# Boosting Robustness Certification of Neural Networks


*Gagandeep Singh, Timon Gehr, Markus Püschel , and Martin Vechev*

Department of Computer Science, ETH Zurich

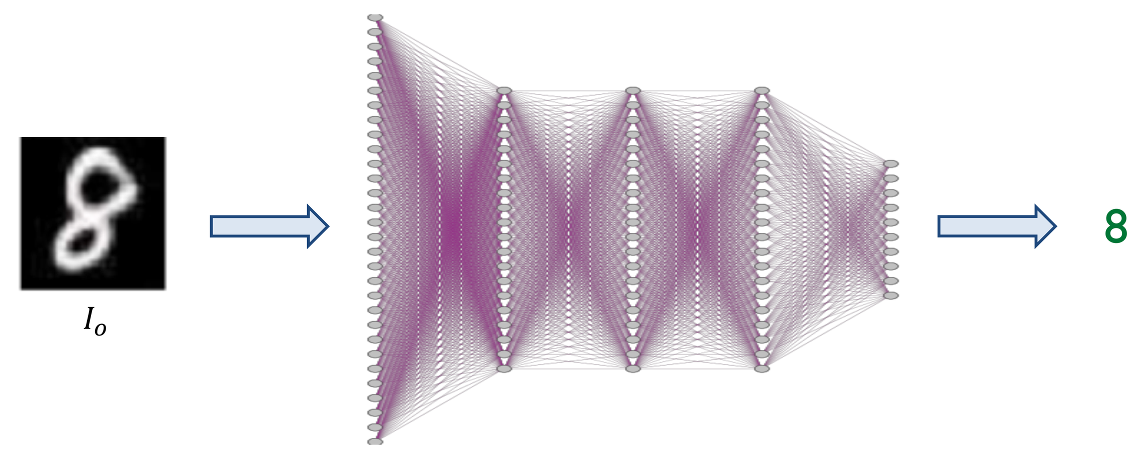
ETH zürich

safeai.ethz.ch

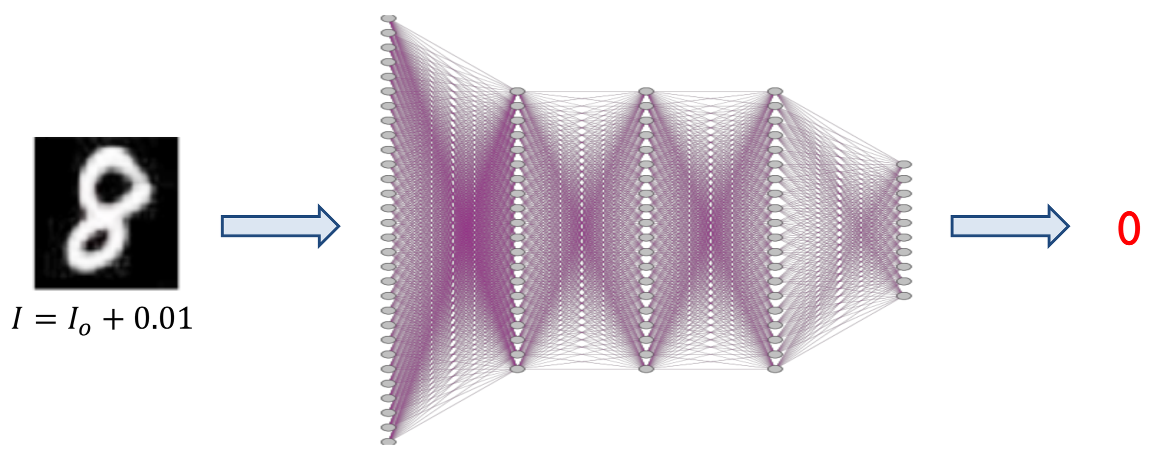

## Problem: Certification of neural network robustness

Small changes in pixel intensities can cause neural networks to misclassify

### $L_\infty$-norm based perturbation




The neural network classifies the image $I_o$ correctly as 8




When $\epsilon = 0.01$ is added to the intensity of each pixel in $I_o$, the network misclassifies the perturbed image $I$ as 0 even though $I$ appears as 8 to the human eye

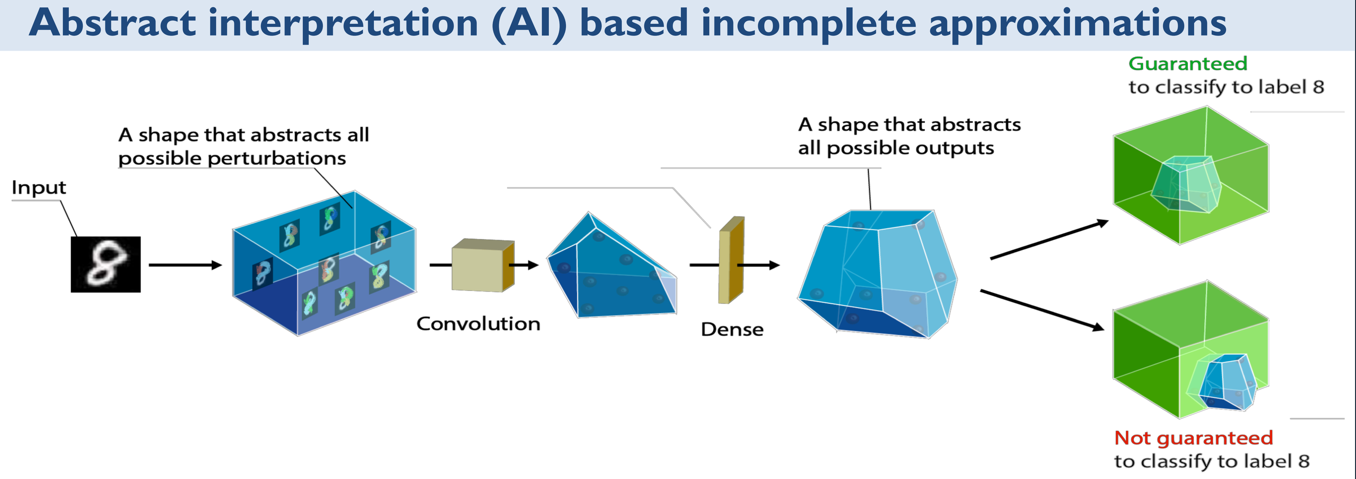
Goal: certify if a given neural network correctly classifies all images $I$ in the $\epsilon$-ball $\mathcal{B}_{(I_0,\infty)}(\epsilon)$ around $I_0$, i.e., all images $I$ where the intensity of each pixel in $I$ differs by at most $\epsilon$ from the corresponding pixel in $I_o$

### Solver based complete approaches

$$\begin{aligned} &min_z\ f^T z \\ &s.t.: z = ReLU(Cy + d) \\ &\quad y = ReLU(Ax + b) \\ &\quad l \leq x \leq u \end{aligned}$$

Precise but often do not scale

### Abstract interpretation (AI) based incomplete approximations



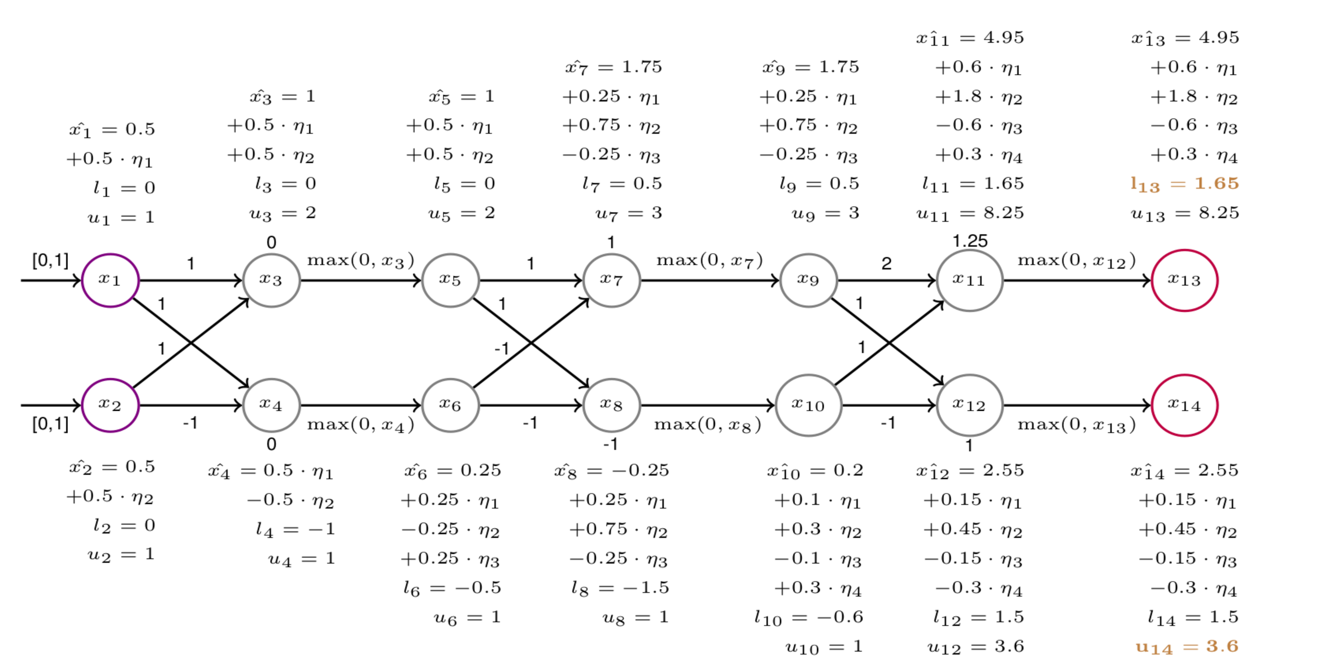

Scale but can be imprecise

In this work, we use the Zonotope based DeepZ [1] as the incomplete verifier

Further reading on efficient abstract interpretation: elina.ethz.ch
[a] Making Numerical Program Analysis Fast, PLDI'15
[b] Fast Polyhedra Abstract Domain, POPL'17

## Key Idea: Best of both worlds Solvers + Abstract Interpretation

### DeepZ on a toy feedforward network with ReLU



### Solvers for computing refined bounds

After affine transformation in every feedforward layer (except the first):

- select neurons that can take positive values as candidates for refinement
- compute refined lower and upper bounds for the candidates using solvers
- solver instances are encoded as either MILP [2] or LP via DeepZ bounds

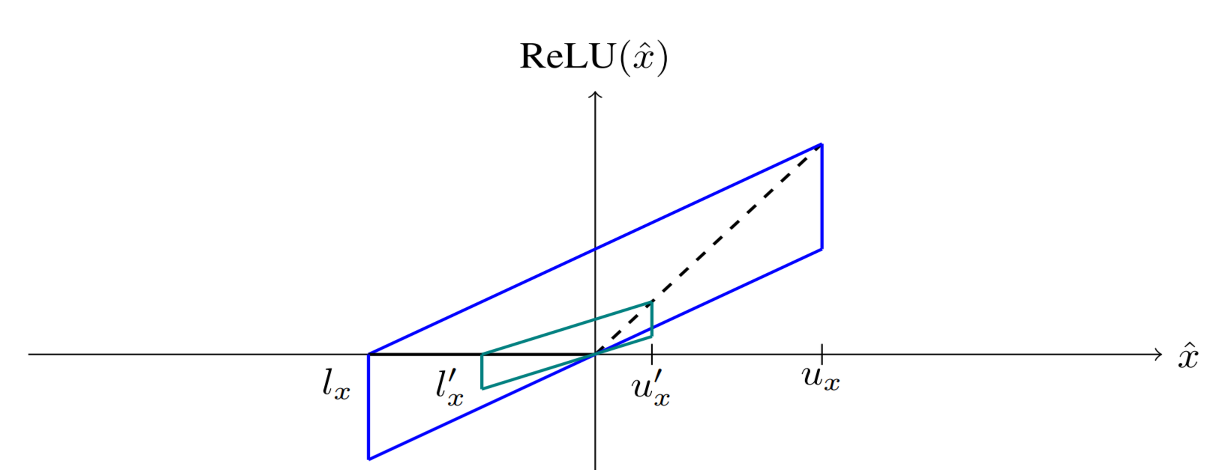
$$\begin{aligned} &l'_8 := min\ x_8 \\ s.t.: &x_8 = x_5 - x_6 - 1, \\ &0 \leq x_5 \leq x_3 - l_3 \cdot (1 - a_3), \\ &0 \leq x_6 \leq x_4 - l_4 \cdot (1 - a_4), \\ &x_3 \leq x_5 \leq u_3 \cdot a_3, x_4 \leq x_6 \leq u_4 \cdot a_4, \\ &x_3 = x_1 + x_2, x_4 = x_1 - x_2, \\ &0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1, \\ &a_3, a_4 \in \{0, 1\}. \end{aligned}$$

MILP formulation

$$\begin{aligned} &l'_8 := min\ x_8 \\ s.t.: &x_8 = x_5 - x_6 - 1, \\ &0 \leq x_5 \leq \frac{u_3}{u_3 - l_3} \cdot x_3 - \frac{l_3 \cdot u_3}{u_3 - l_3}, \\ &0 \leq x_6 \leq \frac{u_4}{u_4 - l_4} \cdot x_4 - \frac{l_4 \cdot u_4}{u_4 - l_4}, \\ &x_3 \leq x_5, x_4 \leq x_6 \\ &x_3 = x_1 + x_2, x_4 = x_1 - x_2, \\ &0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1. \end{aligned}$$

LP formulation

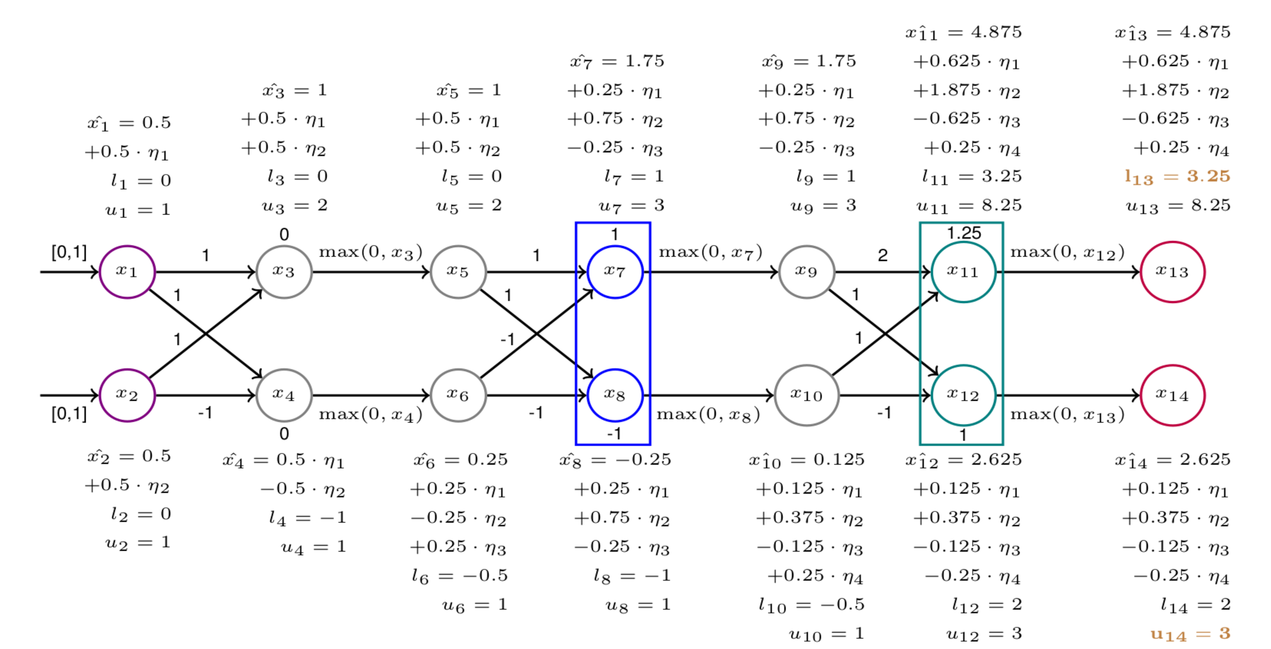
### Our refined ReLU transformer




ReLU transformers, computing an affine form. Here, $l_x$, $u_x$ are the original bounds, whereas $l'_x$, $u'_x$ are the refined bounds. The slope of the two non-vertical parallel blue lines is $\lambda = u_x/(u_x - l_x)$ and the slope of the two non-vertical parallel green lines is $\lambda' = u'_x/(u'_x - l'_x)$. The blue parallelogram is for computing the output affine form in DeepZ, whereas the green parallelogram is for computing the output of the refined ReLU transformer considered in this work.

## RefineZono: Our system for neural network robustness

### Our approach on the toy network



### End to end implementation

- Anytime relaxation
  - refine $\theta$ fraction of neurons in a layer with a timeout $T$ for the solver
  - refine $\delta \in [0, 1 - \theta]$ with a timeout of $\beta.\bar{T}$
  - $\bar{T}$ is the average time for refining $\theta$ fraction of neurons
- Neuron selection heuristic for $\theta$ fraction
  - neurons are sorted by width and the sum of absolute output weights
  - ranks of neurons in both orders are added
  - $\theta$ fraction of neurons with the smallest rank sum are selected
- Refine $k_{MILP}$ layers with MILP and $k_{LP}$ layers with LP
- Implementation is publicly available at safeai.ethz.ch as part of ERAN

### Neural networks

| Dataset | Model | Type | #Neurons | #Layers | Defense |
|---|---|---|---|---|---|
| MNIST | 3 × 50 | feedforward | 160 | 3 | None |
| | 5 × 100 | feedforward | 510 | 5 | None |
| | 6 × 100 | feedforward | 610 | 6 | None |
| | 9 × 100 | feedforward | 910 | 9 | None |
| | 6 × 200 | feedforward | 1,210 | 6 | None |
| | 9 × 200 | feedforward | 1,810 | 9 | None |
| | ConvSmall | convolutional | 3,604 | 3 | DiffAI [3] |
| | ConvBig | convolutional | 34,688 | 6 | DiffAI |
| | ConvSuper | convolutional | 88,500 | 6 | DiffAI |
| CIFAR10 | 6 × 100 | feedforward | 610 | 6 | None |
| | ConvSmall | convolutional | 4,852 | 3 | DiffAI |
| ACAS Xu | 6 × 50 | feedforward | 305 | 6 | None |

### Evaluation

- 3 × 50 FNN and all CNNs on a 2.6 GHz 14 core Intel Xeon CPU E5-2690
- All remaining FNNs on a 3.3 GHz 10 core Intel i9-7900X Skylake CPU
- Benchmarks:
  - property 9 defined in [5] for the ACAS Xu network
  - correctly classified images among the first 100 test images for the rest

## Results with RefineZono: State-of-the-art precision and scalability

### Complete verification

MNIST 3 × 50 Network

Certify with DeepZ first, if it fails then formulate certification as MILP using per-neuron bounds produced by DeepZ

| $\epsilon$ | [2] with Intervals | [2] with LP | RefineZono |
|---|---|---|---|
| 0.03 | 123 sec | 35 sec | 28 sec |

ACAS Xu Network

- Uniformly divide the input region into 6,300 smaller regions
- Run complete certification with RefineZono on each region separately

| Reluplex [5] | Neurify [4] | RefineZono |
|---|---|---|
| > 32 hours | 921 sec | 227 sec |

### Incomplete verification

- RefineZono vs. state-of-the-art incomplete verifiers
  - DeepZ [1]
  - DeepPoly [6]
- Complete verifiers do not scale on these benchmarks
- We chose values of parameters $\theta, T, \delta, \beta, k_{MILP}, k_{LP}$ offering best tradeoff between performance and precision for each network

MNIST Networks

| Model | $\epsilon$ | DeepZ | | DeepPoly | | RefineZono | |
|---|---|---|---|---|---|---|---|
| | | %✅ | time(s) | %✅ | time(s) | %✅ | time(s) |
| 5 × 100 | 0.07 | 38 | 0.6 | 53 | 0.3 | 53 | 381 |
| 6 × 100 | 0.02 | 31 | 0.6 | 47 | 0.2 | 67 | 194 |
| 9 × 100 | 0.02 | 28 | 1.0 | 44 | 0.3 | 59 | 246 |
| 6 × 200 | 0.015 | 13 | 1.8 | 32 | 0.5 | 39 | 567 |
| 9 × 200 | 0.015 | 12 | 3.7 | 30 | 0.9 | 38 | 826 |
| ConvSmall | 0.12 | 7 | 1.4 | 13 | 6.0 | 21 | 748 |
| ConvBig | 0.2 | 79 | 7 | 78 | 61 | 80 | 193 |
| ConvSuper | 0.1 | 97 | 133 | 97 | 400 | 97 | 665 |

CIFAR10 Networks

| Model | $\epsilon$ | DeepZ | | DeepPoly | | RefineZono | |
|---|---|---|---|---|---|---|---|
| | | %✅ | time(s) | %✅ | time(s) | %✅ | time(s) |
| 6 × 100 | 0.0012 | 31 | 4 | 46 | 0.6 | 46 | 765 |
| ConvSmall | 0.03 | 17 | 5.8 | 21 | 20 | 21 | 550 |


References:
[1] Fast and Effective Robustness Certification, NeurIPS'18
[2] Evaluating Robustness of Neural Networks with Mixed Integer Programming, ICLR'19
[3] Differentiable Abstract Interpretation for Provably Robust Neural Networks, ICML'18
[4] Efficient Formal Safety Analysis of Neural Networks, NeurIPS'18
[5] Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks, CAV'17
[6] An Abstract Domain for Certifying Neural Networks, POPL'19