Reliable and Interpretable Artificial Intelligence

Lecture 7: DeepPoly convex relaxation + Abstract Interpretation

Martin Vechev

ETH Zurich

Fall 2020

SRILAB

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

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

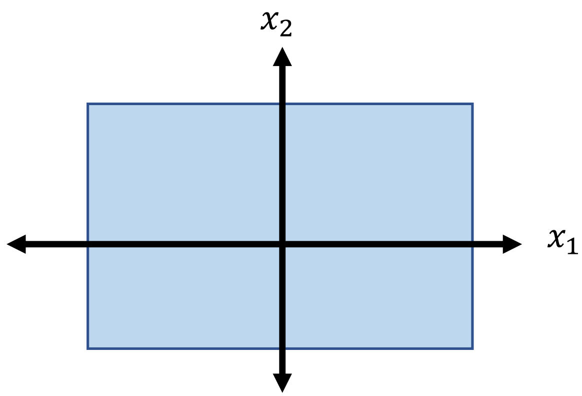
So far in certification…

- Simple box (incomplete) certification, complete certification via MILP solvers and how to use Box to speed-up MILP.

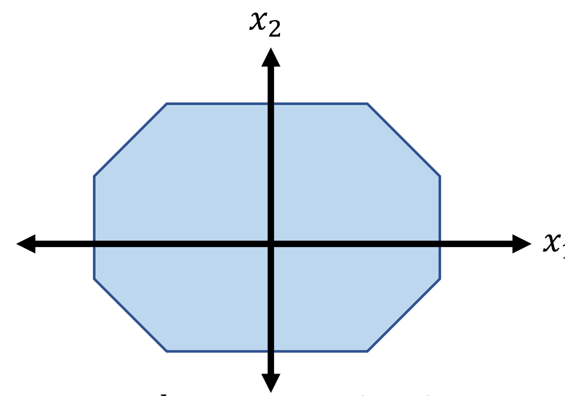
- A more involved and precise relaxation, the Zonotope, which is exact for affine transforms but approximates ReLU, also an incomplete method.

Today: another convex relaxation method, which aims to be more precise than Zonotope when approximating ReLUs.

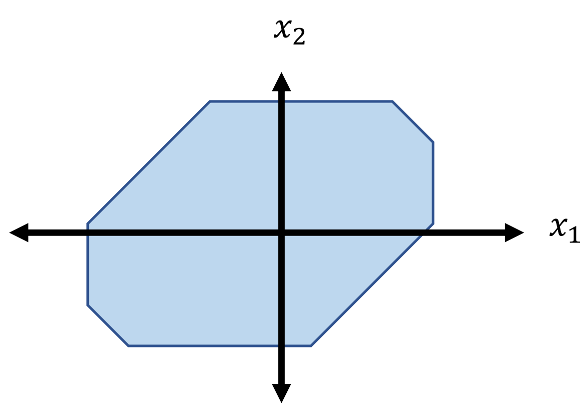
Popular numerical relaxations



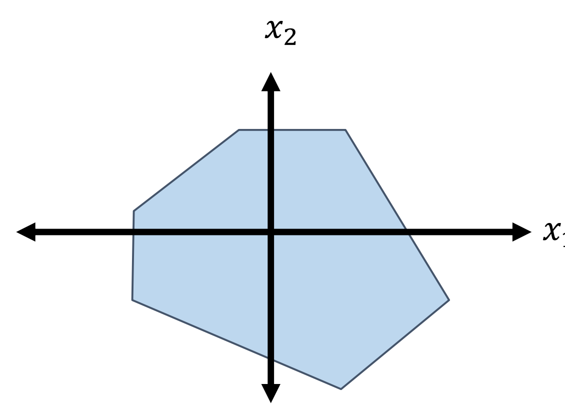
Box: $l_i \leq x_i \leq u_i$



Octagon: $l_i \leq x_i \leq u_i, \pm x_i \pm x_j \leq c_{ij}$



Zonotope: $\hat{x}_i = \alpha_0 + \sum_i \alpha_i \epsilon_i \,, \epsilon_i \in [-1,1]$



Polyhedra: $\sum_i a_i x_i \leq c$

DeepPoly convex relaxation: The Shape

[Singh et. al, POPL'19]

Shape:

for each x_i, we keep:

- An interval constraint: lower bound $l_i \leq x_i$ and upper bound $x_i \leq u_i$

- Two relational constraints: $a_i^{\leq} \leq x_i$ and $x_i \leq a_i^{\geq}$
  where the expressions $a_i^{\leq}, a_i^{\geq}$ are of the form $\sum_j w_j \cdot x_j + v$

DeepPoly convex relaxation: The Shape

[Singh et. al, POPL'19]

Shape:

for each x_i, we keep:

- An interval constraint: lower bound $l_i \leq x_i$ and upper bound $x_i \leq u_i$
- Two relational constraints: $a_i^{\leq} \leq x_i$ and $x_i \leq a_i^{\geq}$
  where the expressions $a_i^{\leq}, a_i^{\geq}$ are of the form $\sum_j w_j \cdot x_j + v$

- less precise than Polyhedra, restriction needed to ensure scalability
- captures affine transformation precisely
- custom transformers for ReLU, sigmoid, tanh, and maxpool activations

DeepPoly convex relaxation: The Shape

[Singh et. al, POPL'19]

Shape:

for each x_i, we keep:

- An interval constraint: lower bound $l_i \leq x_i$ and upper bound $x_i \leq u_i$
- Two relational constraints: $a_i^{\leq} \leq x_i$ and $x_i \leq a_i^{\geq}$
  where the expressions $a_i^{\leq}, a_i^{\geq}$ are of the form $\sum_j w_j \cdot x_j + v$

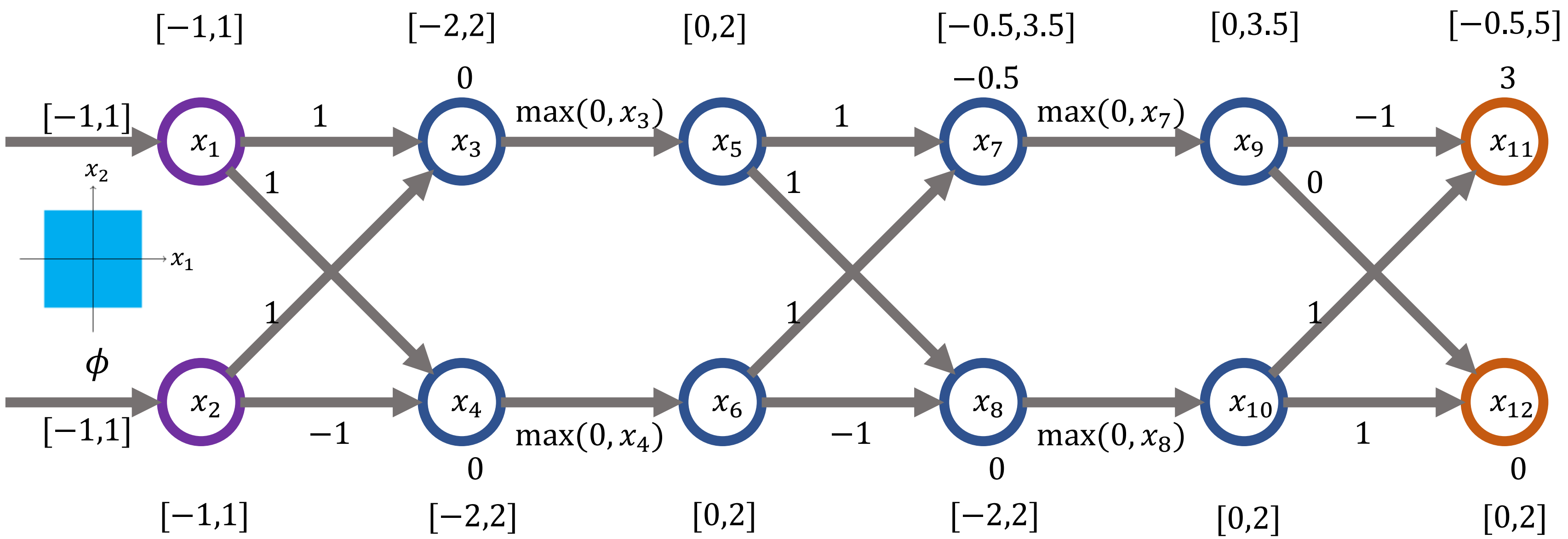
- less precise than Polyhedra, restriction needed to ensure scalability
- captures affine transformation precisely
- custom transformers for ReLU, sigmoid, tanh, and maxpool activations

n: #neurons, m: #constraints

w_{max}: max #neurons in a layer, L: #layers

Transformer	Polyhedra	DeepPoly
Affine	$O(nm^2)$	$O(w_{max}^2 L)$
ReLU	$O(\exp(n,m))$	$O(1)$

Box relaxation (scalable but imprecise)



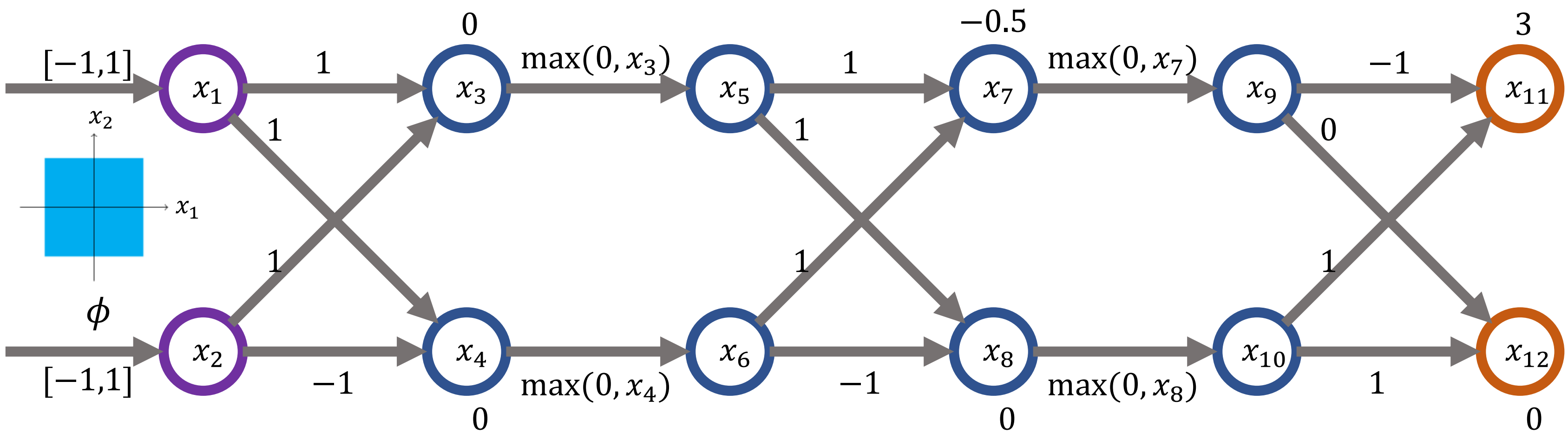
ψ: we want to prove that $x_{11} > x_{12}$ for all values of x_1, x_2 in the input set

Certification with Box fails as it cannot capture relational information

DeepPoly relaxation

$$x_1 \geq -1, \quad x_1 \leq 1, \quad l_1 = -1, \quad u_1 = 1$$

$$x_3 \geq x_1 + x_2, \quad x_3 \leq x_1 + x_2, \quad l_3 = -2, \quad u_3 = 2$$



$$x_2 \geq -1, \quad x_2 \leq 1, \quad l_2 = -1, \quad u_2 = 1$$

$$x_4 \geq x_1 - x_2, \quad x_4 \leq x_1 - x_2, \quad l_4 = -2, \quad u_4 = 2$$

ReLU activation: $x_j \coloneqq \max(0, x_i)$

Single-neuron transformer for $x_j \coloneqq max(0, x_i)$ that uses l_i, u_i

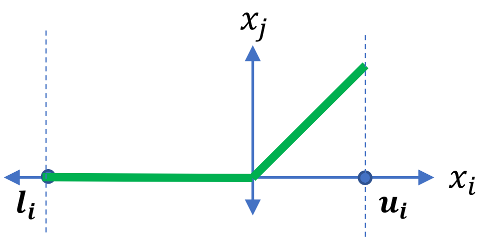
ReLU activation: $x_j \coloneqq \max(0, x_i)$

Single-neuron transformer for $x_j \coloneqq max(0, x_i)$ that uses l_i, u_i

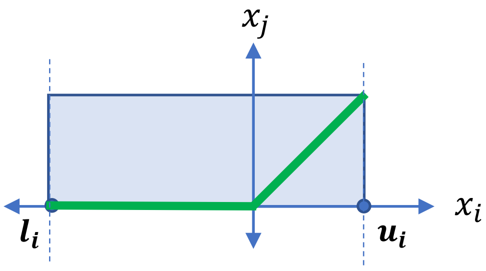
- $if\ u_i \leq 0$: $a_j^{\leq} = a_j^{\geq} = 0, l_j = u_j = 0$ (strictly negative)
- $if\ l_i \geq 0$: $a_j^{\leq} = a_j^{\geq} = x_i, l_j = l_i, u_j = u_i$ (strictly positive)
- $if\ l_i < 0\ and\ u_i > 0$ (crossing ReLU)

Lets discuss the crossing ReLU activation next

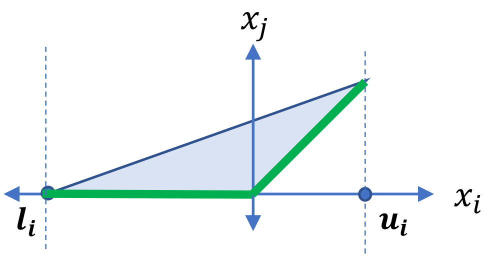
ReLU activation: $x_j := \max(0, x_i)$



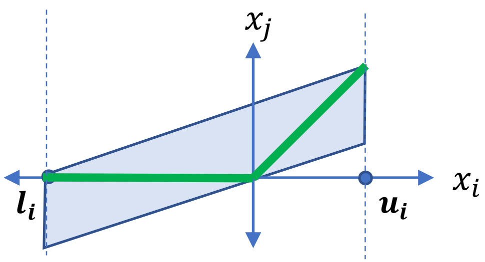
Exact [Katz et al., CAV'17]



Box [Gehr el al. S&P'18]

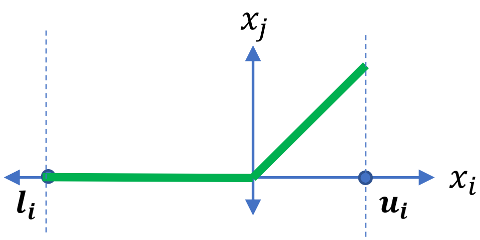


Triangle [Ehlers et al. ATVA'17]

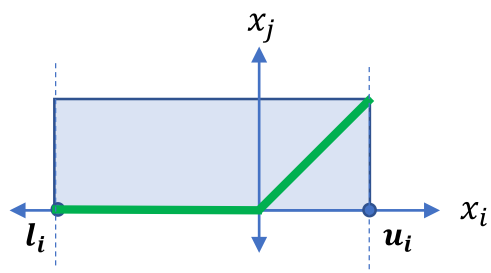


Wong et al. [ICML'18]
FastLin [ICML'18]
DeepZ [NeurIPS'18]

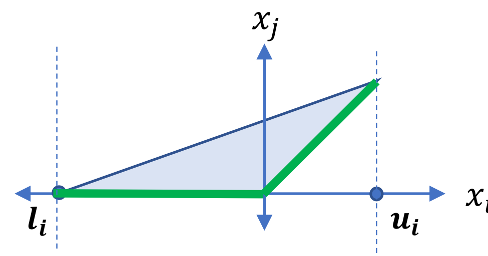
ReLU activation: $x_j := \max(0, x_i)$



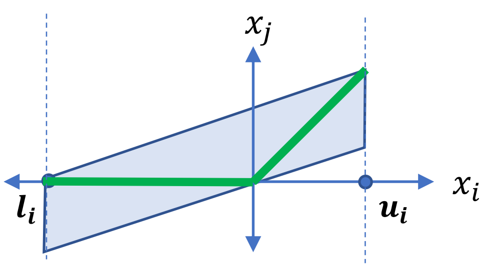
Exact [Katz et al., CAV'17]



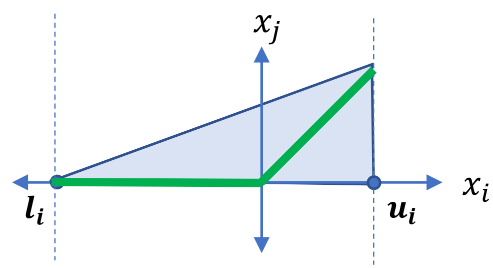
Box [Gehr el al. S&P'18]



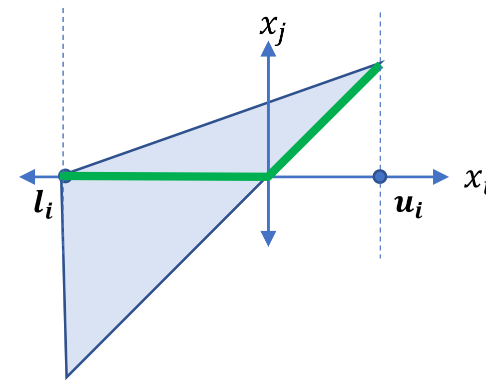
Triangle [Ehlers et al. ATVA'17]



Wong et al. [ICML'18]
FastLin [ICML'18]
DeepZ [NeurIPS'18]



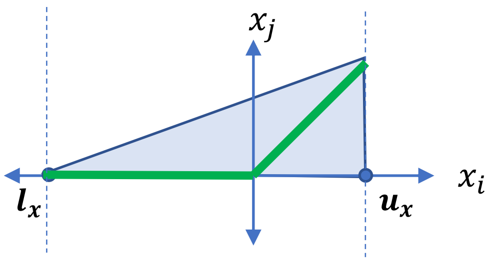
CROWN [NeurIPS'18]
DeepPoly [POPL'19]



CROWN [NeurIPS'18]
DeepPoly [POPL'19]

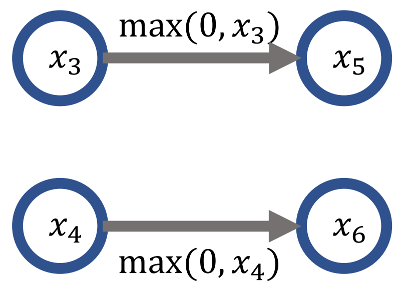
- The choice of DeepPoly shape depends on area (heuristic)
- Note that both approximations are smaller area-wise than the Zonotope

Applying DeepPoly ReLU relaxation



DeepPoly [POPL'19]

$$\begin{gathered} x_3 \geq x_1 + x_2, \\ x_3 \leq x_1 + x_2, \\ l_3 = -2, \\ u_3 = 2 \end{gathered}$$



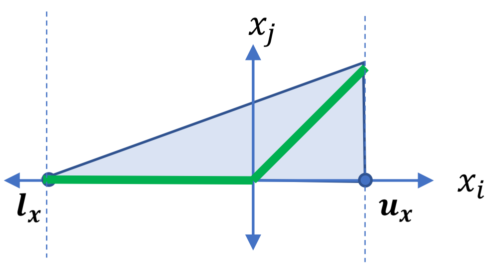
$$\begin{gathered} x_4 \geq x_1 - x_2, \\ x_4 \leq x_1 - x_2, \\ l_4 = -2, \\ u_4 = 2 \end{gathered}$$

Constant runtime with DeepPoly

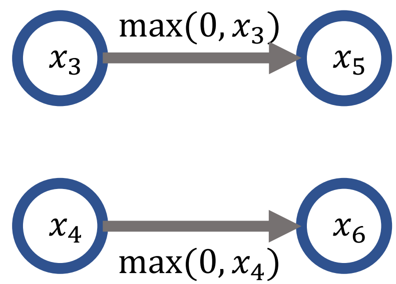
Applying DeepPoly ReLU relaxation

$$x_3 \geq x_1 + x_2,\quad x_3 \leq x_1 + x_2,\quad l_3 = -2,\quad u_3 = 2$$

$$x_5 \geq 0,\quad x_5 \leq 0.5 \cdot x_3 + 1,\quad l_5 = 0,\quad u_5 = 2$$



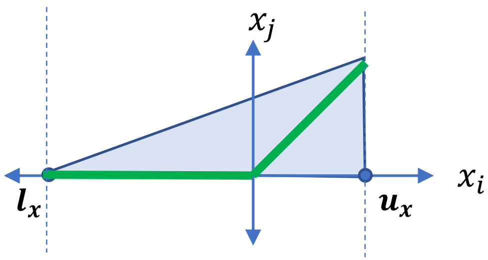
DeepPoly [POPL'19]



$$x_4 \geq x_1 - x_2,\quad x_4 \leq x_1 - x_2,\quad l_4 = -2,\quad u_4 = 2$$

Constant runtime with DeepPoly

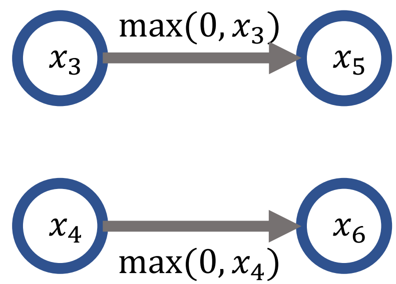
Applying DeepPoly ReLU relaxation



DeepPoly [POPL'19]

$$x_3 \geq x_1 + x_2,\quad x_3 \leq x_1 + x_2,\quad l_3 = -2,\quad u_3 = 2$$

$$x_5 \geq 0,\quad x_5 \leq 0.5 \cdot x_3 + 1,\quad l_5 = 0,\quad u_5 = 2$$



$$x_4 \geq x_1 - x_2,\quad x_4 \leq x_1 - x_2,\quad l_4 = -2,\quad u_4 = 2$$

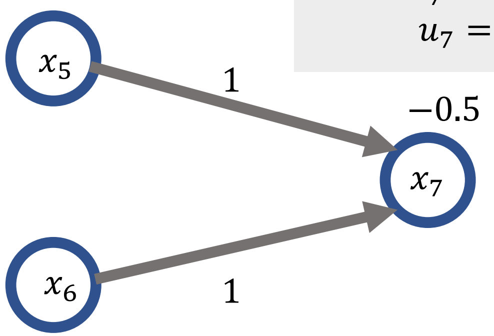
$$x_6 \geq 0,\quad x_6 \leq 0.5 \cdot x_4 + 1,\quad l_6 = 0,\quad u_6 = 2$$

Constant runtime with DeepPoly

Affine transformation after ReLU

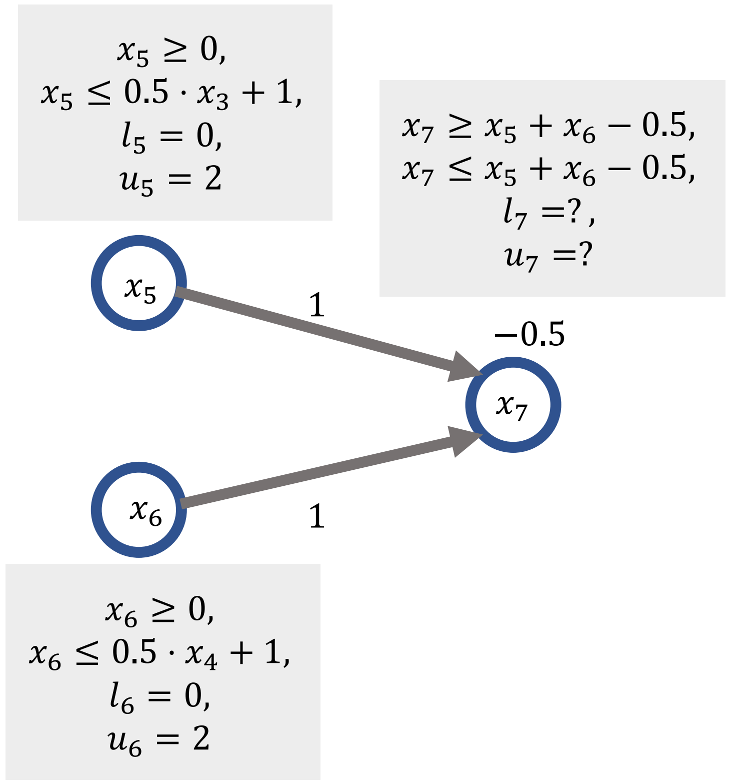
$$x_5 \geq 0,\quad x_5 \leq 0.5 \cdot x_3 + 1,\quad l_5 = 0,\quad u_5 = 2$$

$$x_7 \geq x_5 + x_6 - 0.5,\quad x_7 \leq x_5 + x_6 - 0.5,\quad l_7 = ?,\quad u_7 = ?$$



$$x_6 \geq 0,\quad x_6 \leq 0.5 \cdot x_4 + 1,\quad l_6 = 0,\quad u_6 = 2$$

Affine transformation after ReLU



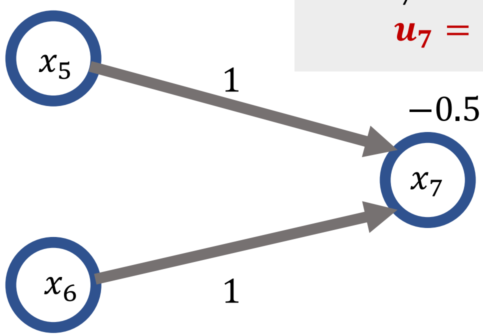
Computing upper bound for neuron x_7:

$$x_7 \leq x_5 + x_6 - 0.5$$
$$\leq 2 + 2 - 0.5 = 3.5$$

Affine transformation after ReLU

$$x_5 \geq 0,\quad x_5 \leq 0.5 \cdot x_3 + 1,\quad l_5 = 0,\quad u_5 = 2$$

$$x_7 \geq x_5 + x_6 - 0.5,\quad x_7 \leq x_5 + x_6 - 0.5,\quad l_7 = -0.5,\quad \boldsymbol{u_7 = 3.5}$$



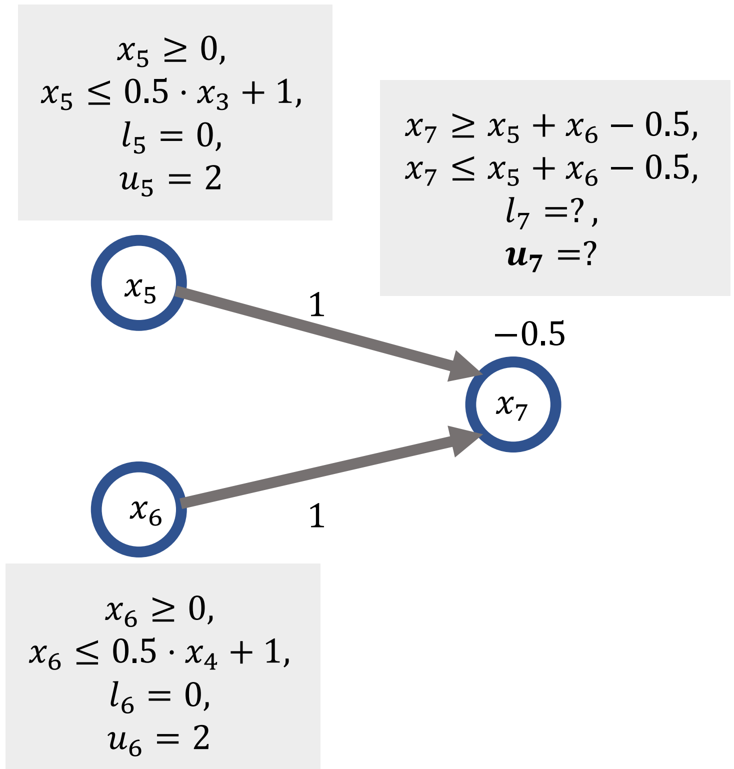
$$x_6 \geq 0,\quad x_6 \leq 0.5 \cdot x_4 + 1,\quad l_6 = 0,\quad u_6 = 2$$

Computing upper bound for neuron x_7:

$$x_7 \leq x_5 + x_6 - 0.5$$
$$\leq 2 + 2 - 0.5 = 3.5$$

Imprecise upper bound u_7 by substituting u_5, u_6 for x_5 and x_6 in $a_7^{\geq}$

Backsubstitution



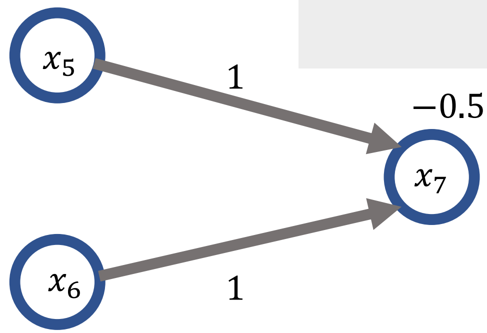
Replace the bounds for x_7 using the ones from the previous layer

Backsubstitution

$$x_5 \geq 0,$$
$$x_5 \leq 0.5 \cdot x_3 + 1,$$
$$l_5 = 0,$$
$$u_5 = 2$$

$$x_7 \geq -0.5,$$
$$x_7 \leq 0.5 \cdot x_3 + 0.5 \cdot x_4 + 1.5,$$
$$l_7 =?,$$
$$u_7 =?$$

Replace the bounds for x_7 using the ones from the previous layer



$$x_6 \geq 0,$$
$$x_6 \leq 0.5 \cdot x_4 + 1,$$
$$l_6 = 0,$$
$$u_6 = 2$$

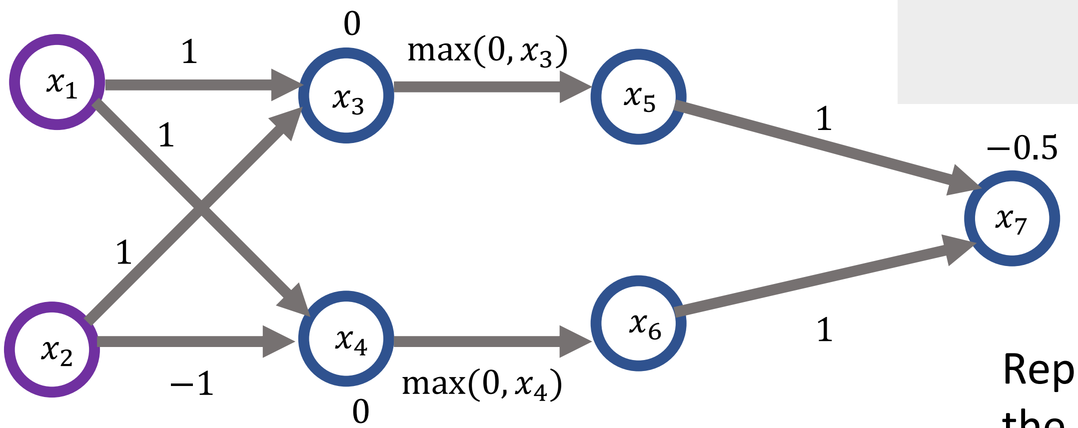
Backsubstitution

$x_1 \geq -1,$
$x_1 \leq 1,$
$l_1 = -1,$
$u_1 = 1$

$x_3 \geq x_1 + x_2,$
$x_3 \leq x_1 + x_2,$
$l_3 = -2,$
$u_3 = 2$

$x_5 \geq 0,$
$x_5 \leq 0.5 \cdot x_3 + 1,$
$l_5 = 0,$
$u_5 = 2$

$x_7 \geq -0.5,$
$x_7 \leq 0.5 \cdot x_3 + 0.5 \cdot x_4 + 1.5,$
$l_7 = ?,$
$u_7 = ?$



$x_2 \geq -1,$
$x_2 \leq 1,$
$l_2 = -1,$
$u_2 = 1$

$x_4 \geq x_1 - x_2,$
$x_4 \leq x_1 - x_2,$
$l_4 = -2,$
$u_4 = 2$

$x_6 \geq 0,$
$x_6 \leq 0.5 \cdot x_4 + 1,$
$l_6 = 0,$
$u_6 = 2$

Replace the bounds for x_7 using the ones from the previous layer:

$$
\begin{aligned}
x_7 &\leq 0.5x_3 + 0.5x_4 + 1.5 \\
&\leq 0.5(x_1 + x_2) + 0.5(x_1 - x_2) + 1.5 \\
&= x_1 + 1.5 \leq 2.5
\end{aligned}
$$

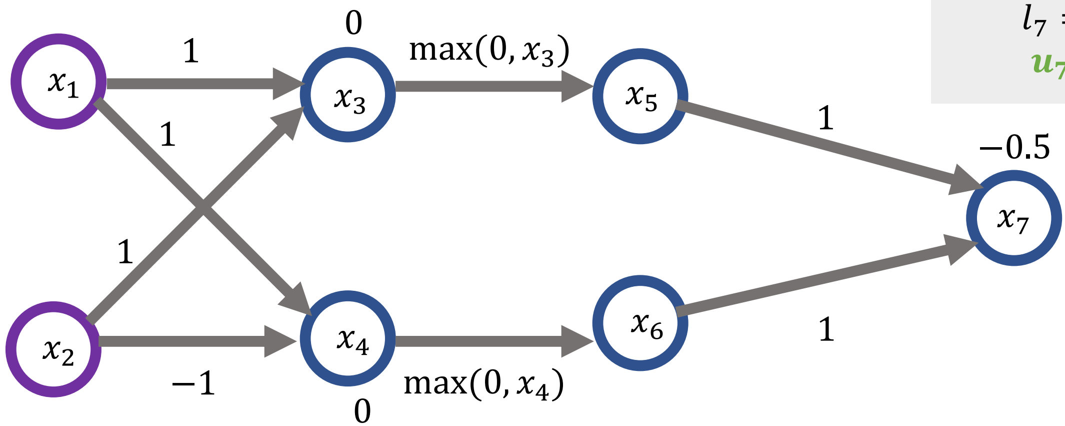
Backsubstitution

$x_1 \geq -1,$
$x_1 \leq 1,$
$l_1 = -1,$
$u_1 = 1$

$x_3 \geq x_1 + x_2,$
$x_3 \leq x_1 + x_2,$
$l_3 = -2,$
$u_3 = 2$

$x_5 \geq 0,$
$x_5 \leq 0.5 \cdot x_3 + 1,$
$l_5 = 0,$
$u_5 = 2$

$x_7 \geq -0.5,$
$x_7 \leq x_1 + 1.5,$
$l_7 = -0.5,$
$\mathbf{u_7 = 2.5}$



$x_2 \geq -1,$
$x_2 \leq 1,$
$l_2 = -1,$
$u_2 = 1$

$x_4 \geq x_1 - x_2,$
$x_4 \leq x_1 - x_2,$
$l_4 = -2,$
$u_4 = 2$

$x_6 \geq 0,$
$x_6 \leq 0.5 \cdot x_4 + 1,$
$l_6 = 0,$
$u_6 = 2$

Affine transformation with backsubstitution is pointwise, complexity: $O\left(w_{max}^2 L\right)$

$$x_1 \geq -1,\quad x_1 \leq 1,\quad l_1 = -1,\quad u_1 = 1$$

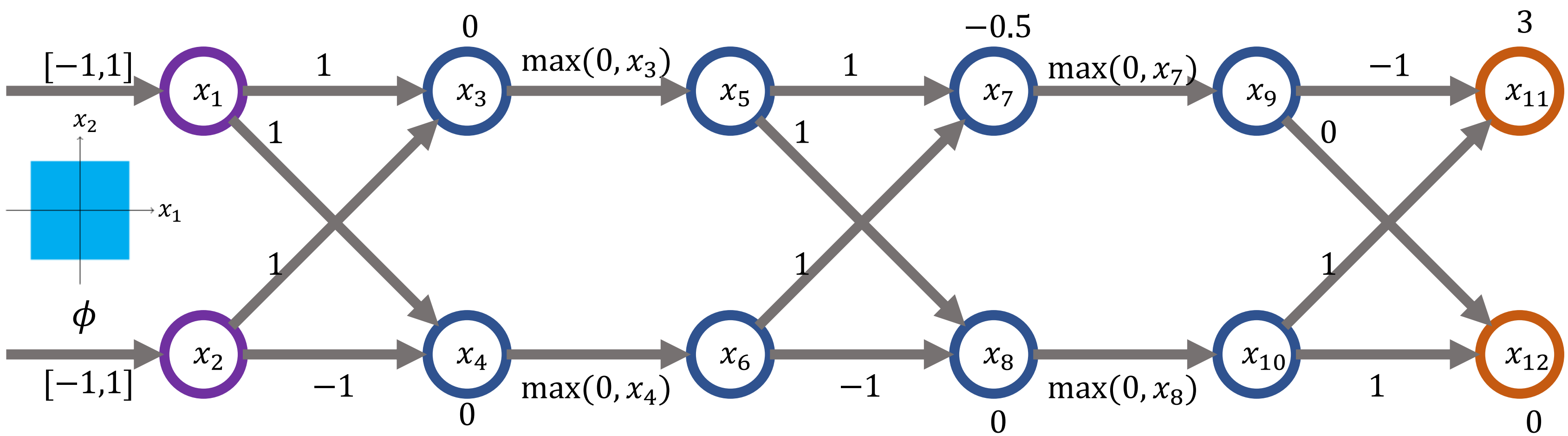
$$x_3 \geq x_1 + x_2,\quad x_3 \leq x_1 + x_2,\quad l_3 = -2,\quad u_3 = 2$$

$$x_5 \geq 0,\quad x_5 \leq 0.5 \cdot x_3 + 1,\quad l_5 = 0,\quad u_5 = 2$$

$$x_7 \geq x_5 + x_6 - 0.5,\quad x_7 \leq x_5 + x_6 - 0.5,\quad l_7 = -0.5,\quad u_7 = 2.5$$

$$x_9 \geq 0,\quad x_9 \leq \frac{5}{6} \cdot x_7 + \frac{5}{12},\quad l_9 = 0,\quad u_9 = 2.5$$

$$x_{11} \geq -x_9 + x_{10} + 3,\quad x_{11} \leq -x_9 + x_{10} + 3,\quad l_{11} = 0.5,\quad u_{11} = 5$$



$$x_2 \geq -1,\quad x_2 \leq 1,\quad l_2 = -1,\quad u_2 = 1$$

$$x_4 \geq x_1 - x_2,\quad x_4 \leq x_1 - x_2,\quad l_4 = -2,\quad u_4 = 2$$

$$x_6 \geq 0,\quad x_6 \leq 0.5 \cdot x_4 + 1,\quad l_6 = 0,\quad u_6 = 2$$

$$x_8 \geq x_5 - x_6,\quad x_8 \leq x_5 - x_6,\quad l_8 = -2,\quad u_8 = 2$$

$$x_{10} \geq 0,\quad x_{10} \leq 0.5 \cdot x_8 + 1,\quad l_{10} = 0,\quad u_{10} = 2$$

$$x_{12} \geq x_{10},\quad x_{12} \leq x_{10},\quad l_{12} = 0,\quad u_{12} = 2$$

Proving the robustness property

Goal: Prove $x_{11} - x_{12} > 0$ for all inputs in $[-1,1] \times [-1,1]$

$$x_{11} \geq -x_9 + x_{10} + 3,\\ x_{11} \leq -x_9 + x_{10} + 3,\\ l_{11} = 0.5,\\ u_{11} = 5$$

$$x_{12} \geq x_{10},\\ x_{12} \leq x_{10},\\ l_{12} = 0,\\ u_{12} = 2$$

Proving the robustness property

Goal: Prove $x_{11} - x_{12} > 0$ for all inputs in $[-1,1] \times [-1,1]$

$$\begin{gathered} x_{11} \geq -x_9 + x_{10} + 3, \\ x_{11} \leq -x_9 + x_{10} + 3, \\ l_{11} = 0.5, \\ u_{11} = 5 \end{gathered}$$

$$\begin{gathered} x_{12} \geq x_{10}, \\ x_{12} \leq x_{10}, \\ l_{12} = 0, \\ u_{12} = 2 \end{gathered}$$

Computing lower bound for $x_{11} - x_{12}$ using l_{11}, u_{12} gives -1.5 which is an imprecise result

Proving the robustness property

Goal: Prove $x_{11} - x_{12} > 0$ for all inputs in $[-1,1] \times [-1,1]$

$$x_{11} \geq -x_9 + x_{10} + 3,$$
$$x_{11} \leq -x_9 + x_{10} + 3,$$
$$l_{11} = 0.5,$$
$$u_{11} = 5$$

$$x_{12} \geq x_{10},$$
$$x_{12} \leq x_{10},$$
$$l_{12} = 0,$$
$$u_{12} = 2$$

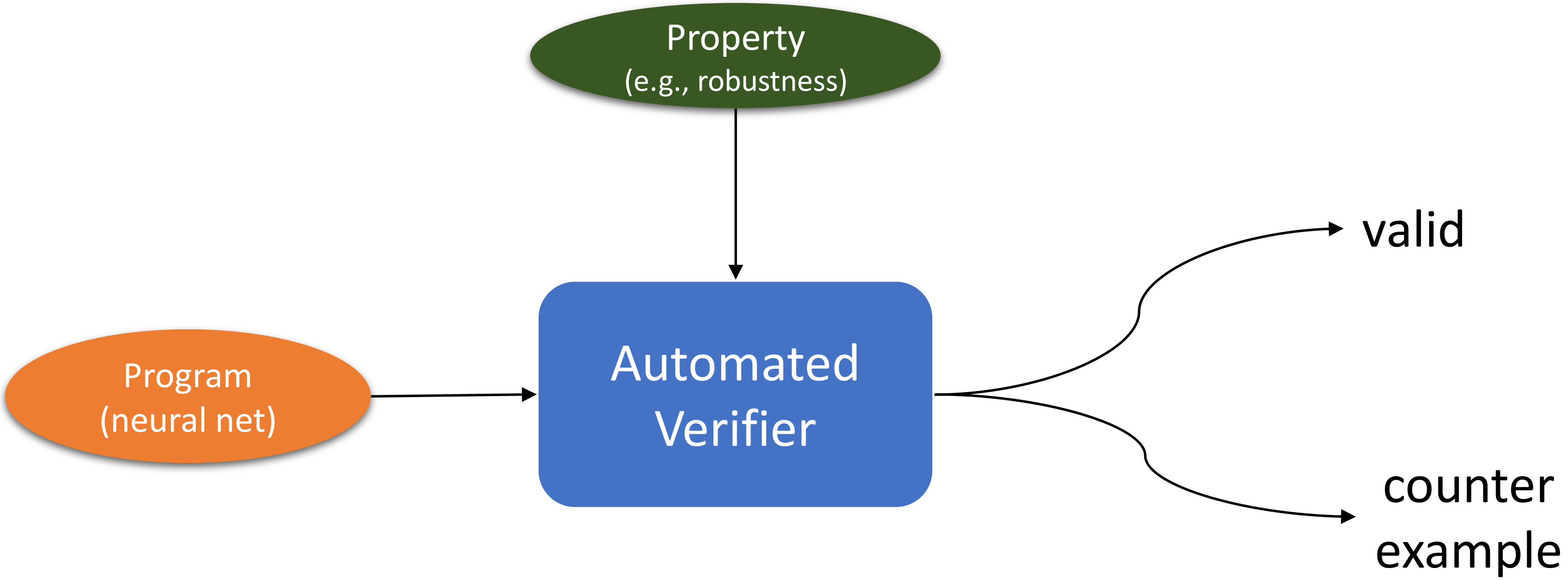
Computing lower bound for $x_{11} - x_{12}$ using l_{11}, u_{12} gives -1.5 which is an imprecise result

With backsubstitution, one gets 0.5 as the lower bound for $x_{11} - x_{12}$, proving the property

By now we have defined several incomplete verifiers and their approximations and the corresponding transformers: Box, Zonotope and DeepPoly. We also introduced a complete method based on MILP.

Before we concude our discussion on deterministic certification of neural networks, it is useful to be aware of the more general theory that these relaxations are an instance of. In particular, in this theory, a particular attention is paid to what sound means and what optimal means.

Why Approximation in the first place? (high-level view)



Minor issue ☺ : general problem is undecidable

Hence: approximation

Abstract Interpretation: a primer

The theory of abstract interpretation is a theory of approximation

Probably one of the most elegant theories in computer science

Patrick and Radhia Cousot
Inventors

- an elegant theoretical framework
- systematic way to build automated analyzers
- a way to think about approximation
- theory invented in late 70s
- started gaining popularity in the 90s
- all commercial tools use some form of A.I.

The principles of approximation are fundamental to reasoning about computation with infinite or very large state spaces.

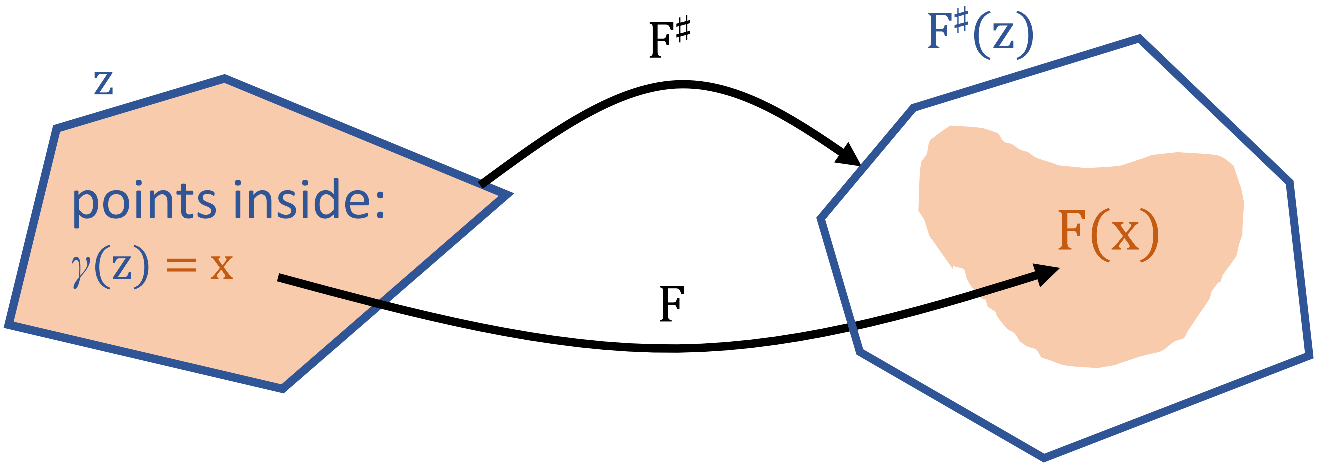
Abstract Interpretation: a primer

A.I. concerns itself with questions such as:

- What is it that we are approximating?
- What does it mean for the approximation to be optimal (or to approximate)?
- What does it mean for an approximation to be sound?
- How do we actually build a correct and efficient verifier?
- Can the process of building an analyzer be automated?

A.I. cheat sheet

blue: abstract   orange: concrete



Important (soundness):

$\gamma(F^\sharp(z)) \supseteq F(x) = F(\gamma(z))$

1. A concrete element x is a set of concrete values.
2. An abstract (symbolic) element z semantically represents a set of concrete values.
3. γ is a concretization: it defines the concrete values an abstract element z represents (the points inside the polygon).
4. F is the concrete transformer. Because the set x is infinite or finite but very large, we generally cannot compute the transformed output set F(x).
5. $F^\sharp$ is the abstract transformer. $F^\sharp(z)$ applies $F^\sharp$ to abstract element z.
6. $F^\sharp$ must be sound (formula on top of slide and visualized in diagram above).

Soundness of Transformers

$$\forall z \,.\, F(\gamma(z)) \subseteq \gamma(F^{\sharp}(z))$$

That is, applying the transformer $F^{\sharp}$ on an abstract element z, and then obtaining the set of concrete values corresponding to the result has to include more points than first concretizing the abstract element and then applying the concrete function F.

Exactness of Transformers

$$\forall z.\, F(\gamma(z)) \;=\; \gamma(F^{\sharp}(z))$$

That is, applying the transformer $F^{\sharp}$ on an abstract element z, and then obtaining the set of concrete values corresponding to the result produces the same set of points as first concretizing the abstract element and then applying the concrete function F.

As we already saw, both, Box and Zonotope transformers are not exact for ReLU. For affine, Box loses precision, while Zonotope is exact.

Optimality of Transformers

A sound transformer F_{best} is called a best transformer if for all sound transformers F', F' is not more precise than F_{best} :

$$\forall z \,.\, \gamma(F'(z)) \not\subset \gamma(F_{best}(z))$$

For Box, both affine and ReLU are optimal. For Zonotope, affine is exact (and optimal) but there is no single best transformer for ReLU.

- We saw several instances of A.I., enough to get a working intuition with it.
- Abstract Interpretation is a rich area with many branches and applications.
- A particular branch we use when analyzing neural networks is numerical domains (e.g., Zonotope, Box, Octagons, Polyhedra, DeepPoly, etc), which trade-off completeness for scalability (while being sound).
- Abstract transformers of these domains can be very tricky to implement efficiently and correctly!
- Good abstract transformers are typically defined for the application-specific operators (e.g., ReLU, sigmoid)
- Scalable and precise verifier is a combination of careful math (e.g., zonotope ReLU) + efficient algorithms and coding.
- Note that in practice we need to ensure floating-point soundness as well!

Summary

- Another incomplete method, the DeepPoly approximation.

- Its abstract transformers for affine and ReLU for DeepPoly.

- Like Zonotope, DeepPoly is exact for affine and is its ReLU transformer produces a smaller area than the Zonotope ReLU transformer.

- A brief look at abstract interpretation and mathematical definition of soundness, exactness and optimality of abstract transformers.