Disclaimer

This document is an exam summary that follows the slides of the *Reliable and Interpretable Artificial Intelligence* lecture at ETH Zurich. The contribution to this is a short summary that includes the most important concepts, formulas and algorithms. This summary was created during the fall semester 2020. Due to updates to the syllabus content, some material may no longer be relevant for future versions of the lecture.

I do not guarantee correctness or completeness, nor is this document endorsed by the lecturers. Feel free to point out any erratas. For the full LaTeX source code, consider `github.com/ymerkli/eth-summaries`.

1 Basics

1.1 Good to know

$||x||_p = \left(\sum_{i=1}^d |x_i|^p\right)^{\frac{1}{p}}$ $\quad ||x||_\infty = \max_{i\in\{1,..,d\}} |x_i|$

Softmax $\sigma(z)_i = e^{z_i} / \sum_{j=1}^D e^{z_j}$

CE loss: $CE(\vec{z}, y) = -\sum_{c=1}^K \mathbb{1}[c=y]\cdot \log z_c$

Implication: $\phi \implies \psi \iff \neg\phi \vee \psi$

Mean value: $f(y) = f(x) + \nabla f(z)^T(y-x)$

Gauss: $\mathcal{N} = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$

CDF: $\Phi(v;\mu,\sigma^2) = \int_{-\infty}^{v} \mathcal{N}(y;\mu,\sigma^2)dy = \Phi(\frac{v-\mu}{\sqrt{\sigma^2}};0,1)$

Line through 2 points: $(a,f(a)),(b,f(b))$, $a<b$: $l(x) = \frac{f(b)-f(a)}{b-a}\cdot x + f(a) - \frac{f(b)-f(a)}{b-a}\cdot a$

Tangent at $(t,f(t))$: $t(x) = f'(t)\cdot x + f(t) - f'(t)\cdot t$

Subadditivity of $\sqrt{\cdot}$: $\sqrt{x+y} \le \sqrt{x} + \sqrt{y}$

Cauchy Schwarz: $\langle x,y\rangle \le ||x||_2 \cdot ||y||_2$

1.2 Abstract Interpretation

Symbolic shape z semantically represents a set of concrete values $x = \gamma(z)$. Concretization γ: defines the concrete values an abstract element z represents. Concrete transformer $F(\cdot)$ ($F(x)$ generally uncomputable), abstract transformer $F^\#(\cdot)$, $F^\#(z)$ applies abstract transformer to z.
Soundness: $F^\#$ needs to over-approximate F: $\gamma(F^\#(x)) \supseteq F(x) = F(\gamma(z))$

$F^\#$ soundness: $\forall z: F(\gamma(z)) \subseteq \gamma(F^\#(z))$

$F^\#$ exactness: $\forall z: F(\gamma(z)) = \gamma(F^\#(z))$

$F^\#_{best}$ optimality: $\forall z: \gamma(F^\#(z)) \not\subset \gamma(F^\#_{best}(z))$.

2 Adversarial Attacks

T-FGSM: $x' = x - \eta,\ \eta = \epsilon\cdot sign(\nabla_x loss_t(x))$

U-FGSM: $x' = x + \eta,\ \eta = \epsilon\cdot sign(\nabla_x loss_s(x))$

Guarantees $\eta \in [x-\epsilon, x+\epsilon]$, η not minimized.

C&W: Find adv sample $x' = x + \eta \in [0,1]^n$ *and* minimize $||\eta||_p$ via relaxation s.t. $f(x') = t$. obj_t: $obj_t(x+\eta) \le 0 \Leftrightarrow f(x+\eta) = t$.
Minim. $||\eta||_p + c\cdot obj_t(x+\eta)$ s.t. $x+\eta \in [0,1]^n$
E.g. $obj_t = \{CE(x',t) - 1; max(0, 0.5 - p_f(x')_t)\}$
$\nabla_\eta ||\eta||_p$ is suboptimal $\rightarrow$ use proxy
l_∞: proxy $L(\eta) = \sum_i max(0, |\eta_i| - \tau)$. Iteratively decrease τ until $L(\eta) > 0$. Then do GD on η: $\eta = \eta - \gamma\nabla_\eta(L(\eta) + c\cdot obj_t(x+\eta))$ until $L(\eta) = 0$, then anneal τ and continue loop.
Constraint $\eta_i \in [-x_i, 1-x_i]$: LBFGS-B, PGD

PGD: Iterative FGSM with projection to find point in $x_o \pm \epsilon$ that max. loss.
1. Init $x' = x + \epsilon\cdot rand[-1,1]$;
2. Repeat: $x' \leftarrow x' + \epsilon_{step}\cdot sgn(\nabla_{x'} loss_s(x'))$ (untargeted) or $x' \leftarrow x' - \epsilon_{step}\cdot sgn(\nabla_{x'} loss_t(x'))$ (targeted); $x' = project(x', x_o, \epsilon)$;

Diffing Networks: find x that diffs 2 networks while($class(f_1(x)) = class(f_2(x))$):
$x = x + \epsilon\nabla_x obj_t(x)$; // $obj_t(x) = f_1(x)_t - f_2(x)_t$

3 Adversarial Defenses

Defense as Optimization: $\min_\theta \rho(\theta)$, $\rho(\theta) = \mathbb{E}_{(x,y)\sim D}\left[\max_{x'\in S(x)} L(\theta, x', y)\right]$, $S(x) = \{x' : ||x - x'||_\infty \le \epsilon\}$

PGD Defense in practice:
1. Select mini batch B from D **2.** $x_{max} = \arg\max_{x'\in S(x)} L(\theta, x', y), \forall(x,y)\in B$
3. $\theta' = \theta - \frac{1}{|B_{max}|}\sum_{(x_{max},y)\in B_{max}} \nabla_\theta L(\theta, x_{max}, y)$

TRADES defense:
$\min_\theta \mathbb{E}_{(x,y)\sim D}[L(\theta,x,y) + \lambda \max_{x'\in\mathbb{B}_\epsilon(x)} L(\theta, x', f_\theta(x))]$

4 Certification of Neural Networks

Given NN N, pre-condition ϕ, post-condition ψ prove: $\forall i \in I : i \vDash \phi \implies N(i) \vDash \psi$ or return a violation.

4.1 Incomplete Methods:

Over-approx. ϕ using relaxation, then push approx. through NN via bound propagation.

Box: $\hat{x_i} = [l,u]$, i.e. $l \le x_i \le u$. $AT^\#$: $[a,b] +^\# [c,d] = [a+b, c+d]$; $-^\#[a,b] = [-b,-a]$; $ReLU^\#[a,b] = [ReLU(a), ReLU(b)]$; $\lambda \cdot^\# [a,b] = [\lambda a, \lambda b]$ $(\lambda > 0)$

Zonotope: $\hat{x_j} = a_0^j + \sum_{i=1}^k a_i^j\epsilon_i, \forall j \in [d]$ $\epsilon_i \in [-1,1]$ shared among d abstract neurons. $[\hat{x_1},..,\hat{x_d}]$ describes Zonotope $\mathcal{Z}$ centered at $\vec{a}_0 = [a_0^1,..,a_0^d]$.
Centering: Flip $X \in \mathcal{Z}$ around $\vec{a}_0$ to flipped point $Y = 2\vec{a}_0 - X$. $Y \in \mathcal{Z}$ and $|X - \vec{a}_0| = |Y - \vec{a}_0|$. $\mathcal{Z}$ is point-symmetric around $\vec{a}_0$.
$AT^\#$: Affine$^\#$ is exact
- Affine$^\#$: $(a_0^p + \sum_{i=1}^k a_i^p\epsilon_i) + (a_0^q + \sum_{i=1}^k a_i^q\epsilon_i) = (a_0^p + a_0^q) + \sum_{i=1}^k (a_i^p + a_i^q)\epsilon_i$
$C\cdot(a_0^p + \sum_{i=1}^k a_i^p\epsilon_i) = C\cdot a_0^p + \sum_{i=1}^k C\cdot a_i^p\epsilon_i$
- ReLU$^\#(\hat{x})$: **1)** Box bounds $\hat{x} = a_0 + \sum_{i=1}^k a_i\epsilon_i \in [l_x, u_x]$. $\mathbf{l_x}$: $\epsilon_i = -1$ if $a_i \ge 0$, $\epsilon_i = +1$ if $a_i < 0$. $\mathbf{u_x}$: $\epsilon_i = +1$ if $a_i \ge 0$, $\epsilon_i = -1$ if $a_i < 0$.
2) $u_x \le 0$: $\hat{y} = 0$; $l_x > 0$: $\hat{y} = \hat{x}$; $l_x < 0, u_x > 0$: $\hat{y}(\hat{x}) = b_0 + \sum_{i=1}^{k+1} b_i\epsilon_i$, $b_0 = \lambda(a_0 - l_x/2)$, $b_i = \lambda a_i, \forall i \in [1,k]$, $b_{k+1} = -\lambda l_x/2$, with $\lambda = u_x/(u_x - l_x)$
$\rightarrow$ error terms increase as NN depth increases!
Note: there are many non-comparable ReLU abstract transformers for Zonotope. The one discussed here is optimal area-wise.

DeepPoly: For each x_i keep:
- interval constraints $l_i \le x_i$, $x_i \le u_i$
- relational constraints: $a_i^\le \le x_i$, $x_i \le a_i^\ge$ where $a_i^\le, a_i^\ge$ are of the form $\sum_j w_j\cdot x_j + \nu$

$AT^\#$: Affine$^\#$ is exact
-Affine$^\#$: rel: $\sum_j w_j^p\cdot x_j + \nu^p + \sum_j w_j^q\cdot x_j + \nu^q = \sum_j (w_j^p + w_j^q)\cdot x_j + (\nu^p + \nu^q)$;
int: backsubstitution up to some layer. Then replace neurons of that layer with its correct interval constraint (like in Zonotope Box bounds). Backsub: sum up all components!
- $x_j = \text{ReLU}^\#(x_i)$: interval constr. $x_i \in [l_i, u_i]$:
$u_i \le 0$: $a_j^\le = a_j^\ge = 0, l_j = u_j = 0$;
$l_i \ge 0$: $a_j^\le = a_j^\ge = x_i, l_j = l_i, u_j = u_i$;
$l_i < 0, u_i > 0$: $\lambda = u_i/(u_i - l_i)$,
$\mathbf{u_i \le -l_i}$: $a_j^\le = 0$, $a_j^\ge = \lambda x_i - \lambda l_i$, $x_j \in [0, u_i]$;
$\mathbf{u_i > -l_i}$: $a_j^\le = x_i$, $a_j^\ge = \lambda x_i - \lambda l_i$, $x_j \in [l_i, u_i]$;

Symbolic bound: when proving $y_2 > y_1$, use abstract shape of $y_2 - y_1$ and prove $l_{y_2 - y_1} > 0$

4.2 Complete Methods

Encode NN as MILP instance.
- Affine: $y = Wx + b$ direct MILP constraint.
- $ReLU(x)$: $y \le x - l_x\cdot(1-a)$, $y \ge x$, $y \le u_x\cdot a$, $y \ge 0$, $a \in \{0,1\}$, for neuron bound $x \in [l,u]$.
  - $a = 0$: $y = 0, x \in [l, 0]$
  - $a = 1$: $y = x, y \in [0, u]$
- $\phi = B_\epsilon^\infty(x)$: $x_i - \epsilon \le x_i' \le x_i + \epsilon, \forall i$
- precomp. Box bounds: $l_i \le x_i^p \le u_i$
- $\psi = o_0 > o_1$: MILP objective $\min o_0 - o_1$. If $\min o_0 - o_1 > 0$, ψ holds.

5 Certified Defenses

5.1 DiffAI

DiffAI Certified PGD Defense: minimize $\rho(\theta) = \mathbb{E}_{(x,y)\sim D}\left[\max_{z\in\gamma(NN^\#(S(x)))} L(\theta, z, y)\right]$

Find output z in concretization $\gamma(NN^\#(S(x)))$ of abstract NN output $NN^\#(S(x))$ that maximizes loss L. Can use any abstract transformer (Box, Zonotope, DeepPoly,...). Essentially automatic differentiation of abstract interpretation.

To find max loss, use abstract loss $L^\#(\vec{z}, y)$, where y = target label, $\vec{z}$ = vector of logits:
- $L(z,y) = max_{q\ne y}(z_q - z_y)$: Compute $d_c = z_c - z_y \forall c \in \mathcal{C}$ where $\mathcal{C}$ set of classes and z_c the abstract logit shape of class i. Then compute box bounds of d_c and compute max upper bound: $\max_{c\in\mathcal{C}}(\max(box(d_c)))$
- $L(z,y) = CE(z,y)$: Compute box bounds $[l_c, u_c]$ of logit shapes z_c. $\forall c \in \mathcal{C}$ pick u_c if $c \ne y$, pick l_c if $c = y$. Then apply softmax to vector $v = [u_0, u_1, .., l_c, .., u_{|\mathcal{C}|}]$ and compute $CE(v', y)$ with $v' = softmax(v)$.

Cheap relaxations (box) scale but introduce lots of infeasible points: substantial drop in standard accuracy. BUT, more precise relaxations (Zonotope) do not actually bring better results in provabililty. Hypothesis: more complex abstractions lead to more difficult optimization problems. $\rightarrow$ bridge the gap between adversarial training and certified defenses..

5.2 COLT

COLT: PGD training with intermediate NN layer shapes S_i. Iterate over layers h_i and find weights θ for layers $h_{i+1},..,h_D$ that minimize the worst-case loss of $x_i \in S_i$. Weights of previous layers $h_1,..,h_i$ are frozen.
$\min_\theta \max_{x_i\in S_i} L(h_D(h_{D-1}(...h_{i+1}(x_i))), y_{true})$
The inner maximization requires projections: projections on Zonotope solved via projection on Box (via change of variables):
$\rightarrow \max_{x\in Z} L(x, y_{true}) = \max_{\epsilon\in[-1,1]^k} L(A\cdot\epsilon, y_{true})$
Each $x \in Z$ has a $\epsilon \in [-1,1]^k$ s.t. $x = A\epsilon$. Instead of projecting $x' \notin Z$ onto Z, write $x' = A\epsilon'$ and project ϵ' onto $[-1,1]^k$ by $clip(\epsilon', -1, 1)$.

6 Geometric Transformation Robustness

Many transformations preserve semantic meaning of the original image while not being covered by a small l_p ball.
Geom. transformation: $T_\kappa : \mathbb{R}^2 \rightarrow \mathbb{R}^2$. Rotation: $T_\phi(x,y) = (x\cos\phi - y\sin\phi, x\sin\phi + y\cos\phi)$.
Translation: $T_{\delta_x,\delta_y}(x,y) = (x+\delta_x, y+\delta_y)$.
Scaling: $T_\lambda(x,y) = (\lambda x, \lambda y)$

Bilinear Interpolation: $I : \mathbb{R}^2 \rightarrow \mathbb{R}^2$
$I(x,y) = p_{x_1,y_1}(x_2 - x)(y_2 - y)$
$+ p_{x_1,y_2}(x_2 - x)(y - y_1) + p_{x_2,y_1}(x - x_1)(y_2 - y)$
$+ p_{x_2,y_2}(x - x_1)(y - y_1)$; where $x_1 \le x \le x_2$, $y_1 \le y \le y_2$ and $x_2 = x_1 + 1, y_2 = y_1 + 1$

Pixel value (x,y) after T_κ given by $I_\kappa : \mathbb{R}^2 \rightarrow \mathbb{R}$, where $I_\kappa(x,y) = I \circ T_\kappa^{-1}(x,y)$

6.1 Certifying geometric robustness

Convex relaxation $C(R_\Phi(O))$ for exact region $R_\Phi(O)$. Use DeepPoly to get *sound, tight* lower and upper constraints *for each pixel*:
$\mathbf{w_l}\kappa + b_l \le I_\kappa(x,y) \le \mathbf{w_u}\kappa + b_u, \forall\kappa \in D$
Minimize Volumes via MC approx:
$L(\mathbf{w_l}, b_l) = \int_{\kappa\in D} I_\kappa(x,y) - (\mathbf{w_l^T}\kappa + b_l)d\kappa$
$\approx \frac{1}{N}\sum_{i=1}^N I_{\kappa^i} - (\mathbf{w_l^T}\kappa^\mathbf{i} + b_l)$
$U(\mathbf{w_u}, b_u) = \int_{\kappa\in D} (\mathbf{w_u^T}\kappa + b_u) - I_\kappa(x,y)d\kappa$
$\approx \frac{1}{N}\sum_{i=1}^N (\mathbf{w_u^T}\kappa^\mathbf{i} + b_u) - I_{\kappa^i}$
subject to finite set of constraints (LP solve)
$\mathbf{w_l^T}\kappa^\mathbf{i} + b_l \le I_{\kappa^i}(x,y) \le \mathbf{w_u^T}\kappa^\mathbf{i} + b_u, \forall i \in 1,..,N$
$\rightarrow$ can obtain over-approximations $\hat{w}_l, \hat{b}_l, \hat{w}_u, \hat{b}_u$
Problem: This is only sound at the finite sample points κ_i. Instead, find upper bounds δ_l, δ_u on the maximum soundness violation on entire D:
lb: $(\mathbf{\hat{w}_l^T}\kappa + \hat{b}_l) - I_\kappa(x,y) \le \delta_l, \forall\kappa \in D$
ub: $I_\kappa(x,y) - (\mathbf{\hat{w}_u^T}\kappa + \hat{b}_u) \le \delta_u, \forall\kappa \in D$
(Intuition: if $(\mathbf{\hat{w}_l^T}\kappa + \hat{b}_l)$ is unsound, then $\exists\kappa$: $\mathbf{\hat{w}_l^T}\kappa + \hat{b}_l > I_\kappa(x,y)$, similarly for ub)
Then, constraints $\mathbf{w_l} = \mathbf{\hat{w}}_l, b_l = \hat{b}_l - \delta_l$, $\mathbf{w_u} = \mathbf{\hat{w}_u}, b_u = \hat{b}_u + \delta_u$ are sound.

Bounding max. violation: compute upper bounds on $f_l(\kappa) = (\hat{\mathbf{w}}_{\mathbf{l}}^{\mathbf{T}}\kappa + \hat{b}_l) - I_\kappa(x,y)$, $f_u(\kappa) = I_\kappa(x,y) - (\hat{\mathbf{w}}_{\mathbf{u}}^{\mathbf{T}}\kappa + \hat{b}_u)$

- Run box propagation to obtain l, u s.t. $f(\kappa) \in [l, u] \to f(\kappa) \le u, \forall \kappa \in D$
- Mean-value theorem, Lipschitz continuity: $f(\kappa) = f(\kappa_c) + \nabla f(\kappa')^T(\kappa - \kappa_c) \le f(\kappa_c) + |L|^T(\kappa - \kappa_c) \le f(\frac{1}{2}(h_u + h_l)) + \frac{1}{2}|L|^T(h_u - h_l)$ with $|\partial_i f(\kappa')| \le |L_i|, \forall \kappa' \in D$, $D = [h_l, h_u]$. Bound L on $\nabla f(\kappa')$? propagate hyper-rectangle D through partial derivative $|\partial_i f(\kappa')$ using some convex relaxation.

In practice: branch D into multiple hyper-rectangles and bound each separately.

7 Visualization

Image similarity: neuron activation $R(x)$, if $||R(x_1) - R(x_2)||_2$ is small, then images x_1, x_2 are similar.

7.1 Feature Visualization

Invert NN, show parts of input causing high activation. Generally, deeper layers encode more complex patterns.

Feature Visualization by Optimization: Find input x that maximizes mean activation of a neuron/channel/layer while minimizing regularizers $R_i(x)$ (regularizers are crucial).

Maximize: $score(x) - \sum_i \lambda_i R_i(x)$ where $score(x) = mean(layer_n[x, y, z])$

7.2 Feature Attribution

Find areas of input that are responsible for classification.

Gradient based: $\frac{\partial logit_i(x)}{\partial x}$ shows how changing single pixel influences logit.

Shapley Values: input $\mathbf{x}$ has feature set P. How much does $i \in P$ contribute to $f(P)$?

$$C_i = \sum_{S \subseteq P \setminus \{i\}} \frac{|S|! \cdot (|P| - |S| - 1)!}{|P|!} [f(S \cup \{i\}) - f(S)]$$

Treat each pixel location as feature, instantiate f with logit of target class. Pixel not contained in S are set to a baseline value (zero, mean,..) $\sum_i C_i = f(x) - f(S = \emptyset)$

Combining Visualization & Attribution: 1) Compute activations in a conv layer 2) Cluster actications into k groups 3) For each group: 1. show which pixel affect if most by upscaling the activation maps (grouped activations) 2. show a visualization for the group of features.

7.3 Visualization and Adversarial Examples

On non-robust NN, gradient based attribution requires prior to enforce a *clean* saliency map. On robust NN, gradient is better aligned with human expectation (since features are better aligned).
$\to$ robust NN rely on different features (better aligned with human perception). Non-robust NN learn non-robust spurious features in the data that statistically are only weakly connected with the input.

8 Logic and Deep Learning (DL2)

8.1 Querying Neural Networks

Use standard logic ($\forall, \exists, \wedge, \vee, f : \mathbb{R}^m \to \mathbb{R}^n, ..$) and high-level queries to impose constraints.

$$(class(NN(i)) = 9) = \bigwedge_{j=1, j \neq 9}^{k} NN(i)[j] < NN(i)[9]$$

Use translation T of logical formulas into differentiable loss function $T(\phi)$ to be solved with gradient-based optimization to minimize $T(\phi)$.

Theorem: $\forall x, T(\phi)(x) = 0 \iff x \vDash \phi$

Logical Formula to Loss:

Logical Term	Translation
$t_1 \le t_2$	$\max(0, t_1 - t_2)$
$t_1 \neq t_2$	$[t_1 = t_2]$
$t_1 = t_2$	$T(t_1 \le t_2 \wedge t_2 \le t_1)$
$t_1 < t_2$	$T(t_1 \le t_2 \wedge t_1 \neq t_2)$
$\phi \vee \psi$	$T(\phi) \cdot T(\psi)$
$\phi \wedge \psi$	$T(\phi) + T(\psi)$

Translation is recursive and $T(\phi)(x) \ge 0, \forall x, \phi$

Box constraints: ineffective in GD. Use L-BFGS-B and give box constraints to optimizer.

8.2 Training NN with Background Knowledge

Incorporate logical property ϕ in NN training.

Problem statement: find θ that maximizes the expected value of property.

Maximize $\rho(\theta) = \mathbb{E}_{s \sim D}[\forall z. \phi(z, s, \theta)]$.

BUT: Universal quantifiers are difficult.

Reformulation: get the worst violation of ϕ and find θ that minimizes its effect.

minimize $\rho(\theta) = \mathbb{E}_{s \sim D}[T(\phi)(z_{worst}, s, \theta)]$

where $z_{worst} = \arg\min_z T(\neg\phi)(z, s, \theta)$

In practice, restrict z to a convex set with efficient projections (closed form). One can then remove the constraint from ϕ that restricts z on the convex set and do PGD while projecting z onto the convex set.

9 Randomized Smoothing for Robustness

Construct a classifier $\mathbf{g}$ from a classifier $\mathbf{f}$ s.t. $\mathbf{g}$ has certain statistical robustness guarantees.

Given base classifier $f : \mathbb{R}^d \to \mathcal{Y}$, construct smoothed classifier g (where $\epsilon \sim \mathcal{N}(, \sigma^2\mathbf{I})$):

$g(x) := \arg\max_{c \in \mathcal{Y}} \mathbb{P}_\epsilon(f(x + \epsilon) = c)$

Robustness Guarantee: suppose $c_A \in \mathcal{Y}$ (most likely class), $\underline{p_A}, \overline{p_B} \in [0, 1]$ satisfy:

$$\mathbb{P}_\epsilon(f(x + \epsilon) = c_A) \ge \underline{p_A} \ge \overline{p_B} \ge \ge \max_{c \neq c_A} \mathbb{P}_\epsilon(f(x + \epsilon) = c)$$

with $\underline{p_A}$ a lower bound on the true highest probability and $\overline{p_B}$ an upper bound on the true second highest probability. In practice, get bounds via sampling which gives statistical guarantees. Then: $g(x + \delta) = c_A$, for all $||\delta||_2 < R$,

$R := \frac{\sigma}{2}(\phi^{-1}(\underline{p_A}) - \phi^{-1}(\overline{p_B})) \ge 0$ with ϕ^{-1} the inverse Gaussian CDF. Certified radius R depends on input x since $\underline{p_A}, \overline{p_B}$ depend on x.

Notes on CDF: If $x \sim \mathcal{N}(0, 1)$, $p \in [0, 1]$, then $\phi^{-1}(p) = \nu$ s.t. $\mathbb{P}_x(x \le \nu) = p$. ϕ^{-1} is monotone, i.e. for $\underline{p_A} \ge \overline{p_B}$, $\phi^{-1}(\underline{p_A}) \ge \phi^{-1}(\overline{p_B})$. $\phi^{-1}(p) = -\phi^{-1}(1 - p), p \in [0, 1]$

Certified Accuracy: Pick target radius T and count #test points whose certified radius is $R \ge T$ and where the predicted c_A matches the test set label.

Standard Accuracy: Instantiate certified accuracy with $T = 0$

9.1 Certification Procedure

```
function CERTIFY(f,σ,x,n_0,n,α)
    counts0 ← SampleUnderNoise(f,x,n_0,σ)
    ĉ_A ← top index in counts0
    counts ← SampleUnderNoise(f,x,n,σ)
    p_a ← LowerConfBound(counts[ĉ_A],n,1-α)
    if p_a > 1/2:
        return prediction ĉ_A, radius σφ^{-1}(p_A)
    else: return ABSTAIN
```

Notes:

- $\hat{c}_A$ is not necessarily the correct test set label
- Sample $2\times$ ($n \gg n_0$) to prevent selection bias.
- SampleUnderNoise evaluates f at $x + \epsilon_i$ for $i \in \{1, .., n\}$, returns dict of class counts.
- LowerConfBound returns probability p_l s.t. $p_l \le p$ with probability $1 - \alpha$, assuming $k \sim Binomial(n, p)$ for unknown p.
- $\underline{p_A} > \frac{1}{2}$ ensures $\overline{p_B} < \frac{1}{2}$, thus $\underline{p_A} \le \overline{p_B}$
- With probability at least $1 - \alpha$, if CERTIFY returns class $\hat{c}_A$ and radius $R = \sigma\phi^{-1}(\underline{p_A})$, then $g(x + \delta) = \hat{c}_A$ for all $||\delta|| < R$.
- To increase R, need to increase $\underline{p_A}$. To increase $\underline{p_A}$, get f to classify more noisy points to $\hat{c}_A$. Increasing the #samples only slowly grows R.

9.2 Inference

```
fuction PREDICT(f,σ,x,n,α)
    counts ← SampleUnderNoise(f,x,n,σ)
    ĉ_A, ĉ_B ← top two indices from counts
    n_A, n_B ← counts[ĉ_A], counts[ĉ_B]
    if BinomPValue(n_A, n_A + n_B, 0.5) ≤ α:
        return ĉ_A
    else: return ABSTAIN
```

Notes:

- Null hypothesis: true probability of success of f returning $\hat{c}_A$ is $q = 0.5$
- BinomPValue returns p-value of null hypothesis, evaluated on n iid samples with i successes.
- Accept null hypothesis if p-value is $> \alpha$
- Reject null hypothesis if p-value is $\le \alpha$
- α small: often accept null hypothesis and ABSTAIN, but more confident in predictions.
- α large: more predictions but more mistakes.
- Can prove that: PREDICT returns wrong class $\hat{c}_A \neq c_A$ with probability at most α

9.3 Generalizing Smoothing

Base classifier $f : \mathbb{R}^d \to \mathcal{Y}$, image transformation $\psi_\alpha : \mathbb{R}^d \to \mathbb{R}^d$, construct smoothed classifier: $g(x) := \arg\max_{c \in \mathcal{Y}} \mathbb{P}_\epsilon(f(\psi_\epsilon(x)) = c)$ where $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ and requiring composition $\psi_\alpha(\psi_\beta) = \psi_{\alpha+\beta}$. ψ is instantiated to geometric transformations.

10 Appendix

DM: $\neg(\phi \wedge \psi) = \neg\phi \vee \neg\psi$; $\neg(\phi \vee \psi) = \neg\phi \wedge \neg\psi$

Normballs: $\mathbb{B}^1_\epsilon \subseteq \mathbb{B}^2_\epsilon \subseteq \mathbb{B}^\infty_\epsilon$ $\quad \mathbb{B}^\infty_\epsilon \subseteq \mathbb{B}^1_{\epsilon \cdot d}$ $\quad \mathbb{B}^\infty_\epsilon \subseteq \mathbb{B}^2_{\epsilon \cdot \sqrt{d}}$ $\quad \mathbb{B}^2_\epsilon \subseteq \mathbb{B}^1_{\epsilon \cdot \sqrt{d}}$

Jensen: g convex: $g(E[X]) \le E[g(X)]$

g concave (e.g. log): $g(E[X]) \ge E[g(X)]$

Bayes: $P(X|Y) = \frac{P(X,Y)}{P(Y)} = \frac{P(Y|X)P(X)}{P(Y)}$

Inv: $A^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

Variance & Covariance

$\mathbb{V}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$\mathbb{V}(X + Y) = \mathbb{V}(X) + \mathbb{V}(Y) + 2Cov(X, Y)$

$\mathbb{V}(AX) = A\mathbb{V}(X)A^T, \mathbb{V}[\alpha X] = \alpha^2\mathbb{V}[X]$

$\mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$

Distributions

$\mathrm{Exp}(x|\lambda) = \lambda e^{-\lambda x}$, $\mathrm{Ber}(x|\theta) = \theta^x(1 - \theta)^{(1-x)}$

Sigmoid: $\sigma(x) = 1/(1 + e^{-x})$

$a\mathcal{N}(\mu_1, \sigma_1^2) + \mathcal{N}(\mu_2, \sigma_2^2) = \mathcal{N}(a\mu_1 + \mu_2, a^2\sigma_1^2 + \sigma_2^2)$

Chebyshev & Consistency

$\mathbb{P}(|X - \mathbb{E}[X]| \ge \epsilon) \le \frac{\mathbb{V}[X]}{\epsilon^2}$

$\lim_{n\to\infty} P(|\hat{\mu} - \mu| > \epsilon) = 0$

Cramer Rao lower bound

$\mathrm{Var}[\hat{\theta}] \ge \mathcal{I}_n(\theta)^{-1}$, $\mathcal{I}_n(\theta) = -\mathbb{E}[\frac{\partial^2 \log p[\mathcal{X}_n|\theta]}{\partial\theta^2}]$,

$\hat{\theta}$ unbiased. Efficiency of $\hat{\theta}$: $e(\theta_n) = \frac{1}{\mathrm{Var}[\hat{\theta}_n]\mathcal{I}_n(\theta)}$

$e(\theta_n) = 1$ (efficient)

$lim_{n\to\infty} e(\theta_n) = 1$ (asymp. efficient)

Derivatives

$(fg)' = f'g + fg'$; $(f/g)' = (f'g - fg')/g^2$

$f(g(x))' = f'(g(x))g'(x)$; $\log(x)' = 1/x$

$\partial_x \mathbf{b}^\top\mathbf{x} = \partial_x \mathbf{x}^\top\mathbf{b} = \mathbf{b}$, $\partial_x \mathbf{x}^\top\mathbf{x} = \partial_x ||\mathbf{x}||_2^2 = 2\mathbf{x}$,

$\partial_x \mathbf{x}^\top\mathbf{A}\mathbf{x} = (\mathbf{A}^\top + \mathbf{A})\mathbf{x}$, $\partial_x(\mathbf{b}^\top\mathbf{A}\mathbf{x}) = \mathbf{A}^\top\mathbf{b}$,

$\partial_X(\mathbf{c}^\top\mathbf{X}\mathbf{b}) = \mathbf{c}\mathbf{b}^\top$, $\partial_X(\mathbf{c}^\top\mathbf{X}^\top\mathbf{b}) = \mathbf{b}\mathbf{c}^\top$,

$\partial_x(\|\mathbf{x} - \mathbf{b}\|_2) = \frac{\mathbf{x} - \mathbf{b}}{\|\mathbf{x} - \mathbf{b}\|_2}$, $\partial_X(\|\mathbf{X}\|_F^2) = 2\mathbf{X}$,

$\partial_x ||\mathbf{x}||_1 = \frac{\mathbf{x}}{|\mathbf{x}|}$, $\partial_x \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{2}(\mathbf{A}^\top\mathbf{A}\mathbf{x} - \mathbf{A}^\top\mathbf{b})$,

MILP encodings

$y = |x|, l \le x \le u$: $y \ge x, y \ge -x$,

$y \le -x + a \cdot 2u, y \le x - (1 - a) \cdot 2l$, $a \in \{0, 1\}$

$y = \max(x_1, x_2), l_1 \le x_1 \le u_1, l_2 \le x_2 \le u_2$:

$y \ge x_1, y \ge x_2$, $y \le x_1 + a \cdot (u_2 - l_1)$,

$y \le x_2 + (1 - a) \cdot (u_1 - l_2), a \in \{0, 1\}$