

Introduction to Neural Network Verification by Abstract Interpretation

LATTICE THEORY FOR PARALLEL PROGRAMMING

Yi-Nung Tsao

yi-nung.tsao@uni.lu

6th Dec 2024

University of Luxembourg



Outline

- Background
- Verifying Neural Networks by Abstract Interpretation
- Summary

Background

Some of Neural Network Applications - Image Generator



prompt: a cat is using laptop with beer

Some of Neural Network Applications - ChatGPT

who are you?



I am ChatGPT, an AI language model created by OpenAI. I am based on the GPT-4 architecture and designed to assist with a wide range of tasks, including answering questions, providing explanations, generating text, and more. How can I assist you today?

Why is the sky blue?



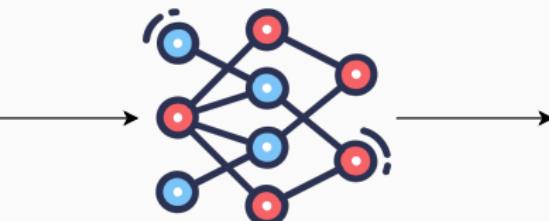
The sky appears blue due to a phenomenon called Rayleigh scattering. This effect occurs because the molecules and small particles in the Earth's atmosphere scatter sunlight in all directions. Here's a more detailed explanation:

1. **Sunlight Composition:** Sunlight, or white light, is made up of many different colors of light, each with its own wavelength. These colors range from violet and blue, which have shorter wavelengths, to red and orange, which have longer wavelengths.

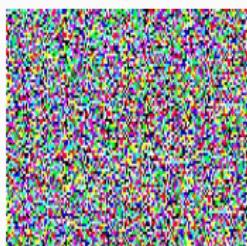
Some of Neural Network Applications - Self-driving car



The Limitation of Neural Network - Robustness Issue

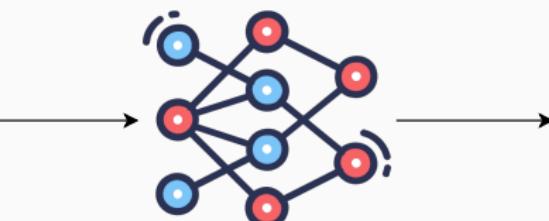


cat



+

II



pig

The Limitation of Neural Network - Robustness Issue

Why is the robustness issue important?



Stop

Max Speed 100

It is a safety and critical issue.

Robustness Issue in Natural Language Process

Task: Sentiment Analysis. **Classifier:** CNN. **Original label:** 99.8% Negative. **Adversarial label:** 81.0% Positive.

Text: I love these awful **awful** **ul** 80's summer camp movies. The best part about "Party Camp" is the fact that it literally **literally** has no **No** plot. The **eliches** **clichs** here are limitless: the nerds vs. the jocks, the secret camera in the girls locker room, the hikers happening upon a nudist colony, the contest at the conclusion, the secretly horny camp administrators, and the embarrassingly **embarrassingly** **foolish** sexual innuendo littered throughout. This movie will make you laugh, but never intentionally. I repeat, never.

Task: Sentiment Analysis. **Classifier:** Amazon AWS. **Original label:** 100% Negative. **Adversarial label:** 89% Positive.

Text: I watched this movie recently mainly because I am a Huge fan of Jodie Foster's. I saw this movie was made right between her 2 Oscar award winning performances, so my expectations were fairly high. Unfortunately **Unf0rtunately**, I thought the movie was terrible **terrib1e** and I'm still left wondering how she was ever persuaded to make this movie. The script is really weak **wea k**.

Task: Toxic Content Detection. **Classifier:** LSTM. **Original label:** 96.7% Toxic. **Adversarial label:** 83.5% Non-toxic.

Text: hello how are you? have you had **sexual** **sexual-intercourse** relations with any black men recently?

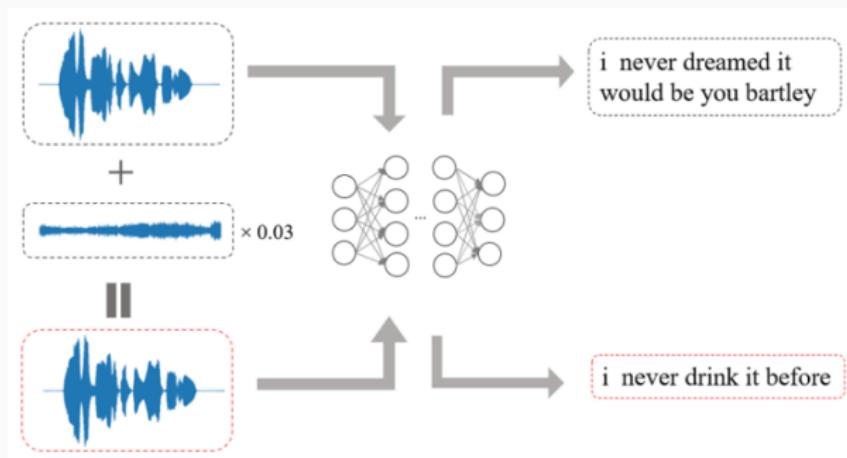
Task: Toxic Content Detection. **Classifier:** Perspective. **Original label:** 92% Toxic. **Adversarial label:** 78% Non-toxic.

Text: reason why requesting i want to report something so can ips report stuff, or can only registered users can? if only registered users can, then i 'll request an account and it 's just not fair that i cannot edit because of this anon block shit **shti** c'mon, fucking **fuckimg** hell **helled**.

1

¹Li et al. (2018). Textbugger: Generating adversarial text against real-world applications. arXiv preprint arXiv:1812.05271.

Robustness Issue in Speech Recognition

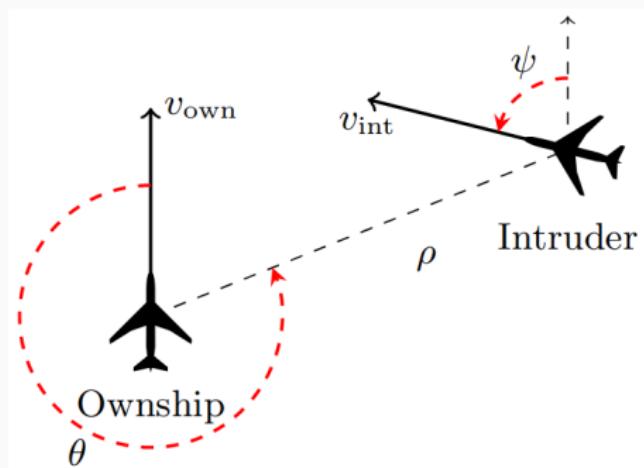


2

²Wang et al. (2020, October). Adversarial examples attack and countermeasure for speech recognition system: A survey. In International Conference on Security and Privacy in Digital Economy.

Beyond Robustness Issue in Neural Networks

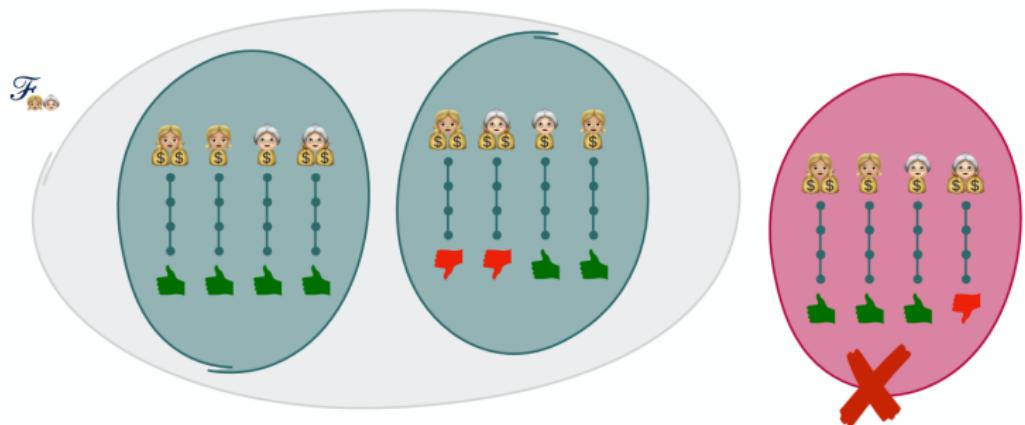
Safety Issue



3

³Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. In Computer Aided Verification: 29th International Conference, CAV 2017.

Dependency Fairness



Objective

In this field, we are not focusing on training or developing neural networks.

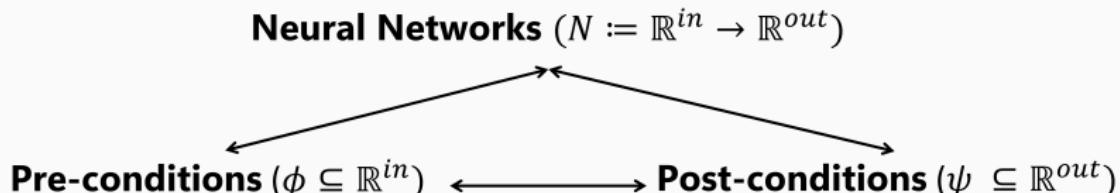
Instead, we aim to identify any critical issues in neural networks before deployment.

How can we do?

Neural Network Verification

Definition of Neural Network Verification (NNV)

There are 3 required components to define neural network verification.



NNV can be defined as:

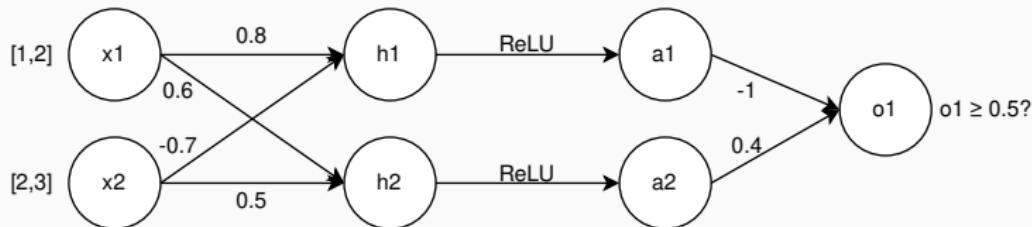
$$N(x) \models \psi, \quad \forall x \in \phi \quad (1)$$

If (1) is true, return **UNSAT**.

Otherwise, return **SAT** and at least one counter example.

A Simple Example for Neural Network Verification

- Neural Network



- Pre-conditions

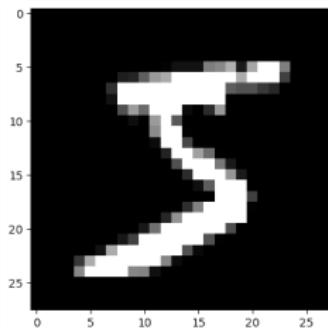
$$1 \leq x_1 \leq 2 \wedge 2 \leq x_2 \leq 3$$

- Post-conditions

$$o_1 \geq 0.5$$

※ In this talk, we only focus the activation function in N is ReLU.

A Realistic Example - NNV with Robustness Property



1. **Neural Network** ($N := \mathbb{R}^{28 \times 28} \rightarrow \mathbb{R}^{10}$)

2. **Pre-conditions** ($\mathbf{x}' \in \phi$)

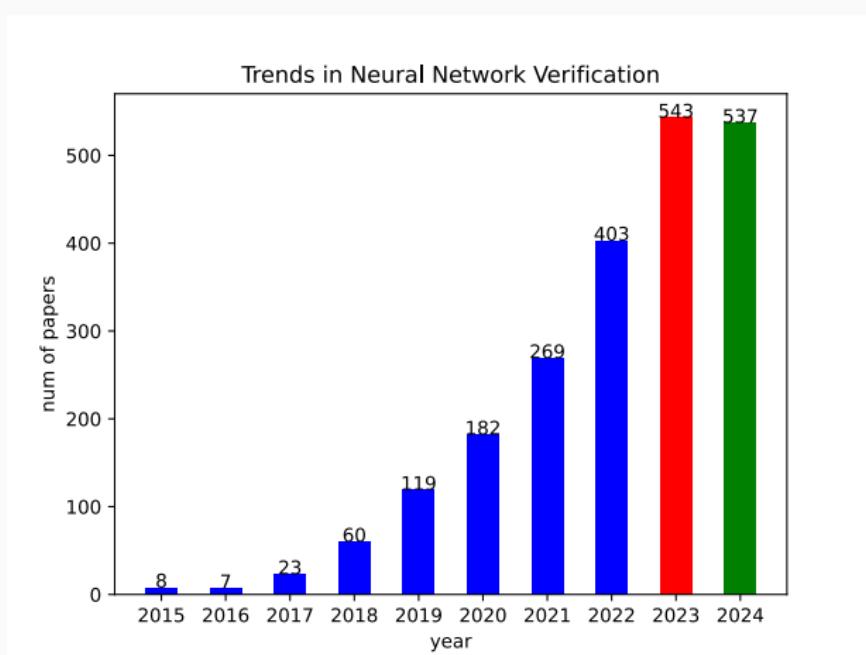
$$\|x_{ij} - x'_{ij}\|_p \leq \epsilon$$

3. **Post-conditions**

Suppose y is true label for x , $\text{argmax } N(\mathbf{x}') = y$

How is this research field growing?

Keywords: "neural network verification" on GoogleScholar until 5th Dec 2024.



How to check all perturbed images in L_p -norm region?

| | | $p \models \mathcal{P}$ | |
|-----------------|-------|-------------------------|---------------|
| | | True | False |
| analysis(p) | True | correct | Type II Error |
| | False | Type I Error | correct |

Definition (Soundness)

For every program $p \in \mathbb{L}$, $\text{analysis}(p) = \text{true} \implies p \models \mathcal{P}$.
 \Leftrightarrow there is no type II error.

Definition (Completeness)

For every program $p \in \mathbb{L}$, $\text{analysis}(p) = \text{true} \iff p \models \mathcal{P}$.
 \Leftrightarrow there is no type I error.

⁴Rival, X., & Yi, K. (2020). Introduction to static analysis: an abstract interpretation perspective. Mit Press.

How to check all perturbed images in L_p -norm region?

Intuitively, we can build a mixed-integer programming model as following.

$$\min_{\mathbf{x}, \hat{\mathbf{x}}} \hat{\mathbf{x}}_n \quad (2a)$$

$$\text{s.t.} \quad \mathbf{x}_0 \in \phi \quad (2b)$$

$$\hat{\mathbf{x}}_{k+1} = W_{k+1} \mathbf{x}_k + b_{k+1} \quad \forall k \in \{1, \dots, |\mathcal{L}| - 1\} \quad (2c)$$

$$\mathbf{x}_k = \sigma(\hat{\mathbf{x}}_k) \quad \forall k \in \mathcal{L} \quad (2d)$$

$$\mathbf{x}_k \in \mathbb{R}, \hat{\mathbf{x}}_k \geq 0 \quad \forall k \in \mathcal{L} \quad (2e)$$

, where $\mathbf{x}_k = \sigma(\hat{\mathbf{x}}_k) \quad \forall k \in \mathcal{L}$

$\Leftrightarrow \mathbf{x}_k \leq M \mathbf{z}_k \wedge \mathbf{x}_k \leq M(1 - \mathbf{z}_k)$, where $\mathbf{z}_k \in \{0, 1\} \quad \forall k \in \mathcal{L}$

How to check all perturbed images in L_p -norm region?

Intuitively, we can build a mixed-integer programming model as following.

$$\min_{\mathbf{x}, \hat{\mathbf{x}}} \hat{\mathbf{x}}_n \quad (2a)$$

$$\text{s.t.} \quad \mathbf{x}_0 \in \phi \quad (2b)$$

$$\hat{\mathbf{x}}_{k+1} = W_{k+1} \mathbf{x}_k + b_{k+1} \quad \forall k \in \{1, \dots, |\mathcal{L}| - 1\} \quad (2c)$$

$$\mathbf{x}_k = \sigma(\hat{\mathbf{x}}_k) \quad \forall k \in \mathcal{L} \quad (2d)$$

$$\mathbf{x}_k \in \mathbb{R}, \hat{\mathbf{x}}_k \geq 0 \quad \forall k \in \mathcal{L} \quad (2e)$$

, where $\mathbf{x}_k = \sigma(\hat{\mathbf{x}}_k) \quad \forall k \in \mathcal{L}$

$$\Leftrightarrow \mathbf{x}_k \leq M \mathbf{z}_k \wedge \mathbf{x}_k \leq M(1 - \mathbf{z}_k), \text{ where } \mathbf{z}_k \in \{0, 1\} \quad \forall k \in \mathcal{L}$$

This approach is **soundness** and **completeness**, but ... **only applicable for very small instances**.

How to check all perturbed images in L_p -norm region?

Verifying properties in deep neural networks with ReLUs is **NP-Complete**⁵

⁵Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. CAV 2017.

How to check all perturbed images in L_p -norm region?

Verifying properties in deep neural networks with ReLUs is **NP-Complete**⁵

- **Enumeration:**

Enumerate all perturbed images in ϕ .

⁵Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. CAV 2017.

How to check all perturbed images in L_p -norm region?

Verifying properties in deep neural networks with ReLUs is **NP-Complete**⁵

- **Enumeration:**

Enumerate all perturbed images in ϕ .

- **Attack:**

Find adversarial example \rightarrow Complete, **Unsound**

⁵Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. CAV 2017.

How to check all perturbed images in L_p -norm region?

Verifying properties in deep neural networks with ReLUs is **NP-Complete**⁵

- **Enumeration:**

Enumerate all perturbed images in ϕ .

- **Attack:**

Find adversarial example \rightarrow Complete, **Unsound**

- Formal Verification - Abstract Interpretation \rightarrow **Sound**

⁵Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. CAV 2017.

How to check all perturbed images in L_p -norm region?

Verifying properties in deep neural networks with ReLUs is **NP-Complete**⁵

- **Enumeration:**

Enumerate all perturbed images in ϕ .

- **Attack:**

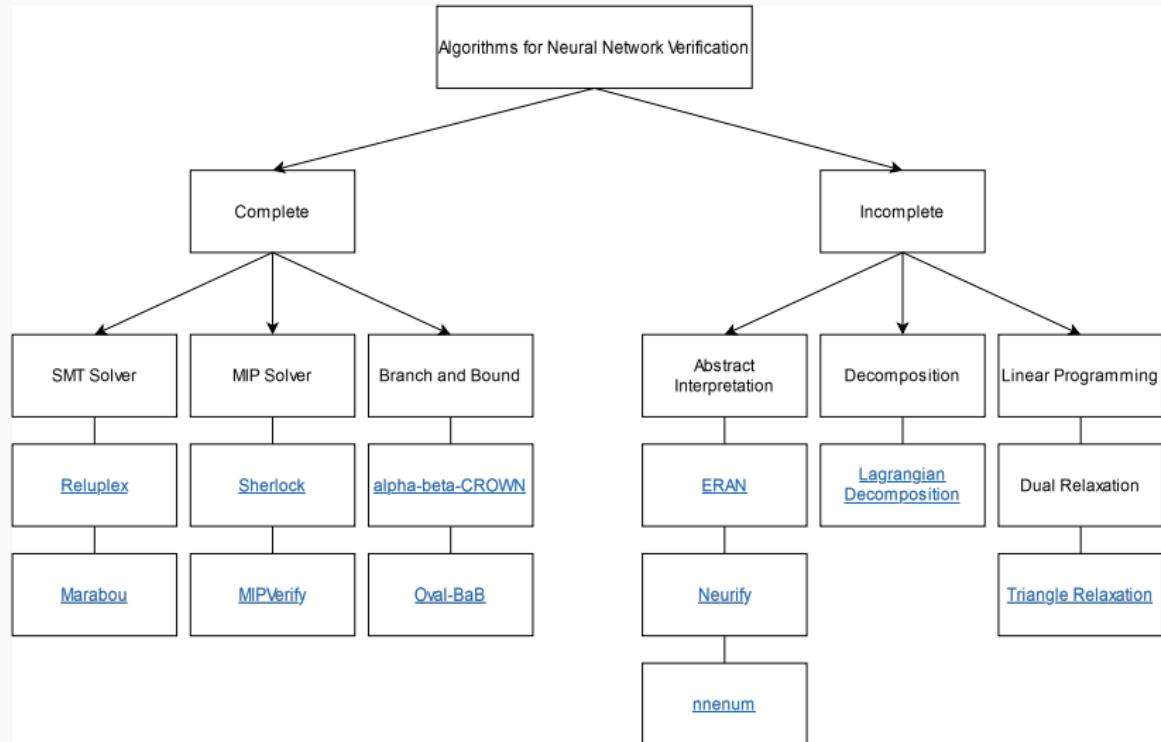
Find adversarial example \rightarrow Complete, **Unsound**

- Formal Verification - Abstract Interpretation \rightarrow **Sound**

- Formal Verification - Abstract Interpretation \rightarrow Incomplete

⁵Katz et al. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. CAV 2017.

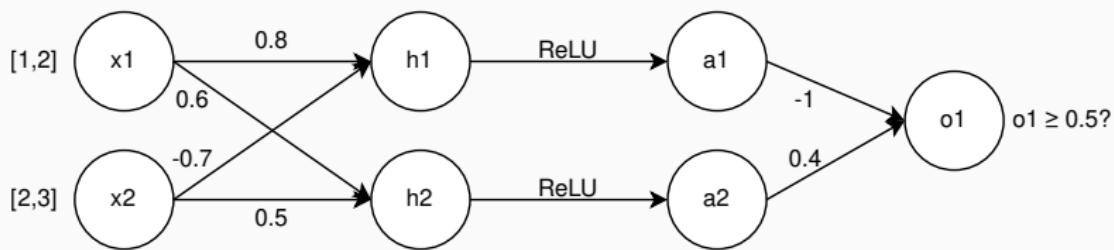
How can we verify neural networks?



Formal Verification - Abstract Interpretation

Concrete Semantics

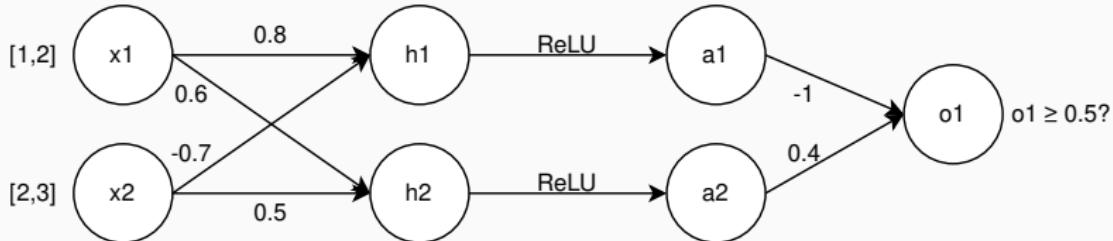
Verifying Feed-forward Neural Network



Concrete Semantics:

The all possible values for each neuron.

Feed-forward Neural Network - Concrete Semantics



Given:

$$1 \leq x_1 \leq 2, 2 \leq x_2 \leq 3$$

1. **Weighted Sum:** $f_s(x_1, \dots, x_n) = \sum_i w_i x_i$

$$h_1 = 0.8x_1 + (-0.7)x_2$$

$$h_2 = 0.6x_1 + 0.5x_2$$

$$o_1 = -1a_1 + 0.4a_2$$

2. **ReLU:** $f_a(x) = \max(0, x)$

$$a_1 = \max(0, h_1)$$

$$a_2 = \max(0, h_2)$$

Abstract Semantics

Interval Domain

Definition

The lattice of interval $\langle \mathcal{I}, \sqsubseteq, \sqcup, \sqcap, \perp, [-\infty, \infty] \rangle$ is defined as:

$$\mathcal{I} \triangleq \{[a, b] \mid a \in \mathbb{Z} \cup \{-\infty\}, b \in \mathbb{Z} \cup \{\infty\}, a \sqsubseteq b\} \cup \{\perp\}$$

with the following operations:

- $[a, b] \sqsubseteq [c, d] \Leftrightarrow a \geq c \wedge b \leq d.$
- $[a, b] \sqcup [c, d] \triangleq [\min(a, c), \max(b, d)].$
- $[a, b] \sqcap [c, d] \triangleq [\max(a, c), \min(b, d)].$

We also define projection functions $\lfloor [a, b] \rfloor \triangleq a$ and $\lceil [a, b] \rceil \triangleq b.$

Interval Domain - Basic Operators

$$\mathcal{I}_1 = [-3, 5], \mathcal{I}_2 = [2, 4]$$

- **Addition**

$$\mathcal{I}_1 + \mathcal{I}_2 = [-3 + 2, 5 + 4] = [-1, 9]$$

- **Subtraction**

$$\mathcal{I}_1 - \mathcal{I}_2 = [-3 - 2, 5 - 4] = [-5, 1]$$

- **Multiplication**

$$\mathcal{I}_1 \times \mathcal{I}_2 = [\min(B), \max(B)] = [-12, 20]$$

, where $B = \{-3 \times 2, -3 \times 4, 5 \times 2, 5 \times 4\}$

Interval Domain - Basic Operators

$$\mathcal{I}_1 = [-3, 5], \mathcal{I}_2 = [2, 4]$$

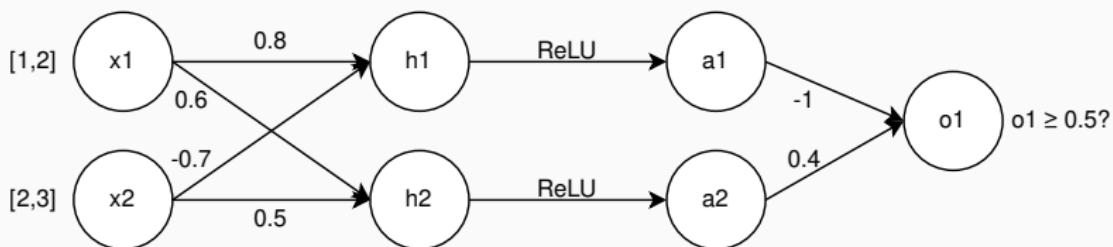
- **Join**

$$\mathcal{I}_1 \sqcup \mathcal{I}_2 = [\min(-3, 2), \max(5, 4)] = [-3, 5]$$

- **Meet**

$$\mathcal{I}_1 \sqcap \mathcal{I}_2 = [\max(-3, 2), \min(5, 4)] = [2, 4]$$

Feed-forward Neural Network - Abstract Operations



Abstract Transformer:

1. Weighted Sum:

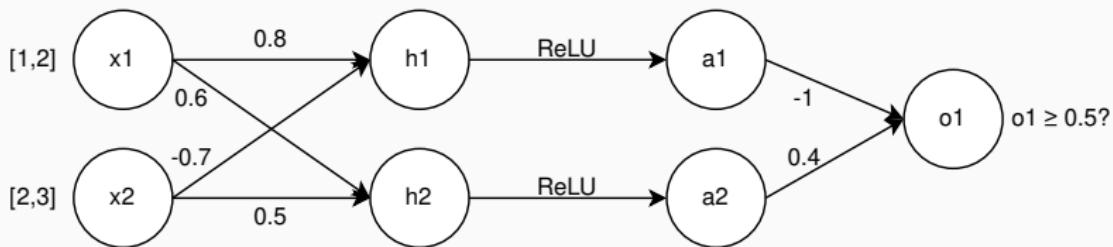
$$f_s^{\#}([l_1, u_1], \dots, [l_n, u_n]) = [\sum_i l'_i, \sum_i u'_i]$$

where $l'_i = \min(w_i l_i, w_i u_i)$ and $u'_i = \max(w_i l_i, w_i u_i)$

2. ReLU:

$$f_a^{\#}([l, u]) = [\max(0, l), \max(0, u)]$$

Example



- **Pre-conditions:**

$$1 \leq x_1 \leq 2$$

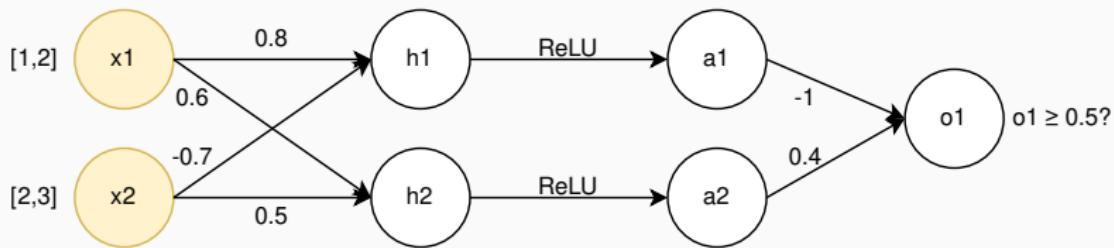
$$2 \leq x_2 \leq 3$$

- **Post-conditions:**

$$o_1 \geq 0.5$$

- Fully Connected & ReLU activation function

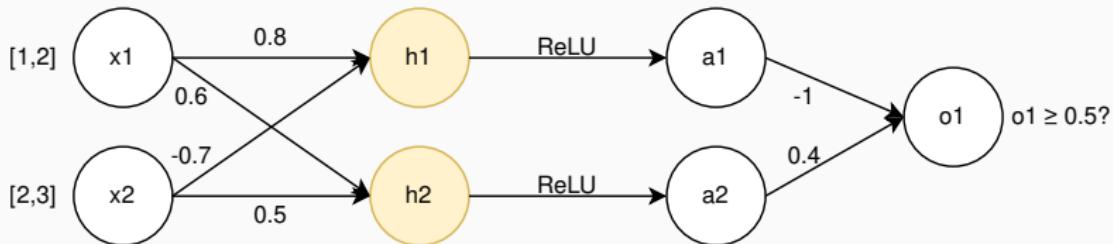
Example - Interval Domain - Input Layer



By pre-conditions:

$$x_1 = [1, 2], x_2 = [2, 3]$$

Example - Interval Domain - Hidden Layer



From Input Layer:

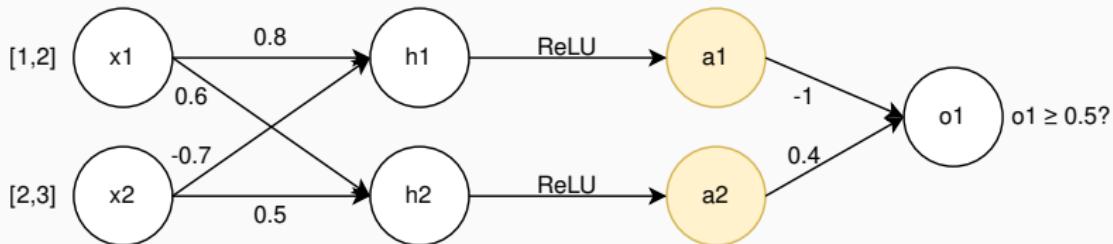
$$x_1 = [1, 2], x_2 = [2, 3]$$

Abstract Affine Function:

$$h_1 = 0.8 \times [1, 2] + (-0.7) \times [2, 3] = [-1.3, 0.2]$$

$$h_2 = 0.6 \times [1, 2] + 0.5 \times [2, 3] = [1.6, 2.7]$$

Example - Interval Domain - Activation Layer



From Hidden Layer:

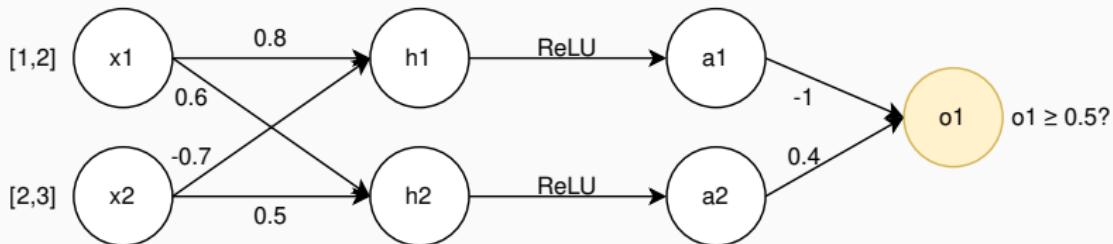
$$h_1 = [-1.3, 0.2], h_2 = [1.6, 2.7]$$

Abstract ReLU Activation Function:

$$a_1 = [\text{ReLU}(-1.3), \text{ReLU}(0.2)] = [0, 0.2]$$

$$a_2 = [\text{ReLU}(1.6), \text{ReLU}(2.7)] = [1.6, 2.7]$$

Example - Interval Domain - Output Layer



From Activation Layer:

$$a_1 = [0, 0.2], a_2 = [1.6, 2.7]$$

Abstract Affine Function:

$$o_1 = -1 \times [0, 0.2] + 0.4 \times [1.6, 2.7] = [0.44, 1.08]$$

UNPROVED!

Summary of Interval Domain

| $p \models \mathcal{P}$ | True | False |
|-------------------------|--------------|---------------|
| analysis(p) | correct | Type II Error |
| True | correct | Type II Error |
| False | Type I Error | correct |

Definition (Soundness)

For every program $p \in \mathbb{L}$, $\text{analysis}(p) = \text{true} \implies p \models \mathcal{P}$.

Definition (Completeness)

For every program $p \in \mathbb{L}$, $\text{analysis}(p) = \text{true} \iff p \models \mathcal{P}$.

Is the property really not satisfied for this neural network? or type I error?

Zonotope Domain

Extended Interval Domain - Zonotope Domain

Definition (Abstraction - Affine Expression)

$$\alpha(x) = \mathcal{Z} = c + \sum_j d_j y_j, \quad c, d_j \in \mathbb{R}, y_j \in [-1, 1]$$

, where c is center, y_j is generator

Definition (Concretization)

$$\gamma(\mathcal{Z}) = [c - \sum_j |d_j|, c + \sum_j |d_j|]$$

Example

$$\alpha(x) = \mathcal{Z} = 1 + 3y_1 - 2y_2$$

$$\gamma(\mathcal{Z}) = [1 - 3 - 2, 1 + 3 + 2] = [-4, 6]$$

Zonotope Domain - Basic Operators

$$\mathcal{Z}_1 = 3 + y_1 - 2y_2, \mathcal{Z}_2 = 1 - y_1 + 3y_2$$

- **Addition**

$$\begin{aligned}\mathcal{Z}_1 + \mathcal{Z}_2 &= 3 + y_1 - 2y_2 + 1 - y_1 + 3y_2 \\ &= 4 + y_2\end{aligned}$$

- **Subtraction**

$$\begin{aligned}\mathcal{Z}_1 - \mathcal{Z}_2 &= 3 + y_1 - 2y_2 - (1 - y_1 + 3y_2) \\ &= 2 + 2y_1 - 5y_2\end{aligned}$$

- **Multiplication**

$$\begin{aligned}\mathcal{Z}_1 \times \mathcal{Z}_2 &= (3 + y_1 - 2y_2) \times (1 - y_1 + 3y_2) \\ &= 3 - 2y_1 + 7y_2 + 5y_1y_2 - y_1^2 - 6y_2^2\end{aligned}$$

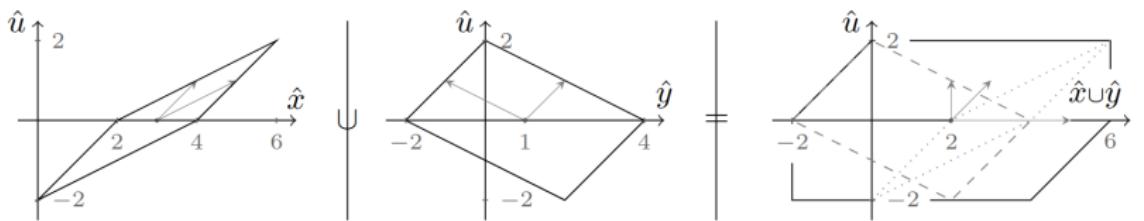
Zonotope Domain - Basic Operators

Definition 1. The join operation $\hat{z} = \hat{x} \cup \hat{y}$ defines an upper bound of \hat{x} and \hat{y} , which is minimal in “generic” situations, and whose interval concretisation is the union of interval concretisations of \hat{x} and \hat{y} :

$$\begin{aligned}\alpha_0^{\hat{z}} &= \text{mid}(\gamma(\hat{x}) \cup \gamma(\hat{y})) && (\text{central value of } \hat{z}) \\ \alpha_i^{\hat{z}} &= \underset{\substack{\min(\alpha_i^x, \alpha_i^y) \leq \alpha \leq \max(\alpha_i^x, \alpha_i^y)}{ \text{argmin}} (|\alpha|), \forall i \geq 1 && (\text{coeff. of } \epsilon_i) \\ \beta^{\hat{z}} &= \sup(\gamma(\hat{x}) \cup \gamma(\hat{y})) - \alpha_0^{\hat{z}} - \sum_{i \geq 1} |\alpha_i^{\hat{z}}| && (\text{coeff. of } \epsilon_U)\end{aligned}$$

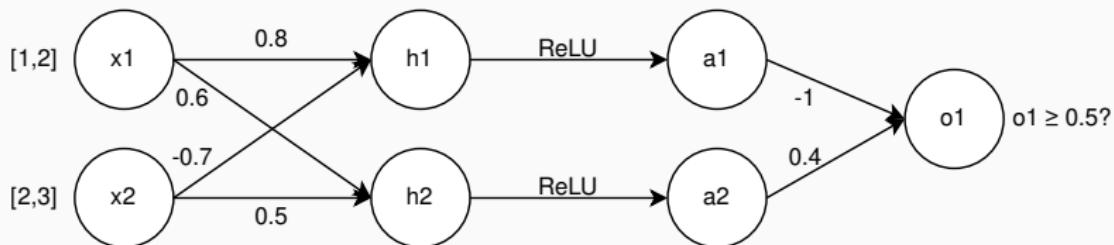
where the γ function returns the interval concretisation of an affine form and $\text{mid}([a, b]) := \frac{1}{2}(a + b)$ and $\text{argmin}(|x|) := \{x \in [a, b], |x| \text{ is minimal}\}$.

$$\left(\begin{array}{l} \hat{x} = 3 + \epsilon_1 + 2\epsilon_2 \\ \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{array} \right) \cup \left(\begin{array}{l} \hat{y} = 1 - 2\epsilon_1 + \epsilon_2 \\ \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{array} \right) = \left(\begin{array}{l} \hat{x} \cup \hat{y} = 2 + \epsilon_2 + 3\epsilon_U \\ \hat{u} \cup \hat{u} = 0 + \epsilon_1 + \epsilon_2 \end{array} \right)$$



⁶Ghorbal et al. (2009). The zonotope abstract domain taylor1+. In Computer Aided Verification: 21st International Conference, CAV 2009.

Zonotope Domain - Neural Network Operators



Abstract Transformer:

1. Weighted Sum:

$$\begin{aligned} f_s^\#(\mathcal{Z}_1, \dots, \mathcal{Z}_n) &= \sum_i w_i \mathcal{Z}_i \\ &= \sum_i w_i (c_i + d_{ji} y_{ji}) \end{aligned}$$

Zonotope Domain - Neural Network Operators

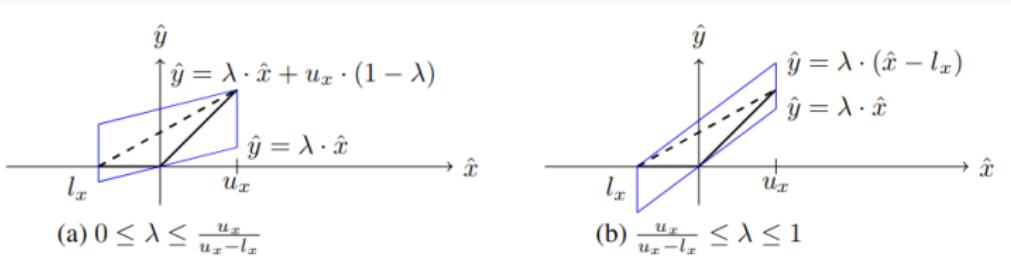


Figure 1: Two zonotope approximations for the ReLU function parameterized by the slope λ .

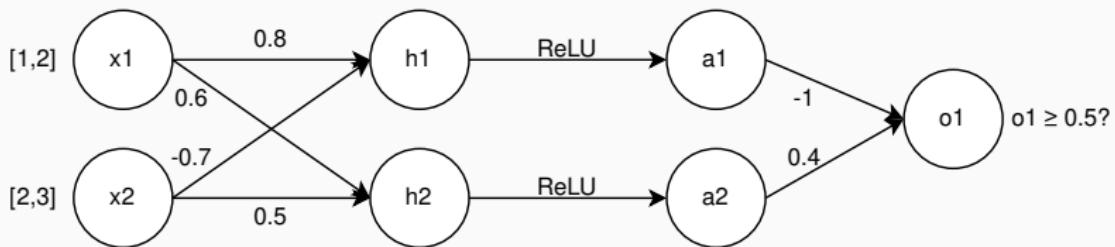
2. ReLU:

$$f_a^\#(\mathcal{Z}) = \begin{cases} \mathcal{Z} & \text{if } l_{\mathcal{Z}} > 0 \\ 0 & \text{if } u_{\mathcal{Z}} \leq 0 \\ \lambda \mathcal{Z} + \mu + \mu y_{\text{new}} & \text{otherwise} \end{cases}$$

, where $\lambda = \frac{u_{\mathcal{Z}}}{u_{\mathcal{Z}} - l_{\mathcal{Z}}}$, $\mu = \frac{-u_{\mathcal{Z}} \times l_{\mathcal{Z}}}{2 \times (u_{\mathcal{Z}} - l_{\mathcal{Z}})}$

⁷Singh et al. (2018). Fast and effective robustness certification. Advances in neural information processing systems, 31.

Example



- **pre-conditions**

$$1 \leq x_1 \leq 2$$

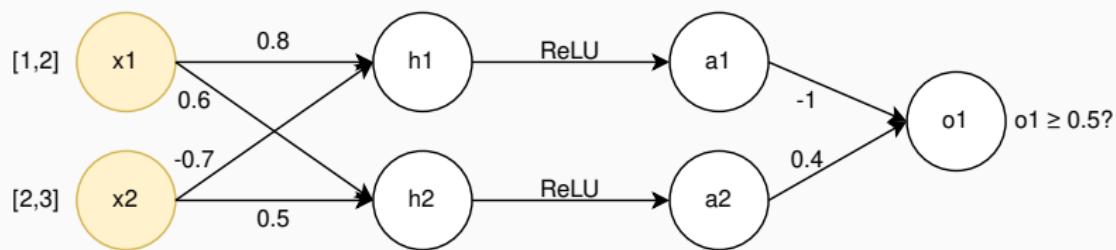
$$2 \leq x_2 \leq 3$$

- **post-conditions**

$$o_1 \geq 0.5$$

- Fully Connected & ReLU activation function

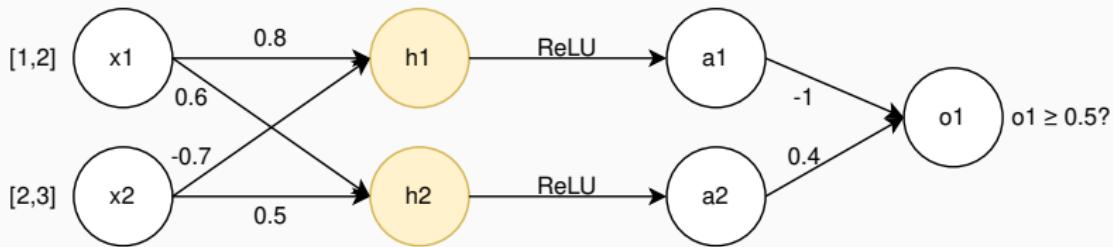
Example - Zonotope Domain - Input Layer



$$x_1 = 1.5 + 0.5y_1$$

$$x_2 = 2.5 + 0.5y_2$$

Example - Zonotope Domain - Hidden Layer



From Input Layer:

$$x_1 = 1.5 + 0.5y_1, x_2 = 2.5 + 0.5y_2$$

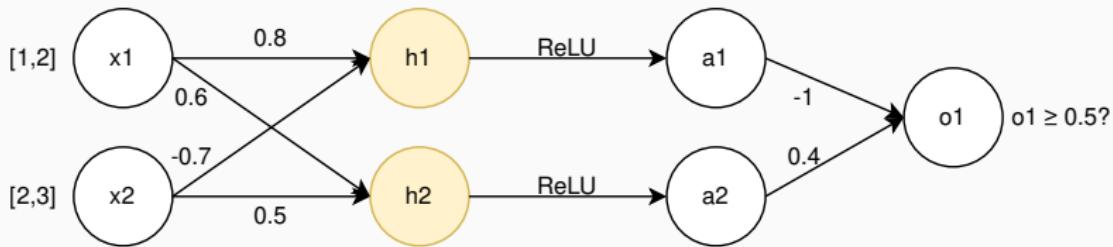
Abstract Affine Function:

$$\begin{aligned}h_1 &= 0.8 \times x_1 + (-0.7) \times x_2 \\&= 0.8 \times (1.5 + 0.5y_1) + (-0.7) \times (2.5 + 0.5y_2) \\&= -0.55 + 0.4y_1 - 0.35y_2\end{aligned}$$

Concretization:

$$[-1.3, 0.2]$$

Example - Zonotope Domain - Hidden Layer



From Input Layer:

$$x_1 = 1.5 + 0.5y_1, x_2 = 2.5 + 0.5y_2$$

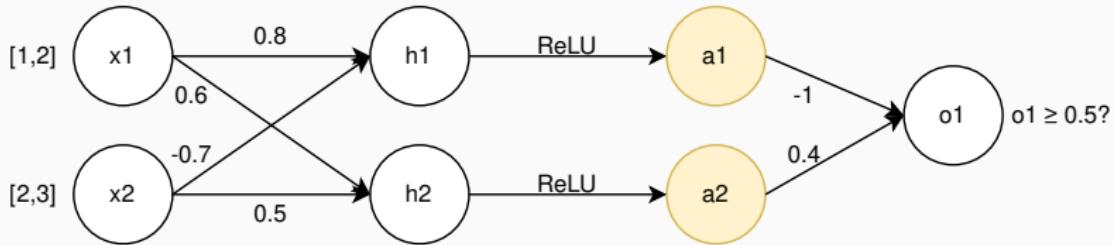
Abstract Affine Function:

$$\begin{aligned}h_2 &= 0.6 \times x_1 + 0.5 \times x_2 \\&= 0.6 \times (1.5 + 0.5y_1) + 0.5 \times (2.5 + 0.5y_2) \\&= 2.15 + 0.3y_1 + 0.25y_2\end{aligned}$$

Concretization:

$$[1.6, 2.7]$$

Example - Zonotope Domain - Activation Layer



From Hidden Layer:

$$h_1 = -0.55 + 0.4y_1 - 0.35y_2, [-1.3, 0.2]$$

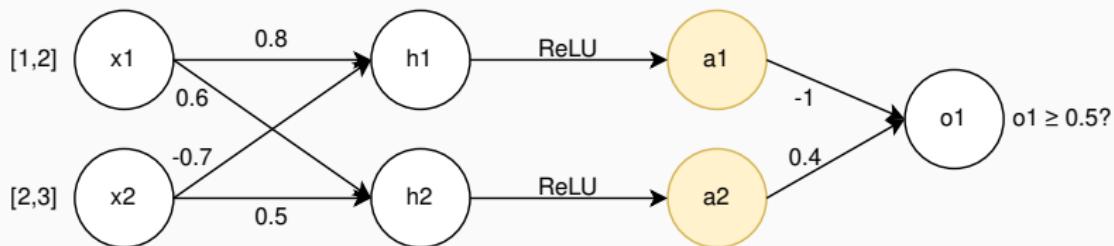
Abstract ReLU Activation Function:

$$\begin{aligned}a_1 &= 0.13 \times (-0.55 + 0.4y_1 - 0.35y_2) + 0.087 + 0.087y_3 \\&= -0.0715 + 0.052y_1 - 0.0455y_2 + 0.087 + 0.087y_3 \\&= 0.0155 + 0.052y_1 - 0.0455y_2 + 0.087y_3\end{aligned}$$

Concretization:

$$[-0.169, 0.1995]$$

Example - Zonotope Domain - Activation Layer



From Hidden Layer:

$$h_2 = 2.15 + 0.3y_1 + 0.25y_2, [1.6, 2.7]$$

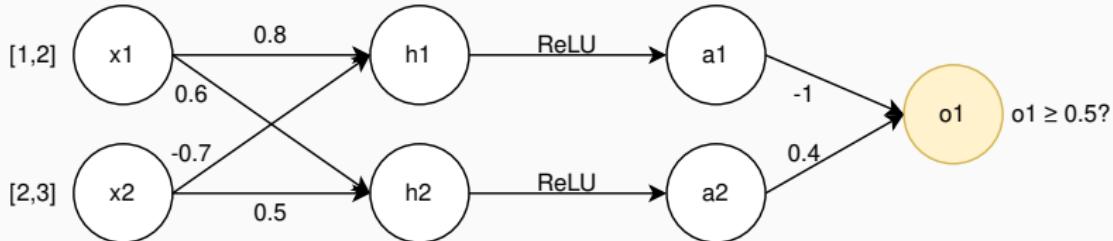
Abstract ReLU Activation Function:

$$a_2 = h_2 = 2.15 + 0.3y_1 + 0.25y_2$$

Concretization:

$$[1.6, 2.7]$$

Example - Zonotope Domain - Output Layer



From Activation Layer:

$$a_1 = 0.0155 + 0.052y_1 - 0.0455y_2 + 0.087y_3$$

$$a_2 = 2.15 + 0.3y_1 + 0.25y_2$$

Abstract Affine Function:

$$\begin{aligned} o_1 &= -1 \times a_1 + 0.4 \times a_2 \\ &= -1 \times (0.0155 + 0.52y_1 - 0.0455y_2 + 0.087y_3) \\ &\quad + 0.4 \times (2.15 + 0.3y_1 + 0.25y_2) \\ &= 0.8445 + 0.068y_1 + 0.1455y_2 - 0.087y_3 \end{aligned}$$

Concretization:

$[0.544, 1.145]$, PROVE!

Summary of Zonotope Domain

- Extend from interval domain.
- More precise than interval domain.
- Implicitly describe *dependency* between different neurons.

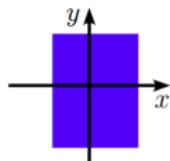
Other Abstract Domains

Course organisation

Course plan (2/8)

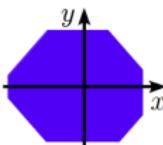
Bricks of abstraction: numerical domains

simple domains



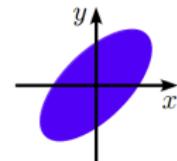
Intervals
 $x \in [a, b]$

relational domains

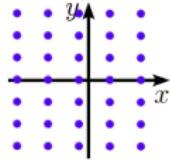


Octagons
 $\pm x \pm y \leq c$

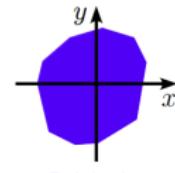
specific domains



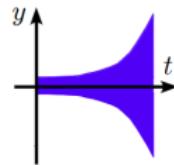
Ellipsoids
digital filters



Congruences
 $x \in a\mathbb{Z} + b$



Polyhedra
 $\sum_i \alpha_i x_i \leq \beta$



Exponentials
rounding errors

Other Neural Network Architectures

- **Layer types**
 1. Convolutional
 2. Residual
 3. ...
- **Activation functions**
 1. Sigmoid
 2. Tahn
 3. ...
- **Dataset**
 1. MNIST
 2. CIFAR-10
 3. CIFAR-100
 4. ImageNet
 5. ...

Verification of Neural Network Competition (VNN-COMP)

The 5th International Verification of Neural Network Competition
(VNN-COMP'24)

The 7th International Symposium on AI Verification (SAIV'24)

The 36th International Conference on Computer Aided Verification (CAV'24)

The standardized comparison:

- .onnx: For storing neural network structure.
- .vnnlib: For defining the pre/post-conditions.

Performance in VNN-COMP

Table 3: Overview of all scored benchmarks.

| Category | Benchmark | Application | Network Types | # Params | Effective Input Dim |
|--------------|---------------------------|---|--|-------------|---------------------|
| Complex | cGAN | Image Generation & Image Prediction | Complex (Conv. + Vision Transformer) | 500k - 68M | 5 |
| | NN4Sys | Dataset Indexing & Cardinality Prediction | Complex (ReLU + Sigmoid) | 33k - 37M | 1-308 |
| | ml4acopf | Power System | Complex (ReLU + Trigonometric + Sigmoid) | 4k-680k | 22 - 402 |
| CNN & ResNet | ViT | Vision | Conv. + Residual + Softmax + BatchNorm | 68k - 76k | 3072 |
| | Collins RUL CNN | Condition Based Maintenance | Conv. + ReLU, Dropout | 60k - 262k | 400 - 800 |
| | VGGNet16 | Image Classification | Conv. + ReLU + MaxPool | 138M | 150k |
| FC | Traffic Signs Recognition | Image Classification | Conv. + Sign + MakPool + BatchNorm | 905k - 1.7M | 2.7k - 12k |
| | TLL Verify Bench | Two-Level Lattice NN | Two-Level Lattice NN (FC. + ReLU) | 17k - 67M | 2 |
| | Acas XU | Collision Detection | FC. + ReLU | 13k | 5 |
| | Dist Shift | Distribution Shift Detection | FC. + ReLU + Sigmoid | 342k - 855k | 792 |

Conclusion

Conclusion

- Introduced neural network verification.
- Demonstrated how to use abstract interpretation to verify neural network.

Welcome to join this research field!

Thanks for your attention!

Q & A

Online Resources

- A Tutorial Website
- Stanford Seminar - Recent progress in verifying neural networks,
Zico Kolter
- AAAI 2022 Tutorial: "Formal Verification of Deep Neural Networks:
Theory and Practice"

References

- [1] Aws Albarghouthi. **Introduction to Neural Network Verification.** 2021.
- [2] Christopher Brix et al. **The Fourth International Verification of Neural Networks Competition (VNN-COMP 2023): Summary and Results.** 2023.
- [3] Timon Gehr et al. **“AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation”.** In: *2018 IEEE Symposium on Security and Privacy (SP)* Not available (May 2018), Not available. ISSN: Not available. DOI: 10.1109/sp.2018.00058. URL: <https://dx.doi.org/10.1109/sp.2018.00058>.

- [4] Khalil Ghorbal, Eric Goubault, and Sylvie Putot. “**The Zonotope Abstract Domain Taylor1+**”. In: *Computer Aided Verification, Lecture Notes in Computer Science* Not available (undefined 2009), pp. 627–633. ISSN: 0302-9743, 1611-3349. DOI: 10.1007/978-3-642-02658-4_47. URL: https://dx.doi.org/10.1007/978-3-642-02658-4_47.
- [5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. **Explaining and Harnessing Adversarial Examples**. 2015.
- [6] Guy Katz et al. “**Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks**”. In: *Computer Aided Verification, Lecture Notes in Computer Science* Not available (undefined 2017), pp. 97–117. ISSN: 0302-9743, 1611-3349. DOI: 10.1007/978-3-319-63387-9_5. URL: https://dx.doi.org/10.1007/978-3-319-63387-9_5.

- [7] Changliu Liu et al. “**Algorithms for Verifying Deep Neural Networks**”. In: *CoRR* abs/1903.06758 (2019). URL: <http://arxiv.org/abs/1903.06758>.
- [8] Xavier Rival and Kwangkeun Yi. **Introduction to Static Analysis**. en. MIT Press, 2020, p. 315. ISBN: 9780262043410.
- [9] Gagandeep Singh et al. “**An abstract domain for certifying neural networks**”. In: *Proceedings of the ACM on Programming Languages* 3 (Jan. 2019), pp. 1–30. ISSN: 2475-1421. DOI: 10.1145/3290354. URL: <https://dx.doi.org/10.1145/3290354>.
- [10] Gagandeep Singh et al. “**Boosting Robustness Certification of Neural Networks**”. In: *International Conference on Learning Representations*. 2018. URL: <https://api.semanticscholar.org/CorpusID:196059499>.

- [11] Gagandeep Singh et al. “**Fast and Effective Robustness Certification**”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf.
- [12] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. “**Evaluating Robustness of Neural Networks with Mixed Integer Programming**”. In: *International Conference on Learning Representations*. 2017. URL: <https://api.semanticscholar.org/CorpusID:47016770>.
- [13] Caterina Urban and Antoine Miné. “**A Review of Formal Methods applied to Machine Learning**”. In: *CoRR* abs/2104.02466 (2021). URL: <https://arxiv.org/abs/2104.02466>.
- [14] Shiqi Wang et al. “**Efficient formal safety analysis of neural networks**”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 6367–6377.

- [15] Huan Zhang et al. “**Efficient Neural Network Robustness Certification with General Activation Functions**”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL:
https://proceedings.neurips.cc/paper_files/paper/2018/file/d04863f100d59b3eb688a11f95b0ae60-Paper.pdf.
- [16] Huan Zhang et al. **General Cutting Planes for Bound-Propagation-Based Neural Network Verification**. 2022.