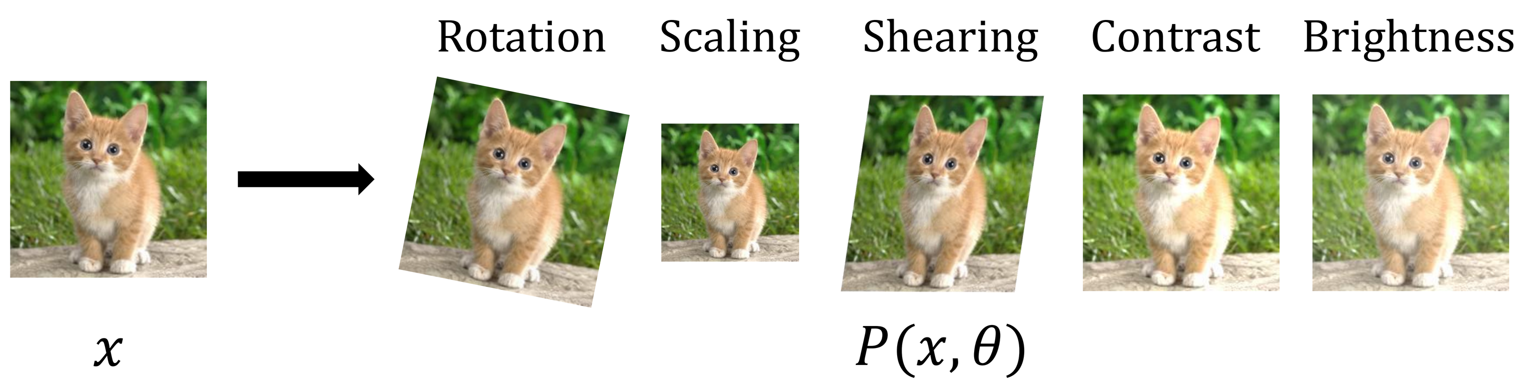
Provable Defense Against Geometric Transformations

Rem Yang, Jacob Laurel, Sasa Misailovic, Gagandeep Singh

Geometric Transformations

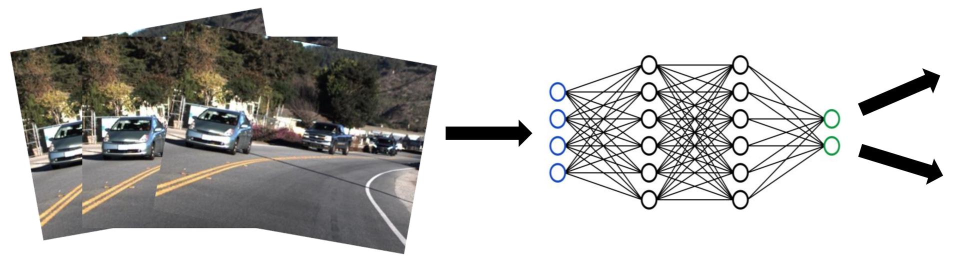


Certified Robustness

Objective

Set of perturbed images X'

Network f



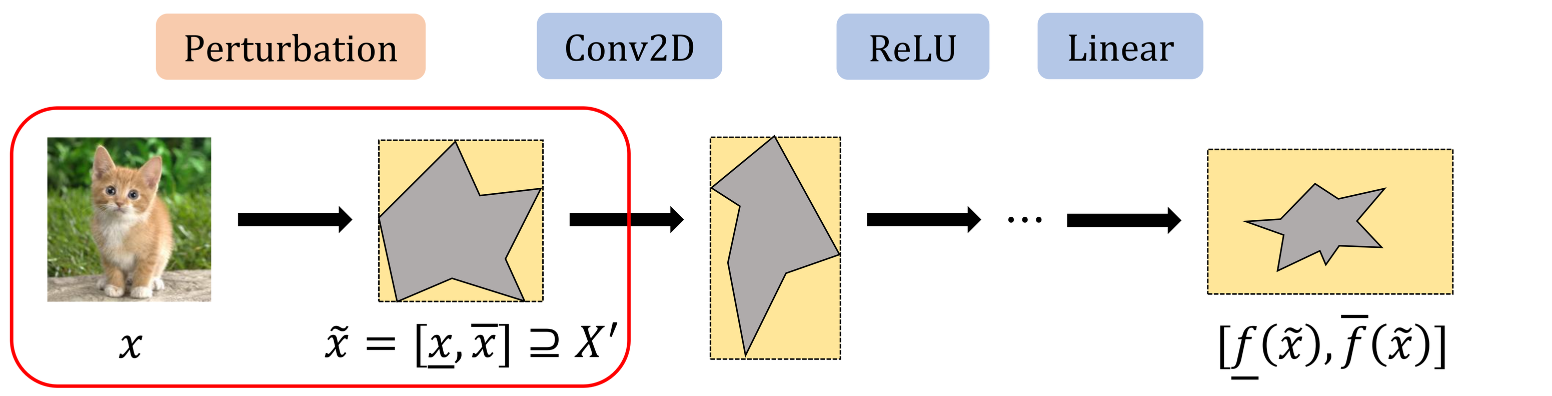
Classification: Provably correct?

$\forall x' \in X'.\ y = \arg\max_i f_i(x')$

Regression: Certified bounds

$\underline{y} \leq \min_{x' \in X'} f(x') \leq \max_{x' \in X'} f(x') \leq \overline{y}$

Interval Bound Propagation



Classification: $\forall j \neq y.\ \underline{f_y}(\tilde{x}) > \overline{f_j}(\tilde{x})$

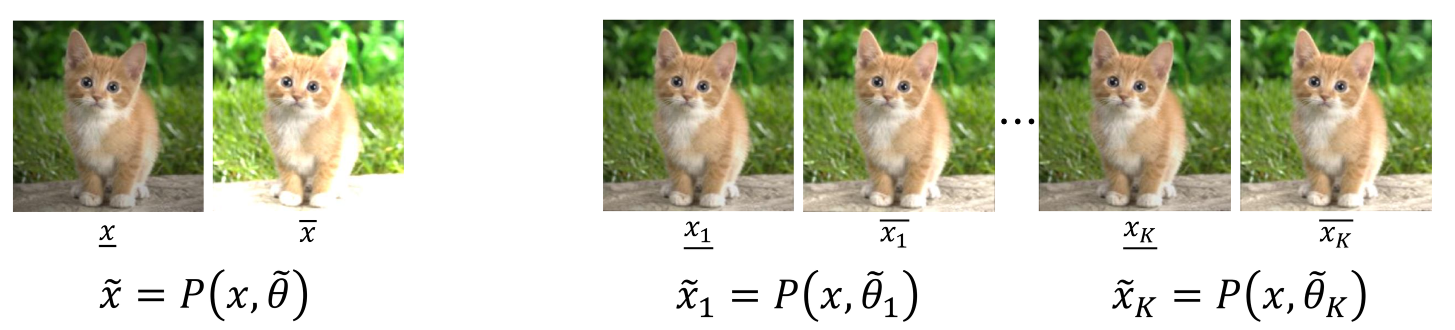
Regression: directly use output bounds

Challenges

Computing Geometric Bounds Is Costly

Dataset	Time to Compute Bounds (s)	Time to Propagate Bounds (s)
CIFAR-10	22.81	0.004
Tiny ImageNet	62.83	0.018

Geometric Certification Requires Splitting



Bounds are too over-approximate

Precision increases after splitting parameter range

Classification

$\forall k \in \{1, 2, \ldots, K\}. \forall j \neq y.\ \underline{f_y}(\tilde{x}_k) > \overline{f_j}(\tilde{x}_k)$

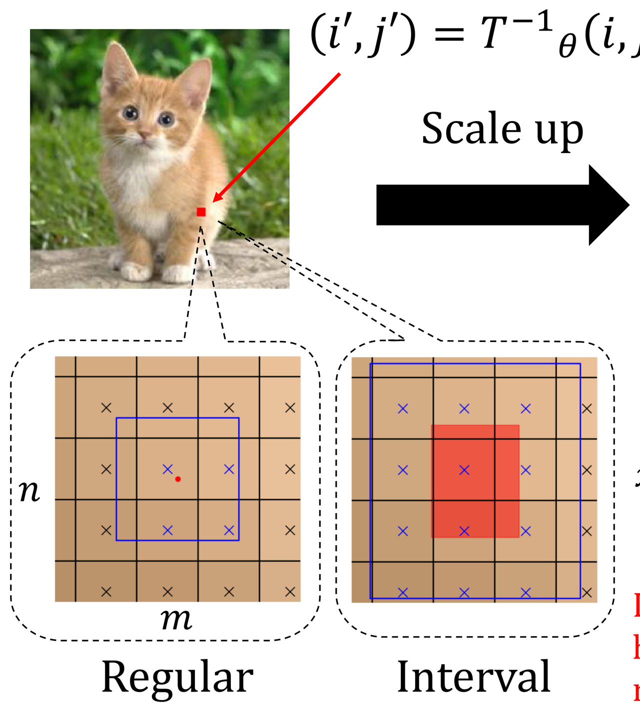
Regression

$\bigcup_{k=1}^{K} \{ f(\tilde{x}_k) \}$

Must compute geometric bounds many times + account for splitting in training

Fast Geometric Verifier

Illustration of Interpolated Transformations



To compute the value of each pixel in the transformed image:

1. Compute its coordinate in the original image via T^{-1}
2. Interpolate values of nearby pixels

Summands with both terms < 1 contribute to sum

$$x'_{i,j} = \sum_{n=0}^{H-1} \sum_{m=0}^{W-1} x_{n,m} \cdot \max(0, 1 - |i' - n|) \cdot \max(0, 1 - |j' - m|)$$

Interpolation distance $d_{n,m}^{i,j}$

Interval interpolation has no data regularity; not GPU-parallelizable as-is!

Algorithm and Running Example

Key insight: Precompute interpolation distances and store in custom sparse representation

① Inverse Coordinates ② Interpolation Grid ③ Exploiting Sparsity ④ Obtaining Final Images

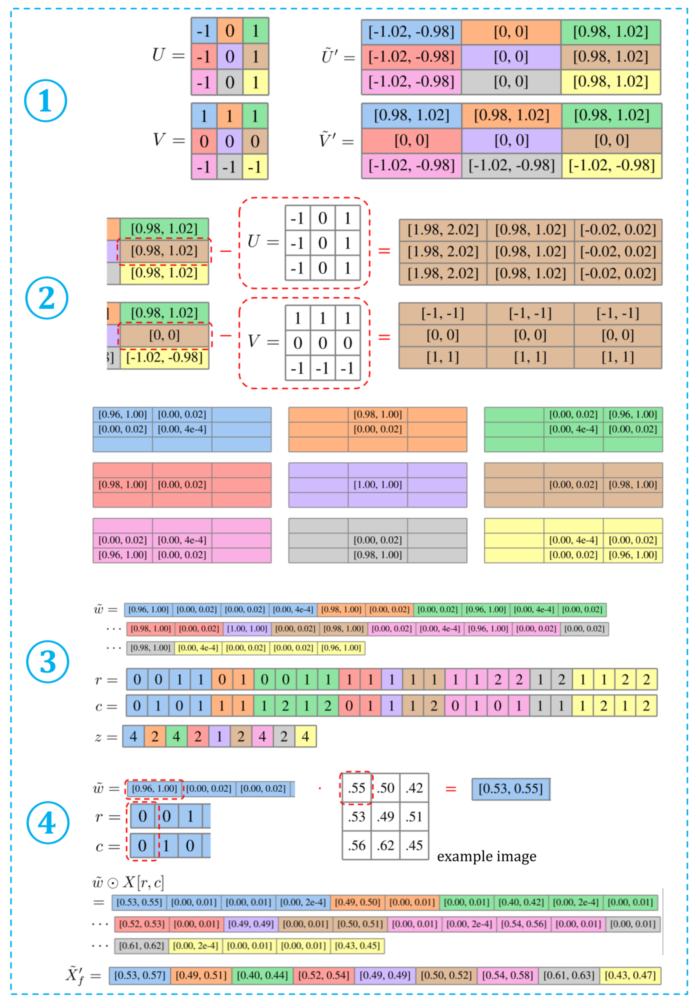
Example. Consider a 3×3 image and scaling by $\tilde{\theta} = \pm 2\%$: $T^{-1}{}_{\tilde{\theta}}(u, v) = \left(\frac{u}{[0.98, 1.02]}, \frac{v}{[0.98, 1.02]} \right)$

Algorithm 1 Fast interval interpolated transformation.

Input: $X \in [0,1]^{N \times C \times H \times W}$, a batch of N images with dimension $C \times H \times W$
$T_{\tilde{\theta}}$, an interpolated transformation with interval parameters $\tilde{\theta}$
Output: $\tilde{X}' \in [[0,1]^{N \times C \times H \times W}, [0,1]^{N \times C \times H \times W}]$

procedure MAKEINTERPGRID($H, W, T_{\tilde{\theta}}$)
① $(i, j) \leftarrow ([0, 1, \ldots, H-1], [0, 1, \ldots, W-1])$
$(u, v) \leftarrow (j - (W-1)/2, (H-1)/2 - i)$
$(U, V) \leftarrow (\underbrace{[u^T, u^T, \ldots, u^T]^T}_{H \text{ times}}, \underbrace{[v^T, v^T, \ldots, v^T]}_{W \text{ times}})$
$(\tilde{U}', \tilde{V}') \leftarrow T_{\tilde{\theta}}^{-1}(U, V)$
② $(\tilde{U}'_r, \tilde{V}'_r) \leftarrow (\tilde{U}'.\text{reshape}(HW, 1, 1), \tilde{V}'.\text{reshape}(HW, 1, 1))$
$\tilde{G} \leftarrow \max(0, 1 - |\tilde{V}'_r - V|) \odot \max(0, 1 - |\tilde{U}'_r - U|)$
③ $z \leftarrow \text{count\_nonzeros}(\tilde{G}, \text{dim} = (1, 2))$
$\tilde{g} \leftarrow \text{flatten}(\tilde{G})$
$q \leftarrow \text{get\_nonzero\_indices}(\tilde{g})$
$\tilde{w} \leftarrow \tilde{g}[q]$
$(r, c) \leftarrow (\lfloor (q \mod HW)/W \rfloor, (q \mod HW) \mod W)$
return $\tilde{G}_s \leftarrow (r, c, \tilde{w}, z)$
end procedure

procedure INTERPOLATE($X, \tilde{G}_s$)
$(r, c, \tilde{w}, z) \leftarrow \tilde{G}_s$
④ $\tilde{S} \leftarrow \tilde{w} \odot X[:, :, r, c]$
$\tilde{X}'_f \leftarrow \text{split\_and\_sum}(\tilde{S}, \text{dim} = 2, \text{sizes} = z)$
return $\tilde{X}' \leftarrow \tilde{X}'_f.\text{reshape}(N, C, H, W)$
end procedure



Achieves $3000 - 5000\times$ speedup over current sequential algorithm for a batch of images!

Geometric Provable Defense Formulation

Existing formulations [worst-case loss over *entire* perturbation region]

$\ell(\hat{f}(\tilde{x}), y)$ where $\hat{f}_y = \underline{f_y}$ and $\forall j \neq y.\ \hat{f}_j = \overline{f_j}$

Our formulation [worst-case loss over *small, sampled* regions]

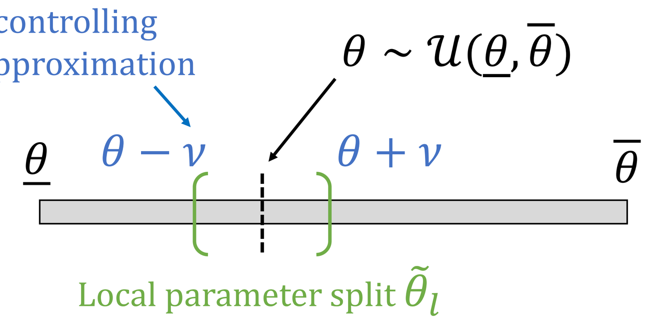
To enforce robustness to P across entire parameter range $\tilde{\theta} = [\underline{\theta}, \overline{\theta}]$:

Classification:

$\ell\left(\hat{f}\left(P(x, \tilde{\theta}_l)\right), y\right)$

Regression:

$\frac{1}{2} \cdot \left(\ell\left(\underline{f}\left(P(x, \tilde{\theta}_l)\right), y\right) + \ell\left(\overline{f}\left(P(x, \tilde{\theta}_l)\right), y\right) \right)$



Experimental Evaluation

Comparison with the State of the Art

We compare 3-layer MNIST CNNs and 4-layer CIFAR-10 CNNs trained and certified with our framework with those in DeepG (Balunovic et al., 2019), the SOTA for deterministic geometric certified robustness.
* denotes DeepG results over 100 test images (since it takes too long to run on the full test set).

MNIST

Transformations	Network	Accuracy (%)	Certified (%)	Certification Time per Image (s)	Our Speedup
Rotate(30°)	DeepG	99.1	86.0*	19.12	–
	Ours	99.1	94.2	0.00045	42623×
TranslateH(2), TranslateV(2)	DeepG	99.1	77.0*	367.82	–
	Ours	99.2	89.8	0.0090	40949×
Scale(5%), Rotate(5°), Contrast(5%), Brightness(.01)	DeepG	99.3	34.0*	155.24	–
	Ours	99.1	92.6	0.0048	32563×
Shear(2%), Rotate(2°), Scale(2%), Contrast(2%), Brightness(.001)	DeepG	99.2	72.0*	71.72	–
	Ours	99.1	96.3	0.024	2933×

CIFAR-10

Transformations	Network	Accuracy (%)	Certified (%)	Certification Time per Image (s)	Our Speedup
Rotate(10°)	DeepG	71.2	65.0*	78.18	–
	Ours	80.5	63.2	0.465	168×
Rotate(2°), Shear(2%)	DeepG	68.5	39.0*	18.92	–
	Ours	70.1	51.0	0.263	72×
Scale(1%), Rotate(1°), Contrast(1%), Brightness(.001)	DeepG	73.2	43.0*	163.26	–
	Ours	71.3	42.3	2.725	60×

Scalability to Larger Datasets

Tiny ImageNet

Network	Transforms	Accuracy (%)	Certified (%)	Certification Time per Image (s)
CNN7	Shear(2%)	27.3	18.7	0.059
	Scale(2%)	26.1	15.2	0.057
	Rotate(5°)	26.0	13.1	0.285
Wide ResNet	Shear(2%)	35.5	25.7	0.214
	Scale(2%)	33.1	21.3	0.205
	Rotate(5°)	32.2	17.4	1.006

No prior results in geometric setting. In ℓ_∞-norm setting with $\epsilon = \frac{1}{255}$, Xu et al., 2020 attain 21.6% clean / 12.7% certified accuracy on CNN7 and 27.8% clean / 15.9% certified accuracy on WideResNet.

Udacity Self-Driving

Green: ground truth label
Blue: network prediction
Red: certified bound under rotation of $\pm 2°$

Training Method	MAE	Certified MAE	Certification Time per Image (s)
Regular	6.07°	97.56°	0.11
Dropout	4.85°	96.65°	0.12
Ours	5.36°	8.05°	0.11

Certified training can **help** network performance; enforcing robustness while regularizing the network!