Adversarial Attacks

FGSM

Targeted: $x' := x - \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}_{\text{target}}(x))$

Untargeted: $x' := x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}_{\text{label}}(x))$

Carlini-Wagner (Minimize Perturbation)

Opt. Prob.: `find` η `minimize` $\|\eta\|_p$ `s.t.` $f(x+\eta) = t, x+\eta \in [0,1]^n$

Relaxed: `find` η `minimize` $\|\eta\|_p + c \cdot \text{obj}_t(x+\eta)$ `s.t.` $x+\eta \in [0,1]^n$

With $\text{obj}_t(x+\eta) \le 0 \Rightarrow f(x+\eta) = t$ (e.g., $\mathcal{L}_t(x) - 1 = -\log_C(p(x)_t) - 1$ or $\max(0, 0.5 - p(x)_t)$)

When using L_∞, gradient of $\|\eta\|_\infty$ is zero at all non-max entries → use $L(\eta) = \sum_i \max(0, |\eta_i - \tau|)$ instead; Start with $\tau = 1$, update η K times, if $L(\eta) = 0$ decrease τ and repeat, otherwise stop and return previous η.

PGD

def PGD$(x, y, k, \epsilon_{\text{step}}, \epsilon)$
- $x' \leftarrow x + \eta$ for random η with $\|\eta\|_\infty \le \epsilon$
- **for** $i = 1, \ldots, k$ **do**
  - $g \leftarrow \nabla_{x'} \mathcal{L}(f(x'), y)$ ▷ *uFGSM(x′, y)*
  - $x' \leftarrow x' + \epsilon_{\text{step}} \cdot \text{sign}(g)$ ▷ *uFGSM(x′, y)*
  - $x' \leftarrow x + \max(\min(x' - x, \epsilon), -\epsilon)$
  - Clip x' to input domain ▷ *e.g.,* $[0,1]^n$
- **return** x'

For general norm $\|\cdot\|$, use

$x' \leftarrow x' + \epsilon_{\text{step}} \cdot \frac{g}{\|g\|}$ ▷ *dir of g, not sign*

if $\|x' - x\|_p > \epsilon$ **then**
- $x' \leftarrow x + \epsilon \frac{x'-x}{\|x'-x\|}$

Diffing Networks

$\text{obj}_t(x) := f(x)_t - g(x)_t$ (or abs. diff. of prob.)

def DIFF_NETS(f, g, ϵ)
- Select x classified as t by both f and g
- **while** $\text{class}(f(x)) = \text{class}(g(x))$ **do**
  - $x \leftarrow x + \epsilon \cdot \nabla_x \text{obj}_t(x)$ ▷ *Make f more confident about t while making g less confident*
- **return** x

Adversarial Defenses

Adversarial accuracy: check if data point in test set is classified correctly *and* network is robust in region around point (e.g., using PGD in ϵ L_∞-ball). Often have tradeoff with standard accuracy.

Opt. Prob.: $\underset{\theta}{\text{argmin}} \underset{(x,y)\sim D}{\mathbb{E}} \left[\max_{x' \in S(x)} \mathcal{L}(\theta; x', y) \right]$

PGD training

1. Select mini-batch B
2. $B_{\max} \leftarrow \{\underset{x' \in S(x)}{\text{argmax}} \mathcal{L}(\theta; x', y) | x \in B\}$
3. $\theta \leftarrow \theta - \frac{1}{|B_{\max}|} \sum_{(x,y)\in B_{\max}} \nabla_\theta \mathcal{L}(\theta; x, y)$

TRADES

$$\underset{\theta}{\text{argmin}} \underset{(x,y)\sim D}{\mathbb{E}} \left[\mathcal{L}(\theta; x, y) + \lambda \cdot \max_{x' \in S(x)} \mathcal{L}(\theta; x', f_\theta(x)) \right]$$

Certification

Soundness When a property is violated, if the method terminates, it always states that the property is violated

Completeness When a property holds, the method is able to prove it

Incompleteness When a property holds, the method might not be able to prove it (e.g. bounds propagation with convex relaxations)

Box Abstract Transformers

- $[a,b] +^\sharp [c,d] = [a+c, b+d]$
- $-^\sharp [a,b] = [-b, -a]$
- $ReLU^\sharp([a,b]) = [ReLU(a), ReLU(b)]$
- $\lambda \cdot^\sharp [a,b] = [\lambda \cdot a, \lambda \cdot b]$ for $\lambda \ge 0$
- $[a,b] \cdot^\sharp [c,d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bc)]$

Certification (Complete Methods)

MILP (complete for ReLU, NP-complete)

Objective $\min_{x_1,\ldots,x_n} c_1 x_1 + \ldots + c_n x_n$

Constraints $a_{i,1} x_1 + \ldots + a_{i,n} x_n \le b_i$, $1 \le i \le m$

Bounds $x_j \in [l_j, u_j]$ or $x_j \in \mathbb{Z}$, $1 \le j \le n$

Affine layer $y = Wx + b$

ReLU layer
$$\left.\begin{array}{l} y \le x - l(1-a) \\ y \le ua \\ y \ge x \\ y \ge 0 \\ a \in \{0,1\} \end{array}\right\} \begin{array}{r} x \in [l,u] \\ a = 0 \Rightarrow (y = 0 \\ \wedge\ x \in [l, 0]) \\ a = 1 \Rightarrow (y = x \\ \wedge\ x \in [0, u]) \end{array}$$

Bounds $x_i \in [l_i, u_i]$ (precomputed box bounds for neurons) or $x'_i \in [x_i - \epsilon, x_i + \epsilon]$ (inputs)

Objective $\min o_{\text{label}} - o_i$ (verification succeeds iff $o_{\text{label}} > o_i$)

Zonotope

Zonotope $\hat{x}_j = a_0^j + \sum_{i=1}^k a_i^j \epsilon_i$, $\epsilon_i \in [-1,1]$

Symmetry center is a_0, $2 \cdot a_0 - X$ is X flipped

Affine Zonotope is linear

ReLU Enumerate $\epsilon_i \in [-1,1]$ to find l_x, u_x. If $u_x \le 0$, $\hat{y} = 0$. If $l_x \ge 0$, $\hat{y} = \hat{x}$. Otherwise, $\hat{y} = \lambda \hat{x} - \epsilon_{\text{new}} \frac{\lambda l_x}{2} - \frac{\lambda l_x}{2}$, $\lambda := \frac{u_x}{u_x - l_x}$

DeepPoly

Def $x_i \in [l_i, u_i]$, $a_i^\le \le x_i \le a_i^\ge$, $a_i^\cdot = \sum_j w_j x_j + v$

Complexity Affine: $\mathcal{O}((\#\text{layers})(\max \#\text{neurons})^2)$, ReLU: $\mathcal{O}(1)$

ReLU

Condition	$a_j^\le$	$a_j^\ge$	l_j	u_j
$u_i \le 0$	0	0	0	0
$l_i \ge 0$	x_i	x_i	l_i	u_i
$l_i < 0 < u_i$	λx_i	$u_i \frac{x_i - l_i}{u_i - l_i}$	λl_i	u_i

For any $\lambda \in [0,1]$

Area heuristic Choose $\lambda = 0$ if $u \le -l$, else $\lambda = 1$

Backsub After affine layer, repeatedly backsubstitute (for l, choose $a^\le$ for $w \ge 0$, $a^\ge$ otherwise)

Verification Backsubstitute $x_{\text{label}} - x_i$ to find lower bound, verifies iff > 0

Abstract Interpretation

Soundness $\forall z \in \mathbb{A}. F(\gamma(z)) \subseteq \gamma(F^\sharp(z))$

Exactness $\forall z \in \mathbb{A}. F(\gamma(z)) = \gamma(F^\sharp(z))$

Optimality $\forall z. \forall F^\sharp$ sound.$\gamma(F^\sharp(z)) \not\subset \gamma(F^\sharp_{\text{best}}(z))$

Domain	Affine Transformer	ReLU Transformer
Box	not exact optimal	not exact optimal
Zonotope	exact optimal	not exact no optimal trafo
DeepPoly	exact optimal	not exact no optimal trafo

Certified Defenses

Train networks to be provably robust (instead of experimentally result as in PGD training).

Opt.: $\underset{\theta}{\operatorname{argmin}} \underset{(x,y)\sim D}{\mathbb{E}} \left[\max_{z\in\gamma(\mathrm{NN}^\sharp(S(x)))} \mathcal{L}(\theta; z, y) \right]$

Loss $\mathcal{L}(z,y) := \max_{q\neq y}(z_q - z_y) = \max_{q\neq y}(\mathrm{box}(z_q - z_y))$

CE loss $\mathcal{L}(z,y) = \mathrm{CE}(z', y)$, with $z'_y := l_y$, $z'_q := u_q$ for $q \neq y$

Universal Approximation For any neural network, there exists a network with the same properties that can be analyzed exactly with Box.

Complexity Using complex relaxations generally leads to worse results in provability than with Box (more complex optimization problem)

COLT For each layer, find $x_l \in S_l$ that maximizes loss in final layer, and use it.

COLT projection Write zonotope as $Z = A \cdot [-1,1]^d$, compute $e = A^{-1} \cdot x$, clip e to $[-1,1]$, projection is $A \cdot e_{\text{clip}}$. This projection is *sound* (result inside zonotope), but *not optimal*.

Certified Robustness to Geometric Trafo

Rotation
$T_\phi(x,y) = (x\cos(\phi) - y\sin(\phi), x\sin(\phi) + y\cos(\phi))$

Translation $T_\delta(x,y) = (x + \delta_x, y + \delta_y)$

Scaling $T_\lambda(x,y) = (\lambda x, \lambda y)$

Interpolation To compute pixel value $I_\kappa(x,y)$ after T_κ: compute pre-image $T_\kappa(x,y)$, then interpolate.

Domain $w_l^T\kappa + b_l \leq I_\kappa(x,y) \leq w_u^T + b_u$, for all κ

Tightness

$$L(w_l, b_l) := \int_{\kappa\in D} \left(I_\kappa(x,y) - (w_l^T\kappa + b_l) \right) d\kappa$$

$$U(w_u, b_u) := \int_{\kappa\in D} \left((w_u^T\kappa + b_u) - I_\kappa(x,y) \right) d\kappa$$

Optimization

- $L(w_l, b_l) \approx \frac{1}{N}\sum_{i=1}^N \left(I_\kappa(x,y) - (w_l^T\kappa^i + b_l) \right)$
- $w_l^T\kappa^i + b_l \leq I_{\kappa^i}(x,y)$, $1 \leq i \leq N$

→ Solve in poly-time with linear programming.

Soundness Find upper bound δ on violation:

$$(w_l^T\kappa^i + b_l) - I_\kappa(x,y) \leq \delta_l \quad \forall\kappa\in D$$
$$I_\kappa(x,y) - (w_u^T\kappa + b_u) \leq \delta_u \quad \forall\kappa\in D$$

Using $b_l - \delta_l$, $b_u + \delta_u$ is sound.

→ **Box** Use Box to compute upper bound on $f(\kappa)$

→ **MVTh** if $|\partial_i f(\kappa')| \leq |L_i| \forall\kappa' \in D = [h_l, h_u]$

$f(\kappa) = f(\kappa_C) + \nabla f(\kappa')^T(\kappa - \kappa_C) \leq$
$f(\kappa_C) + |L|^T(\kappa - \kappa_C) \leq f\left(\frac{h_u+h_l}{2} + |L|^T\frac{(h_u-h_l)}{2}\right)$

Visualization

Opt.: $\operatorname{argmin}_x \mathrm{score}(x) - \sum \lambda_i \mathrm{Regularizer}_i(x)$, $\mathrm{score}(x) := \mathrm{mean}(\mathrm{layer}_l[x])$ (using GD)

Gradient Feature Attribution $\nabla_x \mathrm{logit}_t(x)$

Shapley Values

$$C_i := \sum_{S\subseteq P\setminus\{i\}} \frac{|S|!(|P|-|S|-1)!}{|P|!} \left(f(S\cup\{i\}) - f(S)\right)$$

Need to define $f(S)$ (e.g., set pixels not in S to 0)

Prop. $\sum_i C_i = f(P)$

Robustness Robust NN rely on more robust features, more aligned with human perception

Logic & Deep Learning

Querying

Optimization: $\forall x.T(\phi)(x) = 0 \Leftrightarrow x$ satisfies ϕ

ϕ	$T(\phi)$ (non-negative)
$t_1 \leq t_2$	$\max(0, t_1 - t_2)$
$t_1 \neq t_2$	$[t_1 = t_2]$
$t_1 = t_2$	$T(t_1 \leq t_2 \wedge t_2 \leq t_1$
$t_1 < t_2$	$T(t_1 \leq t_2 \wedge t_1 \neq t_2)$
$\phi \vee \psi$	$T(\phi)\cdot T(\psi)$
$\phi \wedge \psi$	$T(\phi) + T(\psi)$

Box Box constraints are encoded separately (e.g., w/ L-BFGS-B optimizer)

Counter-example Use optimization to find counter-examples to a given property.

Training with Background Knowledge

1. $\underset{\theta}{\operatorname{argmax}} \underset{s\sim D}{\mathbb{E}} [\forall z.\phi(z,s,\theta)]$ (find θ s.t. exp. val. of property increases)
2. $\underset{\theta}{\operatorname{argmin}} \underset{s\sim D}{\mathbb{E}} \left[\max_z \neg\phi(z,s,\theta)\right]$ (find θ s.t. max. violation of ϕ is minimized)
3. $\underset{\theta}{\operatorname{argmin}} \underset{s\sim D}{\mathbb{E}} [T(\phi)(z_{\text{worst}}, s, \theta)]$, $z_{\text{worst}} := \underset{z}{\operatorname{argmin}}\, T(\neg\phi)(z,s,\theta)$ (find worst possible violation of ϕ, then find θ that minimizes its effect)

To solve inner opt. problem efficiently, split into objective and (efficient) projections on convex set (e.g., L_∞-ball)

Randomized Smoothing

Given classifier $f: \mathbb{R}^d \to \mathcal{Y}$, construct smoothed classifier g as $g(x) := \underset{c\in\mathcal{Y}}{\operatorname{argmax}} \underset{\epsilon}{\mathbb{P}}[f(x+\epsilon) = c]$ where $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{1})$

Robustness If $\mathbb{P}[f(x+\epsilon) = c_A] \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c\neq c_A} \mathbb{P}[f(x+\epsilon) = c]$, then $g(x+\delta) = c_A$ for all $\|\delta\|_2 < R := \frac{\sigma}{2}\left(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})\right)$

Certification $\underline{p_A}$ approximate with sampling: if $\underline{p_A} > 0.5$, return radius $\sigma\Phi^{-1}(\underline{p_A})$, otherwise abstain.

Certified Accuracy Pick target radius T, count #points in test set with certified radius $R > T$ and where predicted label matches test set label (Standard accuracy: $T = 0$)

Inference Reject null hypothesis (true prob. of f returning $\hat{c_A}$ is 0.5, i.e., classes are indistinguishable) if estimated p-value $\leq \alpha$, else abstain.
→ returns wrong class $\hat{c_A} \neq c_A$ with prob. $\leq \alpha$

Generalized Smoothing
$g(x) = \operatorname{argmax}_{c\in\mathcal{Y}} \mathbb{P}_\epsilon[f(\psi_\epsilon(x)) = c]$, $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{1})$, $\psi_\alpha(\psi_\beta) = \psi_{\alpha+\beta}$ (e.g., instantiate ψ with geometric transformations).

Summary

- Scales to large networks;
- Relaxes deterministic guarantees into statistical guarantees on robustness;
- May need many samples to obtain higher certified radius. Also requires sampling at inference time;
- Generalizing smoothing to different properties is harder than convex methods.

General

$\leq$	$\cdot\|x\|_1$	$\cdot\|x\|_2$	$\cdot\|x\|_\infty$
$\|x\|_1 = \sum_{i=1}^d \|x_i\|$		$\sqrt{d}$	d
$\|x\|_2 = \sum_{i=1}^d \|x_i\|^2$	1		$\sqrt{d}$
$\|x\|_\infty = \max_{1\leq i\leq d}\|x_i\|$	1	1	

$\mathbb{B}_1^1 \subseteq \mathbb{B}_1^2 \subseteq \mathbb{B}_1^\infty \subseteq \mathbb{B}_{\sqrt{d}}^2 \subseteq \mathbb{B}_d^1$ (Classifier safe for L_2-radius of $\epsilon \Rightarrow$ safe for L_∞-radius of $\frac{\epsilon}{\sqrt{d}}$)

$\mathrm{CE}(p,y) = -\sum_i y_i \cdot \log(p_i)$
$(= -\log(p_{\text{lbl}}) = -\log(o_{\text{lbl}}) + \log(Z)$ for single label)