A Conversation on Local robustness

ESSAI 2025

Julien Girard-Satabin
Zakaria Chihani
Dorin Doncenco

CEA LIST

2025-07-01

This work was supported by the French Agence Nationale de la Recherche (ANR) through SAIF (ANR-23-PEIA-0006) and DeepGreen (ANR-23-DEGR-0001) as part of the France 2030 programme.

Introduction

Me:

- researcher at the *French Commission for Atomic and Alternative Energies* on Trustworthy AI
- one of our mission is « Technological Transfer », as Zakaria showed yesterday
- developer and maintainer of the *CAISAR platform*
- research interests include formal verification for AI, specification languages, formal explanation

My expectations:

- have some of you convinced that Formal Methods can be useful for ML
- present you some of our research
- collect expectations of actual ML practioners
- discuss on your views on Formal Methods and their limitations
- ~~convert you to using FM tools~~ get yourselves a notion of « software engineering » for machine learning

You:

- a diverse set of PhD students on AI, machine learning and/or symbolic with *really* diverse research interests
- knowledgeable *on some degree* on how to operate machine learning (training, checking metrics, finetuning)
- use ML for science, *or* study ML with a certain degree of applicability, *or* apply ML in the industry

Your expectations: Let us discuss! Why did you join this course?

You:

- a diverse set of PhD students on AI, machine learning and/or symbolic with *really* diverse research interests
- knowledgeable *on some degree* on how to operate machine learning (training, checking metrics, finetuning)
- use ML for science, *or* study ML with a certain degree of applicability, *or* apply ML in the industry

Your expectations: Let us discuss! Why did you join this course? If you are more inclined into Y/N questions:

- have you ever setup unit tests for ML?
- do you use Continuous Integration?
- is the word « proof » itches you?
- or does it sparks joy?

Scope of this session:

- Verification of Machine Learning, with a strong focus on Neural Networks
- Local robustness and its declinations
- Abstract Interpretation for Neural Network

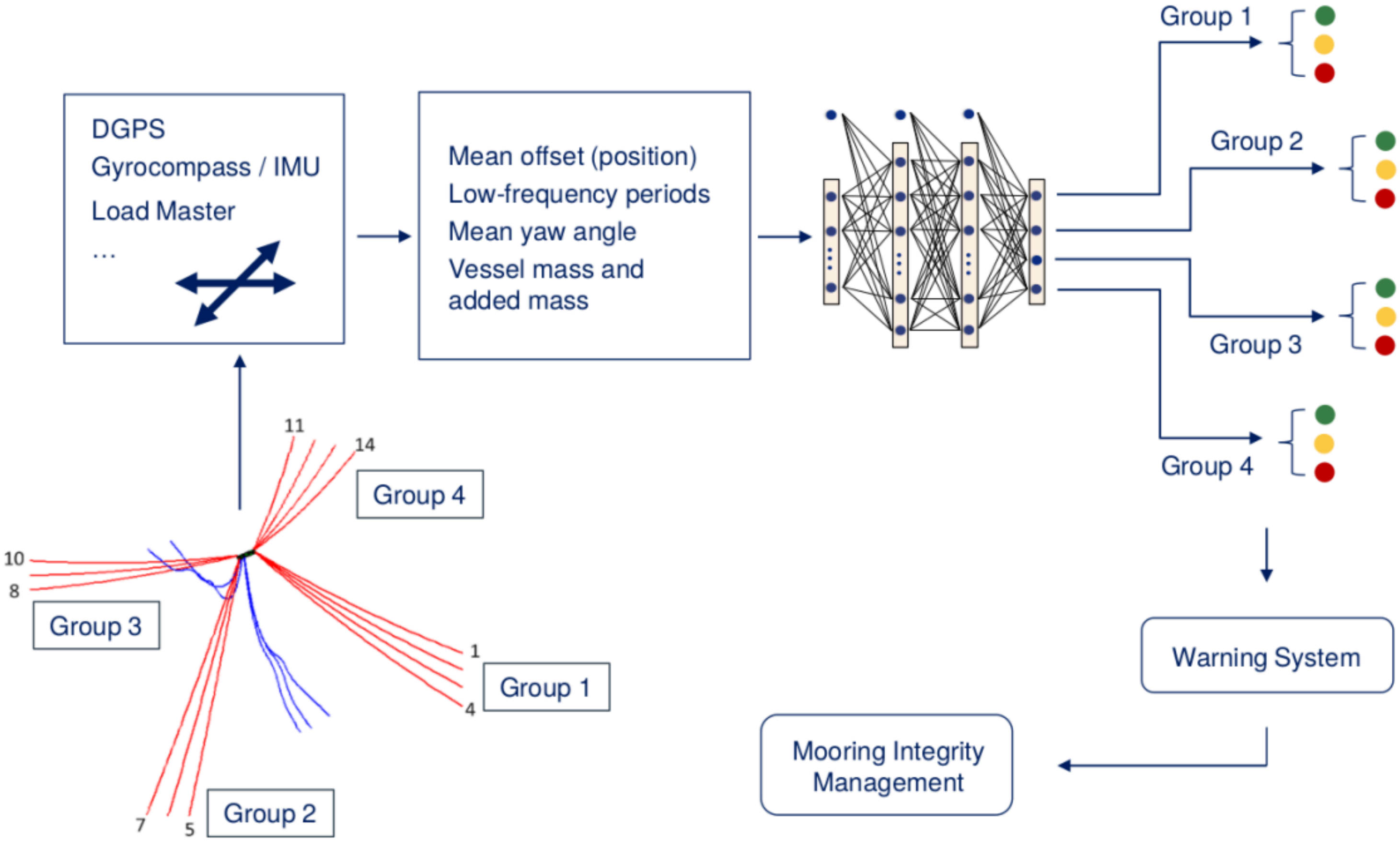
Not covered in this course:

- Training schemes like Differentiable Logics (Ślusarz et al. 2023)
- Purely symbolic systems (plenty of resources already on this topic)
- LLMs (why? Wait for Session 4 🤫)

Tackling small variations on the inputs

Altering the prediction of a neural network

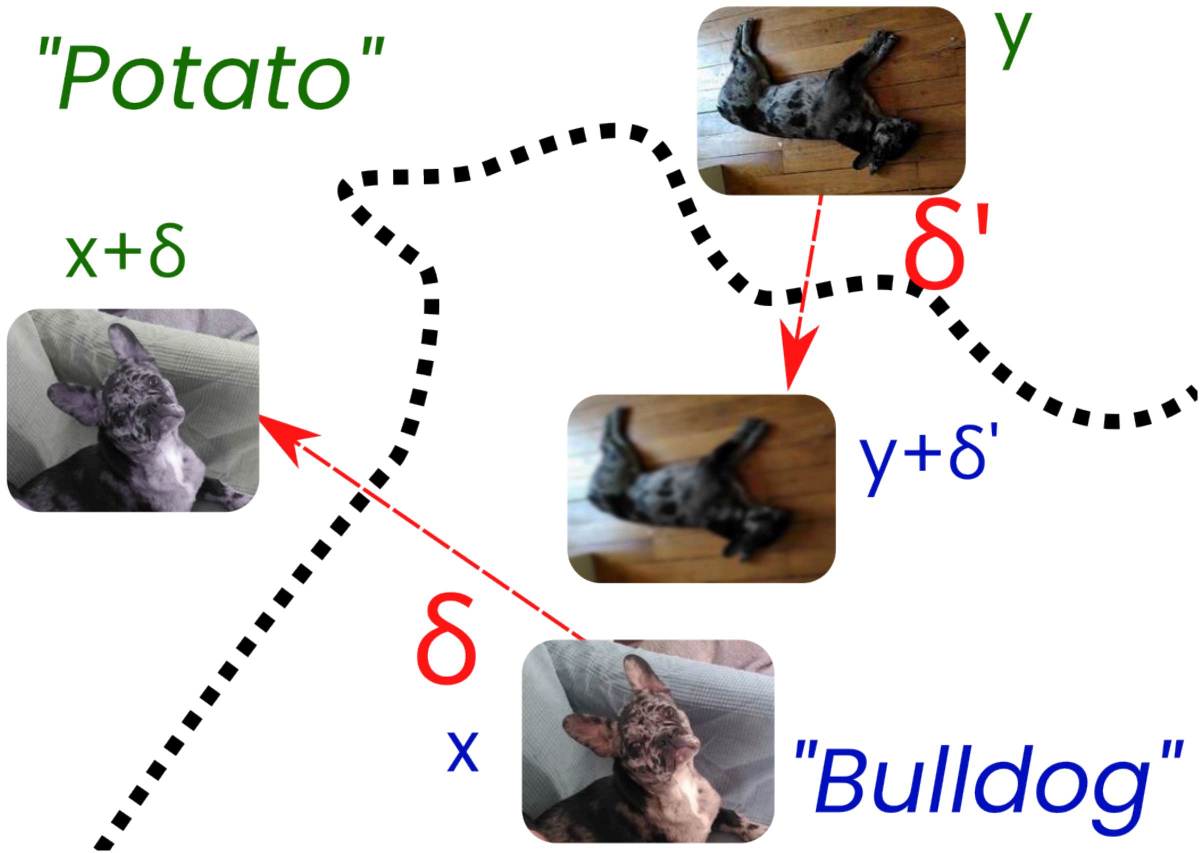
Intuition



Sensor uncertainty (because physical components, data transmission, sometimes low quality of component) may result in critical systems failures

Perpetuating embedded social biases

Intuition



Definition

Local robustness (Katz et al. 2017)

Let a classifier $f : \mathcal{X} \mapsto \mathcal{Y}$. Given $x \in \mathcal{X}$ and $\varepsilon \in \mathbb{R} \lll 1$ the problem of *local robustness* is to prove that $\forall x'. \ \|x - x'\|_p < \varepsilon \rightarrow f(x) = f(x')$

Definition

To prove this property usually means falsifying its negation:

Falsfying local robustness

Let a classifier $f : \mathcal{X} \mapsto \mathcal{Y}$. Given $x \in \mathcal{X}$ and $\varepsilon \in \mathbb{R} \lll 1$, the problem of *finding a point* x' such that $\exists x'.\ \|x - x'\|_p < \varepsilon \Rightarrow f(x) \neq f(x')$

Variations

Numerous variants can be found in the literature (Casadio et al. 2022)

Prediction invariance

Given $\delta \in \mathbb{R}$, $\exists x'.\ \|x - x'\|_p \leq \varepsilon \Rightarrow \|f(x) - f(x')\|_p \leq \delta$

Exact classification is very unlikely to happen

Importance of distance function $\|\cdot\|_p$

Variations

Lipschitz Robustness

Given $L \in \mathbb{R}$, $\forall x'.\ \|x - x'\|_p \leq \varepsilon \Rightarrow \|f(x) - f(x')\|_p \leq L\|x - x'\|_p$

« Smooth » the variation of output with regard to inputs

Variations

Class-robust prediction

$$\forall x'.\ \|x - x'\|_p \leq \varepsilon \Rightarrow \text{class} f(x) = \text{class } f(x')$$

Useful in a classification setting

Given $i \in \{1..c\}$ the set of indices defining possible classes for a classification setting, $\forall x'.\ \|x - x'\|_p \leq \varepsilon \Rightarrow \text{class} f(x) = i = \text{class } f(x')$

Variations

Strong classification Robustness

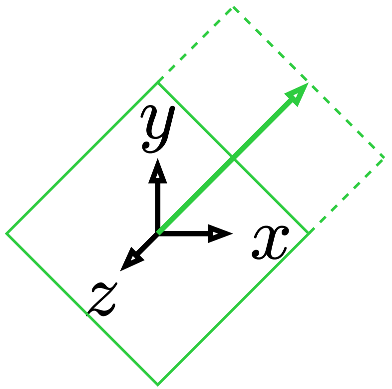
Given $\nu \in \mathbb{R}, \forall x'. \; \|x - x'\|_p \leq \varepsilon \Rightarrow \mathrm{class} f(x') = \mathrm{class} f(x) \wedge \mathrm{class} f(x) \geq \nu$

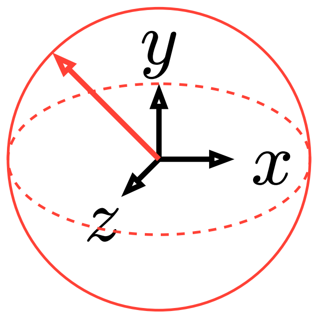
ν usually being above 0.5 to describe a « high confidence » network (softmax value)

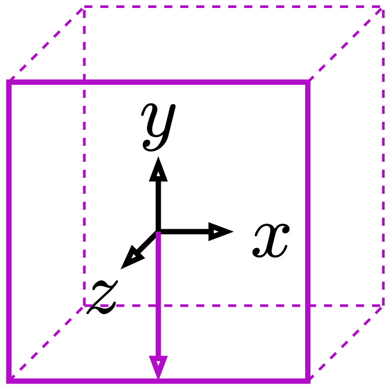
On norms

Which norm to use? The most frequent norms are L_2, L_1 and L_∞:

$$L_1(x) = \sum_i |x_i| \qquad L_2(x) = \sqrt{\sum_i (x_i)^2} \qquad L_\infty(x) = \max_i |x_i|$$







All describe convex sets

Encoding and verifying

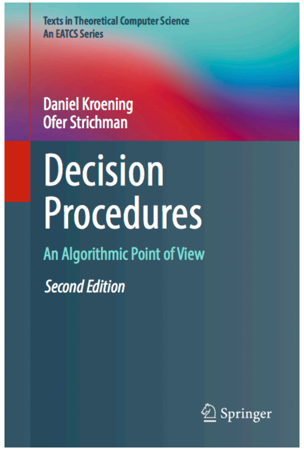
Assessing the local robustness of a neural network is a NP-Complete problem (Katz et al. 2017)

Spheres, hyperboxes… all are convex shapes, right?

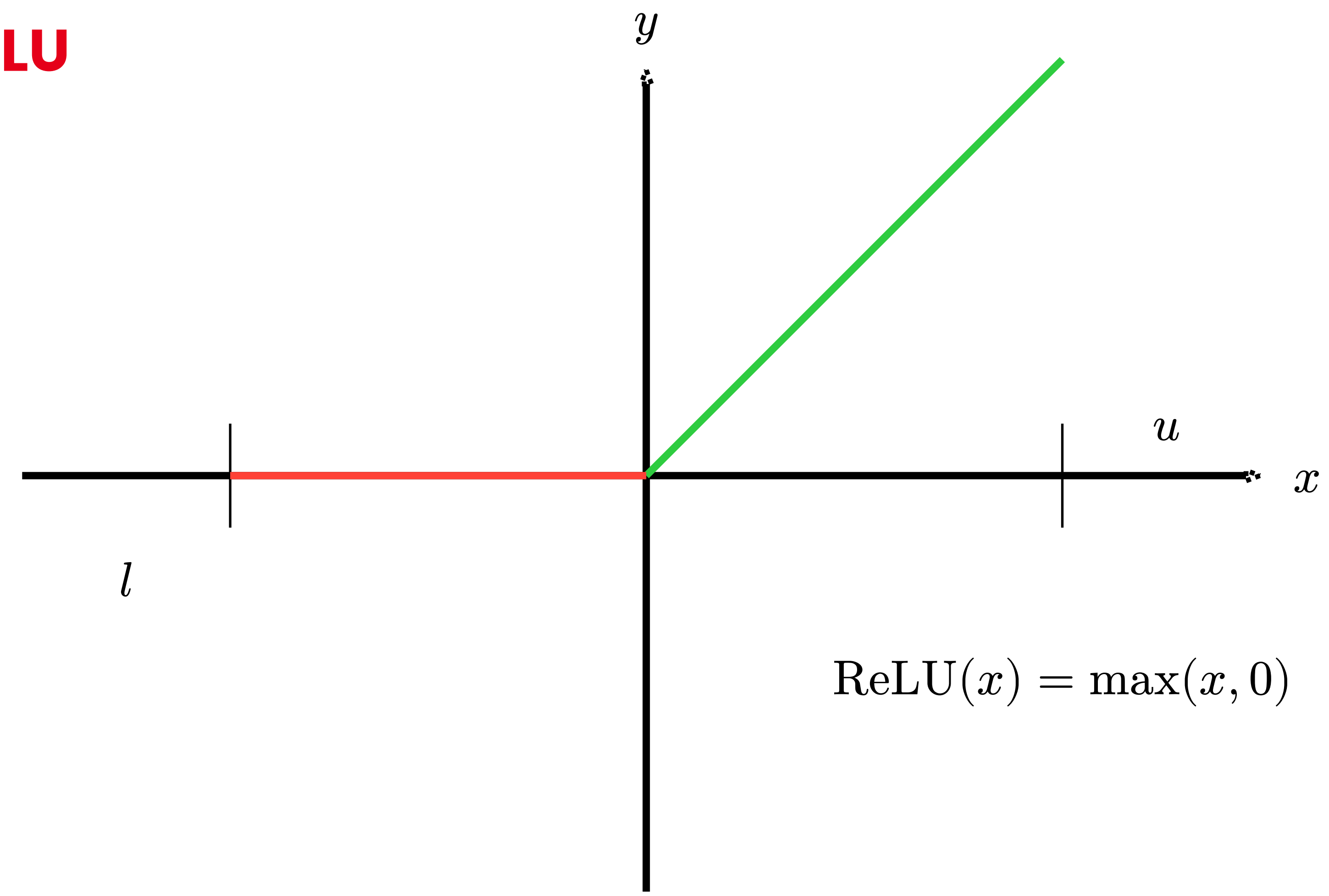
Spheres, hyperboxes… all are convex shapes, right?

Existing approaches to solve convex problems include well-known techniques:

- Satisfaction Modulo Theory (SMT) with Linear Arithmetic (Kroening and Strichman 2016)
- Abstract Interpretation (Mirman, Gehr, and Vechev 2018; Lemesle, Lehmann, and Gall 2024)

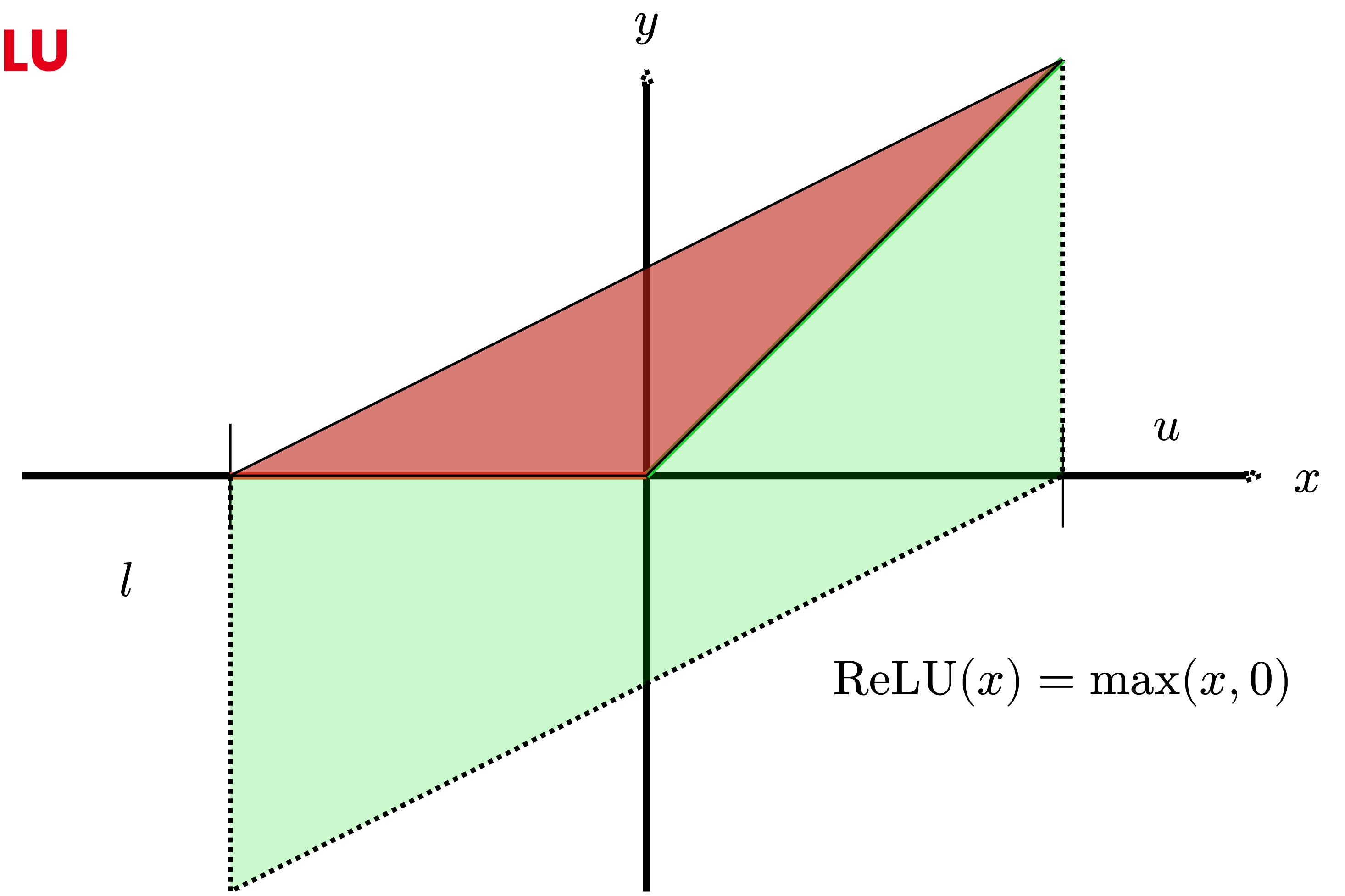


On the ReLU



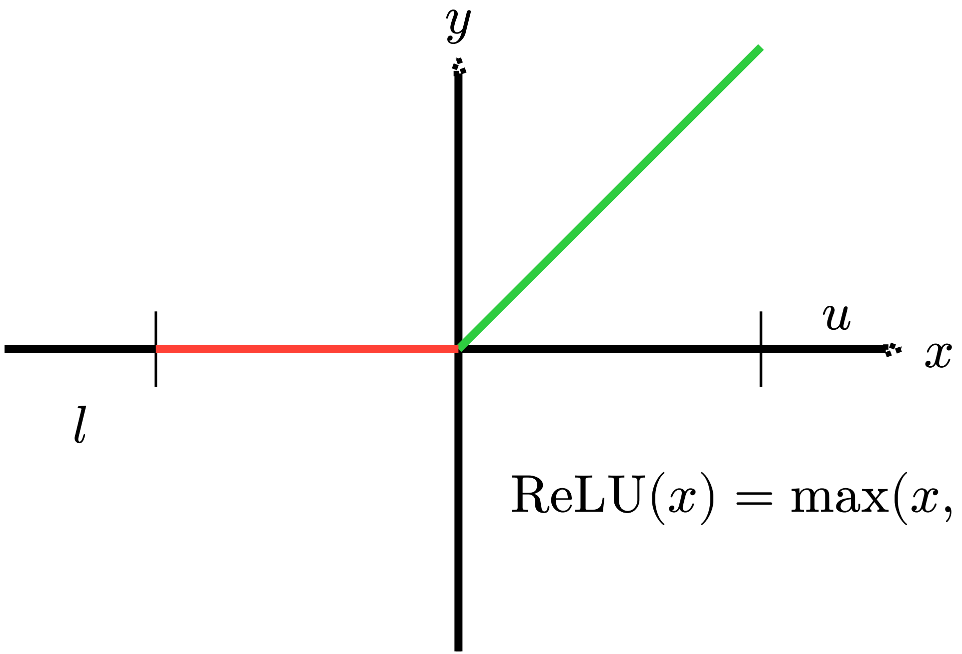
$$\mathrm{ReLU}(x) = \max(x, 0)$$

On the ReLU



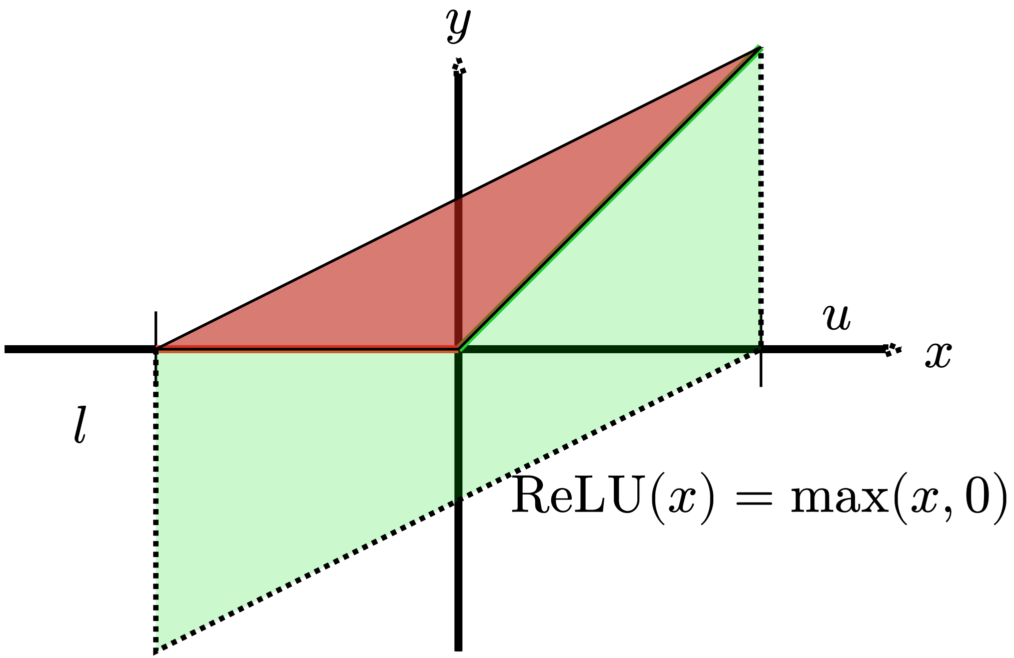
$$\mathrm{ReLU}(x) = \max(x, 0)$$

On the ReLU



$$\mathrm{ReLU}(x) = \max(x, 0)$$

On the ReLU



- Two states: *active* $(x \geq 0)$ or *inactive* $(x < 0)$
- Given a network with n ReLUs, 2^n possible activation states
- Need to approximate as a convex shape $\Rightarrow$ inaccuracies

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. initial answer
   - x_1 : `true`, x_2 : `true`, x_3 : `true`, x_4 : `true`, x_5 : `false`,

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \lor (a = 6)) \land (a \geq 3 \land ((b \leq 2) \lor (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(x_1 \lor x_2) \land (x_3 \land (x_4 \lor x_5))$
3. initial answer
   - $x_1 : a = 4$, $x_2 : a = 6$, $x_3 : a \geq 3$, $x_3 : b \leq 2$, $x_4 : b \leq 3$,

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. initial answer
   - $x_1 : a = 4$, $x_2 : a = 6$, $x_3 : a \geq 3$, $x_3 : b \leq 2$, $x_4 : b \leq 3$,

   $a = 4 \wedge a = 6$ is unfeasible!

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. initial answer
   - $x_1 : a = 4$, $x_2 : a = 6$, $x_3 : a \geq 3$, $x_3 : b \leq 2$, $x_4 : b \leq 3$,

   $a = 4 \wedge a = 6$ is unfeasible!

   Solution: encode a new constraint to the SAT formula: $\neg x_1 \vee \neg x_2$

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. initial answer
   - $x_1 : a = 4$, $x_2 : a = 6$, $x_3 : a \geq 3$, $x_3 : b \leq 2$, $x_4 : b \leq 3$,

   $a = 4 \wedge a = 6$ is unfeasible!

   Solution: encode a new constraint to the SAT formula: $\neg x_1 \vee \neg x_2$

4. New SAT formula: $(\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

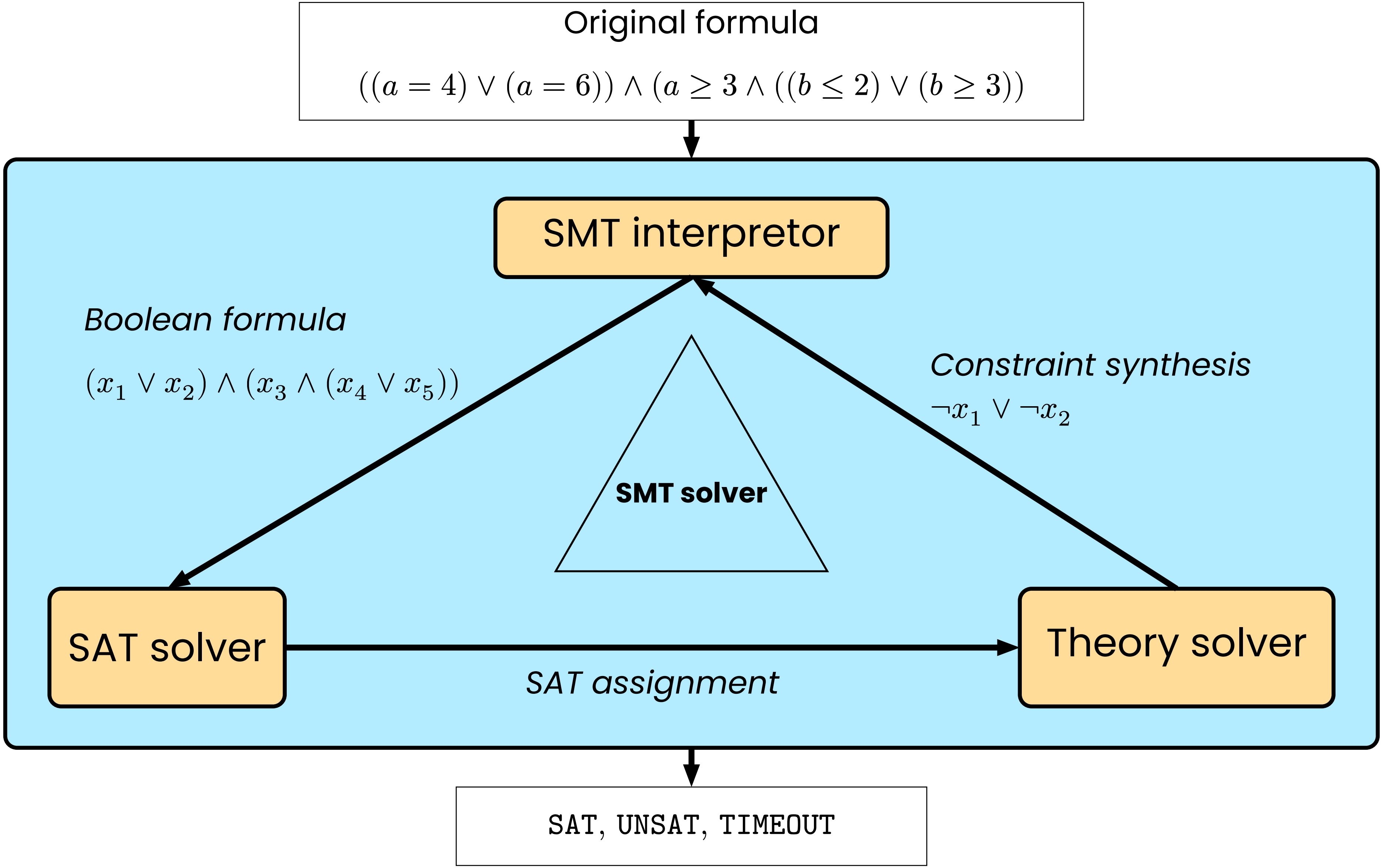
1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(\neg \mathbf{x}_1 \vee \neg \mathbf{x}_2) \wedge (x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. New answer:
   - x_1 : `true`, x_2 : `false`, x_3 : `true`, x_4 : `true`, x_5 : `false`,
4. $a = 4 \wedge b = 2$ satisfy the formula

One-slide primer on SMT calculus

Consider the formula: $((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$

Are there reals numbers making this formula true?

1. Encode constraints as boolean atoms
   - $x_1 : a = 4, x_2 : a = 6, x_3 : a \geq 3, x_4 : b \leq 2, x_5 : b \geq 3$
2. SAT formula: $(\neg \mathtt{x}_1 \vee \neg \mathtt{x}_2) \wedge (x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
3. New answer:
   - $x_1 : a = 4$, $x_2 : a \neq 6$, $x_3 : a \geq 3$, $x_3 : b \leq 2$, $x_4 : b \leq 3$,
4. $a = 4 \wedge b = 2$ satisfy the formula



Original formula
$((a = 4) \vee (a = 6)) \wedge (a \geq 3 \wedge ((b \leq 2) \vee (b \geq 3))$
SMT interpretor
Boolean formula
$(x_1 \vee x_2) \wedge (x_3 \wedge (x_4 \vee x_5))$
Constraint synthesis
$\neg x_1 \vee \neg x_2$
SMT solver
SAT solver
Theory solver
SAT assignment
SAT, UNSAT, TIMEOUT

Mixed-Integer Linear Programming (Tjeng, Xiao, and Tedrake 2019)

$$\text{Minimize } c^{\mathrm{T}}x \text{ subject to}$$
$$Ax \leq b$$
$$l \leq x \leq u$$
$$x_i \in \mathbb{Z}$$

Mixed-Integer Linear Programming (Tjeng, Xiao, and Tedrake 2019)

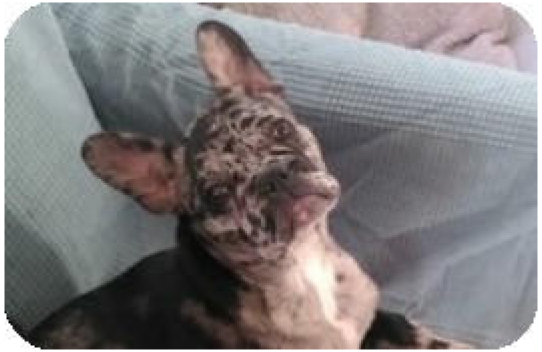
Given $y = \mathrm{ReLU}(z)$, $\underline{z}$ (resp. $\overline{z}$) the lower bound (resp. upper bound) of z, propose the following encoding for the ReLU

$$y \leq z - \underline{z}(1-a)) \wedge y \geq z \wedge (y \leq \overline{z}a) \wedge (y \geq 0) \wedge a \in \{0,1\}$$

Slightly higher-level than SMT solver with `QF_LRA` so usually what is used within most decision procedures

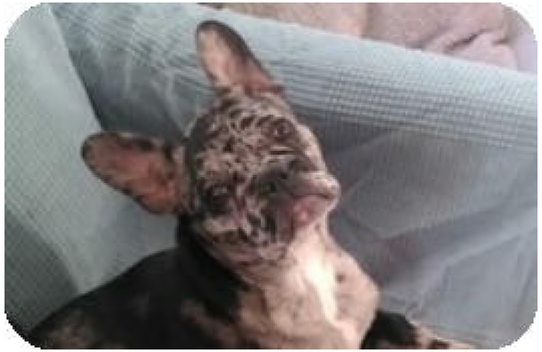
Abstract interpretation

Intuition



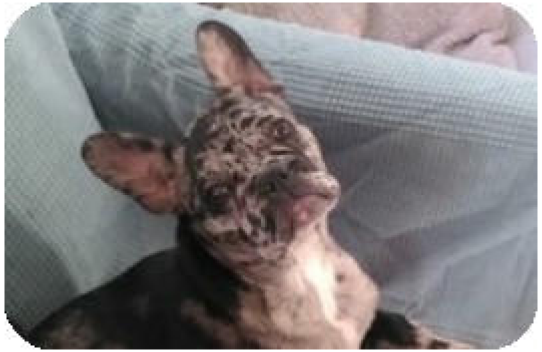
Abstract interpretation

Intuition



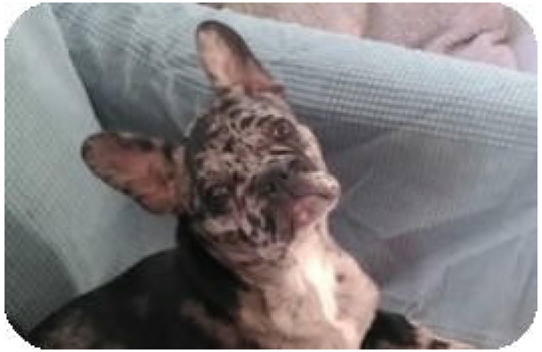
Abstract interpretation

Intuition



Abstract interpretation

Intuition

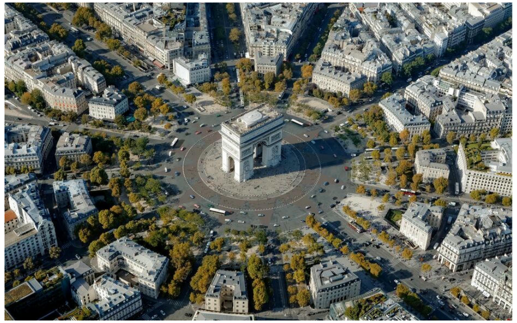
  

Problem to compute: when to unleash Virgule?

Park: good

Empty street: okay

Other animals: meh



Gigantic street: not a chance

Abstract interpretation

Benefits of abstracting a problem

- easier for me to take a decision
  - `if maybe_car then leash else free_puppy`
- guarantees safety for Virgule
- balance to be found between ease to take a decision and Virgule's well-being *(ensure your pets stay hydrated during the hot summer and don't keep them in a car)*

Abstract interpretation

What would be the content of the abstraction?

Abstract interpretation

What would be the content of the abstraction?

- general kind of location (park? street?)
- presence of humans
- presence of cars
- general location of virgule (assuming the dog don't fly)

Abstract interpretation

What would be the content of the abstraction?

- general kind of location (park? street?)
- presence of humans
- presence of cars
- general location of virgule (assuming the dog don’t fly)

```
type abstract_state = (Park | Street, bool, bool, (int, int))
```

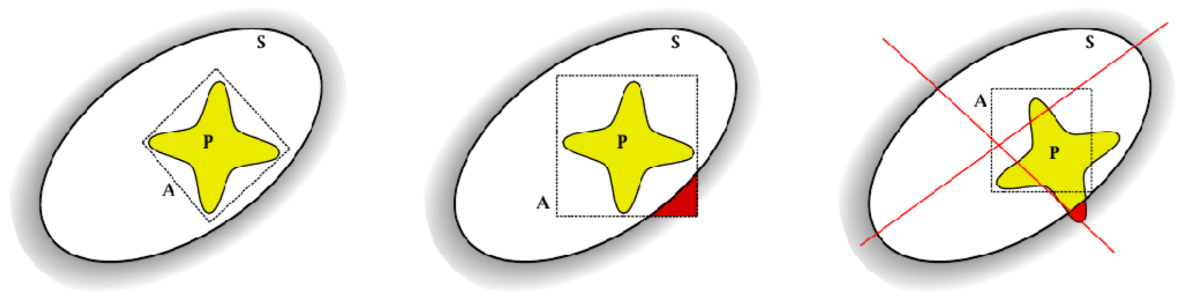
Abstract interpretation

For any possible situation `s : concrete_state`, I need to:

- convert it to an abstract state: `val abstract: concrete_state -> abstract_state`
- perform decision on the abstract state: `val compute:  abstract_state -> abstract_state`
- ensure the decision in my brain is actually happening: `val concretize: abstract_state -> concrete_state`

Abstract interpretation

Abstract interpretation (Miné 2017; Cousot and Cousot 1977) is a theoretical framework to soundly *abstract* a program, use it to perform *abstract, simplified* computations and deduce properties on said program



precise analysis
$A \subseteq S \implies P \subseteq S$
(a)

false alarm
$A \not\subseteq S$ but $P \subseteq S$
(b)

unsound analysis
$A \subseteq S$ but $P \not\subseteq S$
(c)

Abstract interpretation

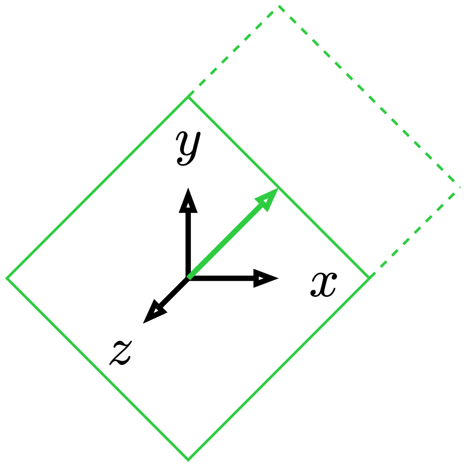
Ensure my model of the real world sufficiently describes the relevant parts, and ensure my decision making process is coherent

Abstract interpretation for neural networks

What kind of rather simple, subspace we saw earlier in the course could be a good candidate?

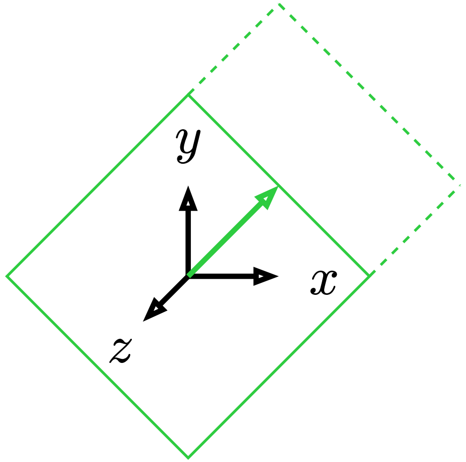
Abstract interpretation for neural networks

What kind of rather simple, subspace we saw earlier in the course could be a good candidate?



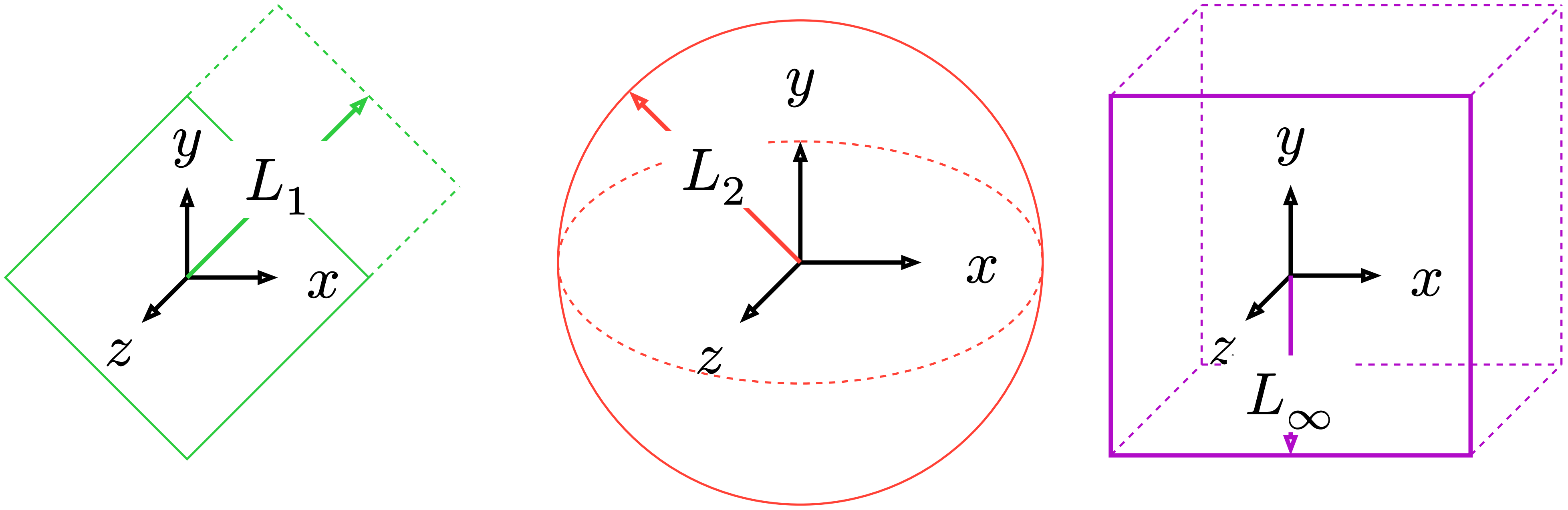
Abstract interpretation for neural networks

What kind of rather simple, subspace we saw earlier in the course could be a good candidate?



For a neural network: compute convex sets!

$L_1(x) = \sum_i |x_i|$ $\quad L_2(x) = \sqrt{\sum_i (x_i)^2}$ $\quad L_\infty(x) = \max_i |x_i|$



Abstract interpretation for neural networks

The input space described by local robustness property is a convex set

What would it look like to compute such set with a neural network?

Abstract interpretation for neural networks

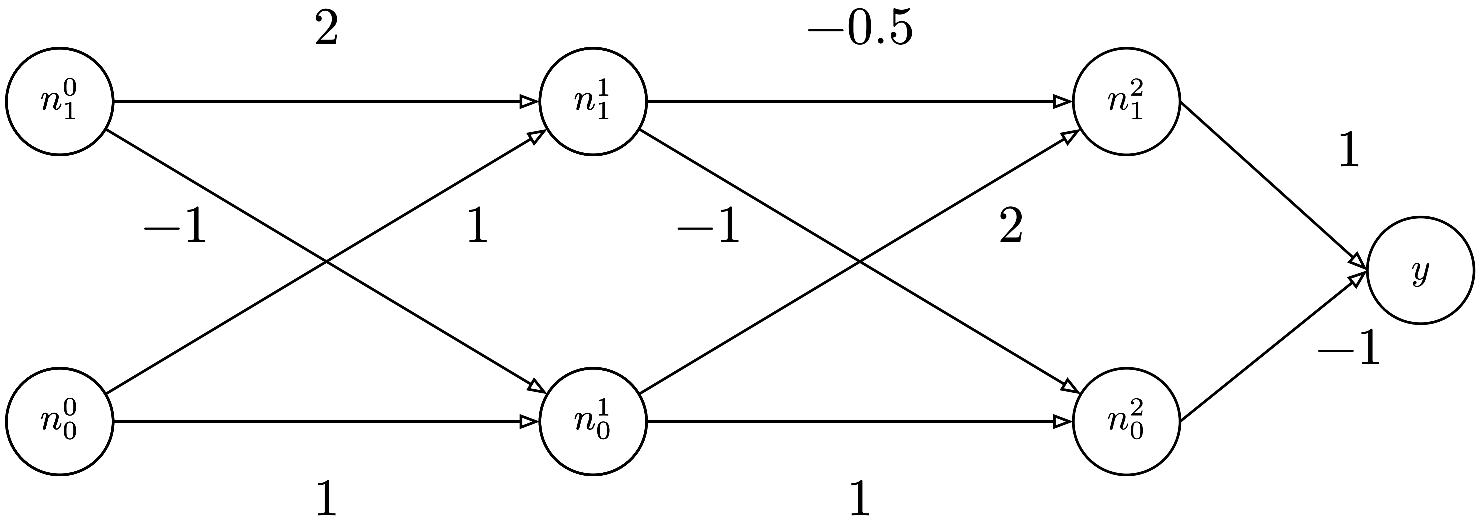
The input space described by local robustness property is a convex set

What would it look like to compute such set with a neural network?

Let us have an example!

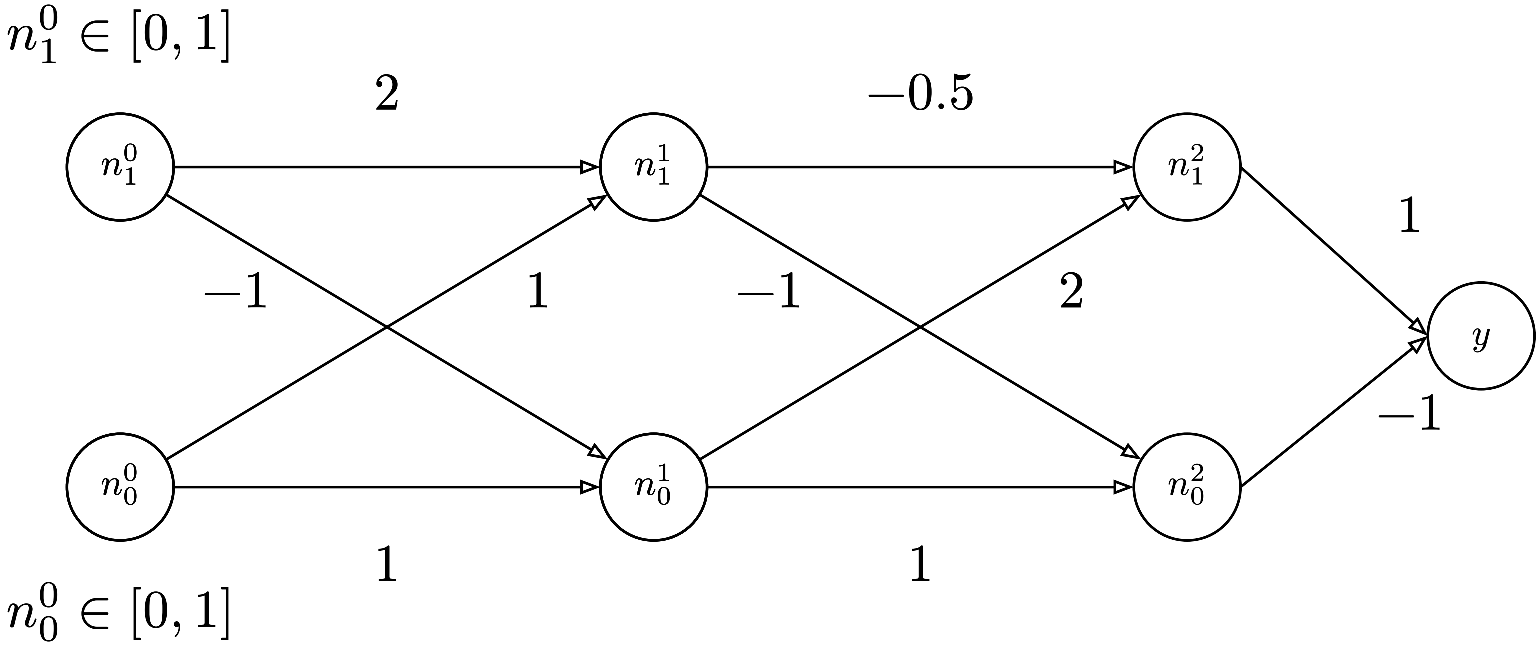
Each layer is separated by a $\mathrm{ReLU} = \max(x, 0)$.

Assume we want to check $y_0 > 0$



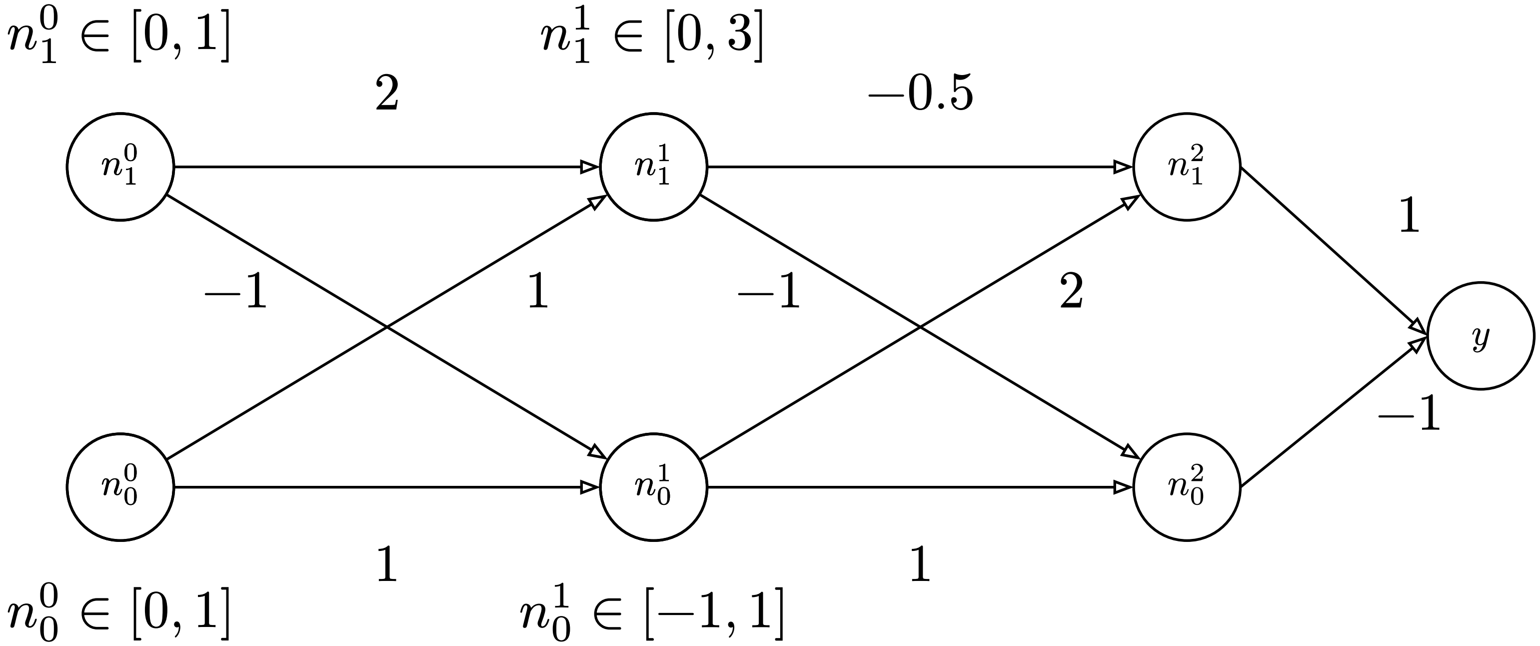
Each layer is separated by a $\mathrm{ReLU} = \max(x, 0)$.

Assume we want to check $y_0 > 0$



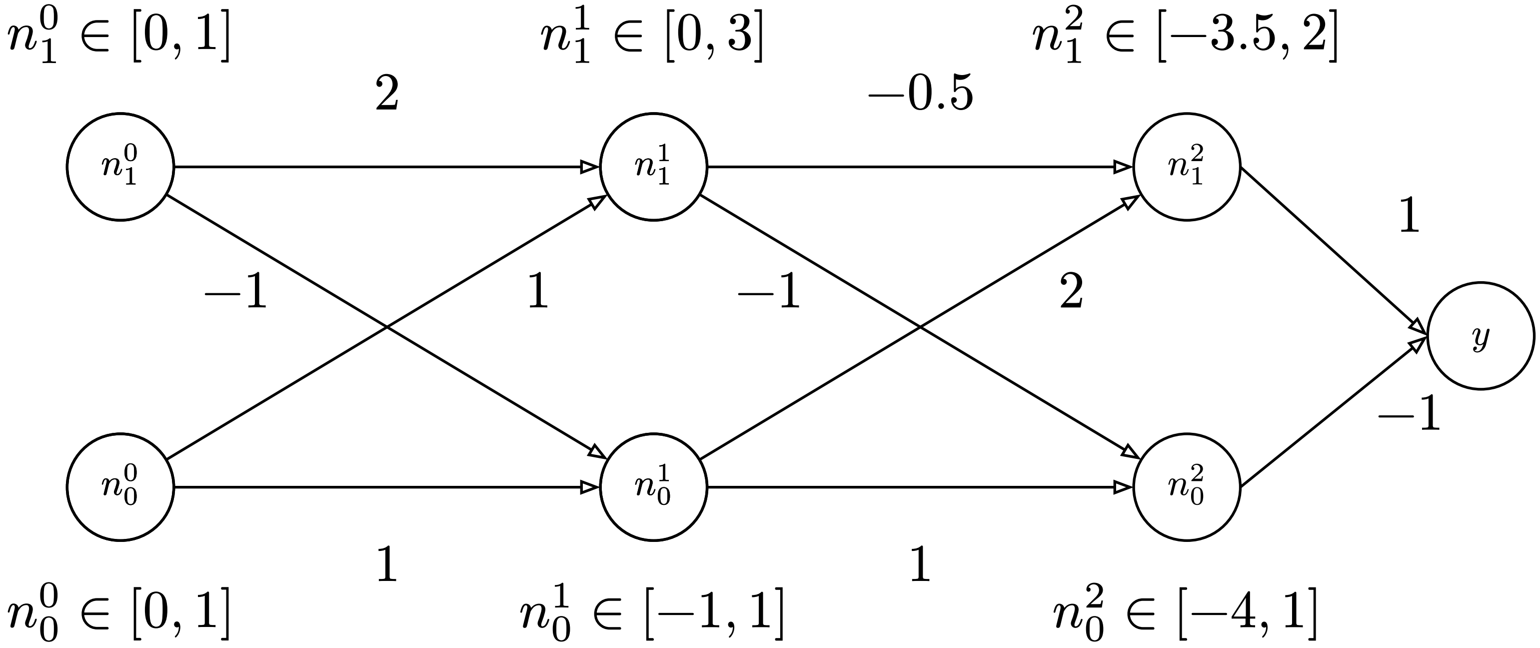
Each layer is separated by a $\mathrm{ReLU} = \max(x, 0)$.

Assume we want to check $y_0 > 0$



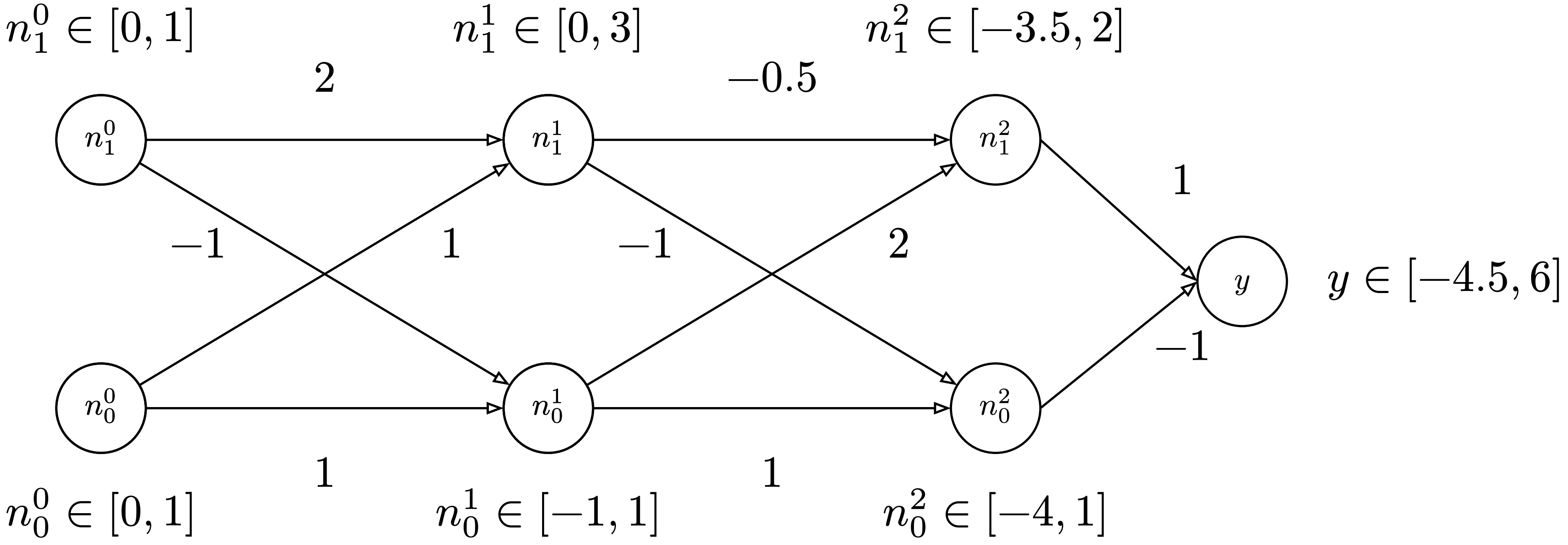
Each layer is separated by a $\mathrm{ReLU} = \max(x, 0)$.

Assume we want to check $y_0 > 0$

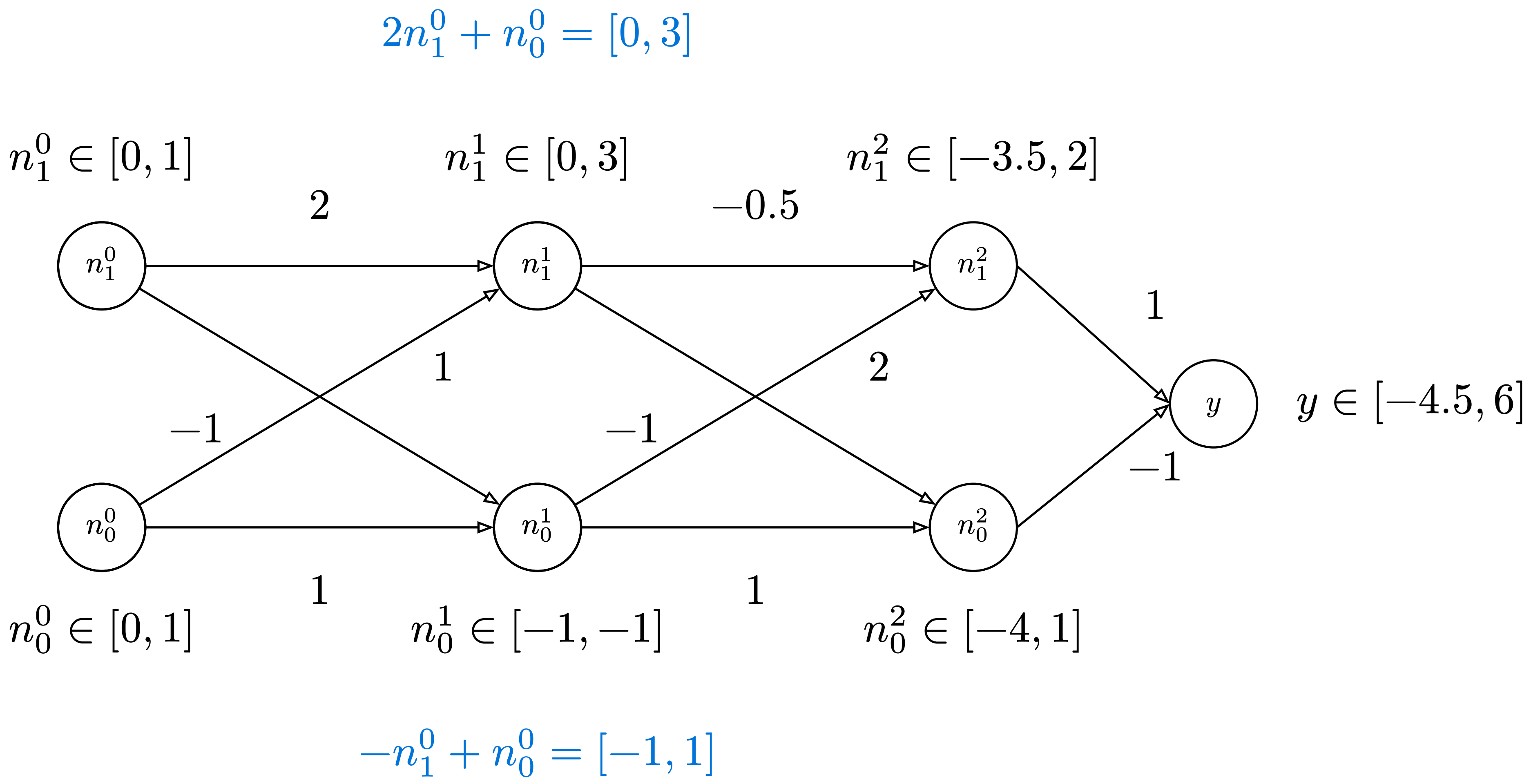


Each layer is separated by a $\mathrm{ReLU} = \max(x, 0)$.

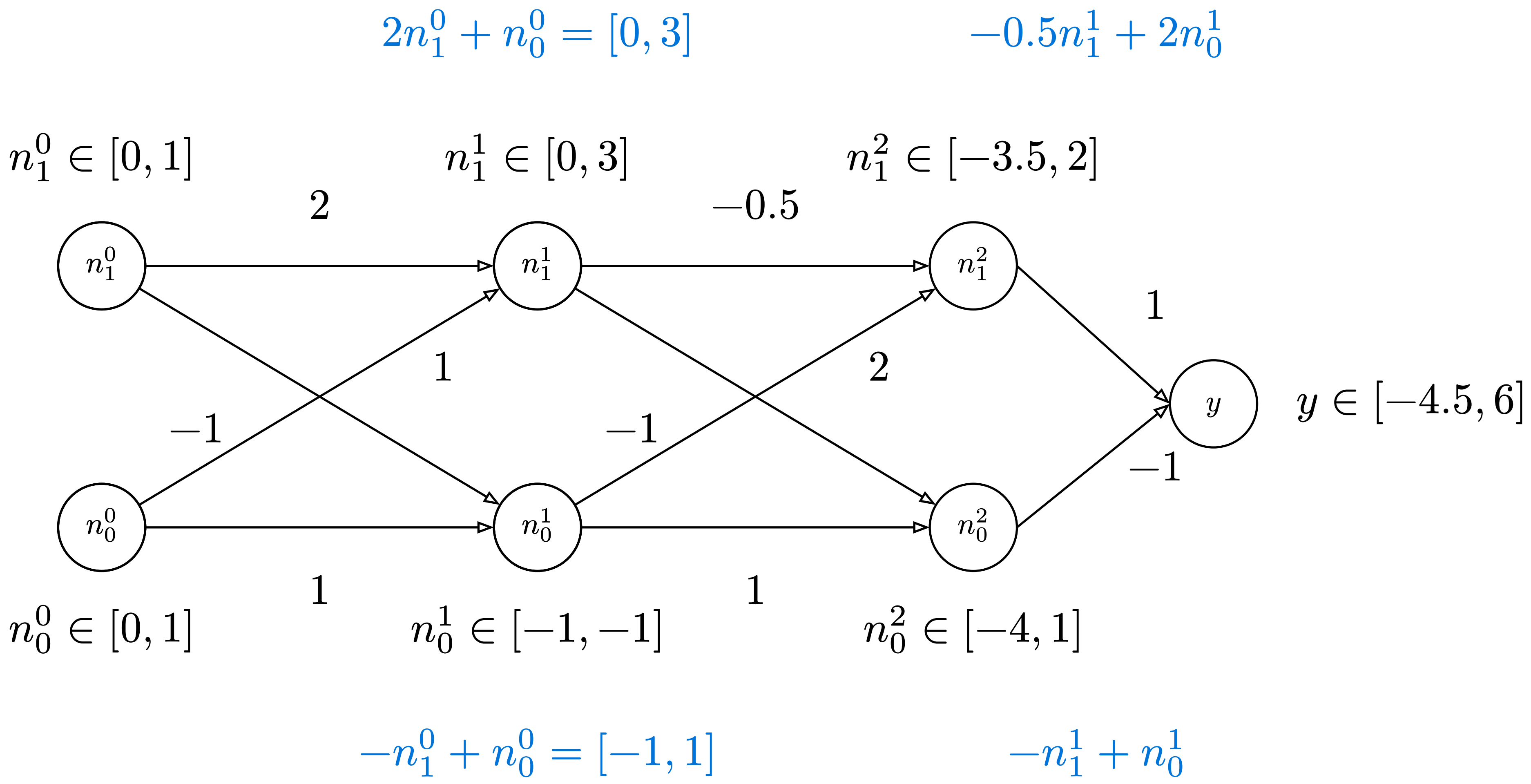
Assume we want to check $y_0 > 0$



A bit imprecise, right?

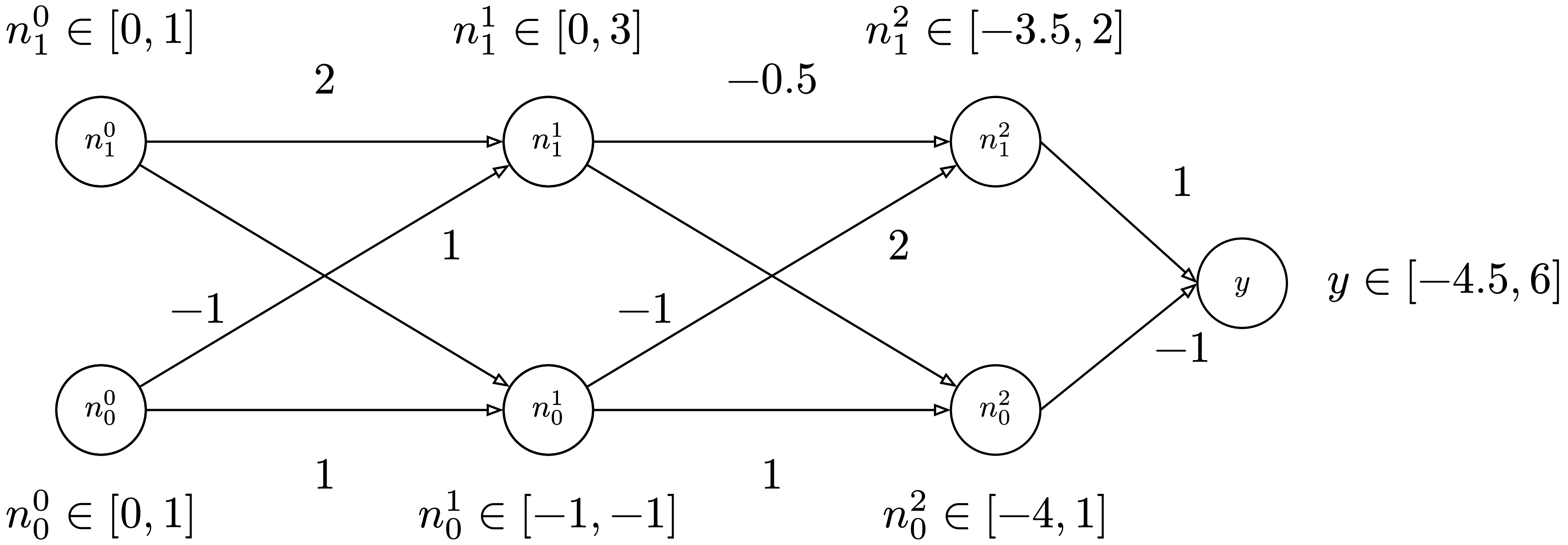


Symbolic propagation of variables *and their relations*



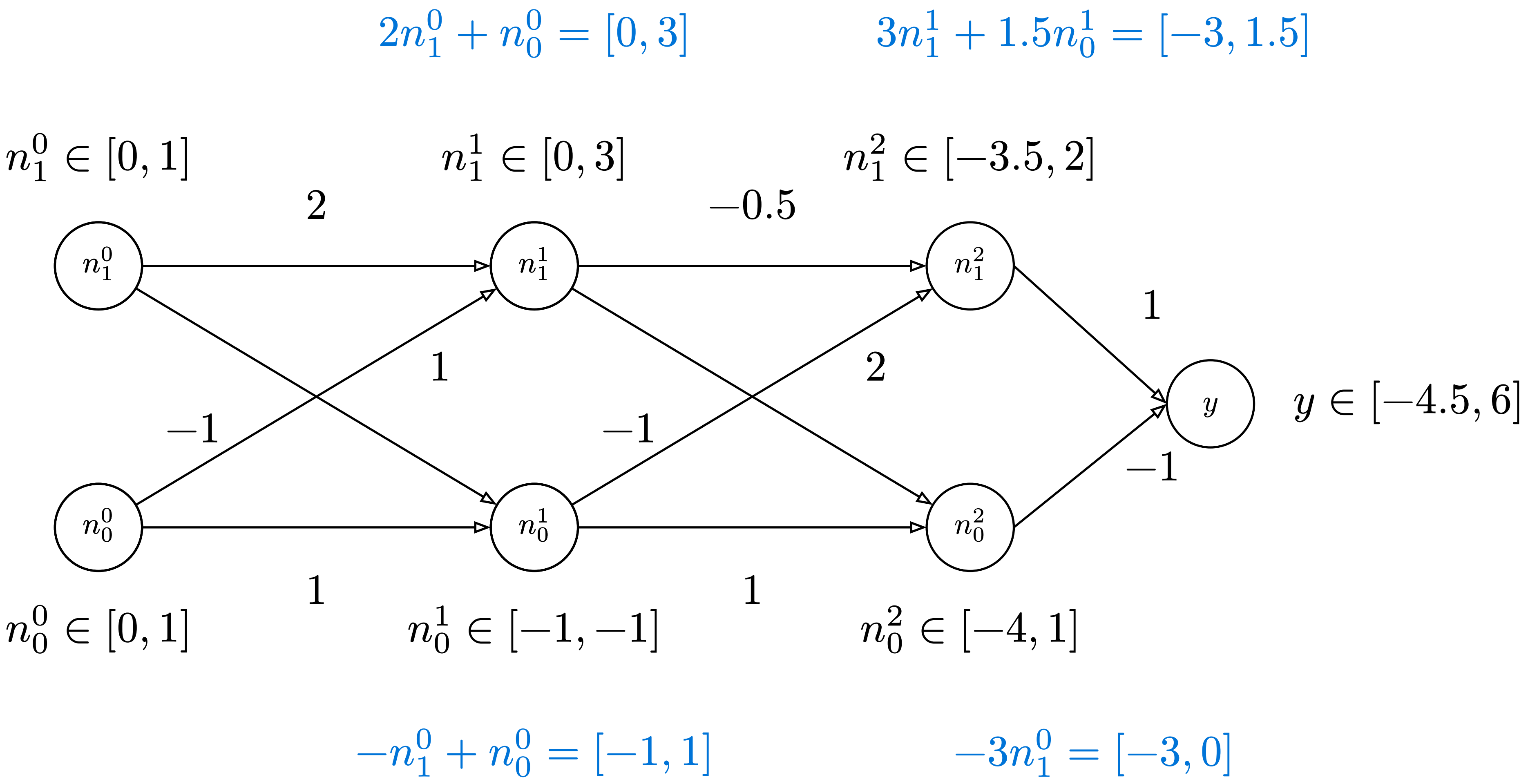
Symbolic propagation of variables *and their relations*

$$2n_1^0 + n_0^0 = [0, 3] \qquad -0.5(2n_1^0 + n_0^0) + 2(-n_1^0 + n_0^0)$$

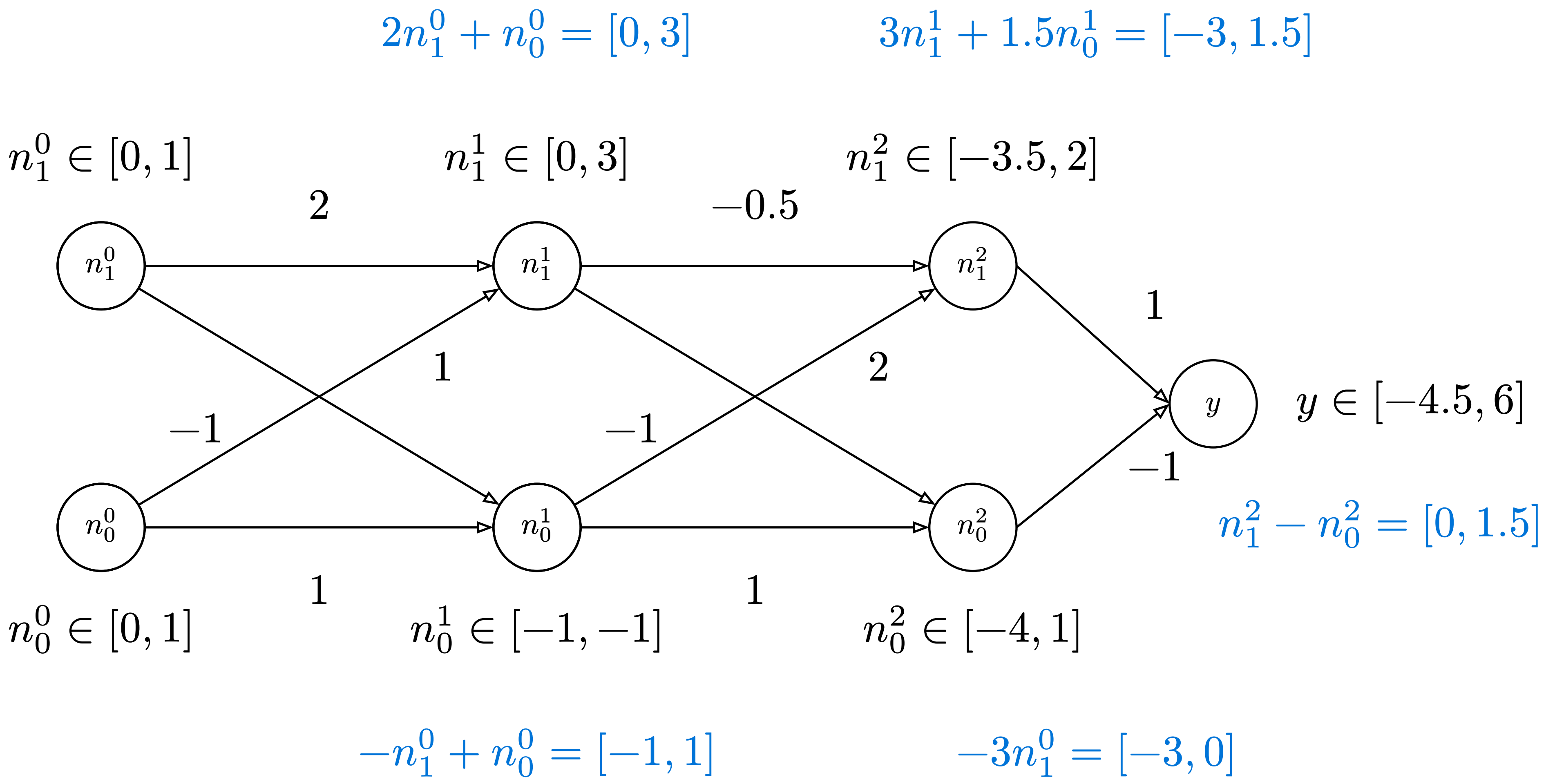


$$-n_1^0 + n_0^0 = [-1, 1] \qquad -(2n_1^0 + n_0^0) - n_1^0 + n_0^0$$

Symbolic propagation of variables *and their relations*



Symbolic propagation of variables *and their relations*



Symbolic propagation of variables *and their relations*

Relational domains

What we just did was using *relational domains*: computing convex sets that keep track of variable bounds *and their relations*.

There exist a real fauna of numerical domains, we will briefly skim through them.

⚠️ next slides might be gory

Relational domains

Zonotopes

A convex polytope with central symmetry

$$x_i = \left\{ \sum_{j=1}^{m} \alpha_{i,j}\varepsilon_j + \beta_i \mid \varepsilon \in [-1,1]^m \right\}$$

Here, $\alpha_{i,j}$ are symbolic relational variables, ε_j are noise symbols and β_j are symbolic variables.

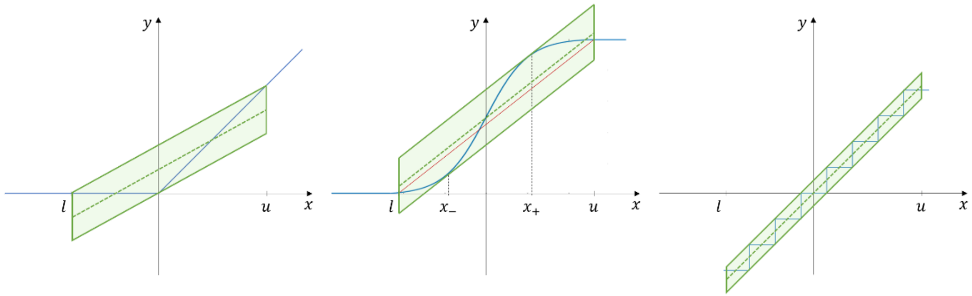


Figure 3: Activation functions and their *zonotope* abstraction (Relu, Sigmoid, Cast)

Zonotope abstraction for various activation functions, extracted from (Lemesle, Lehmann, and Gall 2024)

Relational domains

Constrained Zonotopes

A zonotope with additional constraints as seen in Convex Optimization

$x_i = \left\{\sum_{j=1}^{m} \alpha_{i,j}\varepsilon_j + \beta_i \mid \varepsilon \in [-1,1]^m \mid \forall k \in \{1..,\mathrm{K}\}, \sum_{j=1}^{m} \mathrm{A}_{k,j}\varepsilon_j + b_k \geq 0\right\}$ where $A_{k,j} \in \mathbb{R}$ and $b_k \in \mathbb{R}$

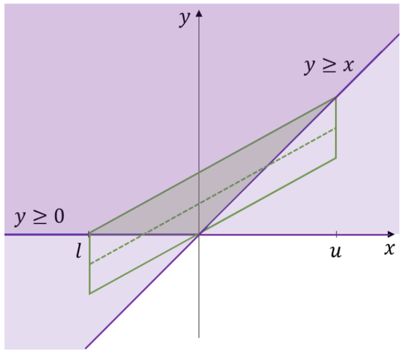


Figure 4: Abstraction of the ReLU with the *constrained zonotope* domain

Constrained Zonotope abstraction for the ReLU, extracted from (Lemesle, Lehmann, and Gall 2024)

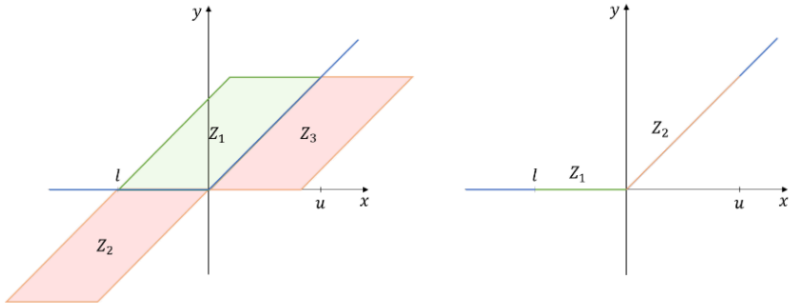
Relational domains

Hybrid Zonotopes

An hybrid zonotope is an union of constraint zonotopes defined by integer-only variables

$$x_i = \left\{ \sum_{j=1}^{m_c} \alpha_{i,j}\varepsilon_j^c + \sum_{j=1}^{m_b} \gamma_{i,j}\varepsilon_j^b + \beta_i \right.$$
$$\mid \varepsilon^c \in [-1,1]^m$$
$$\mid \varepsilon^b \in \{-1,1\}_b^m$$
$$\left. \forall k \in \{1..,\mathrm{K}\}, \sum_{j=1}^{m^c} a_{k,j}\varepsilon_j^c + \sum_{j=1}^{m^b} b_{k,j}\varepsilon_j^b + c_k = 0 \right\}$$

where $\gamma_{i,j}$ are called *binary generators* and $a_{k,j}, b_{k,j}, c_k \in \mathbb{R}$

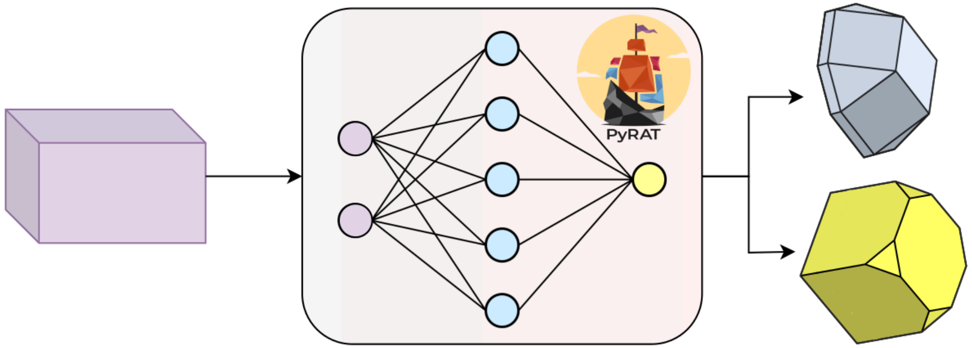


ure 5: *Hybrid zonotope* abstraction of the ReLU function. $H = Z_1 \cap (Z_2 \cup Z_3)$ [51] on the left. $H = Z_1$
n Zhang et al. [50] on the right.

Hybrid Zonotope abstraction for the ReLU, extracted from (Lemesle, Lehmann, and Gall 2024). Note that they can exactly represent piece-wise linear activations, but then require a dedicated solver to find bounds

And what to do with all that?

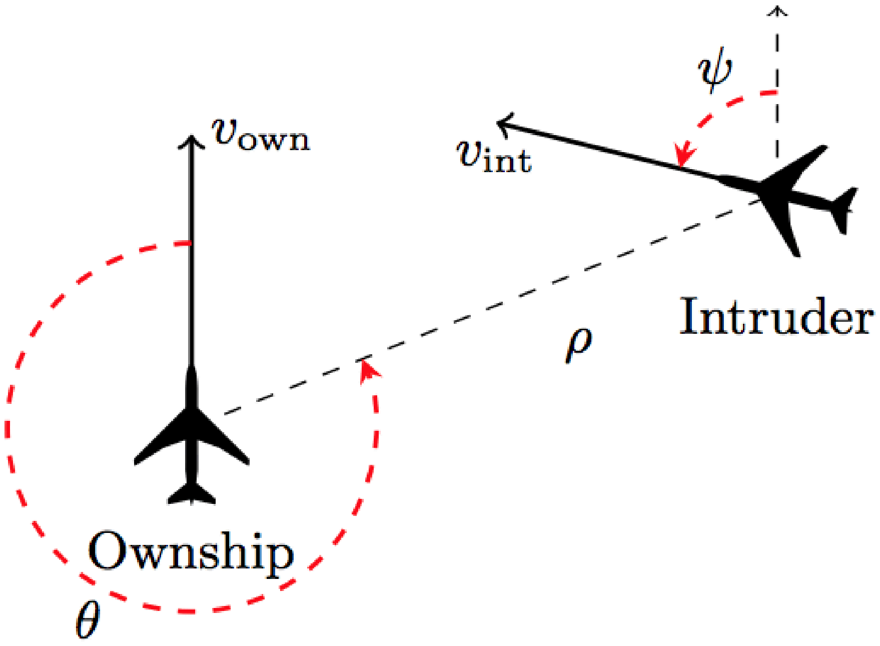
1. *transforms* a neural network f that maps concrete inputs $\mathbb{R}^d$ to concrete outputs $\mathbb{R}^p$ into a « neural network » that maps convex sets to convex sets `let abstract =` $(f : \mathbb{R}^d \to \mathbb{R}^p)$ `->` $(f^\# : \mathcal{X}^\# \xrightarrow{\#} \mathcal{Y}^\#)$
2. *compute* convex sets performing Interval Bound Propagation `let` $\left(\xrightarrow{\#}\right)$ `=` $\mathcal{X}^\#$ `->` $\mathcal{Y}^\#$
3. (during computation) *checks* the abstract outputs within the concrete outputs `let concretize =` $\mathcal{Y}^\#$ `->` $\mathbb{R}^p$
4. *check* that the abstract output lies within a safety set



And what to do with all that?

Choosing the proper abstraction is a crucial tradeoff between precision and speed

Not that this is not limited to local robustness!



Property ϕ_1.

- Description: If the intruder is distant and is significantly slower than the ownship, the score of a COC advisory will always be below a certain fixed threshold.
- Tested on: all 45 networks.
- Input constraints: $\rho \geq 55947.691$, $v_{\text{own}} \geq 1145$, $v_{\text{int}} \leq 60$.
- Desired output property: the score for COC is at most 1500.

Property ϕ_2.

- Description: If the intruder is distant and is significantly slower than the ownship, the score of a COC advisory will never be maximal.
- Tested on: $N_{x,y}$ for all $x \geq 2$ and for all y.
- Input constraints: $\rho \geq 55947.691$, $v_{\text{own}} \geq 1145$, $v_{\text{int}} \leq 60$.
- Desired output property: the score for COC is not the maximal score.

Property ϕ_3.

- Description: If the intruder is directly ahead and is moving towards the ownship, the score for COC will not be minimal.
- Tested on: all networks except $N_{1,7}$, $N_{1,8}$, and $N_{1,9}$.
- Input constraints: $1500 \leq \rho \leq 1800$, $-0.06 \leq \theta \leq 0.06$, $\psi \geq 3.10$, $v_{\text{own}} \geq 980$, $v_{\text{int}} \geq 960$.
- Desired output property: the score for COC is not the minimal score.

Checking a safety property is equivalent of checking for set inclusion

Some results and examples

Briefly, extracted from the VNN-Comp reports (Brix et al. 2023):

- ACAS-Xu property check takes $\sim 11s$ mean
  - **global** proof 🎉
- CIFAR100 Local Robustness takes $\sim 85s$ mean
  - **local** proof 🤔

Extensions

We can make a training scheme out of this!

cf. the whole domain of adversarial attacks and defense on ESSAI courses *Bridging Adversarial Learning and Data-Centric AI for Robust AI* and *AI for Security: Exploring Adversarial Learning and the Transferability of Adversarial Attacks*

Core idea: use formal verification *during training* to train the neural network to predict the correct class against a worst-perturbation ε

More details: (Balunovic and Vechev 2020; Jovanović et al. 2022)

Limitations

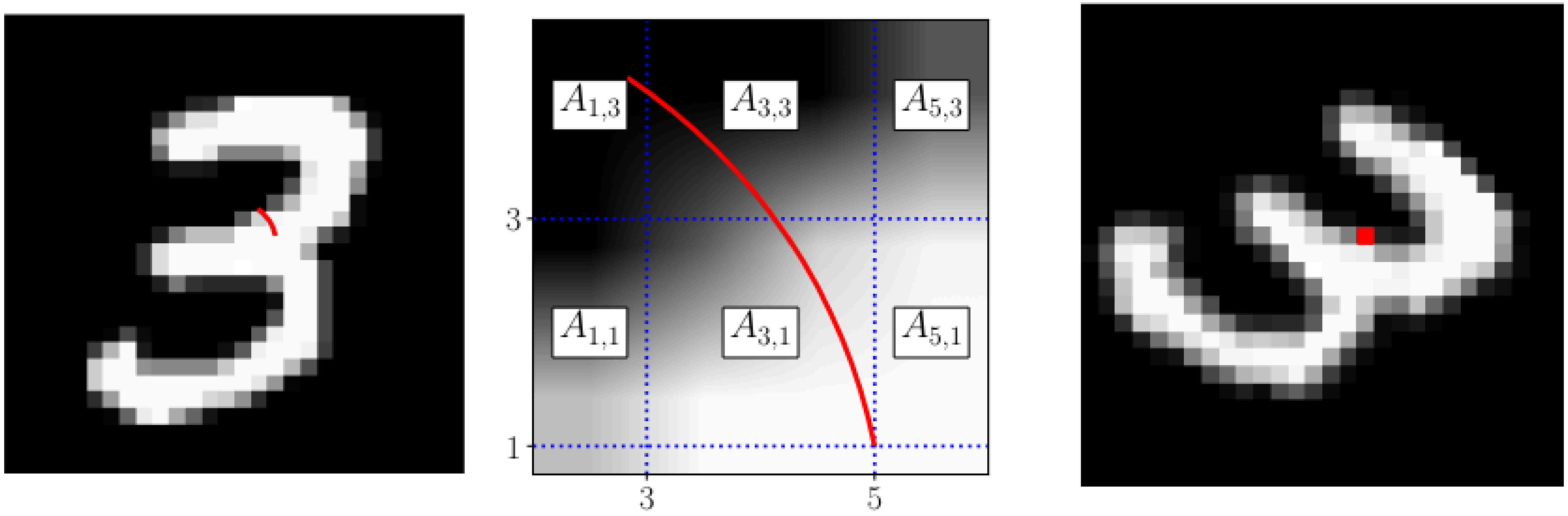
Threat model

Local robustness (Katz et al. 2017)

Let a classifier $f : \mathcal{X} \mapsto \mathcal{Y}$. Given $x \in \mathcal{X}$ and $\varepsilon \in \mathbb{R} \lll 1$ the problem of *local robustness* is to prove that $\forall x'.\ \|x - x'\|_p < \varepsilon \rightarrow f(x) = f(x')$

- choosing the correct ε depends on the use case
- local robustness is only valid up to a given ε
  - usually, $\frac{10}{255}$ on MNIST and $\frac{2}{255}$ on bigger datasets; not much…
  - it costs almost nothing to attack, depending on your threat model
- physical adversarial examples are difficult to protect against

Semantic transformations



(a) Original image (b) Interpolation (c) Rotated image

How to encode image rotations (Balunovic et al. 2019) ?

Hyperproperties

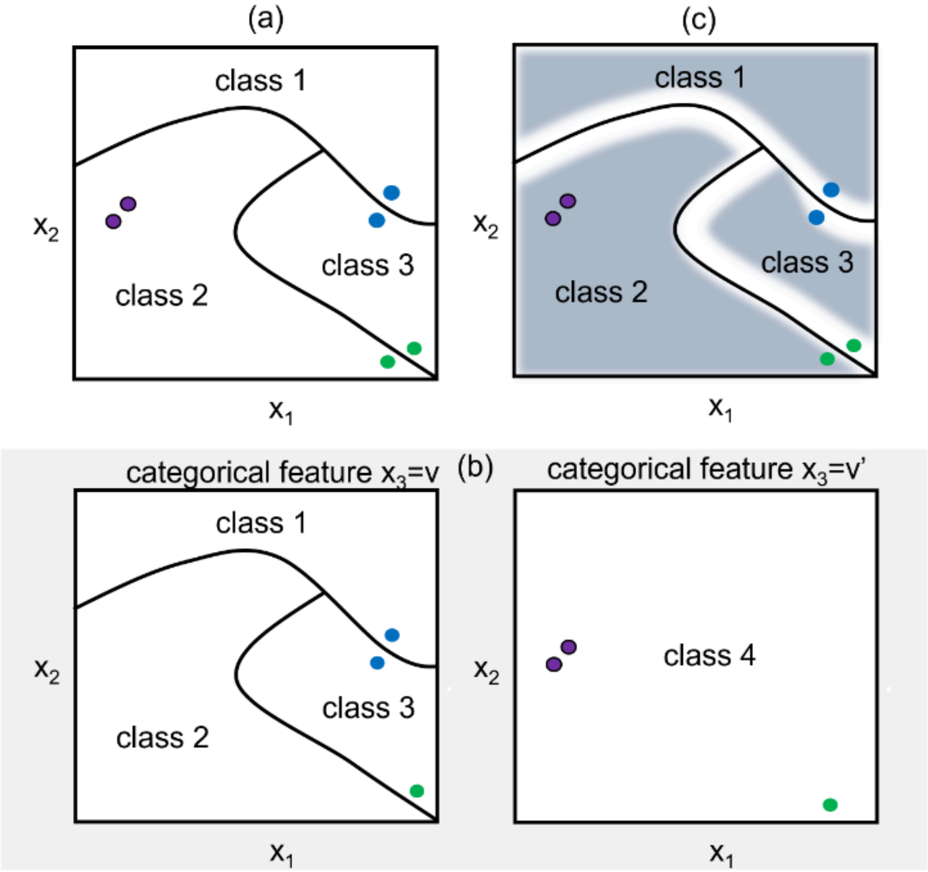
Confidence-based robustness (Athavale et al. 2024)

$$\forall x, x', \mathrm{cond}(x, x', \varepsilon) \land \mathrm{conf}(f(x)) > \kappa \Rightarrow \mathrm{class}(f(x)) = \mathrm{class}(f(x'))$$

For all couple of inputs, as long as the network is confident enough in its prediction, prediction should not change

And a whole family of *hyperproperties* (multiple execution traces)

Global robustness



Next sessions

In session 3

- Formal explanations, or how to use Formal Methods to provide guarantees explanations (Marques-Silva and Huang 2024)

In session 4

- Using formal verification to debug neural networks
- Testing neural networks
- Certified Training ?

In session 5

- Languages to express specifications
- Tool to check those specifications
- Variety of community initiatives (competitions, benchmarks, conferences)
- Open questions and discussions

Thanks for your attention!

Very short feedback

- Was the technical level alright?
- Are there some notions you may want to focus more or discuss further?
- General questions?

Bibliography

Athavale, Anagha, Ezio Bartocci, Maria Christakis, Matteo Maffei, Dejan Nickovic, and Georg Weissenbacher. 2024. "Verifying Global Two-Safety Properties in Neural Networks with Confidence". In *Computer Aided Verification*, edited by Arie Gurfinkel and Vijay Ganesh, 329–351. Cham: Springer Nature Switzerland. doi:*10.48550/arXiv.2405.14400*.

Balunovic, Mislav, and Martin Vechev. 2020. "Adversarial Training and Provable Defenses: Bridging the Gap". In *International Conference on Learning Representations*. *https://openreview.net/forum?id=SJxSDxrKDr*.

Balunovic, Mislav, Maximilian Baader, Gagandeep Singh, Timon Gehr, and Martin Vechev. 2019. "Certifying Geometric Robustness of Neural Networks". In *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc.. *https://*

papers.nips.cc/paper/2019/hash/f7fa6aca028e7ff4ef62d75ed025fe76-Abstract.html.

Brix, Christopher, Stanley Bak, Changliu Liu, and Taylor T. Johnson. 2023. “The Fourth International Verification of Neural Networks Competition (VNN-COMP 2023): Summary and Results.”

Casadio, Marco, Ekaterina Komendantskaya, Matthew L Daggitt, Wen Kokke, Guy Katz, Guy Amir, and Idan Refaeli. 2022. “Neural Network Robustness as a Verification Property: A Principled Case Study”. In *International Conference on Computer Aided Verification*, 219–231.

Cousot, Patrick, and Radhia Cousot. 1977. “Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints”. In *POPL*, 238–252. doi:[10.1145/512950.512973](10.1145/512950.512973).

Jovanović, Nikola, Mislav Balunovic, Maximilian Baader, and Martin Vechev. 2022. “On the Paradox of Certified Training”. *Transactions on Machine Learning Research*, June. https://openreview.net/forum?id=atJHLVyBi8.

Katz, Guy, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. 2017. “Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks”. In *Computer Aided Verification*, 97–117. Springer International Publishing. doi:10.1007/978-3-319-63387-9_5.

Kroening, Daniel, and Ofer Strichman. 2016. *Decision Procedures*. Springer Berlin Heidelberg. doi:10.1007/978-3-662-50497-0.

Lemesle, Augustin, Julien Lehmann, and Tristan Le Gall. 2024. “Neural Network Verification with Pyrat”. arXiv. doi:10.48550/ARXIV.2410.23903.

Marques-Silva, Joao, and Xuanxiang Huang. 2024. "Explainability Is Not a Game". *Communications of the ACM* 67 (7). Association for Computing Machinery (ACM): 66–75. doi:10.1145/3635301.

Miné, Antoine. 2017. "Tutorial on Static Inference of Numeric Invariants by Abstract Interpretation". *Foundations and Trends® in Programming Languages* 4 (3–4): 120–372. doi:10.1561/2500000034.

Mirman, Matthew, Timon Gehr, and Martin Vechev. 2018. "Differentiable Abstract Interpretation for Provably Robust Neural Networks". In , 3578–3586. PMLR. https://proceedings.mlr.press/v80/mirman18b.html.

Tjeng, Vincent, Kai Xiao, and Russ Tedrake. 2019. "Evaluating Robustness of Neural Networks with Mixed Integer Programming". In . https://openreview.net/pdf?id=HyGIdiRqtm.

Ślusarz, Natalia, Ekaterina Komendantskaya, Matthew L. Daggitt, Robert Stewart, and Kathrin Stark. 2023. “Logic of Differentiable Logics: Towards a Uniform Semantics of Dl.”