Reliable and Interpretable Artificial Intelligence

Lecture 8: Certified Defenses

Martin Vechev

ETH Zurich

Fall 2020

SRILAB

http://www.sri.inf.ethz.ch

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Can certification methods benefit training?

Verifying networks which are not meant to be robust will certainly produce worse results (smaller epsilon provability) than verifying networks which are trained to be provably robust.

Note that there is a difference between training the network to be experimentally robust (e.g., PGD defense) vs. training the network to be provably robust (what we see next).

So, can we then use certification for training the network to be robust?

Recall: PGD Defense

find	θ
minimize	$\rho(\theta)$
where	$\rho(\theta) = \mathbf{E}_{(\mathrm{x,y})\sim D}\left[\max_{x' \in S(x)} L(\theta, x', y)\right]$

Outer minimization problem

find $\boldsymbol{\theta}$ that minimizes the high loss, aiming to train a robust classifier

Inner maximization problem

find $\boldsymbol{x}'$ that achieves high loss

D is the underlying distribution

$\mathbf{E}$ is typically estimated with the empirical risk

$S(x)$ denotes the perturbation region around point x, that is, we want all points in $S(x)$ to classify the same as x . We can pick $S(x)$ to be:

$$S(x) = \{\, \boldsymbol{x}' \mid \|\boldsymbol{x} - \boldsymbol{x}'\|_{\infty} < \epsilon \}$$

Madry et.al, 2017

Lets Incorporate Provability

Outer minimization problem

Lets keep this part

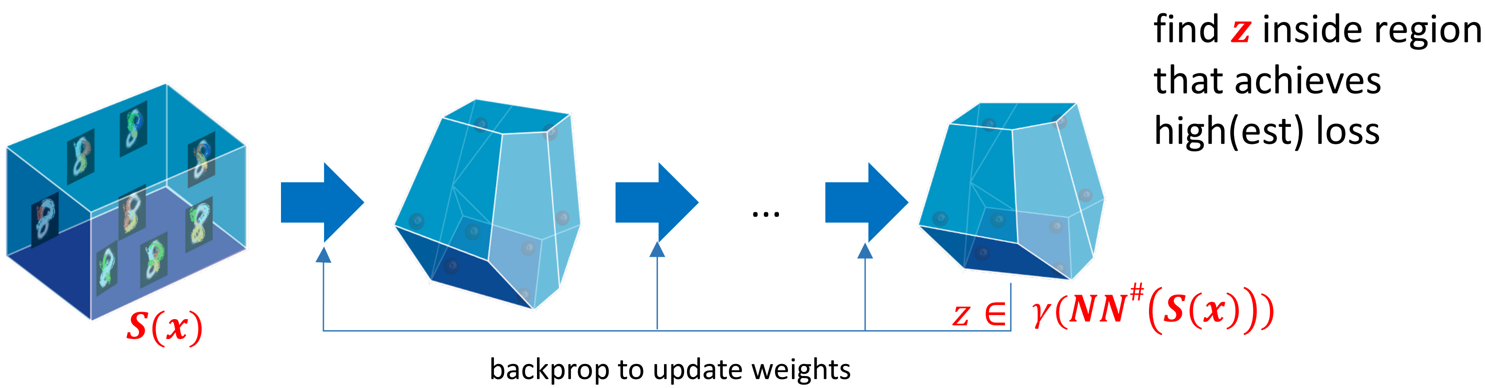
find	θ
minimize	$\rho(\theta)$
where	$\rho(\theta) = \mathbf{E}_{(\mathrm{x},\mathrm{y})\sim D}\left[\max_{z \in \gamma(NN^{\#}(S(x)))} L(\theta, z, y)\right]$

Inner maximization problem

find $\boldsymbol{z}$ that achieves high loss under abstraction

Differentiable Abstract Interpretation for Provably Robust Neural Networks
Mirman, Gehr, V. ICML 2018

Visualization of Certified Training



Essentially: **automatic differentiation of abstract interpretation**

Adversarial Training

find θ
minimize $\rho(\theta)$

where $\rho(\theta) = \mathbf{E}_{(x,y)\sim D}[\max_{\boldsymbol{x}' \in \boldsymbol{S}(\boldsymbol{x})} L(\theta, \boldsymbol{x}', y)]$

Find input $\boldsymbol{x}'$ that achieves high loss

Certified Defense

find θ
minimize $\rho(\theta)$

where $\rho(\theta) = \mathbf{E}_{(x,y)\sim D}[\max_{\boldsymbol{z} \in \gamma(\boldsymbol{NN}^{\#}(\boldsymbol{S}(\boldsymbol{x})))} L(\theta, \boldsymbol{z}, y)]$

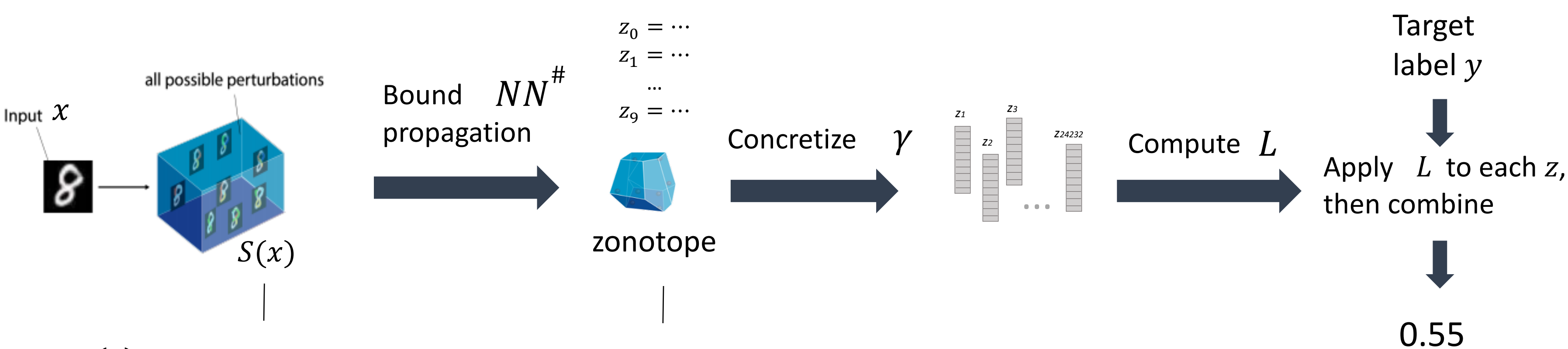
Find output $\boldsymbol{z}$ that achieves high loss (under abstraction)

Certified Defenses: General Method

$$\max_{z \in \gamma(NN^{\#}(S(x)))} L(\theta, z, y)$$

Let us examine the pattern in the concrete first.

The pattern works with any abstract relaxation.



$S(x)$ region captures possible perturbations, e.g., L_∞ ball

Here we picked Zonotope, but can be any relaxation

Differentiable Abstract Interpretation for Provably Robust Neural Networks
Mirman, Gehr, V. ICML 2018

Let us now pick a loss function L

$$L(\boldsymbol{z}, y) = \max_{q \neq y} \; (z_q - z_y)$$

target label

a vector of logits

q ranges over all possible labels

Certified Defenses with a given loss

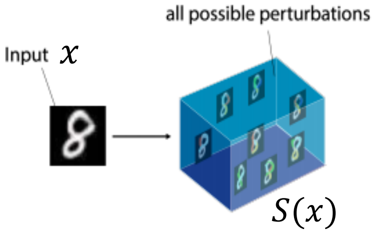
$$\max_{z \in \gamma(NN^{\#}(S(x)))} L(\theta, z, y)$$

Lets define L to be:

$$L(\mathbf{z}, y) = \max_{\mathrm{q} \neq y} (\mathrm{z_q} - \mathrm{z_y})$$

target label

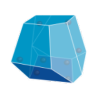
a vector of logits



$S(x)$ region captures possible perturbations, e.g., L_∞ ball

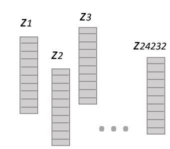
Bound propagation $NN^{\#}$

$z_0 = \cdots$
$z_1 = \cdots$
$\cdots$
$z_9 = \cdots$



zonotope

Concretize γ



Compute L

Target label y

Apply L to each z, then take max one

0.55

Differentiable Abstract Interpretation for Provably Robust Neural Networks
Mirman, Gehr, V. ICML 2018

Certified Defenses with a given loss

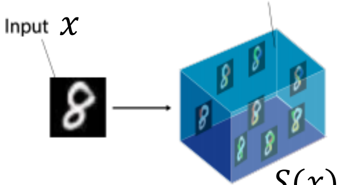
$$\max_{z \in \gamma(NN^{\#}(S(x)))} L(\theta, z, y)$$

Lets define L to be:

$$L(\mathbf{z}, y) = \max_{q \neq y} (z_q - z_y)$$

target label

a vector of logits

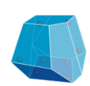


$S(x)$

$S(x)$ region captures possible perturbations, e.g., L_∞ ball

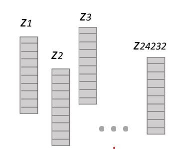
Bound propagation $NN^{\#}$

$z_0 = \cdots$
$z_1 = \cdots$
$\cdots$
$z_9 = \cdots$



zonotope

Concretize γ



Compute L

Target label y

Apply L to each z, then take max one

0.55

Key problem: set of vectors could be infinite or very large, so we cannot just enumerate.

How do we address this?

Certified Defenses in the abstract

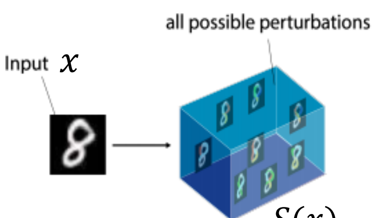
$$\max_{z \in \gamma(NN^{\#}(S(x)))} L(\theta, z, y)$$

Lets define L to be:

$$L(\mathbf{z}, y) = \max_{q \neq y} (z_q - z_y)$$

a vector of logits

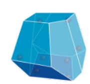
target label



$S(x)$ region captures possible perturbations, e.g., L_∞ ball

Bound propagation $NN^{\#}$

$z_0 = \cdots$
$z_1 = \cdots$
...
$z_9 = \cdots$

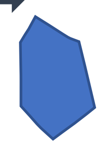


zonotope

$d_0 = z_0 - z_y$

...

$d_9 = z_9 - z_y$



$z_0 = \cdots$
...
$z_9 = \cdots$
$d_0 = \cdots$
...
$d_9 = \cdots$

e.g. [0.1, 0.3]

$\max(box(d_0))$

0.3

e.g. [-0.4, -0.1]

$\max(box(d_9))$

-0.1

0.3

This is an affine transform which is exact for zonotope.

Plug in lower and upper bounds of epsilons into the expression to get the interval.

Defining $\max(box(d_0))$

$$d_0 = 3 + \epsilon_1 - 2\epsilon_2$$

ϵ_1 and ϵ_2 range over $[-1,1]$

$$d_{-1,-1} = 3 - 1 + 2 = 4$$

plug in -1 for both

$$d_{-1,1} = 3 - 1 - 2 = 0$$

plug in -1 for ϵ_1, and 1 for ϵ_2

$$d_{1,-1} = 3 + 1 + 2 = 6$$

plug in 1 for ϵ_1, and -1 for ϵ_2

$$d_{1,1} = 3 + 1 - 2 = 2$$

plug in 1 for both

$d_{box} = [0, 6]$ $\quad \max(d_{box}) = 6$

Of course, to compute max, rather than enumerating combinations, we pick the value for the ϵ depending on its sign in d_0. If positive, pick 1, if negative, pick -1.

Let us keep the same pattern but now pick a different loss, the cross-entropy loss CE

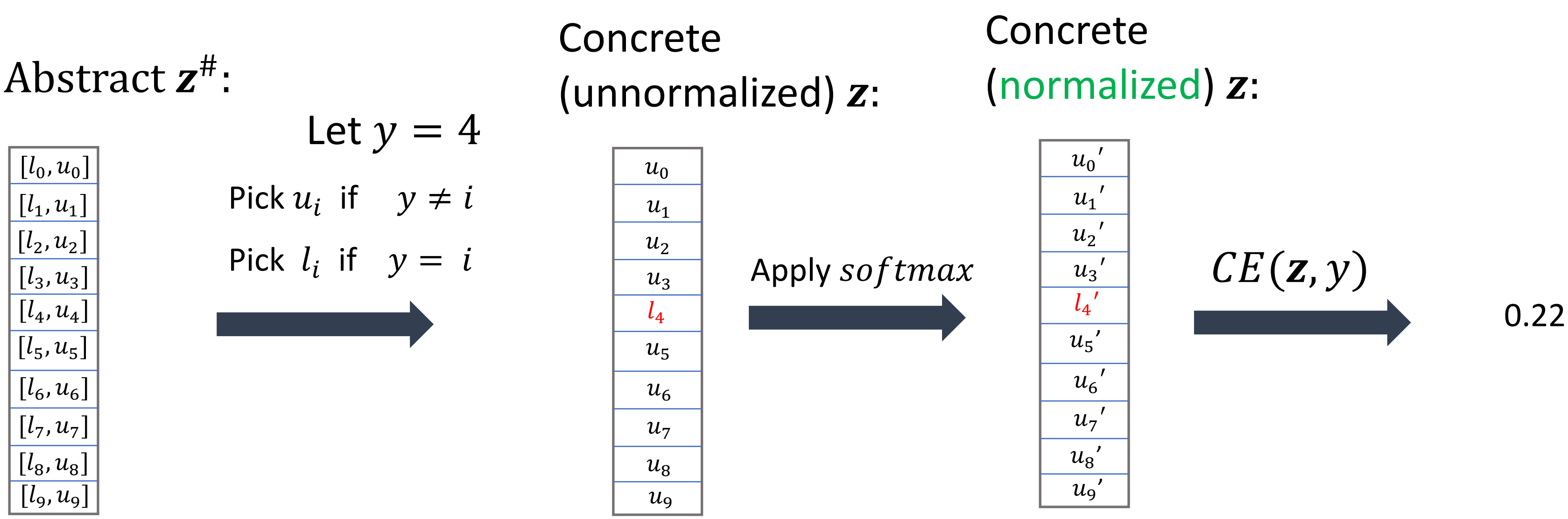
$$L(\mathbf{z}, y) = CE(\mathbf{z}, y)$$

This is in the concrete, but we need to work in the abstract.

On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models, 2018
Gowal, Dvijotham, Stanforth, Bunel, Qin, Uesato, Arandjelovic, Mann, Kohli

Let us keep the same pattern but now pick a different loss, the cross-entropy loss CE

$$L(\boldsymbol{z}, y) = CE(\boldsymbol{z}, y)$$



Few additional tricks in practice

- Annealing on the size of $S(x)$ – start with small region around x (small ϵ) and gradually grow it during training. This was found to be most helpful heuristic.

- Even though the whole propagation is done via Box, IBP processes the last linear layer exactly (e.g., zonotope).

- Dynamically weighing-in the standard CE loss and the correctness CE loss.

These and more implemented in the DiffAI certified training system: https://github.com/eth-sri/diffai

Key observations when using DiffAI scheme in practice:

Using cheap relaxations (e.g., Box) scales to large networks. But the problem is, it introduces a lot of garbage (infeasible points) in the final output shape, meaning the deeper the network is, the more the capacity increases (potential for higher accuracy), but the more we are training w.r.t. assigning labels to garbage points. Thus, typically training with Box scales but accuracy drops substantially.

Naturally we would like to reduce the infeasible points w.r.t to which we are training. However, it turned out that more precise relaxations (e.g., Zonotope) may lead to worse results than Box! This is an unintuitive pathological situation where more precise relaxations during training do not actually bring better results in provability and where further loss tweaking is not enough.

Question I:

Is there a network with **perfect accuracy** s.t.

analyzing it with Box is **exact**?

Question I:

Is there a network with **perfect accuracy** s.t.

analyzing it with Box is **exact**?

Yes.

Universal Approximation with Certified Networks, ICLR'2020
Baader, Mirman, Vechev

(answers the existence question, but construction is still impractical, more work needed)

Question II:

Why are better relaxations not producing better results?

Question II:

Why are better relaxations not producing better results?

Hypothesis: More **complex abstractions** lead to more **difficult optimization problems**.

Why? Intuitively, a relatively **small number of weights** in the network need to control complex relaxations with **many more parameters** (than weights). This is quite unlike normal training.

Question II:

Why are better relaxations not producing better results?

We need a training method that produces a simpler optimization problem

Reminder: optimization problems

Adversarial Training

find θ
minimize $\rho(\theta)$

where $\rho(\theta) = \mathbf{E}_{(x,y)\sim D}[\max_{\boldsymbol{x}' \in \boldsymbol{S}(\boldsymbol{x})} L(\theta, \boldsymbol{x}', y)]$

Find input $\boldsymbol{x}'$ that achieves high loss

Good accuracy

Worse verifiability

Easier optimization

Certified Defense

find θ
minimize $\rho(\theta)$

where $\rho(\theta) = \mathbf{E}_{(x,y)\sim D}[\max_{\boldsymbol{z} \in \boldsymbol{NN}^{\#}(\boldsymbol{S}(\boldsymbol{x}))} L(\theta, \boldsymbol{z}, y)]$

Find output $\boldsymbol{z}$ that achieves high loss (under abstraction)

Worse accuracy

Good verifiability

Harder optimization

Adversarial Training and Provable Defenses: Bridging the Gap

COLT: Balunovic and V, ICLR'20 (oral)

COLT: stands for Convex Layerwise Adversarial Training

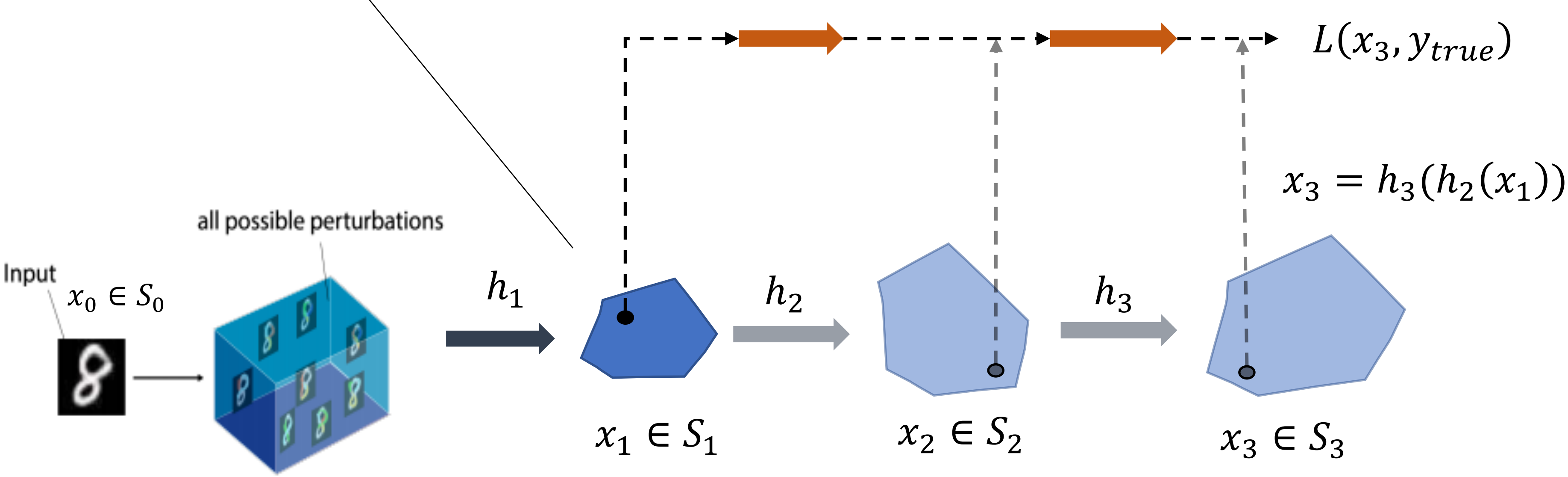
Key challenge:

Find point $x_1 \in S_1$ such that
loss L in the final layer is maximized
Need **projections** again!

Optimization problem (after layer 1):

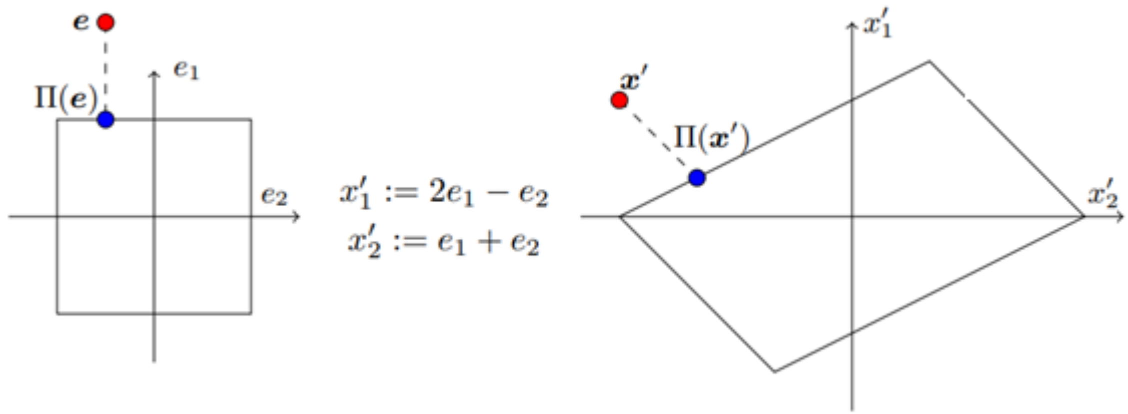
$$\min_{\theta} \max_{x_1 \in S_1} L(\, h_3(h_2(x_1)), y_{true})$$

(high-level view: PGD training but with shapes arising in the middle of the network)



https://github.com/eth-sri/colt

Instantiation of COLT with Zonotope



Solved via standard PGD (projection on Box)

Projection on Zonotope solved via projection on Box (via change of variables)

$$\max_{e \in [0,1]^k} L(Ae, y_{true}) = \max_{x \in Z} L(x, y_{true})$$

Zonotope relaxation (reminder): $Z = A \cdot [-1, 1]^k$

Each $x \in Z$ has a corresponding $e \in [-1, 1]^k$ such that $x = Ae$

Lecture Summary

Certified defenses: using relaxations during training in order to obtain more provable networks

We introduced the DiffAI method and showed how to instantiate it with two loss functions and two relaxations

The DiffAI method and its follow-ups can produce complex optimization problems. Towards that, we introduced COLT, a certified defense that **combines adversarial training and relaxations to produce a simpler optimization problem** where better relaxations can lead to better results.