Certified Denfense to Image Transformation via Randomized Smoothing

Marc Fischer, Maximilian Baader, Martin V echev

ETH Zurich

September 14, 2021

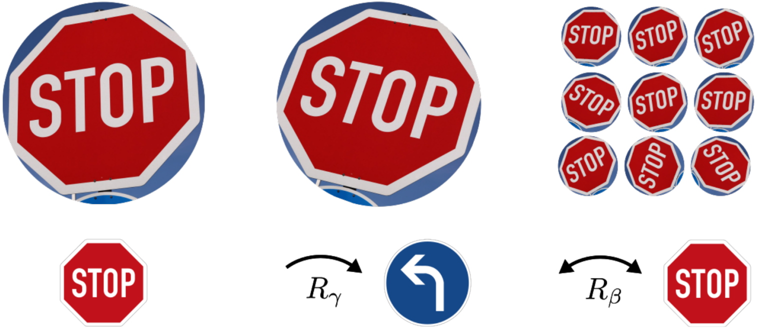
目录

目录

Contribution

- First generalization of randomized smoothing to image transformations
- Presented several certified defenses allowing for both distributional and individual guarantees (relying on statistical error bounds or on efficient inverse computation)

Attack of Parametric Transformations



Randomized Smoothing (RS)

A smoothed classifier $g\colon \mathbb{R}^m \mapsto \mathcal{Y}$ can be constructed out of an ordinary classifier $f\colon \mathbb{R}^m \mapsto \mathcal{Y}$, by calculating the most probable result of $f(\boldsymbol{x}+\epsilon)$ where $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbb{1})$:

$$g(\boldsymbol{x}) := \arg\max_c \mathbb{P}_{\epsilon\sim\mathcal{N}(0,\sigma^2\mathbb{1})}(f(\boldsymbol{x}+\epsilon) = c).$$

One then obtains the following robustness guarantee:

Theorem 3.1 (From [7]). *Suppose* $c_A \in \mathcal{Y}$, $\underline{p_A}, \overline{p_B} \in [0,1]$. *If*

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

then $g(\boldsymbol{x}+\delta) = c_A$ *for all* δ *satisfying* $\|\delta\|_2 \leq \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) =: r_\delta$.

randomized smoothing for parametric transformations (SPT)

to randomized smoothing for parametric transformations (**SPT**):

$$g(x) = \operatorname{argmax}_{c \in \mathcal{Y}} \mathbb{P}\left(f(\psi_\beta(x)) = c\right)$$

$$\text{for classifier } f, \text{ noise } \beta \sim \mathcal{N}(0, \sigma^2 I)$$

$$\text{Then } g(\psi_\gamma(x)) = g(x) \text{ for } \|\gamma\|_2 \leq r_\gamma.$$

$$\textbf{requires } \psi_\alpha \circ \psi_\beta = \psi_{\alpha+\beta}$$

then $g \circ \psi_\gamma(\boldsymbol{x}) = c_A$ *for all* γ *satisfying* $\|\gamma\|_2 \leq \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) =: r_\gamma$. *Further, if* g *is evaluated on a proxy classifier* f' *that behaves like* f *with probability* $1-\rho$ *and else returns an arbitrary answer, then* $r_\gamma := \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A} - \rho) - \Phi^{-1}(\overline{p_B} + \rho))$.

Pratical algorithm

```
# evaluate g at x
function PREDICT(f, σ, x, n, α)
    counts ← SAMPLEUNDERNOISE(f, x, n, σ)
    ĉ_A, ĉ_B ← top two indices in 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
```

- SAMPLEUNDERNOISE(f, x, num, σ) works as follows:
  1. Draw num samples of noise, $\varepsilon_1 \dots \varepsilon_{\text{num}} \sim \mathcal{N}(0, \sigma^2 I)$.
  2. Run the noisy images through the base classifier f to obtain the predictions $f(x+\varepsilon_1), \dots, f(x+\varepsilon_{\text{num}})$.
  3. Return the counts for each class, where the count for class c is defined as $\sum_{i=1}^{\text{num}} \mathbf{1}[f(x+\varepsilon_i) = c]$.
- BINOMPVALUE(n_A, n_A+n_B, p) returns the p-value of the two-sided hypothesis test that $n_A \sim \text{Binomial}(n_A+n_B, p)$. Using `scipy.stats.binom_test`, this can be implemented as: `binom_test(nA, nA + nB, p)`.

Pratical algorithm

 # certify the robustness of g around x
function CERTIFY(f, σ, x, n_0, n, α)
  `counts0` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n_0, σ)
  $\hat{c}_A$ $\leftarrow$ top index in `counts0`
  `counts` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n, σ)
  $\underline{p_A}$ $\leftarrow$ LOWERCONFBOUND(`counts`$[\hat{c}_A]$, n, $1-\alpha$)
  if $\underline{p_A} > \frac{1}{2}$ **return** prediction $\hat{c}_A$ and radius $\sigma\,\Phi^{-1}(\underline{p_A})$
  else return ABSTAIN

LOWERCONFBOUND(k, n, $1-\alpha$) returns a one-sided $(1-\alpha)$ lower confidence interval for the Binomial parameter p given that $k \sim \text{Binomial}(n, p)$. In other words, it returns some number $\underline{p}$ for which $\underline{p} \leq p$ with probability at least $1-\alpha$ over the sampling of $k \sim \text{Binomial}(n, p)$. Following Lecuyer et al. (2019), we chose to use the Clopper-Pearson confidence interval, which inverts the Binomial CDF (Clopper & Pearson, 1934). Using `statsmodels.stats.proportion.proportion_confint`, this can be implemented as

```
proportion_confint(k, n, alpha=2*alpha, method="beta")[0]
```

目录

BaseSPT(Heuristic best effort defense)

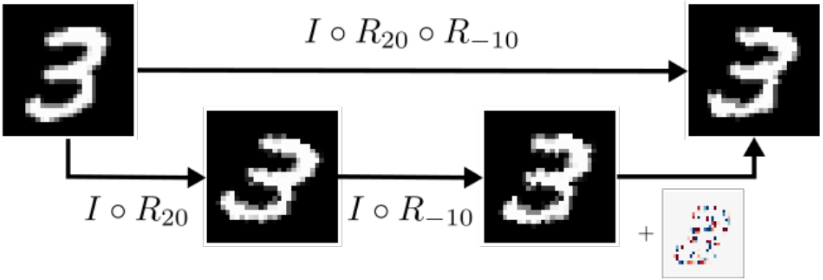
Table 2: Evaluation of BaseSPT. We obtain Acc for b on the test set and evaluate adv. Acc. on 3000 images obtained by the worst-of-100 attack. t denotes the average run time of g.

			Acc.	adv. Acc.		
Dataset	T^I	$\Gamma_\pm$	b	b	g	t [s]
MNIST	R^I	30°	0.99	0.73	0.99	0.97
CIFAR-10	R^I	30°	0.91	0.26	0.85	0.95
ImageNet	R^I	30°	0.76	0.56	0.76	5.43
MNIST	Δ^I	4	0.99	0.03	0.53	0.86
CIFAR-10	Δ^I	4	0.91	0.44	0.79	0.95
ImageNet	Δ^I	20	0.76	0.65	0.75	6.70

- By applying SPT to image rotation we can obtain a heuristic defense as rotations don't compose as required (discussed next).
- Here we show results for adversarial rotations of up to 30° and translations up 4 or 20.
- worst-of-k attack:randomly sample k parameters;choose the three worst(max loss).
- $n_\gamma = 1000, _\gamma = \Gamma_\pm, \alpha_\gamma = 0.01$

目录

Interpolation:Image Rotations don't compose



Certification in the Presence of Interpolation

1. Aim:$g_E \circ T^I_\gamma(x) = g_E(x), \|\gamma\| \leq r_\gamma$
2. $h_E \circ T^I_\beta \circ T^I_\gamma(\mathbf{x}) = h_E \circ T^I_{\beta+\gamma}(\mathbf{x}),\ \forall \beta, \gamma \in \mathbb{R}^d$
3. $f := h_E \circ I, \psi_\beta := T_\beta$
4. $g_E(\mathbf{x}) := \arg\max_c \mathbb{P}_{\beta\sim\mathcal{N}(0,\sigma^2 1)} (h_E \circ I \circ T_\beta(\mathbf{x}) = c)$
5. Such that $g_E \circ T_\gamma(x) = c_A = g_E(x),\ \ \|\gamma\| \leq r_\gamma$

$$
\begin{aligned}
g_E \circ T_\gamma(\boldsymbol{x}) &= \arg\max_c \mathbb{P}_{\beta\sim\mathcal{N}(0,\sigma^2\mathbb{1})} \left(h_E \circ I \circ T_\beta \circ T_\gamma(\boldsymbol{x}) = c\right) \\
&= \arg\max_c \mathbb{P}_{\beta\sim\mathcal{N}(0,\sigma^2\mathbb{1})} \left(h_E \circ T^I_\beta \circ T^I_\gamma(\boldsymbol{x}) = c\right) \\
&= g_E \circ T^I_\gamma(\boldsymbol{x}),
\end{aligned}
$$

Second Step: Construction h_E

$$\epsilon(\beta, \gamma, \boldsymbol{x}) := T^I_\beta \circ T^I_\gamma(\boldsymbol{x}) - T^I_{\beta+\gamma}(\boldsymbol{x}),$$

$$\text{bounded by } E \in \mathbb{R}^{\geq 0} \text{ s.t. } \forall \beta, \gamma \in \mathbb{R}^d.\ \|\epsilon(\beta, \gamma, \boldsymbol{x})\|_2 \leq E$$

Thus if h_E is l_2-robust with radius E around $T^I_{\beta+\gamma}(x)$, **interpolation invariance** holds.

Obtaining probabilistic guarantees from Theorem 3.2

$$\mathbb{P}_{\beta\sim\mathcal{N}(0,\sigma^2 1)}\left(\|\epsilon(\beta,\gamma,\boldsymbol{x})\|_2 \leq E\right) \geq 1-\rho_E \quad \forall\gamma\in\mathbb{R}^d$$

- h'_E: like h_E with probability at least $1-\rho_E$
- $f' = h'_E \circ I$ like f with probability at least $1-\rho_E$

For a fixed $E \in \mathbb{R}^{\geq 0}$, $\rho_E \in [0, 1]$, the probability that ϵ is bounded by E for $\boldsymbol{x} \sim \mathcal{D}$ is

$$q_E := \mathbb{P}_{\boldsymbol{x} \sim \mathcal{D}}(\mathbb{P}_{\beta \sim \mathcal{N}(0, \sigma^2 \mathbb{1})}(\max_{\gamma \in \Gamma} \|\epsilon(\beta, \gamma, \boldsymbol{x})\|_2 \leq E\ ) \geq 1 - \rho_E). \tag{6}$$

Evaluate q_E:$DistSPT^D$

1. Sampling multiple realizations of β
2. Computing their corresponding error ϵ and checking how many are successfully bounded by E
3. Bounding the inner probability using Clopper-Pearson.
4. Counts x and apply Clopper-Pearson and obtain a lower bound q_E with the desired confidence
5. For $DistSPT^x$, Only compute inner probability ($q_E = 1$)

maximization interpolation error over γ

1. method: interval analysis;propagate lower and upper bounds
2. obtain a lower and upper bound for the norm Calculation,pick maximum:

$$\max_{\gamma\in\Gamma} \|\epsilon(\beta,\gamma,x)\|_2 \leq \max \left\| T^I_\beta \circ T^I_\Gamma(x) - T^I_{\beta+\Gamma}(x) \right\|_2$$

3. Refine: splitting the hyperrectangle Γ into smaller hyperrectangles Γ_k

$$\max_{\gamma\in\Gamma} \|\epsilon(\beta,\gamma,\boldsymbol{x})\|_2 \leq \max_{k\in\{0,\ldots,N\}} \max \left\| T^I_\beta \circ T^I_{\Gamma_k}(\boldsymbol{x}) - T^I_{\beta+\Gamma_k}(\boldsymbol{x}) \right\|_2$$

4. E:For each sample (x, β), we simply keep the values;
5. $DistSPT^D$:pick an E that bounds many of these values, choosing ρ_E to be small,and Theorem hold with q_E. $DistSPT^x$:suitable E

$DistSPT^{D}$

Table 3: Evaluation of DISTSPT$^{\mathcal{D}}$ for $T^I := R^I$. We show the test set accuracy of b, certified accuracy of g at different radii r_γ, along with the average run time t. # denotes values obtained by sampling. Each certificate hold with overall confidence 0.99.

				g cert. acc at r_γ					
Dataset	E	q_E	b acc.	0°	10°	20°	30°	t [s]	n_γ
MNIST	0.45	0.99	0.98	0.89	0.88	0.87	0.85	21.56	200
CIFAR-10	0.55	0.99	0.56	0.31	0.28	0.25	0.19	89.75	50
CIFAR-10	0.55	0.99	0.56	0.32	0.30	0.28	0.25	351.47	200
RImageNet	$1.20^{\#}$	0.97	0.78	0.74	0.72	0.68	0.61	100.73	50
RImageNet	$1.35^{\#}$	0.99	0.78	0.64	0.62	0.56	0.50	100.13	50
ImageNet	$0.95^{\#}$	0.75	0.38	0.30	0.24	0.18	0.12	100.21	50
ImageNet	$1.20^{\#}$	0.97	0.38	0.23	0.19	0.13	0.09	100.73	50
ImageNet	$1.35^{\#}$	0.99	0.38	0.16	0.12	0.08	0.06	100.44	50

- obtain E:Sample the interpolation error using 1000 images
- test for $\rho_E = 10^{-3}$,obtain q_E with $1-\alpha_E = 0.999, (x = 1000, \beta = 8000)$
- optimization over γ for many images is computationally expensive $\rightarrow$ 10 sample for γ

$DistSPT^x$

Table 4: Evaluation of DISTSPT^{x} for $T^I := R^I$. We show the test set certified accuracy of g at different radii r_γ, along with the average E estimated, the average time t_E to estimate E and average time t_{RS} to apply randomized smoothing. # denotes values obtained by sampling. * we use a server with 128 threads on an AMD EPYC 7601 processor, on the same system as the other results these take 766 s. Each certificate hold with overall confidence 0.99.

			g cert. acc at r_γ								
Dataset	$\Gamma_\pm$	σ_γ	0°	10°	20°	30°	50°	avg. E	t_E [s]	t_{RS} [s]	n_γ
MNIST	50°	30	0.93	0.92	0.91	0.90	0.82	0.34	53.33	20.56	200
CIFAR-10	30°	40	0.35	0.30	0.27	0.22	-	0.34	81.83	91.72	50
CIFAR-10	10°	10	0.43	0.37	-	-	-	0.34	51.12	92.83	50
ImageNet	30°	30	0.31	0.25	0.17	0.11	-	0.86#	73.58*	100.47	50
ImageNet	30°	30	0.32	0.29	0.22	0.16	-	0.86#	73.58*	396.50	200

- individual bounds are much lower than distribution bounds for $E \rightarrow$ better accuracy
- 100 sample of β to guess E; 400 sample of β to test ρ_E
- In theory, $DistSPT^D$ can perform better if ρ_E is lower (e.g. when more samples are used)
- However, in practice this is offset by the tighter error bound attained by $DistSPT^x$

目录

Flow

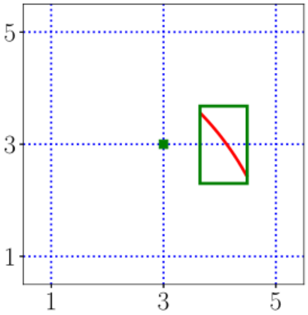
- we are given $x' := T^I_\gamma(x)$ but neither the original x nor the parameter $\gamma \in \Gamma$,and we would like to certify that $g_E(x') = g_E(x)$
- $\mathbb{P}_{\beta \sim \mathcal{N}(0,\sigma^2 1)} \left(\|\epsilon(\beta, \gamma, \boldsymbol{x})\|_2 \leq E \right) \geq 1 - \rho_E \quad \forall \gamma \in \mathbb{R}^d$
- compute an upper bound on the max term without having access to x
- obtain E:$\max_{\gamma \in \Gamma} \|\epsilon(\beta, \gamma, \boldsymbol{x})\|_2 \leq \max \left\| T^I_\beta(\boldsymbol{x}') - T^I_{\beta+\gamma} \circ \left(T^I_\gamma\right)^{-1}(\boldsymbol{x}') \right\|_2$
- KEY:$\left(T^I_\gamma\right)^{-1} \to \left(T^I_\Gamma\right)^{-1}(\boldsymbol{x}') := \left\{ \boldsymbol{x} \in \mathbb{R}^m \mid T^I_\gamma(\boldsymbol{x}) = \boldsymbol{x}', \gamma \in \Gamma \right\}$
- $\max_{\gamma \in \Gamma} \|\epsilon(\beta, \gamma, \boldsymbol{x})\|_2 \leq \max \left\| T^I_\beta(\boldsymbol{x}') - T^I_{\beta+\Gamma} \circ \left(T^I_\Gamma\right)^{-1}(\boldsymbol{x}') \right\|_2$

base

- even image height and width;center it at 0 on $G := (2\mathbb{Z}+1)\times(2\mathbb{Z}+1)$
- pixel at (i,j): $p_{(i,j)} \in [0,1]$
- (v,w)-interpolation region: $[v, v+2] \times [w, w+2]$
- Bilinear interpolation:$I(x,y) =$
  $p_{v,w}\frac{2+v-x}{2}\frac{2+w-y}{2} + p_{v,w+2}\frac{2+v-x}{2}\frac{y-w}{2} + p_{v+2,w}\frac{x-v}{2}\frac{2+w-y}{2} + p_{v+2,w+2}\frac{x-v}{2}\frac{y-w}{2}$
- Example: assume attack $\gamma \in [23°, 26°]$,calculate the constraint for pixel (3,3) of the original image.

Step1

For every $(i', j') \in G$, we over-approximate the region the pixel value $p'_{i',j'}$ could have been interpolated from, which is
$c_{i',j'} := T_{\Gamma}^{-1}(i', j')$, $C := \{c_{i',j'} \mid (i', j') \in G\}$.
We illustrate the calculation of the set C for
$c_{5,1} := R^{-1}_{[23°,26°]} \begin{pmatrix} 5 \\ 1 \end{pmatrix} = \begin{pmatrix} [4.06, 4.21] \\ [2.85, 3.11] \end{pmatrix}$.



Step2

the pixel value$p'_{i',j'}$ yields constraints for value $p_{i,j}$

$$[x_l, x_u] \times [y_l, y_u] := c_{i',j'} \cap [i, i+2] \times [j, j+2]$$

plug this into the interpolation I

$$\begin{aligned} p'_{i',j'} \in I\left([x_l, x_u], [y_l, y_u]\right) = & p_{i,j} \frac{2+i-[x_l, x_u]}{2} \frac{2+j-[y_l, y_u]}{2} + p_{i,j+2} \frac{2+i-[x_l, x_u]}{2} \frac{[y_l, y_u]-j}{2} \\ & + p_{i+2,j} \frac{[x_l, x_u]-i}{2} \frac{2+j-[y_l, y_u]}{2} + p_{i+2,j+2} \frac{[x_l, x_u]-i}{2} \frac{[y_l, y_u]-j}{2} \end{aligned}$$

Step2

- Solve for the pixel value of interest $p_{i,j}$:replace other three with the (trivial) [0,1] constraint, covering all possible pixel values.
- instantiating $[x_l, x_u]$ and $[y_l, y_u]$ with its corner (x, y) furthest from (i, j), yields still a sound but more precise constraint $q_{i,j}$for $p_{i,j}$.
- choose x_u, y_u,then $p_{i,j}$ is:

$$q_{i,j} = \left[p'_{i',j'} - \left(\frac{2+i-x_u}{2}\frac{y_u-j}{2} + \frac{x_u-i}{2}\frac{2+j-y_u}{2} + \frac{x_u-i}{2}\frac{y_u-j}{2}\right), p'_{i',j'}\right]\left(\frac{2+i-x_u}{2}\frac{2+j-y_u}{2}\right)^{-1}$$

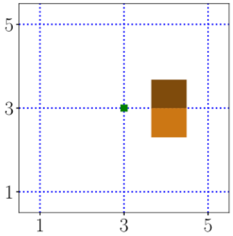
Step2

Step 2 The only non-empty intersections of $c_{5,1}$ with interpolation regions (blue squares in Fig. 3), cornering $(3,3)$ are the $(3,1)$ and the $(3,3)$-interpolation regions, hence we omit $q_{1,1}$ and $q_{1,3}$. The intersection with the $(3,3)$-interpolation region yields $[x_l, x_u] = [4.06, 4.21]$ and $[y_l, y_u] = [3, 3.11]$ (dark brown rectangle in Fig. 3b), hence at the furthest corner $(x, y) = (4.21, 3.11)$, we get

$$q_{3,3} = [0.73, 2.48] = \left[p'_{5,1} - \left(\tfrac{5-x}{2}\tfrac{y-3}{2} + \tfrac{x-3}{2}\tfrac{5-y}{2} + \tfrac{x-3}{2}\tfrac{y-3}{2}\right), p'_{5,1}\right]\left(\tfrac{5-x}{2}\tfrac{5-y}{2}\right)^{-1},$$

and the intersection with the $(3,1)$-interpolation region yields $[x_l, x_u] = [4.06, 4.21]$ and $[y_l, y_u] = [2.85, 3]$ (light brown rectangle in Fig. 3b), hence at the furthest corner $(x, y) = (4.21, 2.85)$, we get

$$q_{3,1} = [0.72, 2.48] = \left[p'_{5,1} - \left(\tfrac{5-x}{2}\tfrac{3-y}{2} + \tfrac{x-3}{2}\tfrac{3-y}{2} + \tfrac{x-3}{2}\tfrac{y-1}{2}\right), p'_{5,1}\right]\left(\tfrac{5-x}{2}\tfrac{y-1}{2}\right)^{-1}.$$



Step3

- In order to be sound, we need to take the union $q_{i,j}$
- To gain precision, we can intersect all of those unions and [0,1]
- $[a, b] \sqcup [c, d] := [\min(a, c), \max(b, d)]$
- $$p_{i,j} \in [0, 1] \cap \left(\bigcap_{c_{i',j'} \in C} q_{i-2,j-2}\left(c_{i',j'}\right) \sqcup q_{i,j-2}\left(c_{i',j'}\right) \sqcup q_{i-2,j}\left(c_{i',j'}\right) \sqcup q_{i,j}\left(c_{i',j'}\right) \right)$$
- In example: $q_{3,1} \sqcup q_{3,1} = [0.72, 2.48] \rightarrow \cap[0, 1] \rightarrow$ constraints from the other $c'_{i',j'} \rightarrow p_{3,3} \in [0.73, 1]$

Experiment

- **Refine**: 10 times
- E:large due to loss of precision
- g was correct on 82% of attacked images;81% we could certify on MNIST
- analysis of E took on average 0.26s and the randomized smoothing 25.03s
- Challenges on larger datasets

further Experiment

Table 11: Maximum observed errors and without gaussian blur (G) and without vignetting (V).

Dataset	Both	-V	-G	-V-G
MNIST	**0.36**	**0.36**	2.47	2.51
CIFAR-10	**0.51**	6.08	2.66	18.17
ImageNet	**0.91**	70.66	9.25	75.69

Table 12: Correct classifications and by the model and verifications by DeepG [11], with and without vignetting (V), out of 100 images.

Model	Correct	[11]	[11]+V
MNIST	98	86	87
CIFAR-10	74	65	32
CIFAR-10+V	78	63	23

we apply a circular or rectangular vignette for rotation and translation respectively, to reduce error estimates in areas of the image where information is lost.

Thank you