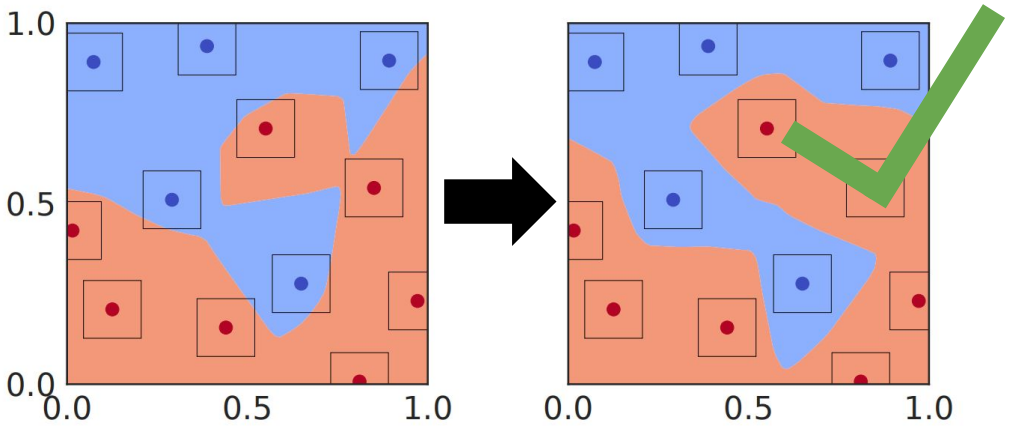
Provable Adversarial Robustness



Game of cat and mouse

- Many proposed defenses and detectors have been broken
  - “Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods”
  - “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples”
  - “Evaluating and Understanding the Robustness of Adversarial Logit Pairing”
- How can we be sure no future attacks will break a defense?
- Theoretical performance guarantees!

Training provably robust networks

Two important papers

ICLR

- “Certified Defenses Against Adversarial Examples”
  - [Raghunathan et al. ICLR 2018]
- “Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope”
  - [Wong and Kolter. ICML 2018]

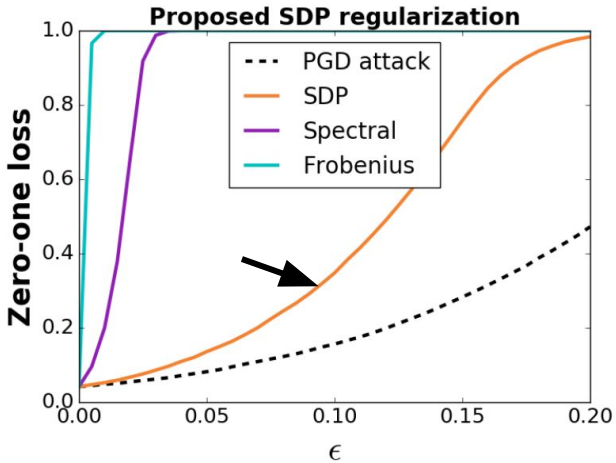
ICML

Certified Defenses Against Adversarial Examples

Aditi Raghunathan, Jacob Steinhardt, Percy Liang

Overview

- Tractable bound on the adversarial error for 2-layer networks
  - Exciting because of the universal approximation theorem
- Improves on prior art for certified robustness on MNIST
  - Certificate of 35% test error for ℓ_∞ adversaries with $\epsilon = 0.1$.



Notation and setup

Score-based classifiers. Our goal is to learn a mapping $C : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^d$ is the input space (e.g., images) and $\mathcal{Y} = \{1, \ldots, k\}$ is the set of k class labels (e.g., object categories). Assume C is driven by a scoring function $f^i : \mathcal{X} \to \mathbb{R}$ for all classes $i \in \mathcal{Y}$, where the classifier chooses the class with the highest score: $C(x) = \arg\max_{i \in \mathcal{Y}} f^i(x)$. Also, define the *pairwise margin* $f^{ij}(x) \stackrel{\text{def}}{=} f^i(x) - f^j(x)$ for every pair of classes (i, j). Note that the classifier outputs $C(x) = i$ iff $f^{ij}(x) > 0$ for all alternative classes $j \neq i$. Normally, a classifier is evaluated on the 0-1 loss $\ell(x, y) = \mathbb{I}[C(x) \neq y]$.

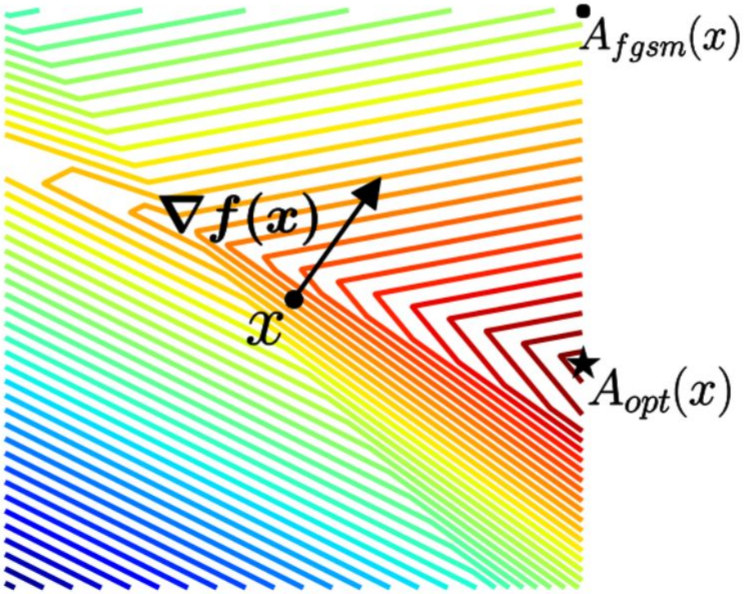
Many possible adversaries

Arbitrary Attacker

$$A\colon \mathcal{X} \to \mathcal{X} \qquad A(x) \in B_\epsilon(x)$$

Optimal Attacker

$$A_{\text{opt}}(x) = \text{argmax}_{\tilde{x} \in B_\epsilon(x)} \max_i f^{iy}(\tilde{x})$$



More setup!

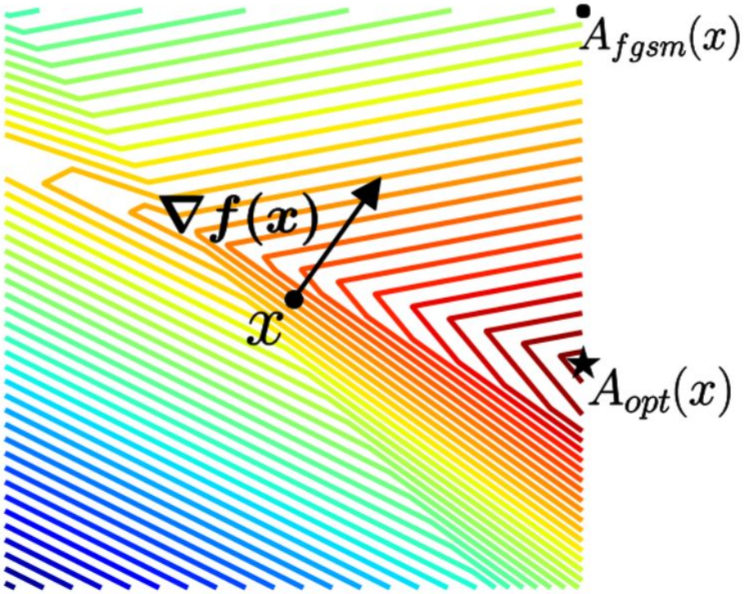
w.l.o.g. We consider binary classifiers

$f(x) = f^1(x) - f^2(x)$

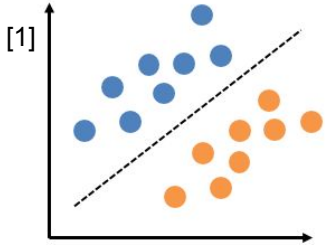
$A_{\text{opt}}(x) = \text{argmax}_{\tilde{x} \in B_\epsilon(x)} f(\tilde{x})$

We are interested in bounds of the form

$f(A_{\text{opt}}(x)) < \text{something}$



Bound for a linear classifier



Using Hölder's inequality, we get that for any $\tilde{x} \in B_\epsilon(x)$,

$$f(\tilde{x}) = f(x) + (W_1 - W_2)^\top (\tilde{x} - x) \leq f(x) + \epsilon \|W_1 - W_2\|_1.$$

Theorem (Hölder's inequality). Let (S, Σ, μ) be a measure space and let $p, q \in [1, \infty]$ with $1/p + 1/q = 1$. Then, for all measurable real- or complex-valued functions f and g on S,

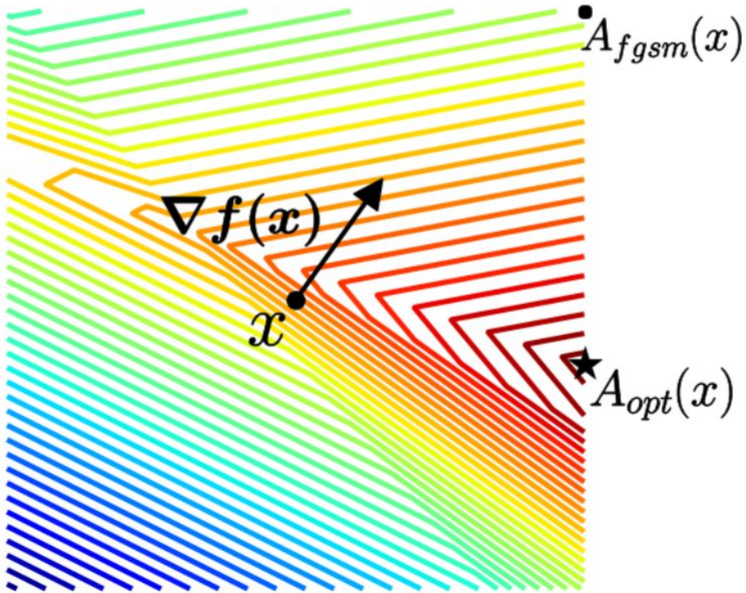
$$\|fg\|_1 \leq \|f\|_p \|g\|_q.$$

Bound for nonlinear classifiers

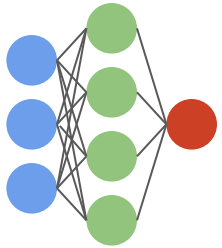
An integration trick gives an intuitive bound

$$f(\tilde{x}) = f(x) + \int_0^1 \nabla f\big(t\tilde{x} + (1-t)x\big)^\top \big(\tilde{x} - x\big) dt$$
$$\leq f(x) + \max_{\tilde{x} \in B_\epsilon(x)} \epsilon \|\nabla f(\tilde{x})\|_1,$$

Computing this is hard in general, and it must be repeated for every input.



Bound for a two-layer neural network

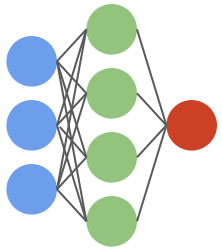


For a 2-layer network, we can get somewhere.

$$\begin{aligned}
\|\nabla f(\tilde{x})\|_1 &\overset{(i)}{=} \|W^\top \operatorname{diag}(v)\sigma'(W\tilde{x})\|_1 \\
&\overset{(ii)}{\leq} \max_{s\in[0,1]^m} \|W^\top \operatorname{diag}(v)s\|_1 \\
&\overset{(iii)}{=} \max_{s\in[0,1]^m, t\in[-1,1]^d} t^\top W^\top \operatorname{diag}(v)s,
\end{aligned}$$

Crucially, this does not depend on the input.

Bound for a two-layer neural network



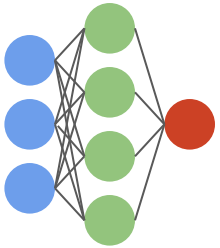
This gives a quadratic program bound on the adversarial margin:

$$f(A_{\text{opt}}(x)) \leq f(x) + \epsilon \max_{\tilde{x} \in B_\epsilon(x)} \|\nabla f(\tilde{x})\|_1$$

$$\leq f(x) + \epsilon \max_{s \in [0,1]^m, t \in [-1,1]^d} t^\top W^\top \operatorname{diag}(v) s \stackrel{\text{def}}{=} f_{\text{QP}}(x).$$

This is NP-hard and similar to the MAXCUT problem.

Bound for a two-layer neural network



Mirroring a previous approach to relax MAXCUT to an SDP, the authors derive

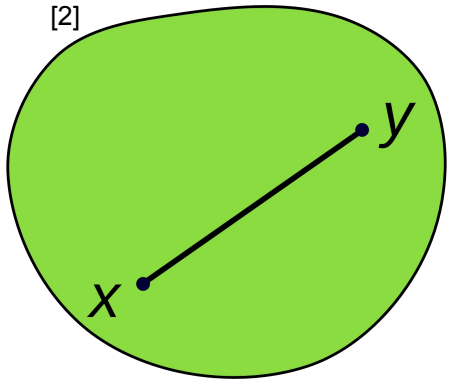
$$f_{\mathrm{QP}}(x) \leq f_{\mathrm{SDP}}(x) \stackrel{\mathrm{def}}{=} f(x) + \frac{\epsilon}{4} \max_{P \succeq 0, \mathrm{diag}(P) \leq 1} \langle M(v, W), P \rangle .$$

The matrices M and P are defined in the paper.

This is a convex, SDP relaxation of the QP.

What is an SDP?

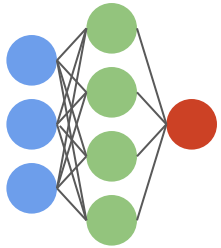
- Linear objective function
- Variable is a positive semi-definite (PSD) matrix
- Linear constraints



The space of PSD matrices is convex, so this is a convex relaxation.

$$\begin{aligned} \min_{X \in \mathbb{S}^n} \quad & \langle C, X \rangle_{\mathbb{S}^n} \\ \text{subject to} \quad & \langle A_k, X \rangle_{\mathbb{S}^n} \leq b_k, \quad k = 1, \ldots, m \\ & X \succeq 0 \end{aligned}$$

Bound for a two-layer neural network



Unfortunately, this is still too slow.

8 hours per network on a CPU.

Not feasible for training neural networks.

Efficiency from duality

The authors obtain a version suitable for training from the theory of duality.

$$\max_{P \succeq 0, \operatorname{diag}(P) \leq 1} \left\langle M^{ij}(V, W), P \right\rangle = \min_{c^{ij} \in \mathbf{R}^D} D \cdot \lambda^{+}_{\max}\left(M^{ij}(V, W) - \operatorname{diag}(c^{ij})\right) + \mathbf{1}^{\top} \max(c, 0), \tag{14}$$

where $D = (d + m + 1)$ and $\lambda^{+}_{\max}(B)$ is the maximum eigenvalue of B, or 0 if all eigenvalues are negative. This dual formulation allows us to introduce additional dual variables $c^{ij} \in \mathbb{R}^D$ that are optimized *at the same time as the parameters V and W*, resulting in an objective that can be trained efficiently using stochastic gradient methods.

Duality proof

In this section we justify the duality relation (14). Recall that the primal program is

$$\begin{aligned} &\text{maximize } \langle M, P \rangle \\ &\text{subject to } P \succeq 0, \operatorname{diag}(P) \leq 1. \end{aligned} \tag{19}$$

Rather than taking the dual directly, we first add the redundant constraint $\operatorname{tr}(P) \leq d + m + 1$ (it is redundant because the SDP is in $d + m + 1$ dimensions and $\operatorname{diag}(P) \leq 1$). This yields

$$\begin{aligned} &\text{maximize } \langle M, P \rangle \\ &\text{subject to } P \succeq 0, \operatorname{diag}(P) \leq 1, \operatorname{tr}(P) \leq d + m + 1. \end{aligned} \tag{20}$$

We now form the Lagrangian for the constraints $\operatorname{diag}(P) \leq 1$, leaving the other two constraints as-is. This yields the equivalent optimization problem

$$\begin{aligned} &\text{maximize } \min_{c \geq 0} \langle M, P \rangle + c^\top (\mathbf{1} - \operatorname{diag}(P)) \\ &\text{subject to } P \succeq 0, \operatorname{tr}(P) \leq d + m + 1. \end{aligned} \tag{21}$$

Now, we apply minimax duality to swap the order of min and max; the value of (21) is thus equal to

$$\begin{aligned} &\text{minimize } \max_{\substack{P \succeq 0, \\ \operatorname{tr}(P) \leq d+m+1}} \langle M, P \rangle + c^\top (\mathbf{1} - \operatorname{diag}(P)) \\ &\text{subject to } c \geq 0. \end{aligned} \tag{22}$$

Duality proof

The inner maximum can be simplified as

$$\mathbf{1}^\top c+(d+m+1)\cdot\Big(\max_{P\succeq 0,\mathrm{tr}(P)\leq 1}\langle M-\mathrm{diag}(c),P\rangle\Big)=\mathbf{1}^\top c+(d+m+1)\lambda^+_{\max}(M-\mathrm{diag}(c)). \quad (23)$$

Therefore, (22) simplifies to

$$\begin{aligned}&\text{minimize } \mathbf{1}^\top c+(d+m+1)\lambda^+_{\max}(M-\mathrm{diag}(c)) \quad (24)\\ &\text{subject to } c\geq 0.\end{aligned}$$

This is almost the form given in (14), except that c is constrained to be non-negative and we have $\mathbf{1}^\top c$ instead of $\mathbf{1}^\top \max(c,0)$. However, note that for the $\lambda^+_{\max}$ term, it is always better for c to be larger; therefore, replacing c with $\max(c,0)$ means that the optimal value of c will always be non-negative, thus allowing us to drop the $c\geq 0$ constraint and optimize c in an unconstrained manner. This finally yields the claimed duality relation (14).

Final method

Generalization to multiple classes. The preceding arguments all generalize to the pairwise margins f^{ij}, to give:

$$f^{ij}(A(x)) \leq f^{ij}_{\text{SDP}}(x) \stackrel{\text{def}}{=} f^{ij}(x) + \frac{\epsilon}{4} \max_{P \succeq 0, \text{diag}(P) \leq 1} \left\langle M^{ij}(V, W), P \right\rangle, \text{ where} \tag{11}$$

$M^{ij}(V, W)$ is defined as in (9) with $v = V_i - V_j$. The adversarial loss of any attacker, $\ell_A(x, y) = \mathbb{I}[\max_{i \neq y} f^{iy}(A(x)) > 0]$, can be bounded using the fact that $f^{iy}_{\text{SDP}}(x) \geq f^{iy}(A(x))$. In particular,

$$\ell_A(x, y) = 0 \text{ if } \max_{i \neq y} f^{iy}_{\text{SDP}}(x) < 0. \tag{12}$$

How they measure adversarial error using the margin bound

The final objective. Using (14), we end up optimizing the following training objective:

$$(W^\star, V^\star, c^\star) = \arg\min_{W,V,c} \sum_n \ell_{\text{cls}}(V, W; x_n, y_n) + \sum_{i \neq j} \lambda^{ij} \cdot \left[D \cdot \lambda^+_{\max}(M^{ij}(V, W) - \text{diag}(c^{ij})) + \mathbf{1}^\top \max(c^{ij}, 0) \right]$$

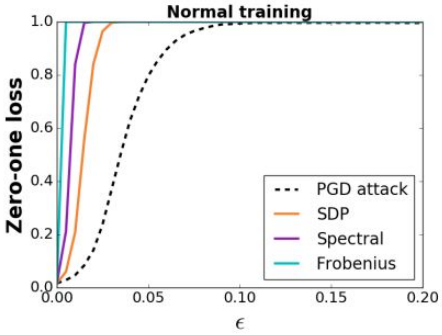
Other upper bounds

The Frobenius norm and spectral norm of a linear layer give bounds. Letting these bounds cascade across all the layers gives a layer-wise bound on the adversarial error.

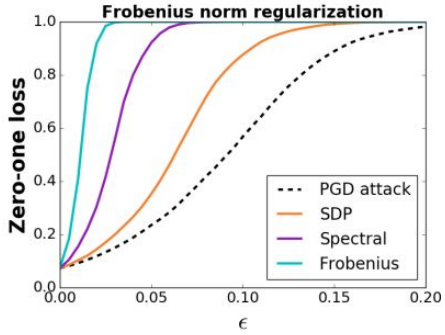
These are typically weak bounds, though they are fast to compute.


SUN DEVILS
40
ARIZONA
5
SUN DEVILS
21

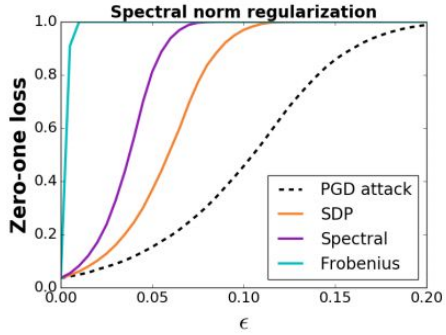
Results



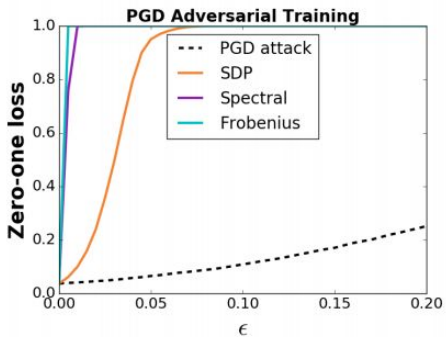
(a) NT-NN



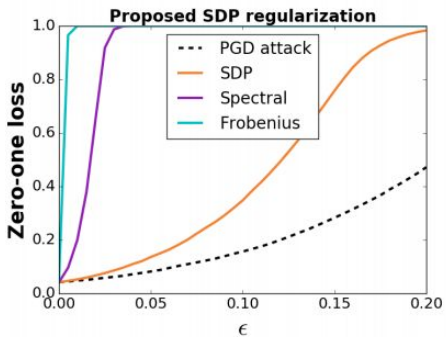
(b) Fro-NN



(c) Spe-NN

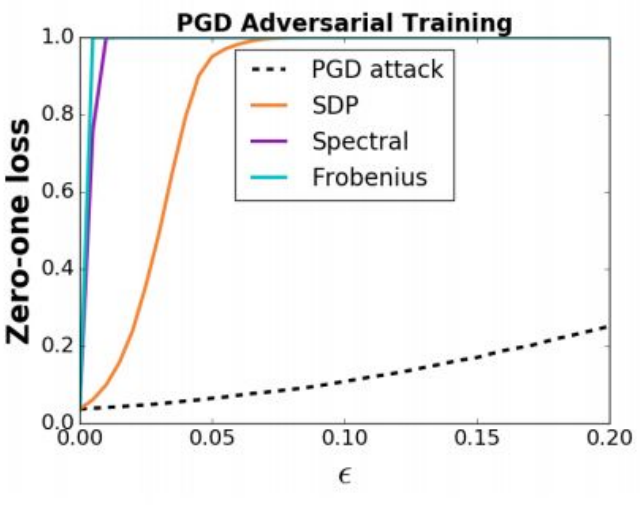


(d) AT-NN

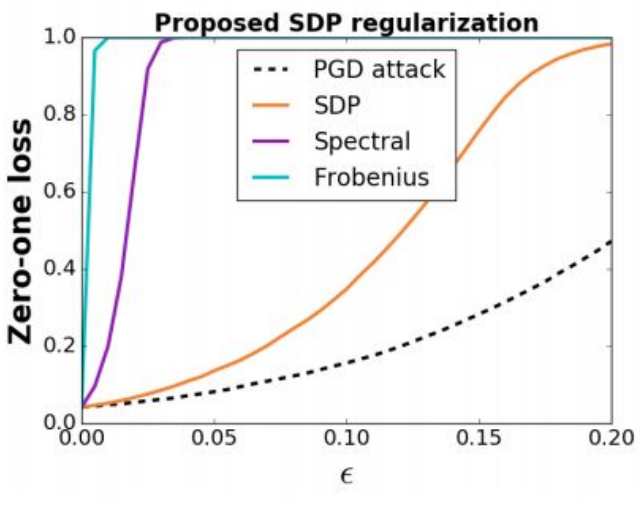


(e) SDP-NN

Results



(d) AT-NN



(e) SDP-NN

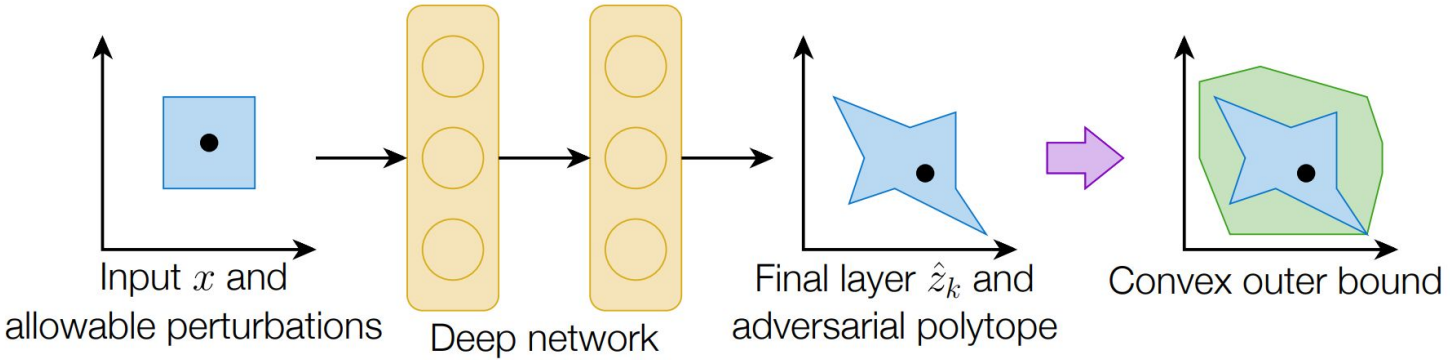
Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope

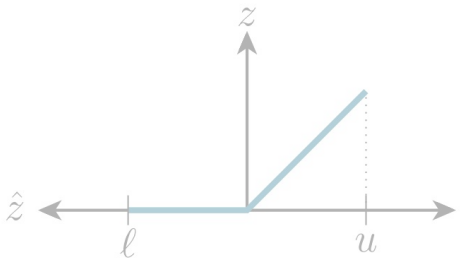
Eric Wong, Zico Kolter

Overview

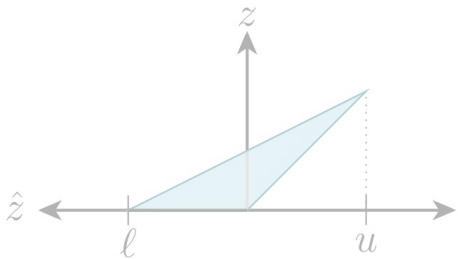
- Tractable bound on the adversarial loss for *deep networks*
- Elegant, highly flexible approach
  - Can be applied to convolutional networks
- SOTA provable robustness at time of publication
  - Certificate of 5.8% test error for ℓ_∞ adversaries with $\epsilon = 0.1$.

Convex outer adversarial polytopes



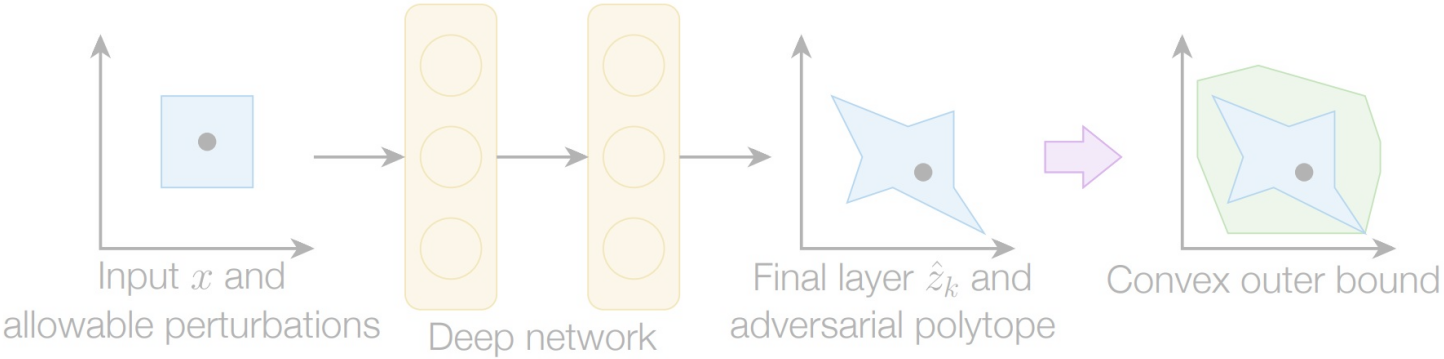


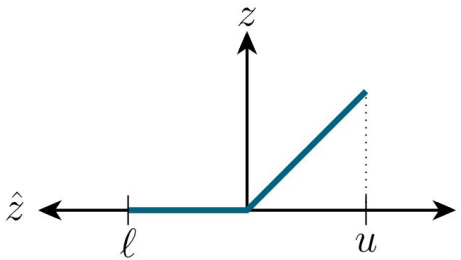
Bounded ReLU set



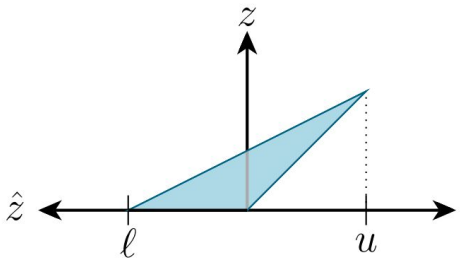
Convex relaxation

Convex outer adversarial polytopes



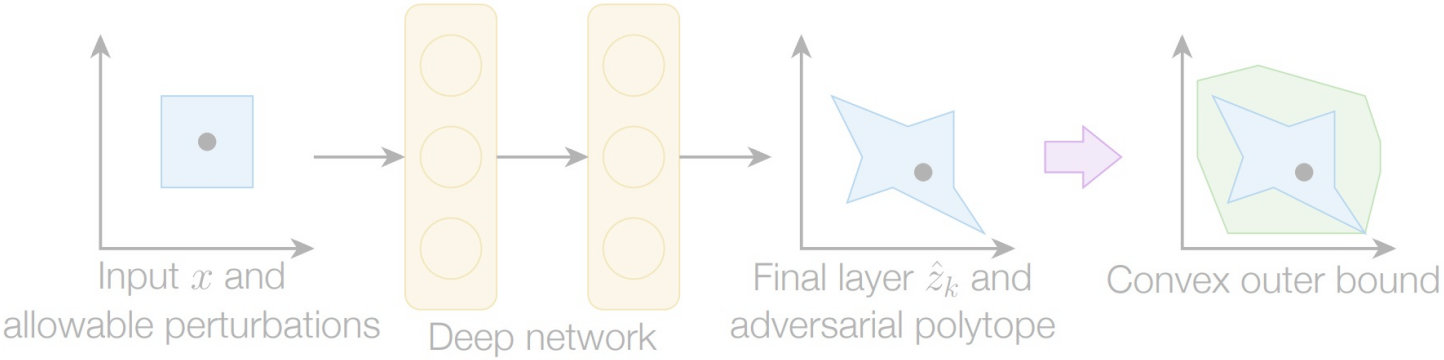


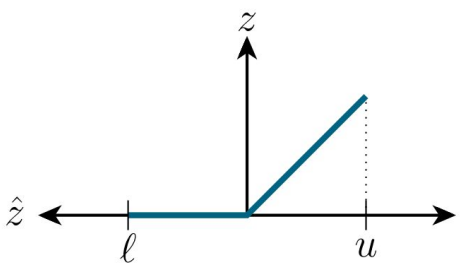
Bounded ReLU set



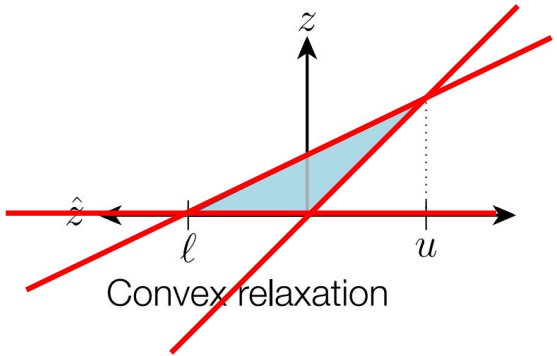
Convex relaxation

Convex outer adversarial polytopes





Bounded ReLU set



Convex relaxation

Linear program

They take a slightly different approach:

- Each activation is a variable
- The activations are constrained to mimic the network

$$\underset{\hat{z}_k}{\text{minimize}}\, c^T \hat{z}_k, \quad \text{subject to}$$

$$\hat{z}_{i+1} = W_i z_i + b_i, \; i = 1, \ldots, k-1$$

$$z_1 \leq x + \epsilon$$

$$z_1 \geq x - \epsilon$$

$$z_{i,j} = 0, \; i = 2, \ldots, k-1, j \in \mathcal{I}_i^-$$

$$z_{i,j} = \hat{z}_{i,j}, \; i = 2, \ldots, k-1, j \in \mathcal{I}_i^+$$

$$\left.\begin{array}{l} z_{i,j} \geq 0, \\ z_{i,j} \geq \hat{z}_{i,j}, \\ \Big((u_{i,j} - \ell_{i,j}) z_{i,j} \\ \quad - u_{i,j} \hat{z}_{i,j}\Big) \leq -u_{i,j}\ell_{i,j} \end{array}\right\} i = 2, \ldots, k-1, j \in \mathcal{I}_i$$

(19)

Linear program

Problems:

- Not efficient enough
- We need a way to compute the lower and upper bounds for the ReLU relaxations

$$
\begin{aligned}
&\underset{\hat{z}_k}{\text{minimize}}\, c^T \hat{z}_k, \quad \text{subject to} \\
&\hat{z}_{i+1} = W_i z_i + b_i, \; i = 1, \ldots, k-1 \\
&z_1 \leq x + \epsilon \\
&z_1 \geq x - \epsilon \\
&z_{i,j} = 0, \; i = 2, \ldots, k-1, j \in \mathcal{I}_i^- \\
&z_{i,j} = \hat{z}_{i,j}, \; i = 2, \ldots, k-1, j \in \mathcal{I}_i^+ \\
&\left.\begin{aligned}
&z_{i,j} \geq 0, \\
&z_{i,j} \geq \hat{z}_{i,j}, \\
&\Big((u_{i,j} - \ell_{i,j}) z_{i,j} \\
&- u_{i,j} \hat{z}_{i,j}\Big) \leq -u_{i,j} \ell_{i,j}
\end{aligned}\right\} i = 2, \ldots, k-1, j \in \mathcal{I}_i
\end{aligned}
\tag{19}
$$

Efficiency from the dual network

“Crucially, we show that the feasible set of the dual problem can itself be expressed as a deep network, and one that is very similar to the standard backprop network.”

Efficiency from the dual network

Theorem 1. *The dual of (4) is of the form*

$$\begin{array}{ll} \underset{\alpha}{\text{maximize}} & J_\epsilon(x, g_\theta(c, \alpha)) \\ \text{subject to} & \alpha_{i,j} \in [0,1],\ \forall i,j \end{array} \tag{5}$$

where $J_\epsilon(x,\nu)$ is equal to

$$-\sum_{i=1}^{k-1} \nu_{i+1}^T b_i - x^T \hat{\nu}_1 - \epsilon \|\hat{\nu}_1\|_1 + \sum_{i=2}^{k-1} \sum_{j \in \mathcal{I}_i} \ell_{i,j} [\nu_{i,j}]_+ \tag{6}$$

and $g_\theta(c,\alpha)$ is a k layer feedforward neural network given by the equations

$$\begin{aligned} \nu_k &= -c \\ \hat{\nu}_i &= W_i^T \nu_{i+1}, \quad \text{for } i = k-1, \ldots, 1 \\ \nu_{i,j} &= \begin{cases} 0 & j \in \mathcal{I}_i^- \\ \hat{\nu}_{i,j} & j \in \mathcal{I}_i^+ \\ \frac{u_{i,j}}{u_{i,j} - \ell_{i,j}} [\hat{\nu}_{i,j}]_+ - \alpha_{i,j} [\hat{\nu}_{i,j}]_- & j \in \mathcal{I}_i, \end{cases} \\ & \qquad \text{for } i = k-1, \ldots, 2 \end{aligned} \tag{7}$$

Efficiency from the dual network

The "dual network" from (7) in fact is almost identical to the backpropagation network, except that for nodes j in $\mathcal{I}_i$ there is the additional free variable $\alpha_{i,j}$ that we can optimize over to improve the objective. In practice, rather than optimizing explicitly over α, we choose the fixed, dual feasible solution

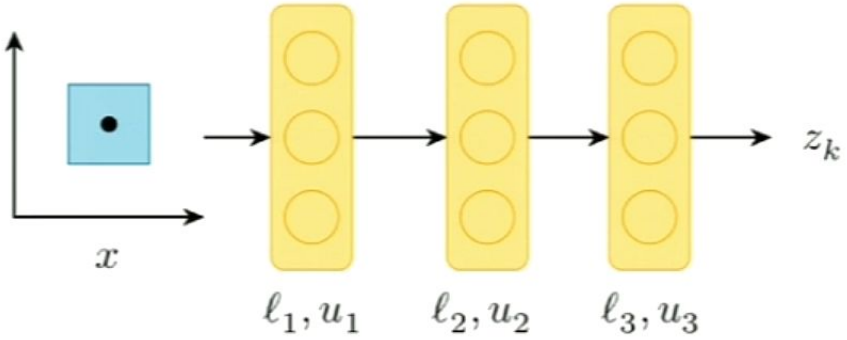
$$\alpha_{i,j} = \frac{u_{i,j}}{u_{i,j} - \ell_{i,j}}. \tag{8}$$

Finding bounds for ReLU relaxation

A meaningful bound requires good lower and upper bounds ℓ_i, u_i, but these just requires solving a similar optimization problem

$$\ell_{i',j} = \min_z \; e_j^T z_k$$
$$\text{subject to} \quad \|z_1 - x\|_\infty \leq \epsilon$$
$$(z_{i+1}, W_i z_i + b_i) \in \mathcal{C}(\ell_i, u_i)$$

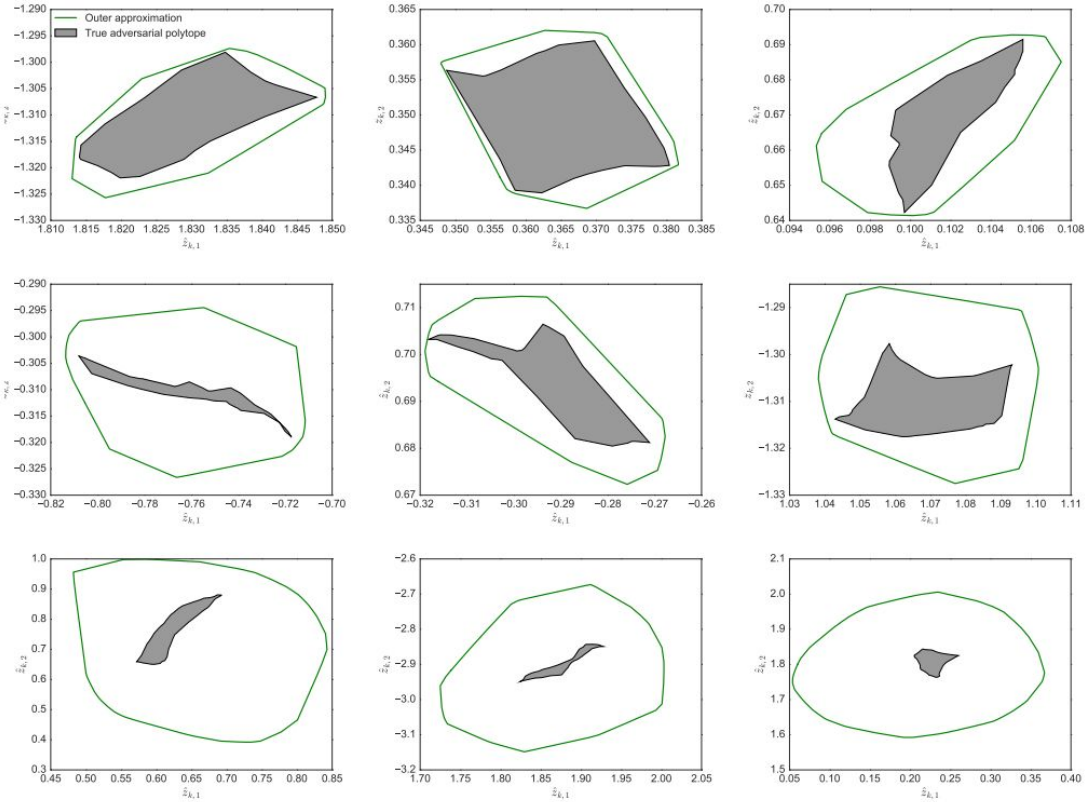
Incrementally build bounds by solving LP for each activation (requires some tricks like using a particular value for dual variables α)



Results

Problem	Robust	ϵ	Test error	FGSM error	PGD error	Robust error bound
MNIST	$\times$	0.1	1.07%	50.01%	81.68%	100%
MNIST	$\surd$	0.1	1.80%	3.93%	4.11%	5.82%
Fashion-MNIST	$\times$	0.1	9.36%	77.98%	81.85%	100%
Fashion-MNIST	$\surd$	0.1	21.73%	31.25%	31.63%	34.53%
HAR	$\times$	0.05	4.95%	60.57%	63.82%	81.56%
HAR	$\surd$	0.05	7.80%	21.49%	21.52%	21.90%
SVHN	$\times$	0.01	16.01%	62.21%	83.43%	100%
SVHN	$\surd$	0.01	20.38%	33.28%	33.74%	40.67%

Analysis



Extensions

“Training for Faster Adversarial Robustness Verification Via Inducing ReLU Stability” [Xiao et al. ICLR 2019]

- Directly train the networks to have more stable ReLUs that rarely cross zero
- Observe a 4-13x speedup in verification times

Similarity to SDP approach

- Unlike SDP approach, the polytope approach scales well with the number of classes.
- The bound from each method does not certify a network trained with the other method.

Network	PGD error	SDP bound	LP bound
SDP-NN	15%	35%	99%
LP-NN	22%	93%	26%

- The SDP approach attains better PGD error for a fixed network size, but a higher provable error.

Citations

- [1]: https://jtsulliv.github.io/perceptron/
- [2]: https://commons.wikimedia.org/wiki/File:Convex_polygon_illustration1.svg