathcal{L}(\boldsymbol{x}'_j, y_j), -1, 1)$$

   - then update x

$$\boldsymbol{x}'_j \leftarrow \boldsymbol{a}_l + \boldsymbol{A}_l \boldsymbol{e}_j$$

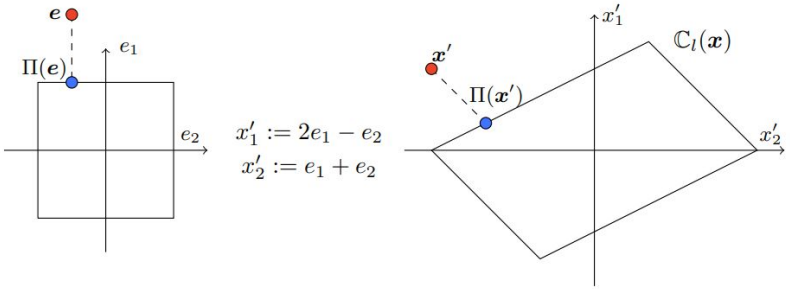


Figure 2: Projection to a region based on linear relaxation using change of variables.

2. **Efficient computations of Convex Regions**
3. **Regularization**

Convex Layerwise Adversarial Training Using Linear Relaxations

Important TRICKS and speedups !!

1. **Projections to Linear Convex Regions**

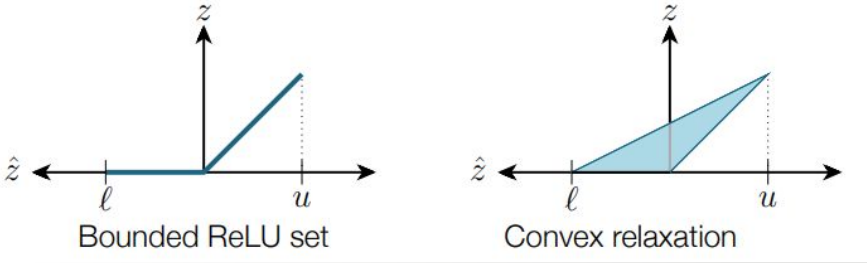
2. **Efficient computations of Convex Regions**
   - Matrix A is sparse, Initialize with Identity.
   - **(more efficient)**, Use approximate computation for obtaining the convex regions.

3. **Regularization**
   - L1 and ReLU Stabilization Regularization.

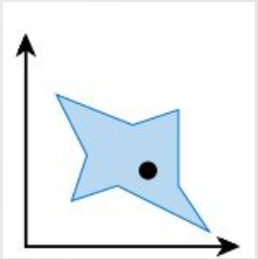
Certification of Neural Networks

SpeedUps: Certification Computations

1. **Refinement of Linear Approximation**
   - Prior work chooses relaxation slope greedily.
   - Instead, optimize for relaxation slope, to minimize the max loss in the convex region.

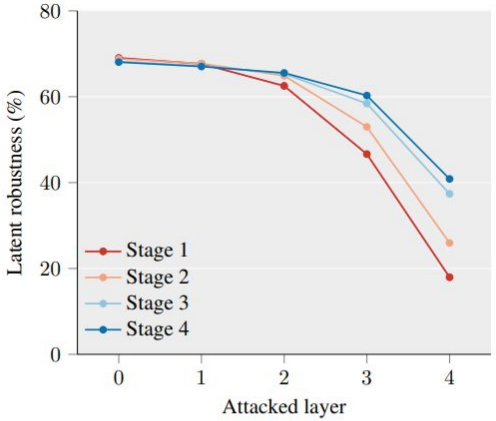


2. **Combining Convex Relaxations with Exact Bound Propagation**
   - Use MILP (Mixed Integer Linear Programming) solver to get exact bounds.

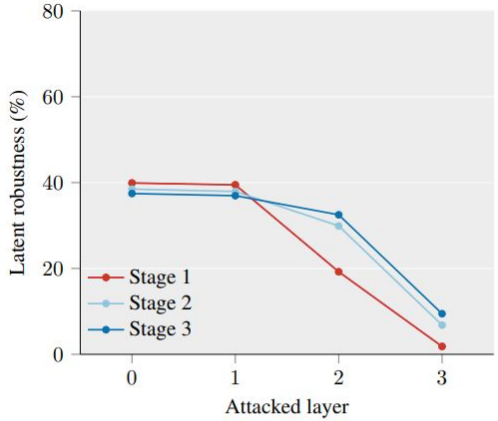


Experiments

- **COLT training in multiple stages (based on the layers).**
- **Evaluation on 2 Conv Nets: 3 layer and 4 layer.**
- **CIFAR-10**



(a) CIFAR-10, 2/255



(b) CIFAR-10, 8/255

Figure 3: Effect of proposed convex layerwise adversarial training. After each stage of the training, we attack the model with a latent adversarial attack on each of the layers. Note that layer 0 represents standard PGD attack (attack in the input space).

Experiments and Results

Table 1: Evaluation on CIFAR-10 dataset with L_∞ 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_∞ 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

Experiments and Results

Table 8: Evaluation on SVHN dataset with L_∞ perturbation 0.01

Method	Accuracy (%)	Certified robustness (%)
Our work	**88.5**	**70.2**
Gowal et al. (2018)	85.2	62.4
Wong & Kolter (2018)	79.6	59.3
Dvijotham et al. (2018a)	83.4	62.4

Table 9: Evaluation on MNIST dataset with L_∞ perturbation 0.1

Method	Accuracy (%)	Certified robustness (%)
Our work	**99.2**	97.1
Gowal et al. (2018)	98.9	**97.7**
Zhang et al. (2019)	99.0	94.4
Wong et al. (2018)	98.9	96.3
Dvijotham et al. (2018a)	98.8	95.6
Mirman et al. (2019)	98.7	95.8
Xiao et al. (2019)	99.0	95.6

Experiments and Results

Table 10: Evaluation on MNIST dataset with L_∞ perturbation 0.3

Method	Accuracy (%)	Certified robustness (%)
Our work	97.3	85.7
Gowal et al. (2018)	**98.3**	91.9
Zhang et al. (2020)	98.2	**93.0**
Wong et al. (2018)	85.1	56.9
Mirman et al. (2019)	96.6	89.3
Xiao et al. (2019)	97.3	80.7

Conclusions

- **COLT:** a new provable adversarial training approach which is empirically promising.
- SoTA robust and standard accuracy on multiple datasets, with low perturbation.
  - Point out approximations (used to avoid time/ memory overhead) as the hindering factor for less than SoTA performance on high perturbation.

- Main Takeaway:
  - Tight convex relaxations with a low memory footprint for efficient projection is important !
- Major Limitation:
  - Methodologies limited to ReLU activations.
  - Scalability (time and memory overheads)

Questions ?

More Data Can Expand the Generalization Gap Between Adversarially Robust and Standard Models

Paper by Lin Chen, Yifei Min, Mingrui Zhang, and Amin Karbasi

Presented by Mike Andrews

Outline

- Introduction to adversarial robustness
- Introduction to the work
- Methods and results
- Conclusions

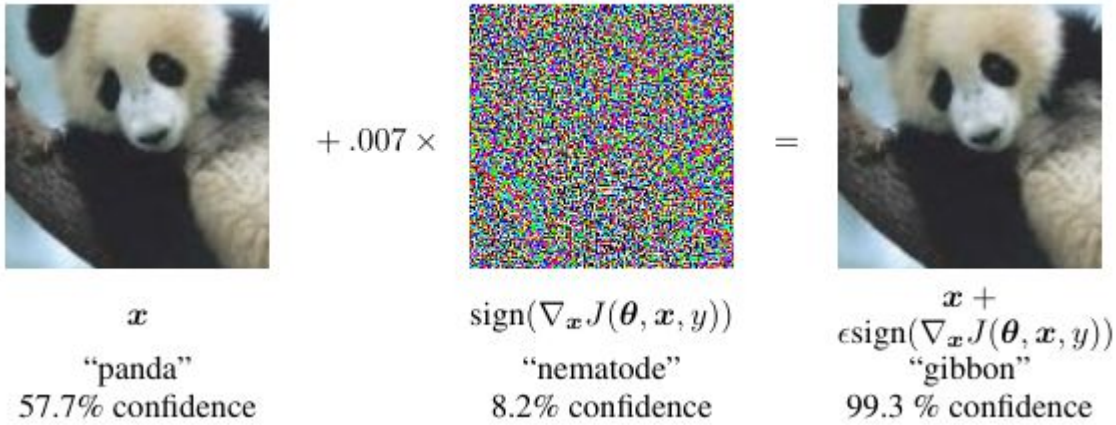
Intro: What is adversarial robustness?

- What if I wanted to cheat facial recognition software?
  - How about tricking self-driving cars?
  - Bank fraud detection software?

- Machine learning is increasingly important
  - Controlling and tricking algorithms is starting to become useful

- We should figure out if this is a thing

Intro: Adversarial Example



Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv preprint arXiv:1412.6572* (2014).

Intro: What does this mean?

- We need methods that can handle “adversarial” examples

- Current methods are extremely vulnerable
  - **Even if** the attacker don’t have access to the underlying model
  - **Even if** you’re classifying an image of a printed adversarial pattern

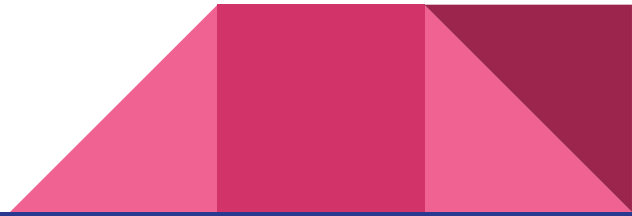
- Current methods can mitigate this, at a cost.

Intro: Adversarial Training

- Current methods trade off accuracy on **real** examples for accuracy on **adversarial** examples.

- They do this by also training their models on adversarial examples

- Adversarially trained models tend to underperform



This Paper: Adversarial Training

- Adversarial training lets an adversary corrupt example data to make the model more robust

$$w_n^{\text{std}} = \arg\min_{w \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; w)\,,$$

Normal training

$$w_n^{\text{rob}} = \arg\min_{w \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\tilde{x}_i \in B_{x_i}^{\infty}(\varepsilon)} \ell(\tilde{x}_i, y_i; w)\,.$$

Adversarial training

- Adversary can only change data a *little bit*.
  - Here, the adversary can change all dimensions by at most ε.

This Paper: The Generalization Gap

- To study the difference, this paper studies **the generalization gap**.
  - <performance of normal models> - <performance of adversarially trained models>

$$g_n = \mathbb{E}_{\{(x_i,y_i)\}_{i=1}^{n} \overset{\text{i.i.d.}}{\sim} \mathcal{D}} \left[L_{\text{test}}(w^{\text{rob}}) - L_{\text{test}}(w^{\text{std}}) \right]$$

$$L_{\text{test}}(w) = \mathbb{E}\left[\frac{1}{n'} \sum_{i=1}^{n'} \ell(x_i', y_i'; w) \right]$$

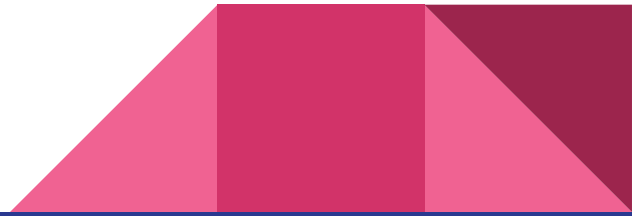
- Some definitions
  - w^{rob} is the adversarially trained model's parameters
  - w^{std} is the normally trained model's parameters
  - loss depends on problem, typically MeanSquaredError or CrossEntropy.

This Paper: Studying The Generalization Gap

- In practice, the generalization gap **shrinks** when you add **more data**
  - This means that, eventually, the adversarially trained model starts to catch up to the regularly trained model.

- Is this always the case? Can we prove it?

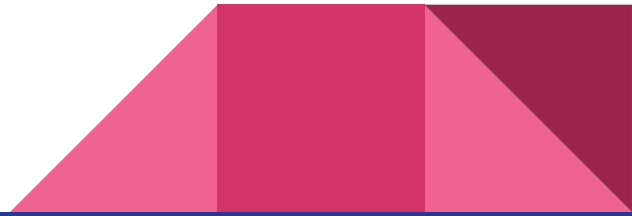
- Result: **No**, and **sometimes things get worse**.



This Paper: Studying The Generalization Gap

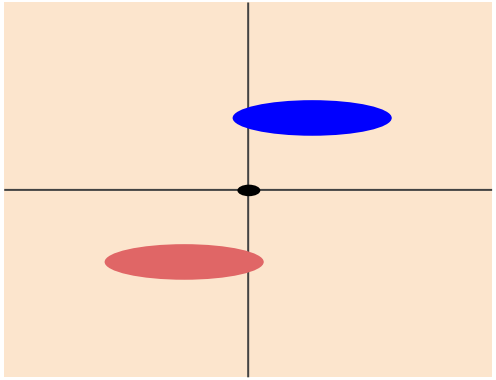
- This paper studies the behavior of three simple models
  - Linear classification of mixture of Gaussian
  - Linear classification of Bernoulli hypercube
  - Linear regression task

- This paper studies how the generalization gap g_n behaves
  - Attempts to bound g_n
  - Attempts to identify g_n in limit with infinite data
  - Characterizes g_n behavior and transition points
  - Empirically studies g_n behavior in regression problem



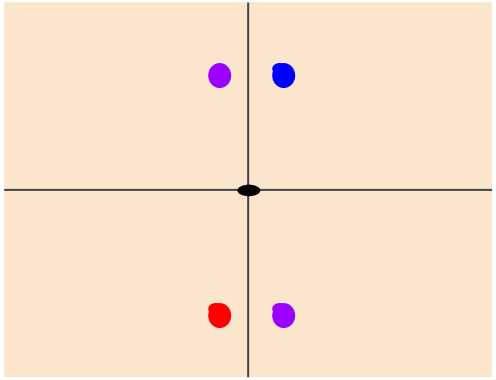
Methods: Gaussian Classification

- Gaussian model
  - **P(x | y) ~ N(yμ, Σ)**
  - Variance is diagonal: $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_d^2)$
  - Matrix mean is nonnegative
  - Solve with linear classification
  - TL; DR: **Two gaussian classes flipped over the center.**



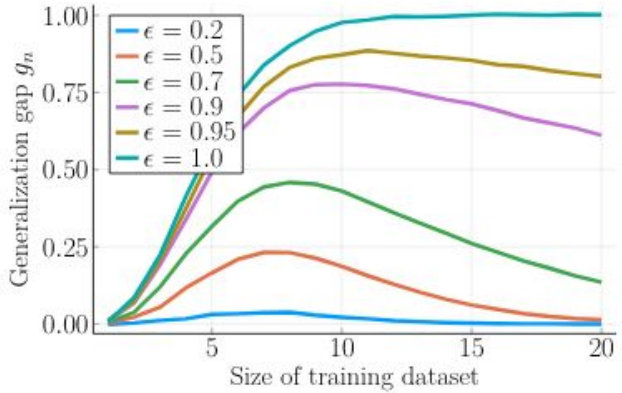
Methods: Gaussian Classification

- Bernoulli model
  - **$P(x_j \mid y) \sim y\,\theta_j$ with probability $(1+\tau)/2$**
  - **$\sim -y\,\theta_j$ with probability $(1-\tau)/2$**
  - All points are on a **hyperrectangle** defined by **θ**
  - “Signal strength” parameter **τ** can flip signs randomly
  - Solve with linear classification
  - TL; DR: **Hypercube fading from -x-y-z to +x+y+z.**

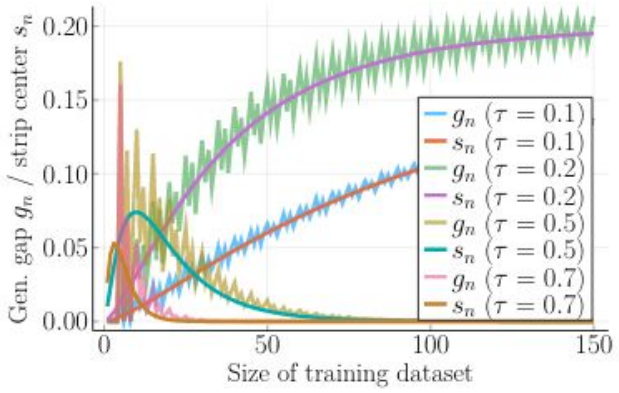


Results: Generalization Gap and Dataset Size

- Two apparent regimes: Catching up, and not catching up.



(a) Gaussian model



(b) Bernoulli model

- Will this catch up?

Results: Generalization Gap and Dataset Size

- Will this catch up? **Not necessarily.**
- In the **Gaussian** case, the authors prove:
  - The gap is nonnegative
  - The final gap size is: $$\lim_{n\to\infty} g_n = 2W \sum_{j\in[d]:\mu(j)>0} \mu(j) H\left(\frac{\varepsilon}{\mu(j)} - 1\right),$$
  - If the adversary is “too strong” (ε is larger than gaussian center), the gap continues growing
    - (Strictly increasing towards bound)
  - If the adversary is “too weak” (ε is smaller than smallest gaussian center dimension), gap **grows**, then **shrinks**

$$n < \min_{\substack{j\in[d]:\\ \mu(j)>0}} \max\left\{\frac{3}{2}, 2\log\frac{1}{1-\varepsilon/\mu(j)}\right\}\left(\frac{\sigma(j)}{\mu(j)}\right)^2$$

$$n \geq \max_{\substack{j\in[d]:\\ \mu(j)>0}} \left(K_0 + 2\log\frac{1}{1-\varepsilon/\mu(j)}\right)\left(\frac{\sigma(j)}{\mu(j)}\right)^2$$

Results: Generalization Gap and Dataset Size

- Will this catch up? **Not necessarily.**
- In the **Bernoulli** case, the authors prove:
  - The gap is nonnegative
  - The final gap size is: $\lim_{n\to\infty} g_n = 2W\tau \sum_{j\in[d]:\theta(j)>0} \theta(j) H\left(\frac{\varepsilon}{\theta(j)\tau} - 1\right)$
  - If the adversary is “too strong” (ε is larger than the largest hypercube dimension), the gap continues growing
    - (Strictly increasing towards bound)
  - If the adversary is “too weak” (ε is smaller than smallest hypercube dimension), gap **grows**, then **shrinks**

$$n < \left(\frac{1}{\tau^2} - 1\right) \max\left\{\frac{3}{2}, 2 \min_{\substack{j\in[d]:\\ \theta(j)>0}} \log \frac{1}{1-\frac{\varepsilon}{\theta(j)\tau}}\right\} \qquad n \geq \left(\frac{1}{\tau^2} - 1\right)\left(K_0 + 2 \max_{\substack{j\in[d]:\\ \theta(j)>0}} \log \frac{1}{1-\frac{\varepsilon}{\theta(j)\tau}}\right)$$

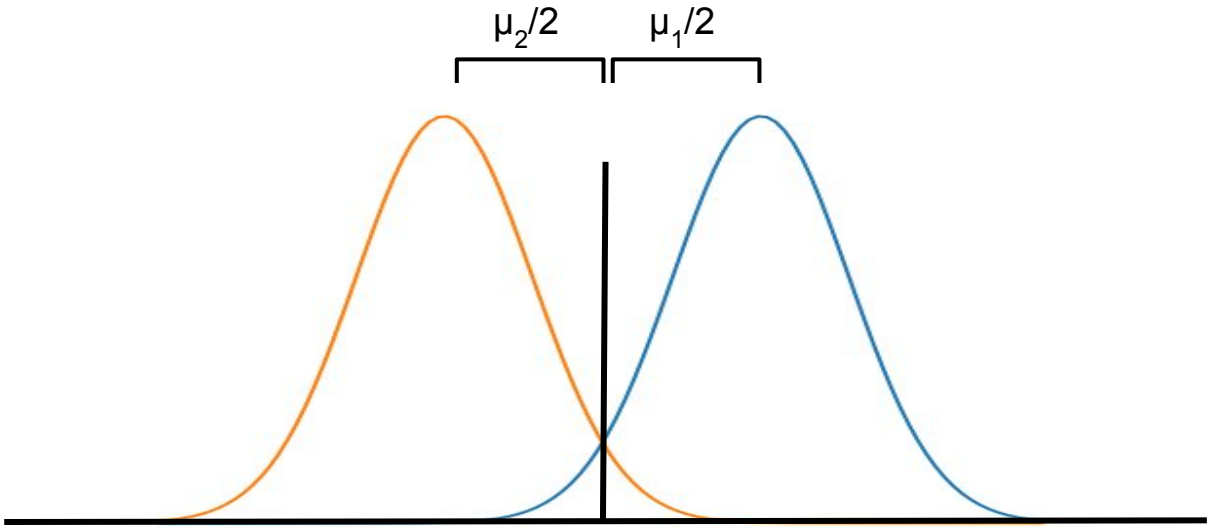
Results: Generalization Gap and Dataset Size

- Two terms show up repeatedly
  - Gaussian: **ε / μ**
  - Bernoulli: **ε / (μ τ)**

- These represent the distance to 0 along an axis

- If an adversary can perturb most data points past the center point, performance drops and training needs more data.

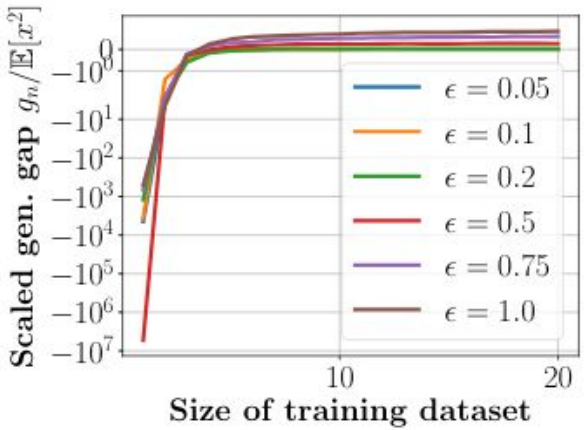
Results: Intuition



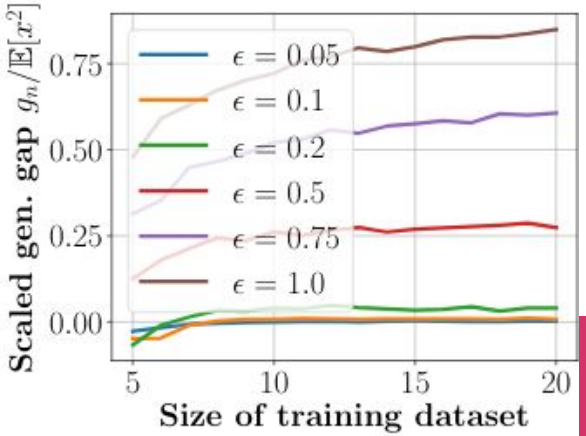
If the adversary can cross the middle in most cases, not much can save us.

Methods and Results: Regression

- The authors also studied a linear regression problem with gaussian noise.
- No strong theoretical results, but the authors found these graphs interesting.
- When spacing data samples in X using a normal distribution:



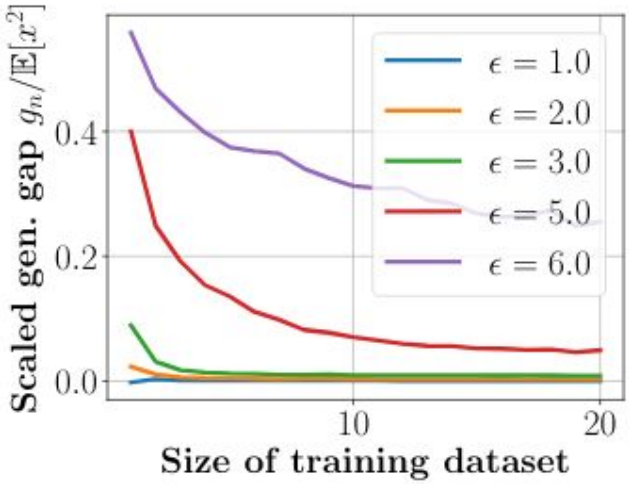
(a) $x \sim \mathcal{N}(0,1)$, $1 \leq n \leq 20$.



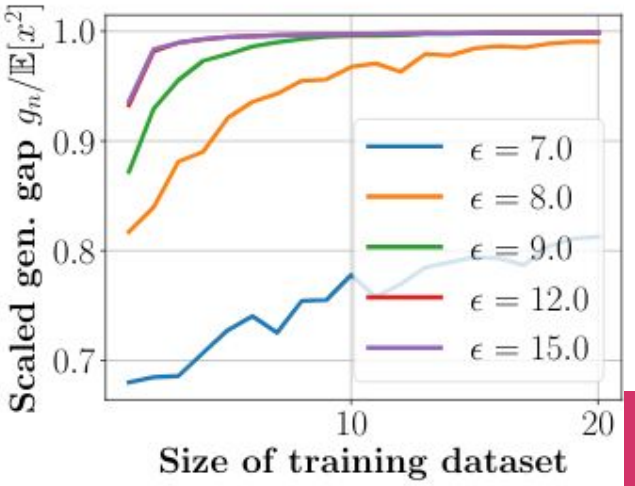
(b) $x \sim \mathcal{N}(0,1)$, $5 \leq n \leq 20$.

Methods and Results: Regression

- When spacing data samples in X using a Poisson distribution:



(c) $x \sim \text{Poisson}(5) + 1$, small ε.



(d) $x \sim \text{Poisson}(5) + 1$, large ε.

Methods and Results: Regression

- The authors continue to justify some of the observations in the graphs using extremely reduced sample problems.

- Additionally, the authors boil down some equations into the following forms:

$$w_n^{\mathrm{rob}} = \operatorname*{arg\,min}_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left(|y_i - \langle w, x_i \rangle| + \varepsilon \sum_{j=1}^{d} |w(j)| \right)^2$$

$$g_n = \|w_n^{\mathrm{rob}} - w^*\|^2_{\mathrm{E}_{x \sim P_X}[xx^\top]} - \|w_n^{\mathrm{std}} - w^*\|^2_{\mathrm{E}_{x \sim P_X}[xx^\top]}.$$

Conclusions

- Interesting outcome!
  - For **strong** adversaries, in multiple simple models, **more training data does not always solve the generalization gap**.
  - For **weak** agents, the gap may widen for a time, then catch up.

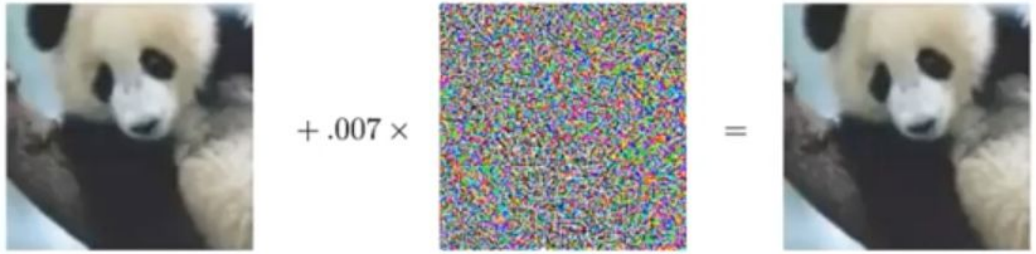
- For multiple simple models, the ratio of the distance between class centers and the adversarial strength determines training difficulty.
- Specific bounds and limits are available on the generalization gap for two simple models.

Understanding and Mitigating the Tradeoff Between Robustness and Accuracy

Aditi, Michael, Fanny, John and Percy

Koushik R Bukkasamudram

Problem 1: Adversarial Examples



"panda"
57.7% confidence

"nematode"
8.2% confidence

"gibbon"
99.3 % confidence

[Goodfellow et al. 2015]

CIFAR10 Image Classification:

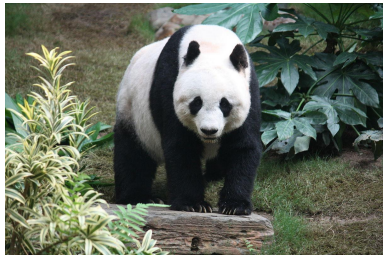
95%

0%

Normal test

Adversarially perturbed test

Panda + Nematode => Gibbon



What if?

Setup of Adversarial Examples

- Standard error: average over distribution

$$\mathsf{P}[f(x) \neq y]$$

- Robust error: average over worst-case perturbations

$$\mathsf{P}[\exists \tilde{x} \in B(x) \text{ such that } f(\tilde{x}) \neq y]$$

$$B(x) = \{\tilde{x} \mid \|\tilde{x} - x\|_\infty \leq \epsilon\}$$

- Standard training: find f to optimize standard error on training data
- Robust training: find f to optimize robust error on training data

Robust training increases standard error

Dataset: CIFAR10

Method	Robust Accuracy	Standard Accuracy
Standard Training	0%	95.2%
Adversarial Training (Zhang et al. 2019)	55.4%	84.0%

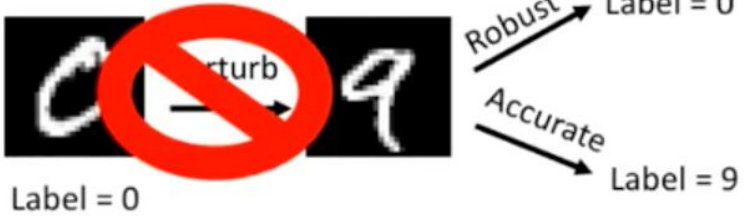
Robust training increases standard error

Dataset: CIFAR10

Method	Robust Accuracy	Standard Accuracy
Standard Training	0%	95.2%
Adversarial Training (Zhang et al. 2019)	55.4%	84.0%

Prior work on robustness vs. accuracy

- Optimal predictor (in terms of accuracy) is itself not robust [Tsipras et al. 2019]



Consistent Perturbations

- Hypothesis class not expressive enough [Nakkiran et al. 2019]

Neural networks typically expressive enough to represent arbitrary functions

Well-specified

Is the tradeoff Inherent

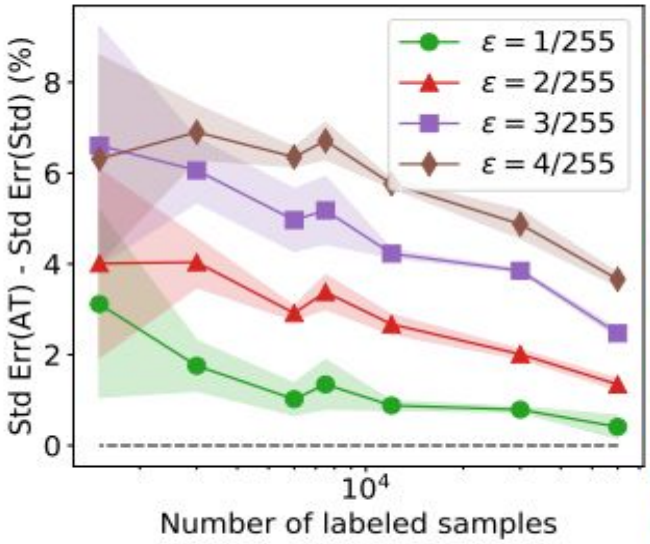
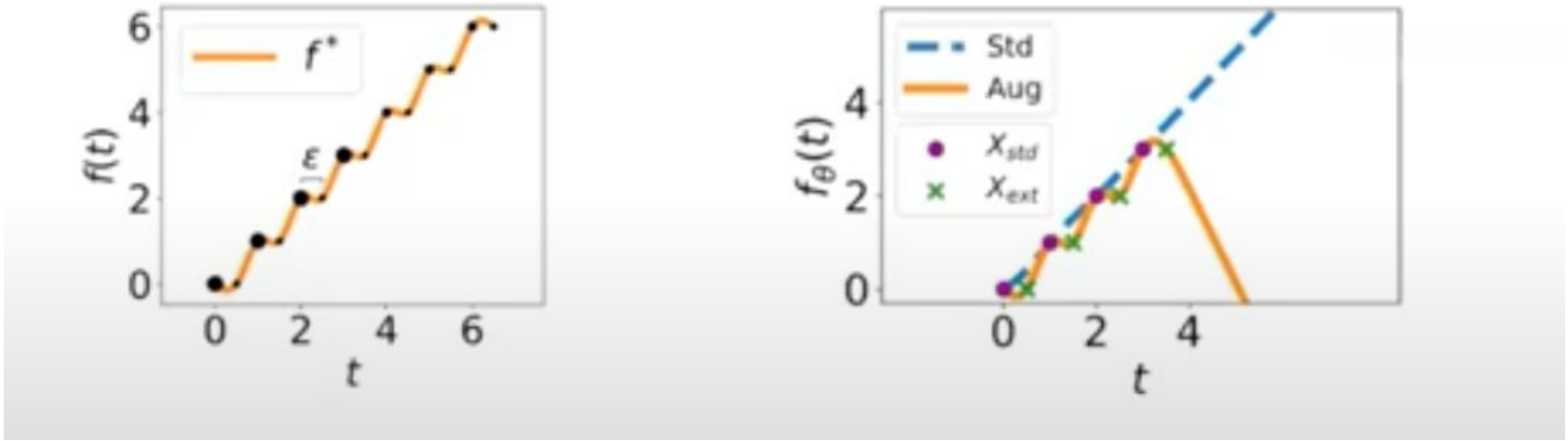


Figure 1. Gap between the standard error of adversarial training (Madry et al., 2018) with ℓ_∞ perturbations, and standard training. The gap decreases with increase in training set size, suggesting that the tradeoff between standard and robust error should disappear with infinite data.

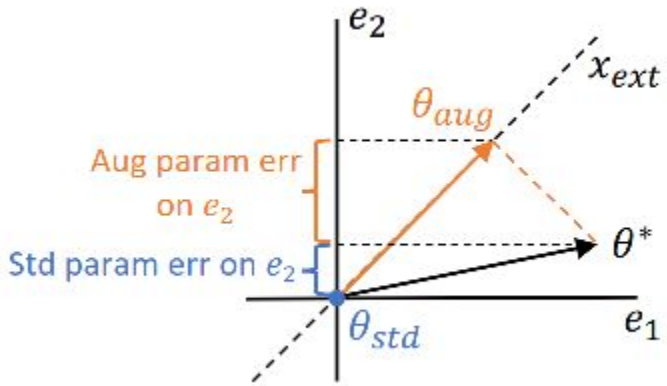
Introduction of Adversarial Examples



General linear model

- Model (noiseless): $y = x^\top \theta^*$ Well-specified
- Standard data: $X_{std} \in \mathbb{R}^{n \times d}$, $y_{std} = X_{std}\theta^*$, $n \ll d$ (overparameterized)
- Extra data (adv examples): $X_{ext} \in \mathbb{R}^{m \times d}$, $y_{ext} = X_{ext}\theta^*$ Consistent perturbations
- We study the following **min-norm interpolants**:
  - $\theta_{std} = \operatorname{argmin}_\theta \{\|\theta\|_2 : X_{std}\theta = y_{std}\}$
  - $\theta_{aug} = \operatorname{argmin}_\theta \{\|\theta\|_2 : X_{std}\theta = y_{std}, X_{ext}\theta = y_{ext}\}$

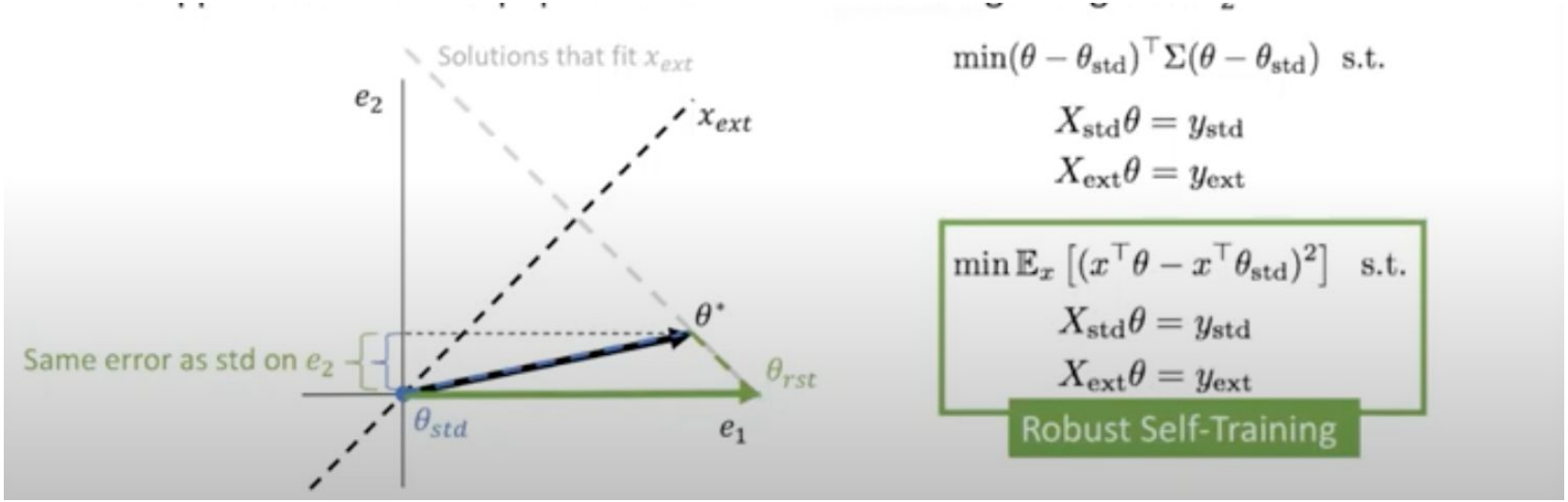
Question: Why would extra data add to the error



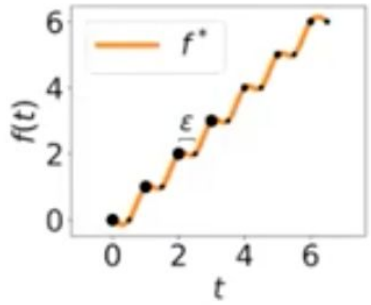
- Std error: $L_{std}(\theta) = (\theta - \theta^*)^T \Sigma (\theta - \theta^*)$ for population covariance Σ
- If Σ has high weight on e_2 direction, errors in e_2 are more costly

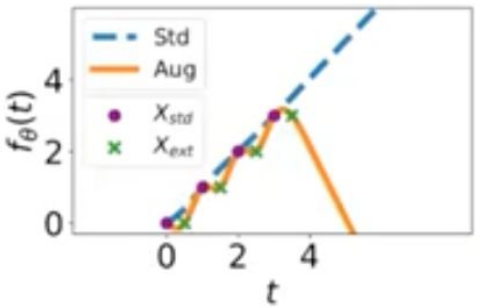
$$\theta_{std} = \text{argmin}_\theta \{\|\theta\|_2 : X_{std}\theta = y_{std}\}$$
$$\theta_{aug} = \text{argmin}_\theta \{\|\theta\|_2 : X_{std}\theta = y_{std}, X_{ext}\theta = y_{ext}\}$$

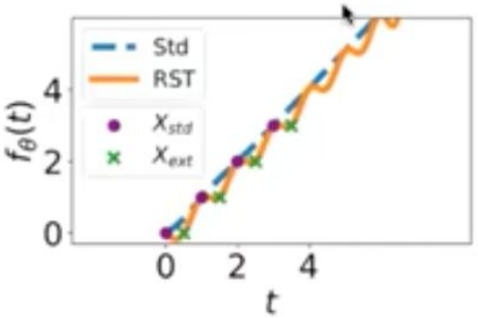
Mitigating the tradeoff



Splines Revisited: How RST improves







Loss Functions

	Standard	Robust
Labeled	$(y - x^\top \theta)^2$ Noiseless targets	$\max_{x_{\text{adv}} \in T(x)} (x^\top \theta - x_{\text{adv}}^\top \theta)^2$ Consistent perturbations
Unlabeled	$(\tilde{y} - \tilde{x}^\top \theta)^2$ Imperfect pseudo-labels	$\max_{\tilde{x}_{\text{adv}} \in T(\tilde{x})} (\tilde{x}^\top \theta - \tilde{x}_{\text{adv}}^\top \theta)^2$ Consistent perturbations

Theorem

Theorem (informal): for noiseless linear regression, $L_{std}(\theta_{rst}) \leq L_{std}(\theta_{std})$ and $L_{rob}(\theta_{rst}) \leq L_{rob}(\theta_{aug})$.

Results

Method	Robust Test Acc.	Standard Test Acc.	
Standard Training	0.8%	95.2%	Vanilla Supervised
PG-AT (Madry et al., 2018)	45.8%	87.3%	
TRADES (Zhang et al., 2019)	55.4%	84.0%	
Standard Self-Training	0.3%	96.4%	Semisupervised with same unlabeled data
Robust Consistency Training (Carmon et al., 2019)	56.5%	83.2%	
RST + PG-AT (this paper)	**58.5%**	**91.8%**	
RST + TRADES (this paper)	**63.1%**	**89.7%**	
(Carmon et al., 2019)			
Interpolated AT (Lamb et al., 2019)[3]	45.1%	93.6%	Modified supervised
Neural Arch. Search (Cubuk et al., 2017)	50.1%	93.2%	

Method	Robust Test Acc.	Standard Test Acc.	
Standard Training	0.2%	94.6%	Vanilla Supervised
Worst-of-10	73.9%	95.0%	
Random	67.7%	95.1%	
RST + Worst-of-10 (this paper)	**75.1%**	**95.8%**	Semisupervised
RST + Random (this paper)	**70.9%**	**95.8%**	
Worst-of-10 (Engstrom et al., 2019)[4]	69.2%	91.3%	Existing baselines (smaller model)
Random (Yang et al., 2019)[5]	58.3%	91.8%	

Summary

- Adding valid data can increase the standard error
- Inductive Bias plays an important role
- RST improves both standard accuracy and robustness is still a future work to match the state-of-the-art ones

Thankyou! Have a good one


Thank You
University of Massachusetts Amherst
BE REVOLUTIONARY™