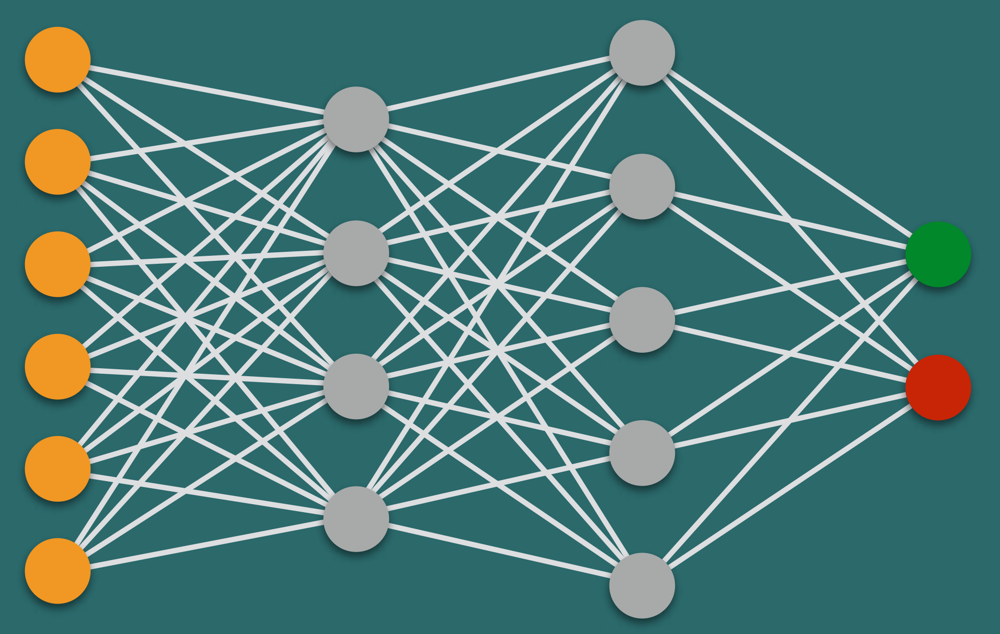
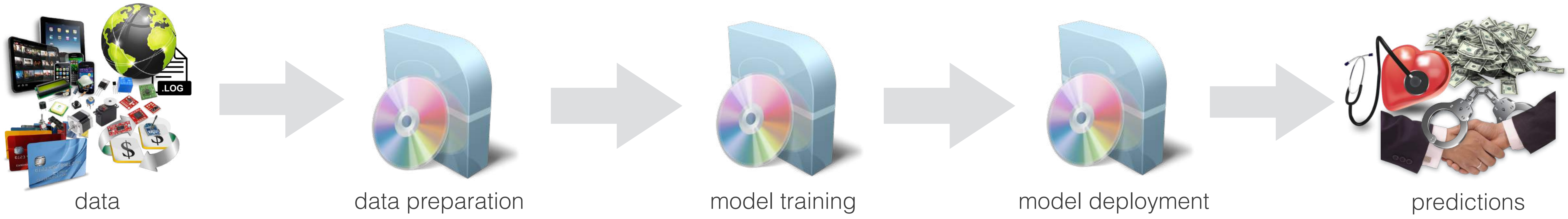
# Formal Methods for Robust Artificial Intelligence
## State of the Art

**Caterina Urban**
ANTIQUE Research Team, Inria & École Normale Supérieure | Université PSL



# Artificial Intelligence Development Process

## Artificial Intelligence Pipeline

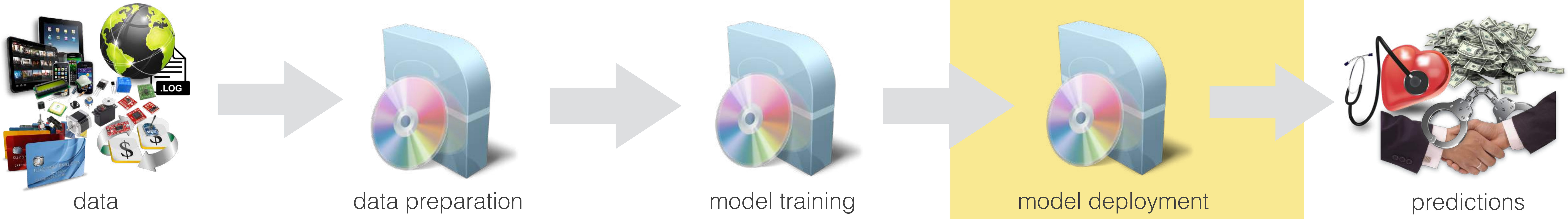


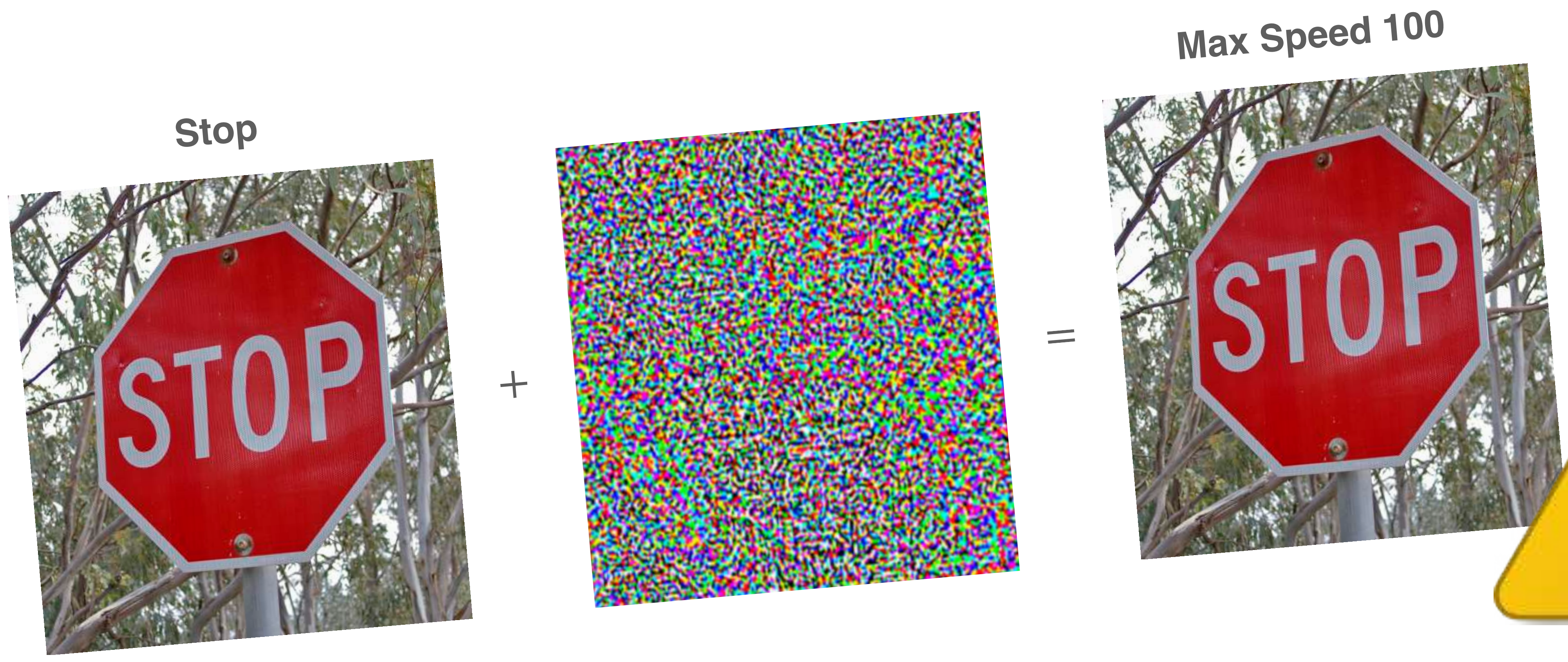
# Model Training is Highly Non-Deterministic



no **predictability** and **traceability**

# Models Only Give Probabilistic Guarantees




not sufficient for guaranteeing
an **acceptable failure rate**
**under any circumstances**

# Safety-Critical Artificial Intelligence

[1] STAT+ [2]

**IBM's Watson supercomputer recommended 'unsafe and incorrect' cancer treatments, internal documents show**

*By* Casey Ross[3] @caseymross[4] *and* Ike Swetlitz

July 25, 2018

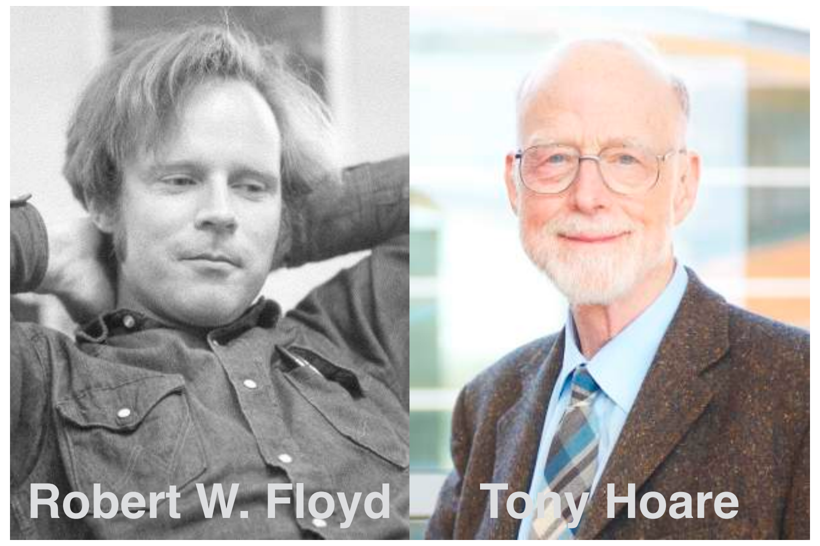
A self-driving Uber ran a red light last December, contrary to company claims

**Feds Say Self-Driving Uber SUV Did Not Recognize Jaywalking Pedestrian In Fatal Crash**

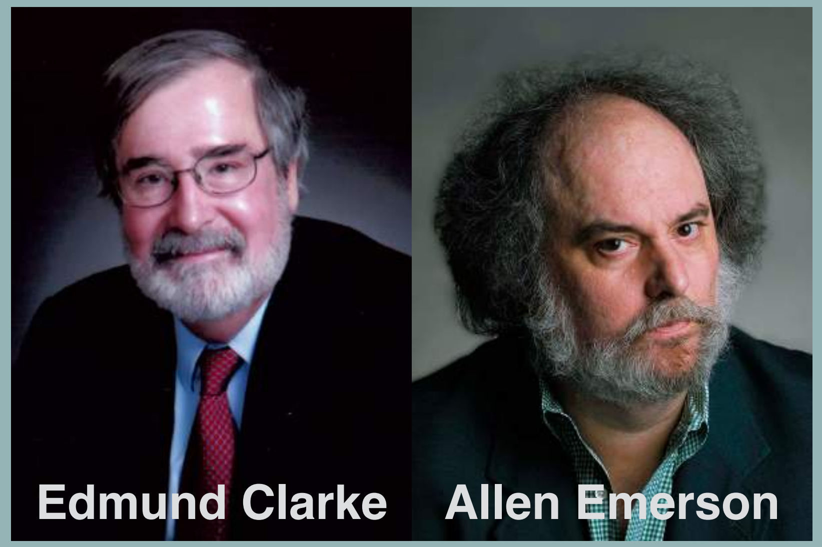
Richard Gonzales November 7, 201910:57 PM ET

# Formal Methods

## Mathematical Guarantees of Safety



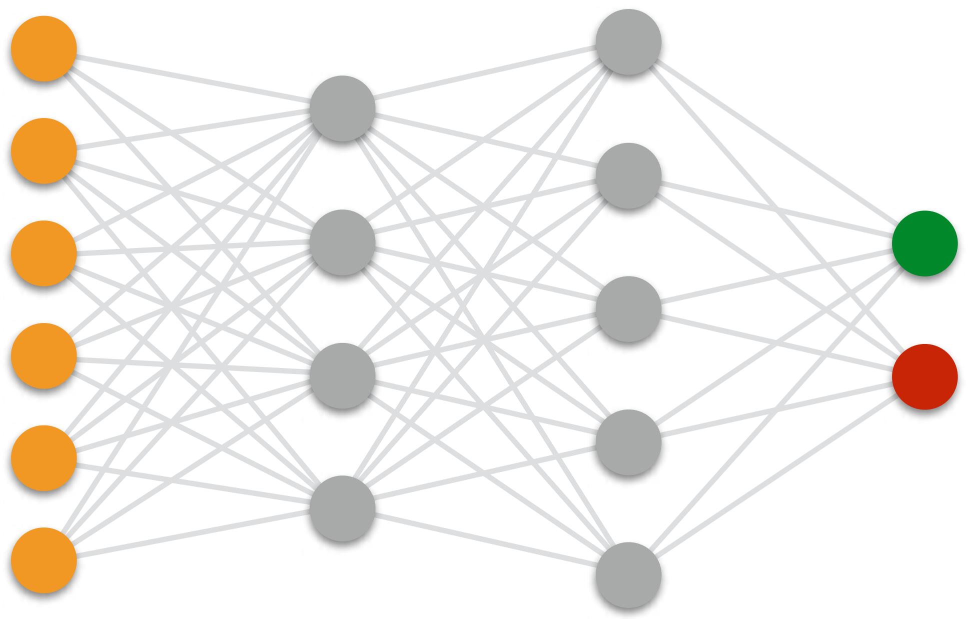

### Deductive Verification
- extremely **expressive**
- **relies on the user** to guide the proof




### Model Checking
- analysis of a **model** of the software
- **sound and complete with respect to the model**

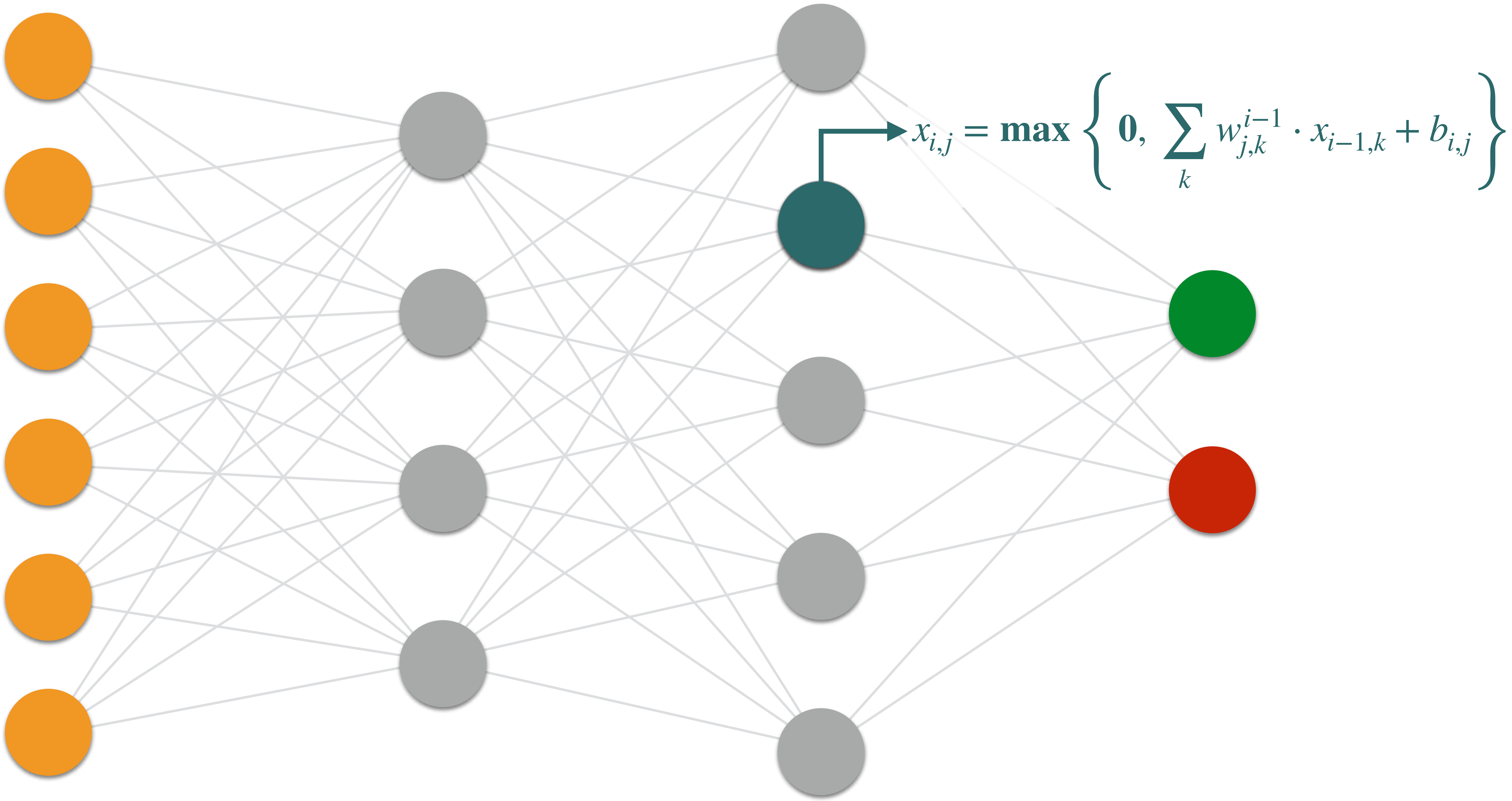

### Static Analysis
- analysis of the software at some level of **abstraction**
- fully **automatic** and **sound** by construction
- generally **not complete**

# Neural Network Models

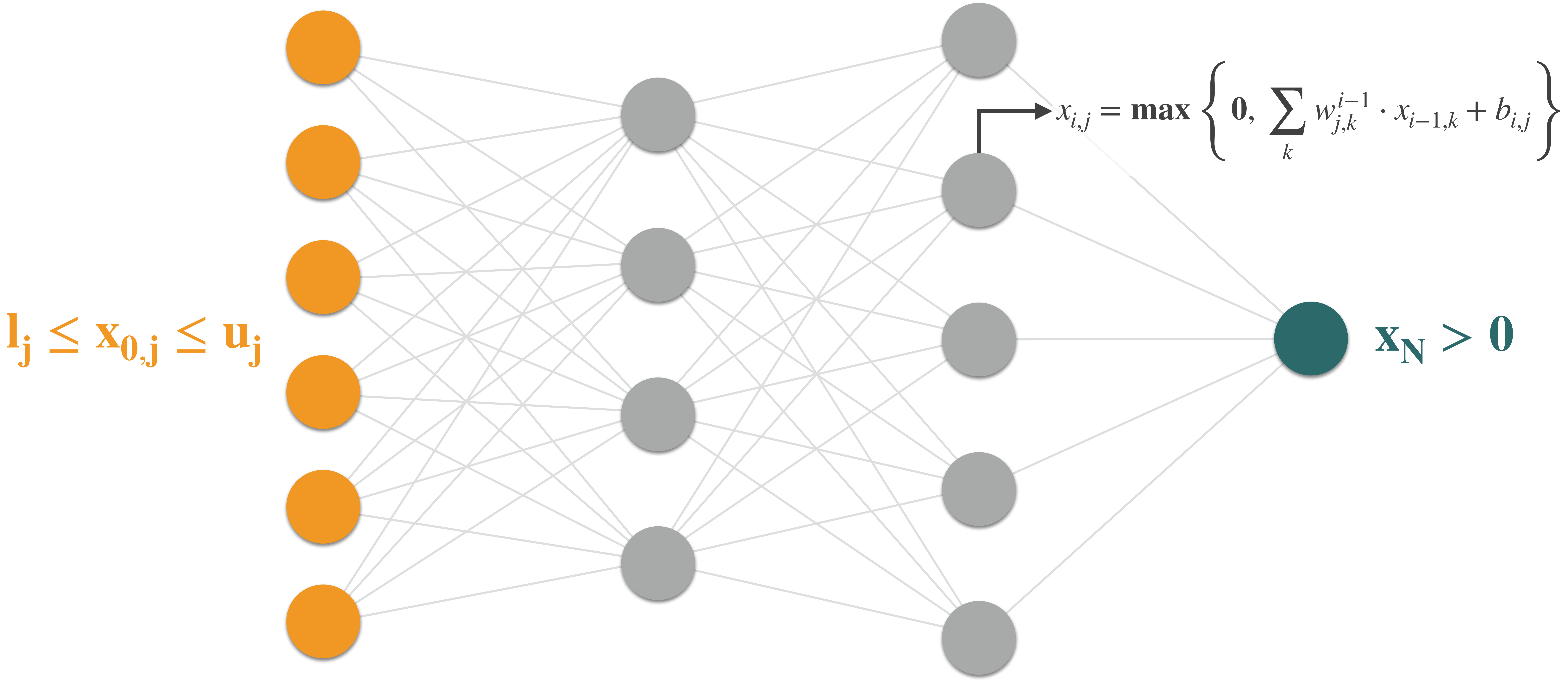


# Feed-Forward Neural Networks

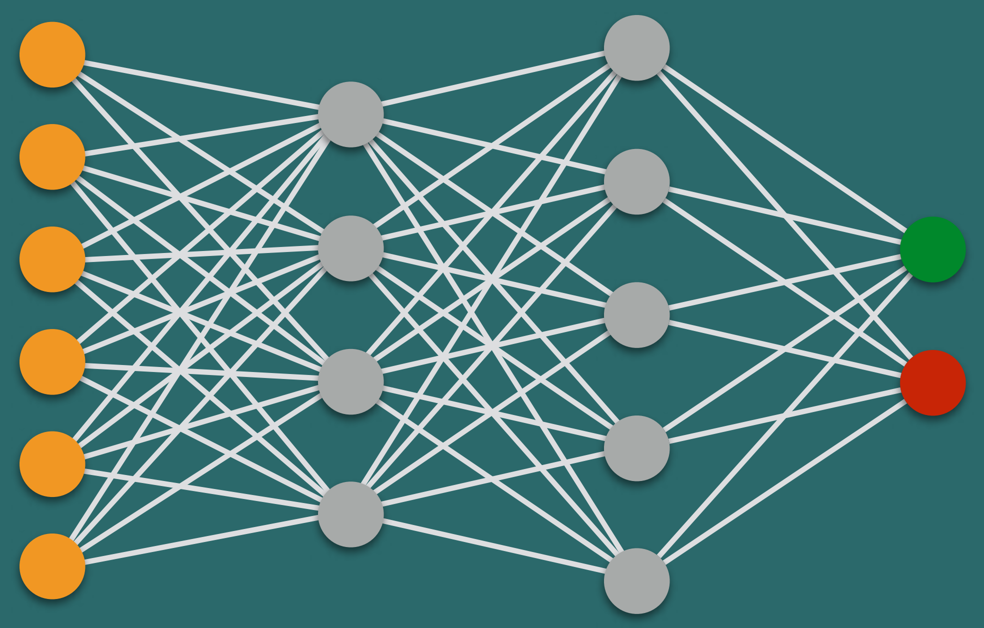
## Fully-Connected Layers with ReLU Activation Functions



# Safety Verification



# Model Checking Methods



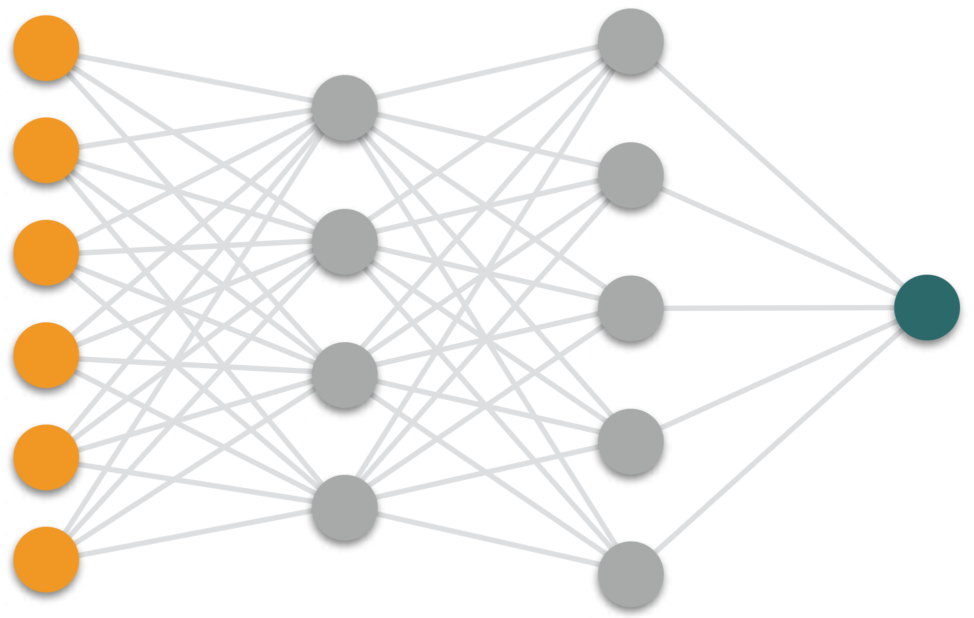
# SMT-Based Methods

## Safety Verification Reduced to Constraint Satisfiability

$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$ $\quad j \in \{0, \ldots, |\mathbf{X}_0|\}$ **input specification**

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$x_{i,j} = \max\{0, \hat{x}_{i,j}\} \qquad i \in \{1, \ldots, n-1\}, j \in \{0, \ldots, |\mathbf{X}_i|\}$$



$\mathbf{x_N} \leq \mathbf{0}$

(negation of)
**output specification**

**satisfiable** → ✗ **counterexample**
otherwise → ✓ **safe**

# Planet

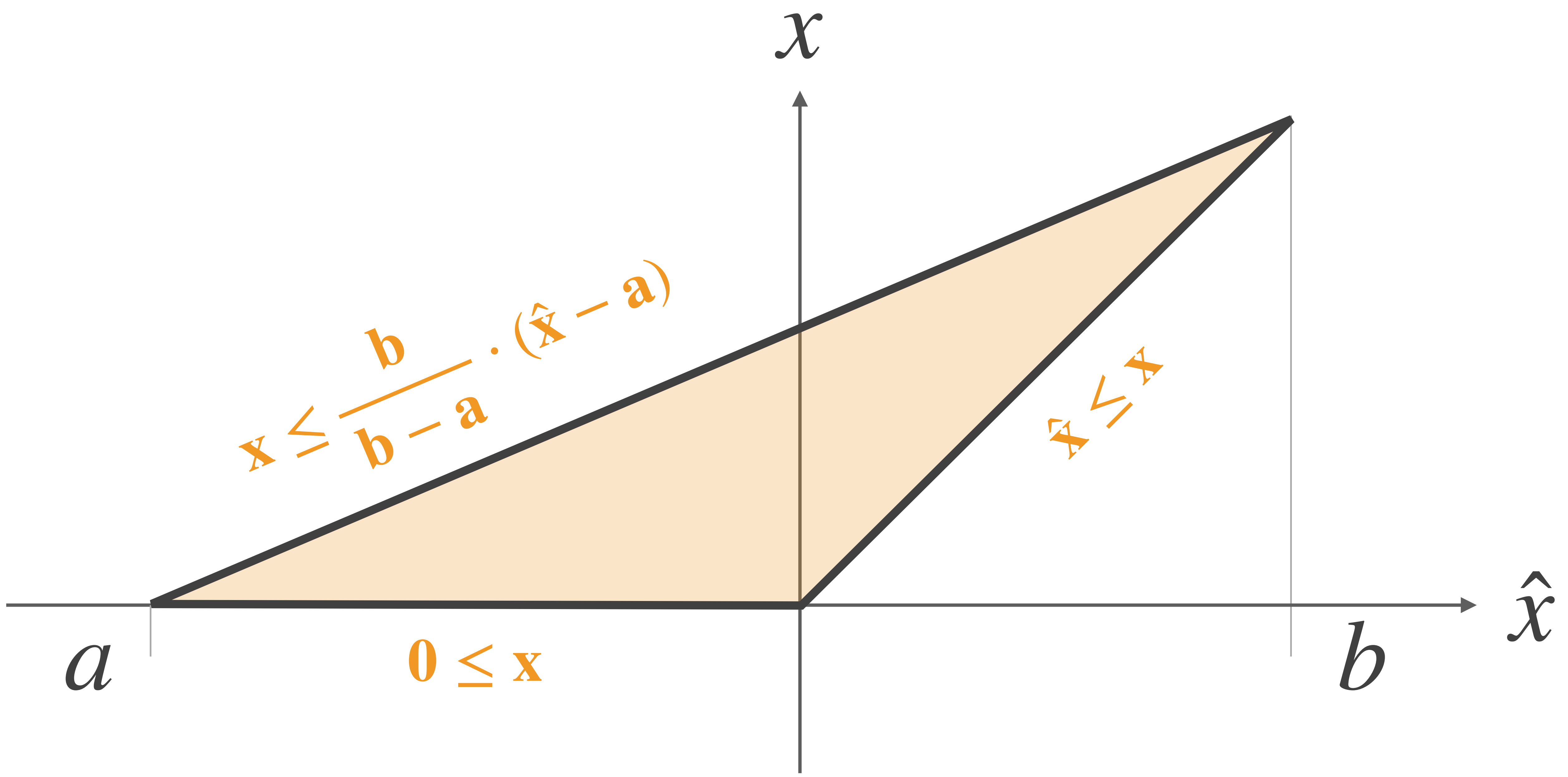
use **approximations** to reduce the solution search space

$$x_{i,j} = \max\{0, \hat{x}_{i,j}\}$$

$$0 \leq x_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j}$$

$$x_{i,j} \leq \frac{b_{i,j}}{b_{i,j} - a_{i,j}} \cdot (\hat{x}_{i,j} - a_{i,j})$$




R. Ehlers - Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks (ATVA 2017)

# Reluplex

based on the **simplex algorithm** extended to support ReLU constraints

| Variable | Value |
|---|---|
| $\mathbf{x}_{00}$ | $v_{00}$ |
| $\dots$ | $\dots$ |
| $\hat{\mathbf{x}}_{\mathbf{ij}}$ | $\hat{v}_{ij}$ |
| $\mathbf{x}_{\mathbf{ij}}$ | $v_{ij}$ |
| $\dots$ | $\dots$ |
| $\mathbf{x}_{\mathbf{N}}$ | $v_N$ |

| Variable | Value |
|---|---|
| $\mathbf{x}_{00}$ | $v_{00}$ |
| $\dots$ | $\dots$ |
| $\hat{\mathbf{x}}_{\mathbf{ij}}$ | $\hat{v}'_{ij}$ |
| $\mathbf{x}_{\mathbf{ij}}$ | $v_{ij}$ |
| $\dots$ | $\dots$ |
| $\mathbf{x}_{\mathbf{N}}$ | $v_N$ |

| Variable | Value |
|---|---|
| $\mathbf{x}_{00}$ | $v_{00}$ |
| $\dots$ | $\dots$ |
| $\hat{\mathbf{x}}_{\mathbf{ij}}$ | $\hat{v}'_{ij}$ |
| $\mathbf{x}_{\mathbf{ij}}$ | $\hat{v}'_{ij}$ |
| $\dots$ | $\dots$ |
| $\mathbf{x}_{\mathbf{N}}$ | $v_N$ |

$\dots$

| Variable | Value |
|---|---|
| $\mathbf{x}_{00}$ | $v_{00}$ |
| $\dots$ | $\dots$ |
| $\hat{\mathbf{x}}_{\mathbf{ij}}$ | $\hat{v}'_{ij}$ |
| $\mathbf{x}_{\mathbf{ij}}$ | $0$ |
| $\dots$ | $\dots$ |
| $\mathbf{x}_{\mathbf{N}}$ | $v_N$ |

$\dots$

G. Katz et al. - Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks (CAV 2017)

# Reluplex

based on the sim

extended to supp

**Follow-up Work**

G. Katz et al. - The Marabou Framework for Verification and Analysis of Deep Neural Networks (CAV 2019)

| Variable | Value |
|---|---|
| $x_{00}$ | $v_{00}$ |
| … | … |
| $\hat{x}_{ij}$ | $\hat{v}_{ij}$ |
| $x_{ij}$ | $v_{ij}$ |
| … | … |
| $x_N$ | $v_N$ |

| Variable | Value |
|---|---|
| $x_{00}$ | $v_{00}$ |
| … | … |
| $\hat{x}_{ij}$ | $\hat{v}'_{ij}$ |
| $x_{ij}$ | $v_{ij}$ |
| … | … |
| $x_N$ | $v_N$ |

| Variable | Value |
|---|---|
| $x_{00}$ | $v_{00}$ |
| … | … |
| $\hat{x}_{ij}$ | $\hat{v}'_{ij}$ |
| $x_{ij}$ | $\hat{v}'_{ij}$ |
| … | … |
| $x_N$ | $v_N$ |

| Variable | Value |
|---|---|
| $x_{00}$ | $v_{00}$ |
| … | … |
| $\hat{x}_{ij}$ | $\hat{v}'_{ij}$ |
| $x_{ij}$ | 0 |
| … | … |
| $x_N$ | $v_N$ |

…

G. Katz et al. - Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks (CAV 2017)

# Other SMT-Based Methods

- L. Pulina and A. Tacchella - An Abstraction-Refinement Approach to Verification of Artificial Neural Networks (CAV 2010)
  **the first formal verification method for neural networks**

- O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi - Measuring Neural Net Robustness with Constraints (NeurIPS 2016)
  **an approach for finding the nearest adversarial example according to the $L_\infty$ distance**

- X. Huang, M. Kwiatkowska, S. Wang, and M. Wu - Safety Verification of Deep Neural Networks (CAV 2017)
  **an approach for proving local robustness to adversarial perturbations**

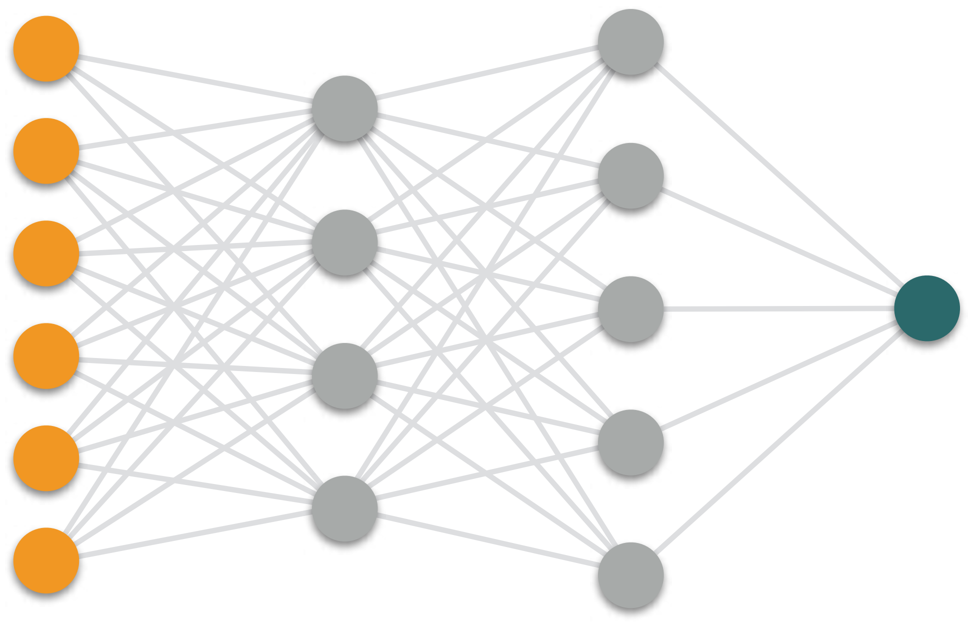
- N. Narodytska, S. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh - Verifying Properties of Binarized Deep Neural Networks (AAAI 2018)
  C. H. Cheng, G. Nührenberg, C. H. Huang, and H. Ruess - Verification of Binarized Neural Networks via Inter-Neuron Factoring (VSTTE 2018)
  **approaches focusing on binarized neural networks**

# MILP-Based Methods

## Safety Verification Reduced to Mixed Integer Linear Program

$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$ $\quad j \in \{0, \ldots, |\mathbf{X}_0|\}$ **input specification**

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$



$$x_{i,j} = \delta_{\mathbf{i,j}} \cdot \hat{x}_{i,j}$$

$$\delta_{\mathbf{i,j}} = 1 \Rightarrow \hat{x}_{i,j} \geq 0$$

$$\delta_{\mathbf{i,j}} = 0 \Rightarrow \hat{x}_{i,j} < 0$$

$\delta_{\mathbf{i,j}} \in \{\mathbf{0}, \mathbf{1}\}$

$i \in \{1, \ldots, n-1\}$

$j \in \{0, \ldots, |\mathbf{X}_i|\}$

**min** $\mathbf{x_N}$ **objective function**

**min** $\mathbf{x_N} \leq \mathbf{0} \rightarrow$ ✗ **counterexample**

otherwise $\rightarrow$ ✓ **safe**

# MILP-Based Methods

## Bounded MILP Encoding with Symmetric Bounds

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0,\ldots,n-1\}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

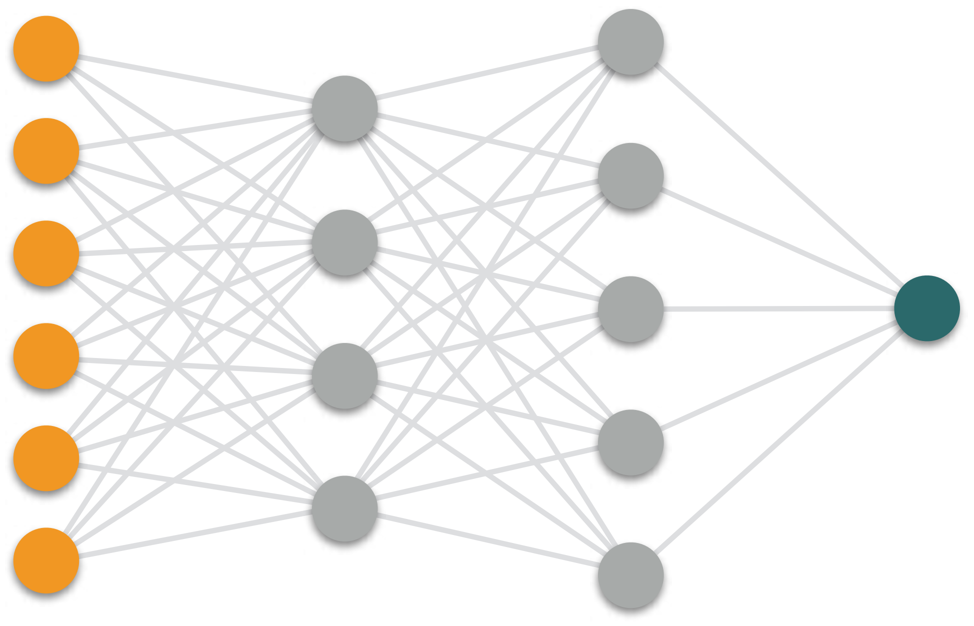
$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \max\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$\delta_{i,j} \in \{0,1\}$

$i \in \{1,\ldots,n-1\}$

$j \in \{0,\ldots,|\mathbf{X}_i|\}$



# Sherlock

## Output Range Analysis

use **local search**
speed up the MILP solver

$\mathbf{l_j \leq x_{0,j} \leq u_j}$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w_{j,k}^{i} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

**min** $\mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# Sherlock

## Output Range Analysis

use **local search**
speed up the MILP solver

$$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{min}\ \mathbf{x_N}$$

**sample** random input $\mathbf{X}$
and evaluate output $\mathbf{L}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# Sherlock

## Output Range Analysis

use **local search**
speed up the MILP solver

$$\mathbf{l_j \leq x_{0,j} \leq u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w_{j,k}^{i} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{x_N < L}$$

**sample** random input $\mathbf{X}$
and evaluate output $\mathbf{L}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# Sherlock

## Output Range Analysis

use **local search**
speed up the MILP solver

$$\mathbf{l_j \leq x_{0,j} \leq u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w_{j,k}^{i} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j} = \max\{-l_i, u_i\}}$$

$$\mathbf{x_N < L}$$

**find** another input $\hat{\mathbf{X}}$
such that $\hat{\mathbf{L}} \leq \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# Sherlock

## Output Range Analysis

use **local search**
speed up the MILP solver

$$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{x_N} < \hat{\mathbf{L}}$$

**find** another input $\hat{\mathbf{X}}$
such that $\hat{\mathbf{L}} \leq \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# Sherlock

## Output Range Analysis

use **local search**
**speed up the MILP solver**

$$\mathbf{l_j} \le \mathbf{x_{0,j}} \le \mathbf{u_j}$$

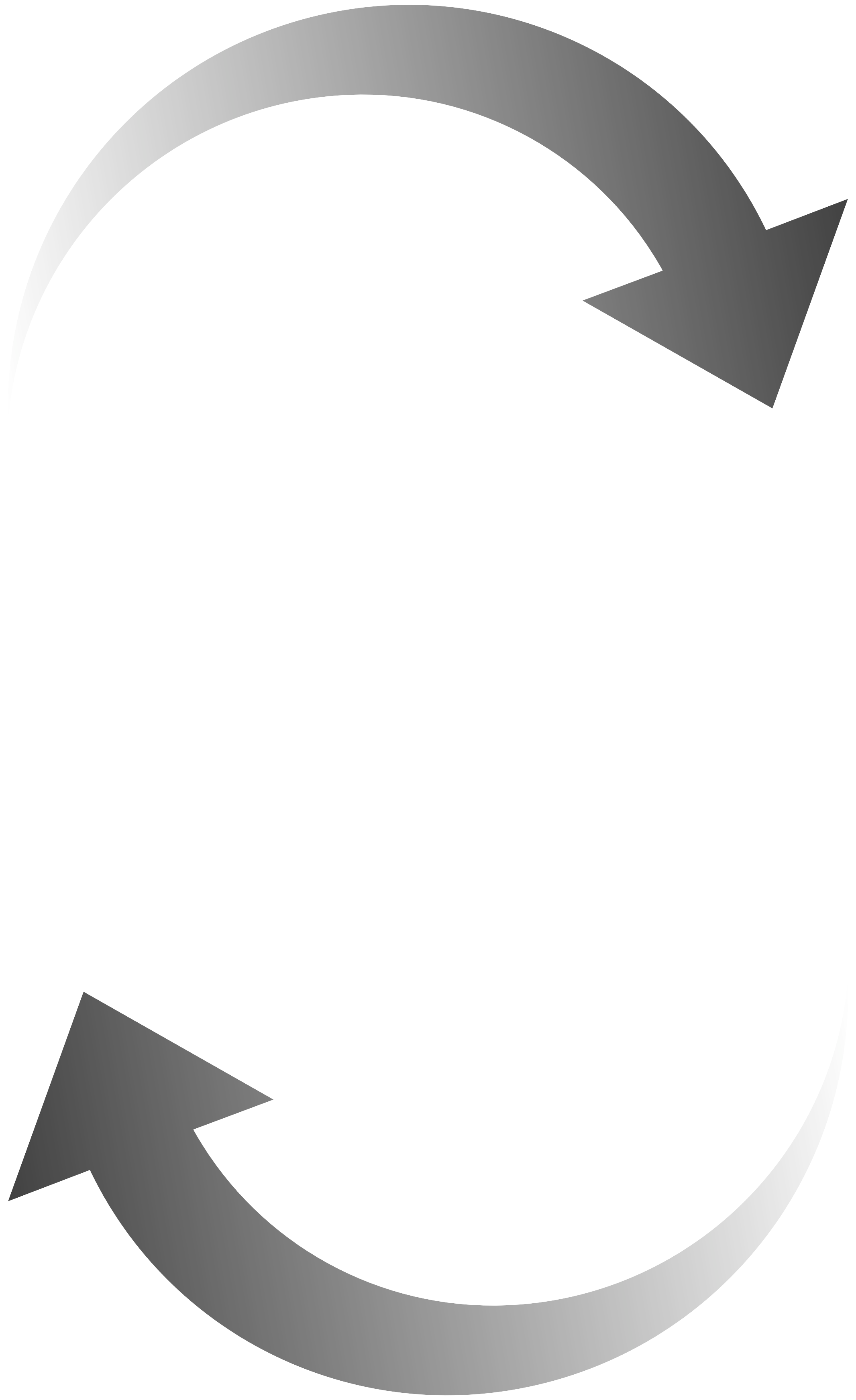
$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \le x_{i,j} \le \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \le x_{i,j} \le \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{x_N} < \hat{\mathbf{L}}$$



**find** another input $\hat{\mathbf{X}}$
such that $\hat{\mathbf{L}} \le \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

# MILP-Based Methods

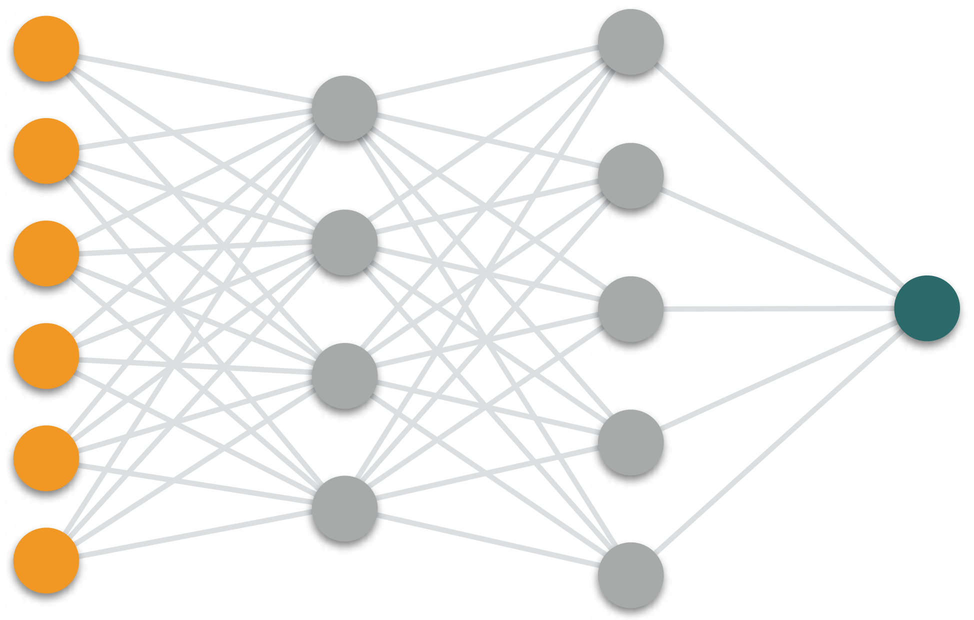
## Bounded MILP Encoding with Asymmetric Bounds

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0,\ldots,n-1\}$$

$$0 \leq x_{i,j} \leq \mathbf{u_{i,j}} \cdot \delta_{i,j} \qquad \delta_{i,j} \in \{0,1\}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{l_{i,j}} \cdot (1 - \delta_{i,j}) \qquad i \in \{1,\ldots,n-1\}$$

$$j \in \{0,\ldots,|\mathbf{X}_i|\}$$



# MIPVerify

## Finding Nearest Adversarial Example

$\min_{\mathbf{X}'} \mathbf{d}(\mathbf{X}, \mathbf{X}')$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w_{j,k}^{i} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{u_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{l_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{x_N} \neq \mathbf{O}$$

---

V. Tjeng, K. Xiao, and R. Tedrake - Evaluating Robustness of Neural Networks with Mixed Integer Programming (ICLR 2019)

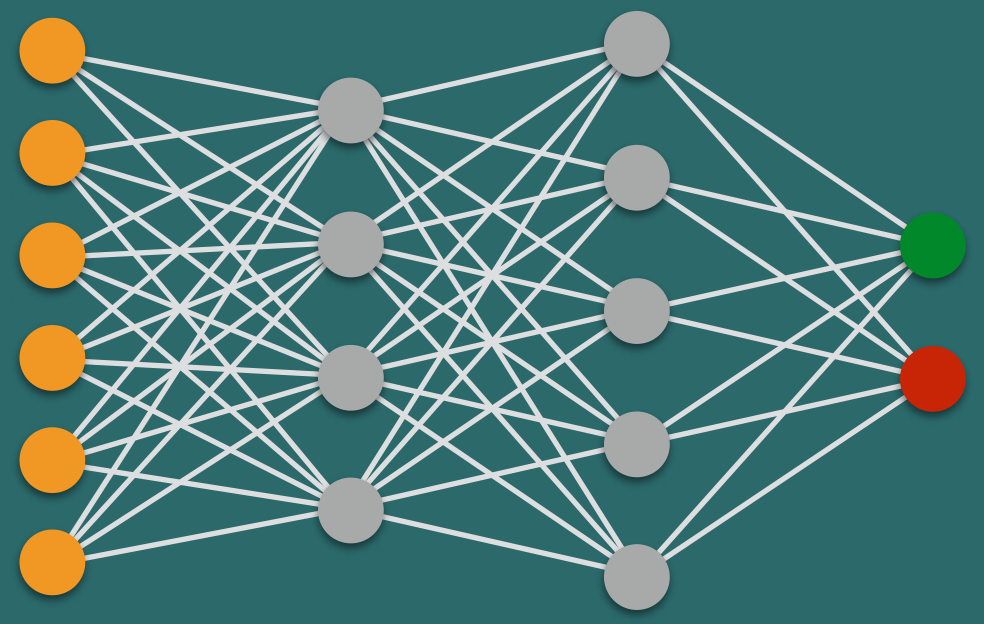
# Other MILP-Based Methods

- R. Bunel, I. Turkaslan, P. H. S. Torr, P. Kohli, and M. P. Kumar - A Unified View of Piecewise Linear Neural Network Verification (NeurIPS 2018)
  **a unifying verification framework for piecewise-linear ReLU neural networks**

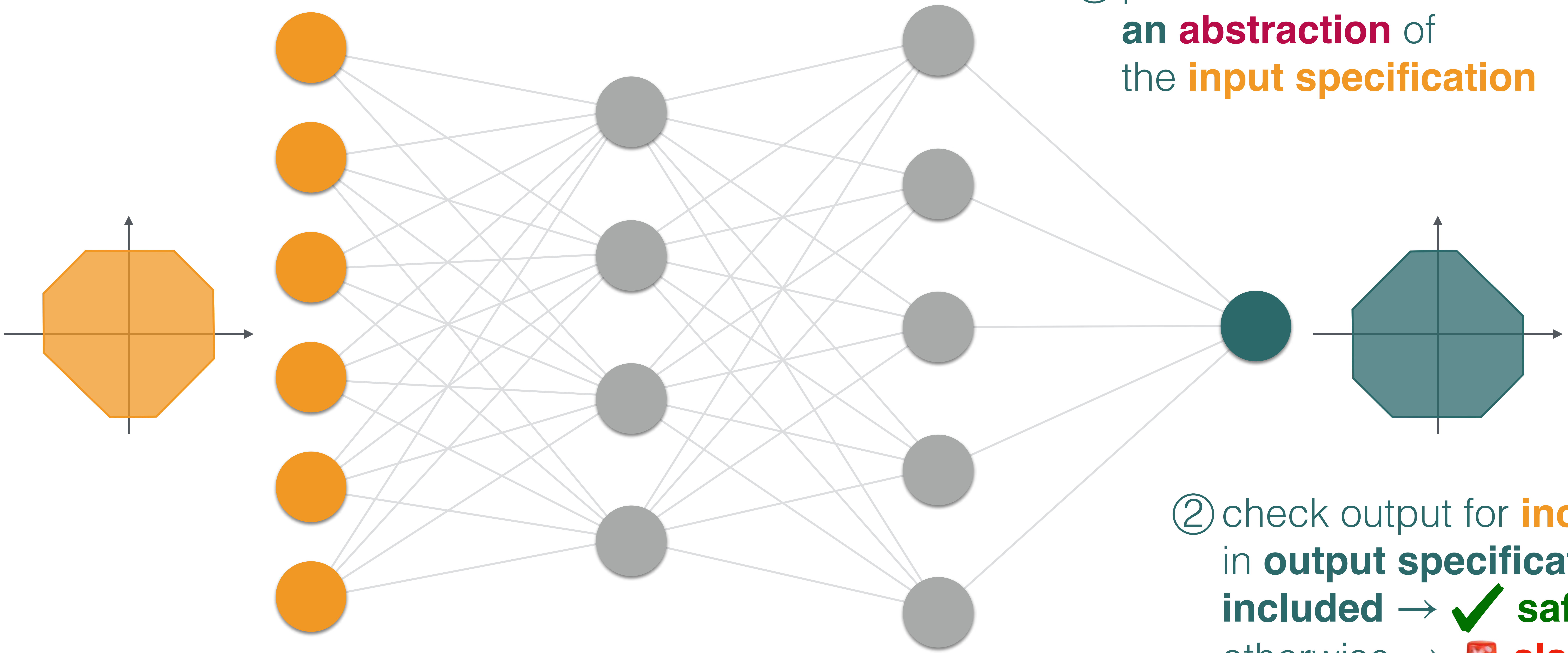
- C.-H. Cheng, G. Nührenberg, and H. Ruess - Maximum Resilience of Artificial Neural Networks (ATVA 2017)
  **an approach for finding a lower bound on robustness to adversarial perturbations**

- M. Fischetti and J. Jo - Deep Neural Networks and Mixed Integer Linear Optimization (2018)
  **an approach for feature visualization and building adversarial examples**

# Static Analysis Methods



# Abstract Interpretation-Based Methods



① proceed **forwards from an abstraction** of the **input specification**

② check output for **inclusion** in **output specification**:
**included** → ✔ **safe**
otherwise → 🚨 **alarm**

# Symbolic Propagation

represent each neuron as a **linear combination** of the **inputs** and the **ReLUs in previous layers**

$$x_{i,j} \mapsto \begin{cases} \sum_{k=0}^{i-1} \mathbf{c}_k \cdot \mathbf{x}_k + \mathbf{c} & \mathbf{c}_k, \mathbf{c} \in \mathscr{R}^{|\mathbf{X}_k|} \\ [a,\ b] & a, b \in \mathscr{R} \end{cases}$$

$$x_{i-1,0} \mapsto \mathbf{E_{i-1,0}}$$
$$\ldots$$
$$x_{i-1,j} \mapsto \mathbf{E_{i-1,j}}$$
$$\ldots$$

$$x_{i,j} = \sum_k w_{j,k}^{i-1} \cdot x_{i-1,k} + b_{i,j}$$

$$x_{i,j} \mapsto \sum_k w_{j,k}^{i-1} \cdot \mathbf{E_{i-1,k}} + b_{i,j}$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a, b]} \end{cases}$$

ReLU

$$x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a, b]} \end{cases} \qquad 0 \le a$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{x_{i,j}} \\ \mathbf{[0, b]} \end{cases} \qquad a < 0 \wedge 0 < b$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{0} \\ \mathbf{[0, 0]} \end{cases} \qquad b \le 0$$

J. Li et al. - Analyzing Deep Neural Networks with Symbolic Propagation (SAS 2019)

# DeepPoly

maintain **symbolic lower- and upper-bounds** for each neuron
+ **convex ReLU approximations**

$$x_{i+1,j} \mapsto \begin{cases} [\sum_k c_{i,k} \cdot x_{i,k} + c,\ \sum_k d_{i,k} \cdot x_{i,k} + d] & c_{i,k}, c, d_{i,k}, d \in \mathcal{R} \\ [a,\ b] & a, b \in \mathcal{R} \end{cases}$$

$$x_{i,j} \mapsto \begin{cases} [\mathbf{L_{i,j}}, \mathbf{U_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU, $0 \leq a$:

$$x_{i,j} \mapsto \begin{cases} [\mathbf{L_{i,j}}, \mathbf{U_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU, $a < 0 \wedge 0 < b$:

$b \leq -a$:

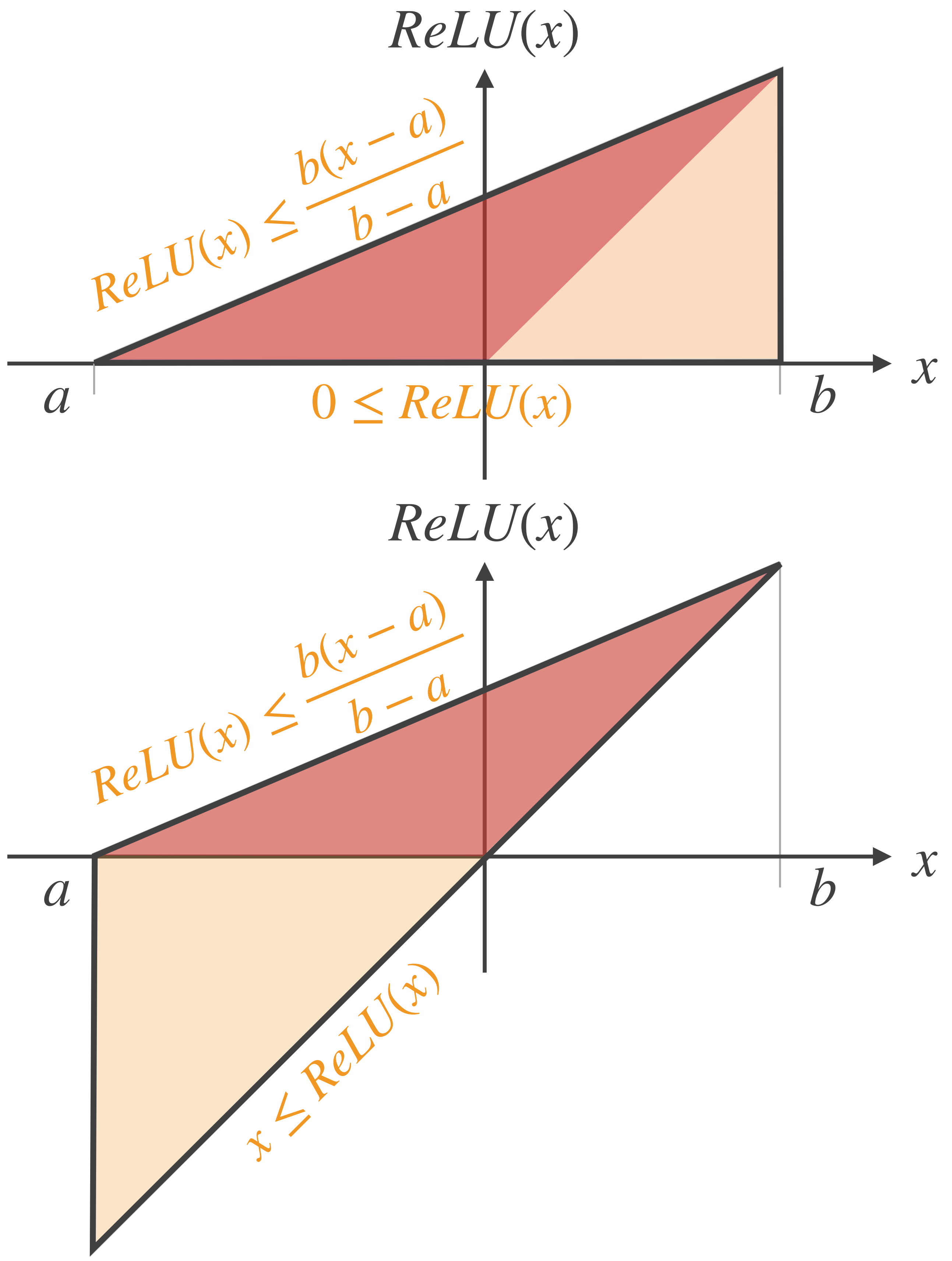
$$x_{i,j} \mapsto \begin{cases} \left[\mathbf{0}, \frac{\mathbf{b(x_{i,j} - a)}}{\mathbf{b - a}}\right] \\ [\mathbf{0}, \mathbf{b}] \end{cases}$$

$-a < b$:

$$x_{i,j} \mapsto \begin{cases} \left[\mathbf{x_{i,j}}, \frac{\mathbf{b(x_{i,j} - a)}}{\mathbf{b - a}}\right] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU, $b \leq 0$:

$$x_{i,j} \mapsto \begin{cases} [\mathbf{0}, \mathbf{0}] \\ [\mathbf{0}, \mathbf{0}] \end{cases}$$




G. Singh, T. Gehr, M. Püschel, and M. Vechev - An Abstract Domain for Certifying Neural Networks (POPL 2019)

# Other Abstract Interpretation Methods

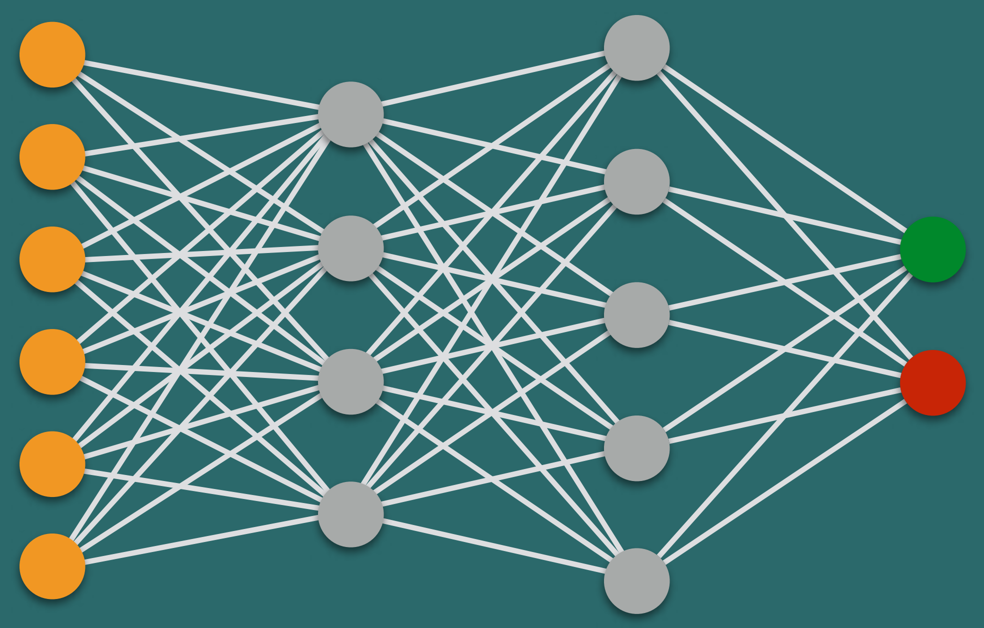
- T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev - AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation (S&P 2018)
  **the first use of abstract interpretation for verifying neural networks**

- G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev - Fast and Effective Robustness Certification (NeurIPS 2018)
  **a custom zonotope domain for certifying neural networks**

- G. Singh, R. Ganvir, M. Püschel, and M. Vechev - Beyond the Single Neuron Convex Barrier for Neural Network Certification (NeurIPS 2019)
  **a framework to jointly approximate k ReLU activations**

- C. Urban, M. Christakis, V, Wüstholz, and F. Zhang - Perfectly Parallel Fairness Certification of Neural Networks (OOPSLA 2020)
  **an approach for verifying fairness of neural network classifiers for tabular data**

# Other Complete Methods



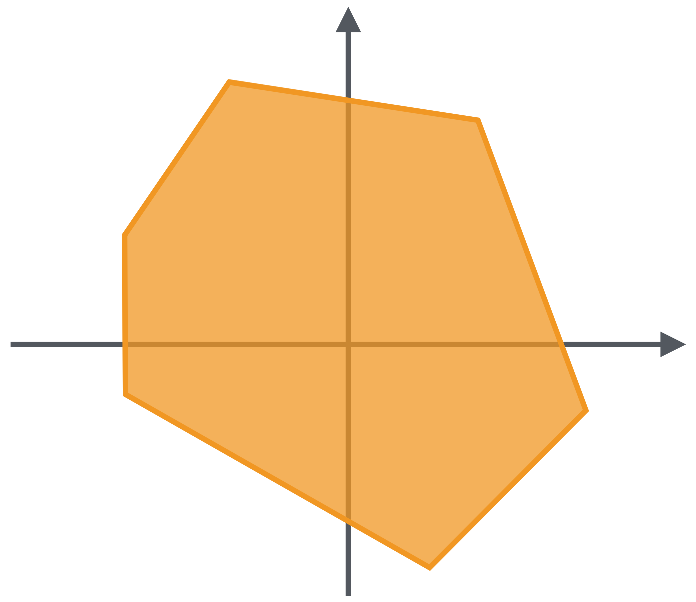
# Star Sets

## Exact Static Analysis Method

use **union** of
**efficient representations**
of bounded convex polyhedra

$\Theta \overset{\text{def}}{=} \langle c, V, P \rangle$

$c \in \mathcal{R}^n$: center
$V = \{v_1, \ldots, v_m\}$: basis vectors in $\mathcal{R}^n$
$P: \mathcal{R}^m \rightarrow \{ \bot, \top \}$: predicate



$$[\![\Theta]\!] = \{x \mid x = c + \sum_{i=1}^{m} \alpha_i v_i \text{ such that } P(\alpha_1, \ldots, \alpha_m) = \top \}$$

- fast and cheap **affine mapping operations** → neural network layers
- inexpensive **intersections with half-spaces** → ReLU activations

---

H.-D. Tran et al. - Star-Based Reachability Analysis of Deep Neural Networks (FM 2018)

# Star Sets

## Exact Static Analysis Method

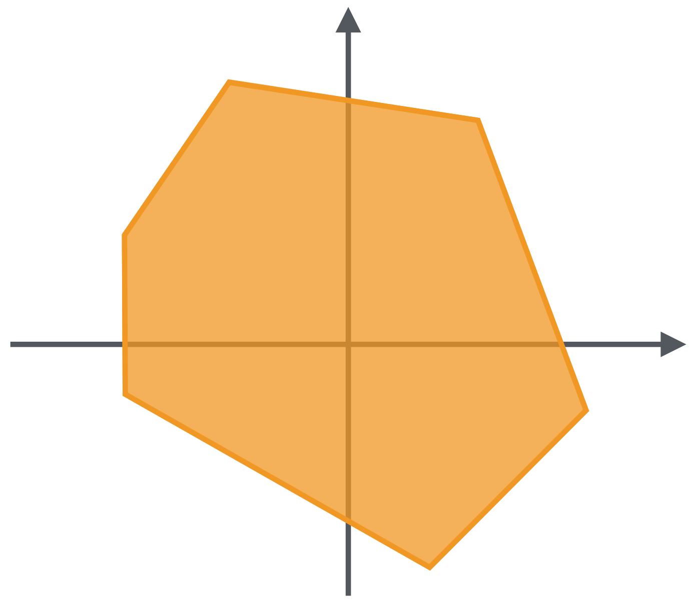
use
**efficient r**
**of bounded**

**Follow-up Work**

H.-D. Tran et al. - Verification of Deep Convolutional Neural Networks Using ImageStars (CAV 2020)

$$\Theta \stackrel{\text{def}}{=} \langle c, V, P \rangle$$

$c \in \mathscr{R}^n$: center
$V = \{v_1, \ldots, v_m\}$: basis vectors in $\mathscr{R}^n$
$P: \mathscr{R}^m \rightarrow \{ \bot, \top \}$: predicate



$$[\![\Theta]\!] = \{x \mid x = c + \sum_{i=1}^{m} \alpha_i v_i \text{ such that } P(\alpha_1, \ldots, \alpha_m) = \top \}$$

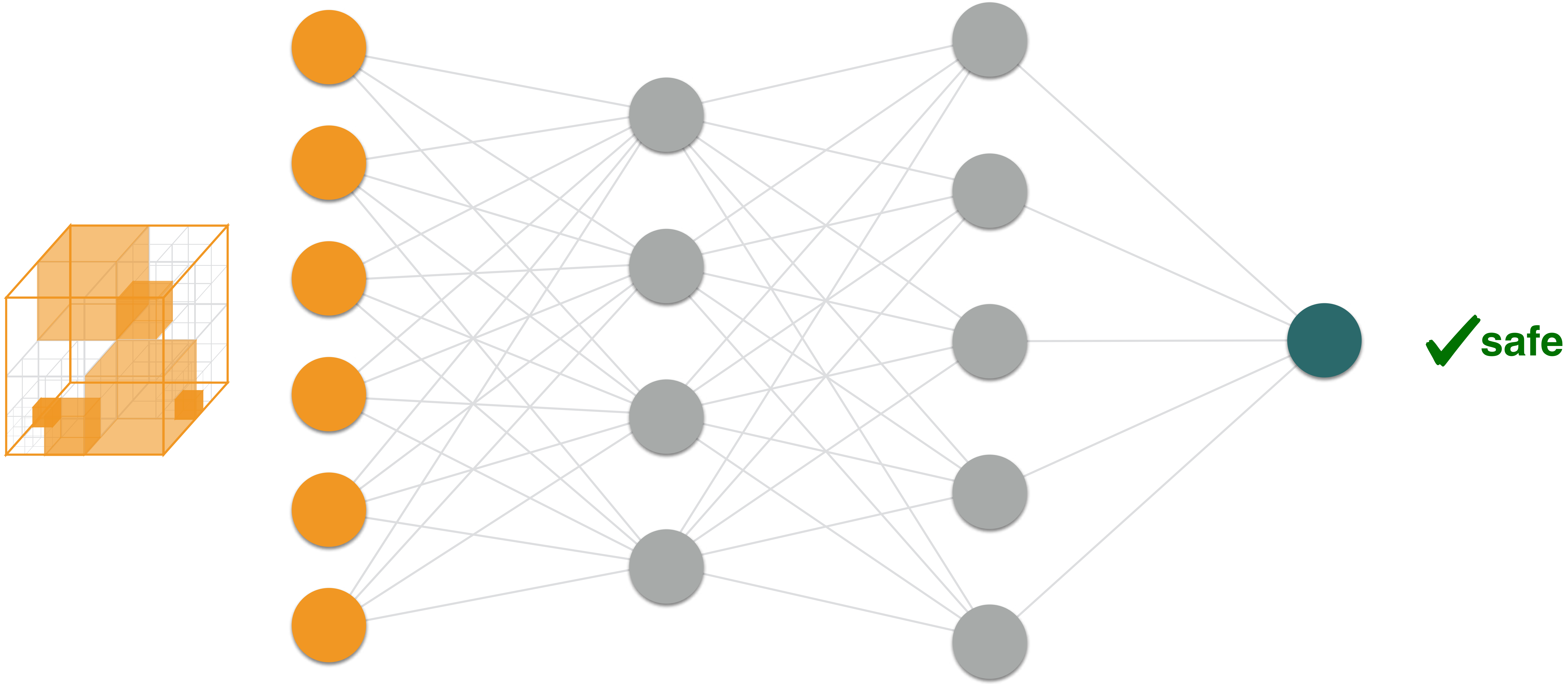
- fast and cheap **affine mapping operations** → neural network layers
- inexpensive **intersections with half-spaces** → ReLU activations

H.-D. Tran et al. - Star-Based Reachability Analysis of Deep Neural Networks (FM 2018)

# ReluVal

## Asymptotically Complete Method

use symbolic propagation
+ **iterative** input **refinement**




S. Wang et al. - Formal Security Analysis of Neural Networks Using Symbolic Intervals (USENIX Security 2018)

# Neurify

## Asymptotically Complete Method

use symbolic propagation
+ **convex ReLU approximation**
+ **iterative** input and ReLU **refinement**

$$x_{i,j} \mapsto \begin{cases} [\sum_k c_{0,k} \cdot x_{0,k} + c, \ \sum_k d_{0,k} \cdot x_{0,k} + d] & c_{0,k}, c, d_{0,k}, d \in \mathscr{R} \\ [a, \ b] & a, b \in \mathscr{R} \end{cases}$$

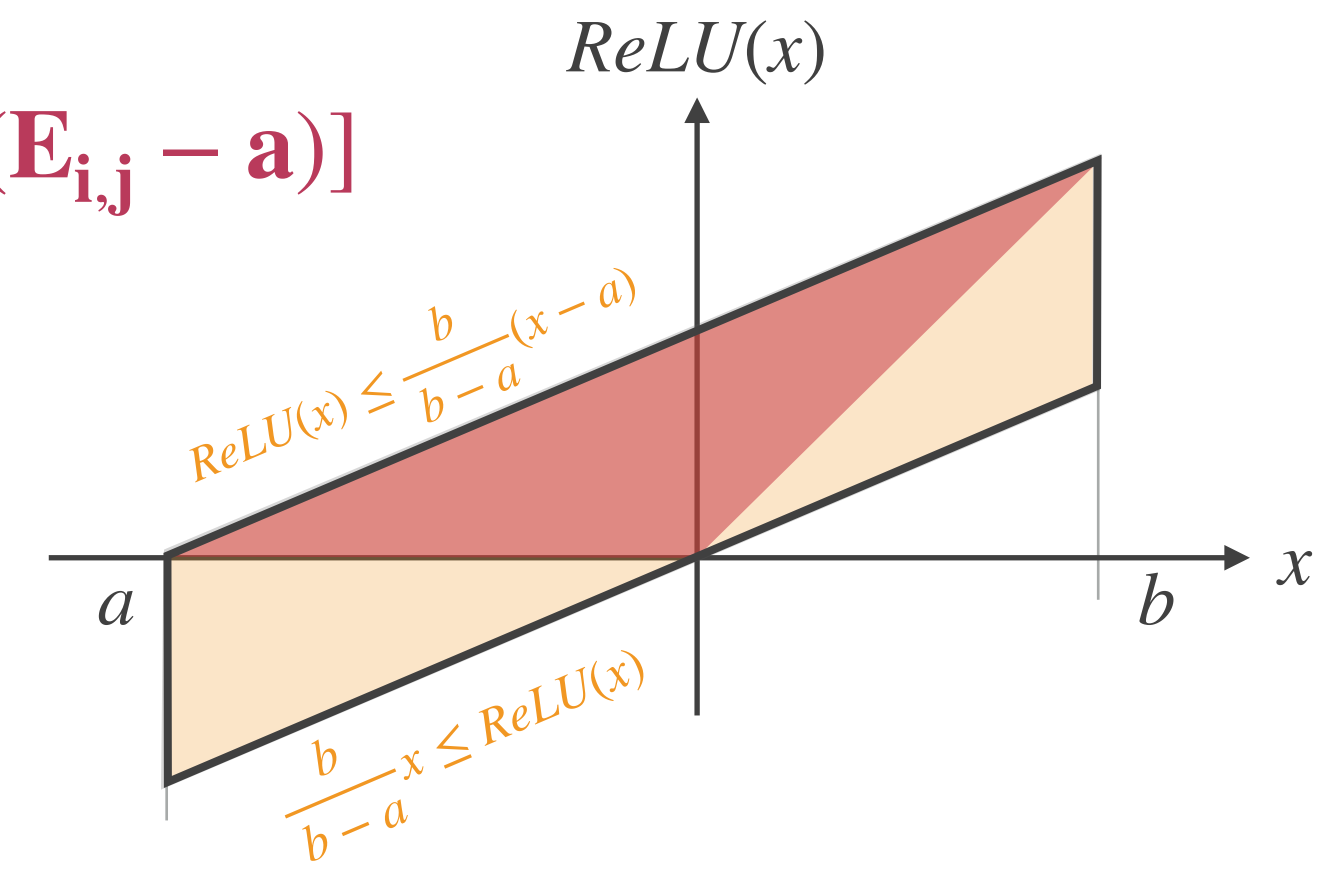
$$x_{i,j} \mapsto \begin{cases} [\mathbf{E_{i,j}}, \mathbf{E_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU ($0 \leq a$):

$$x_{i,j} \mapsto \begin{cases} [\mathbf{E_{i,j}}, \mathbf{E_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU ($a < 0 \wedge 0 < b$):

$$x_{i,j} \mapsto \begin{cases} [\frac{\mathbf{b}}{\mathbf{b-a}}\mathbf{E_{i,j}}, \frac{\mathbf{b}}{\mathbf{b-a}}(\mathbf{E_{i,j}} - \mathbf{a})] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$




ReLU ($b \leq 0$):

$$x_{i,j} \mapsto \begin{cases} [\mathbf{0}, \mathbf{0}] \\ [\mathbf{0}, \mathbf{0}] \end{cases}$$

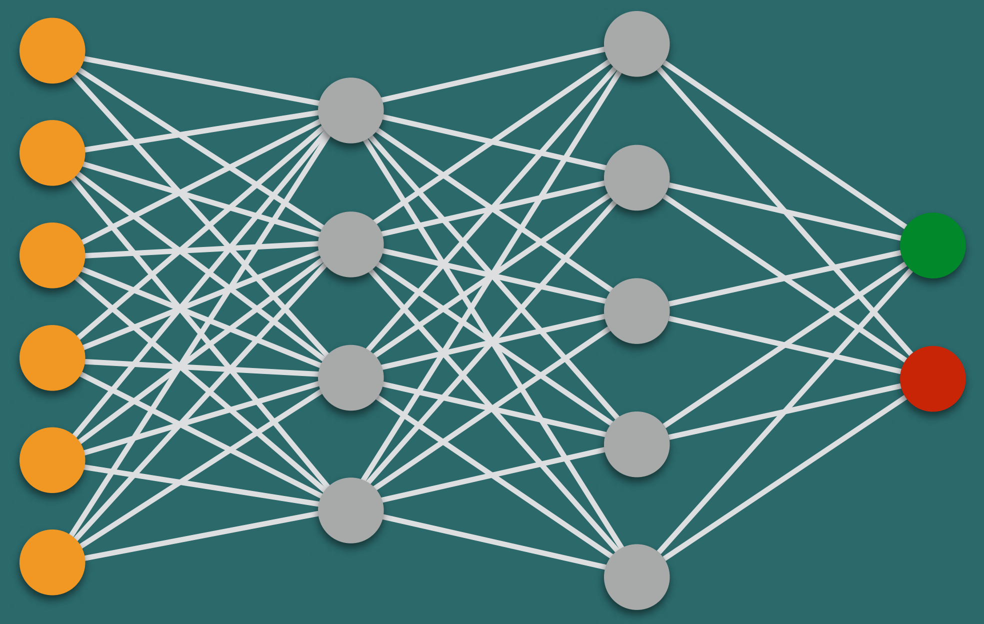
S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana - Efficient Formal Safety Analysis of Neural Networks (NeurIPS 2018)

# Other Complete Methods

- W. Ruan, X. Huang, and Marta Kwiatkowska - Reachability Analysis of Deep Neural Networks with Provable Guarantees (IJCAI 2018)
  **a global optimization-based approach for verifying Lipschitz continuous neural networks**

- G. Singh, T. Gehr, M. Püschel, and M. Vechev - Boosting Robustness Certification of Neural Networks (ICLR 2019)
  **an approach combining abstract interpretation and (mixed integer) linear programming**

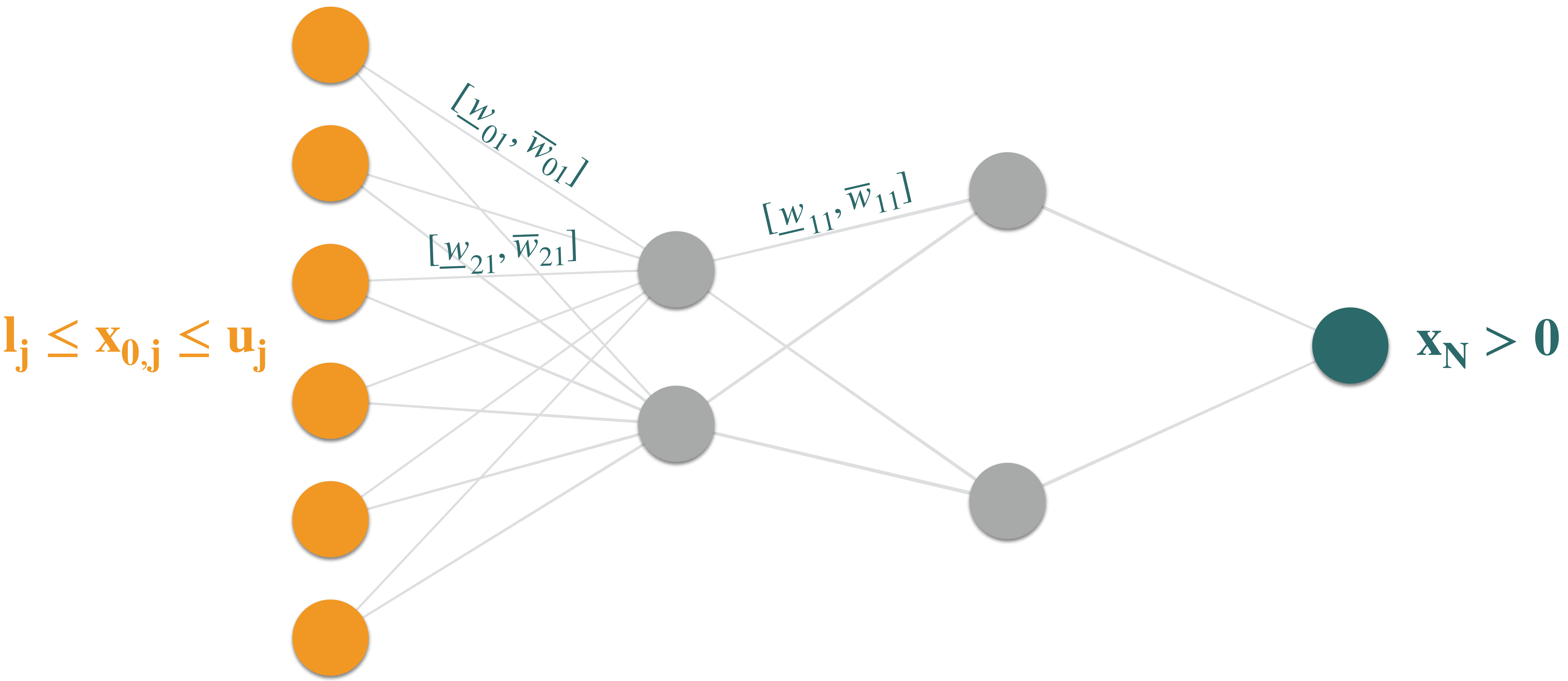
# Other Incomplete Methods



# Interval Neural Networks

## Abstraction-Based Method

**merge neurons layer-wise** according to partitioning strategy and **replace weights with intervals**




P. Prabhakar and Z. R. Afza - Abstraction based Output Range Analysis for Neural Networks (NeurIPS 2019)

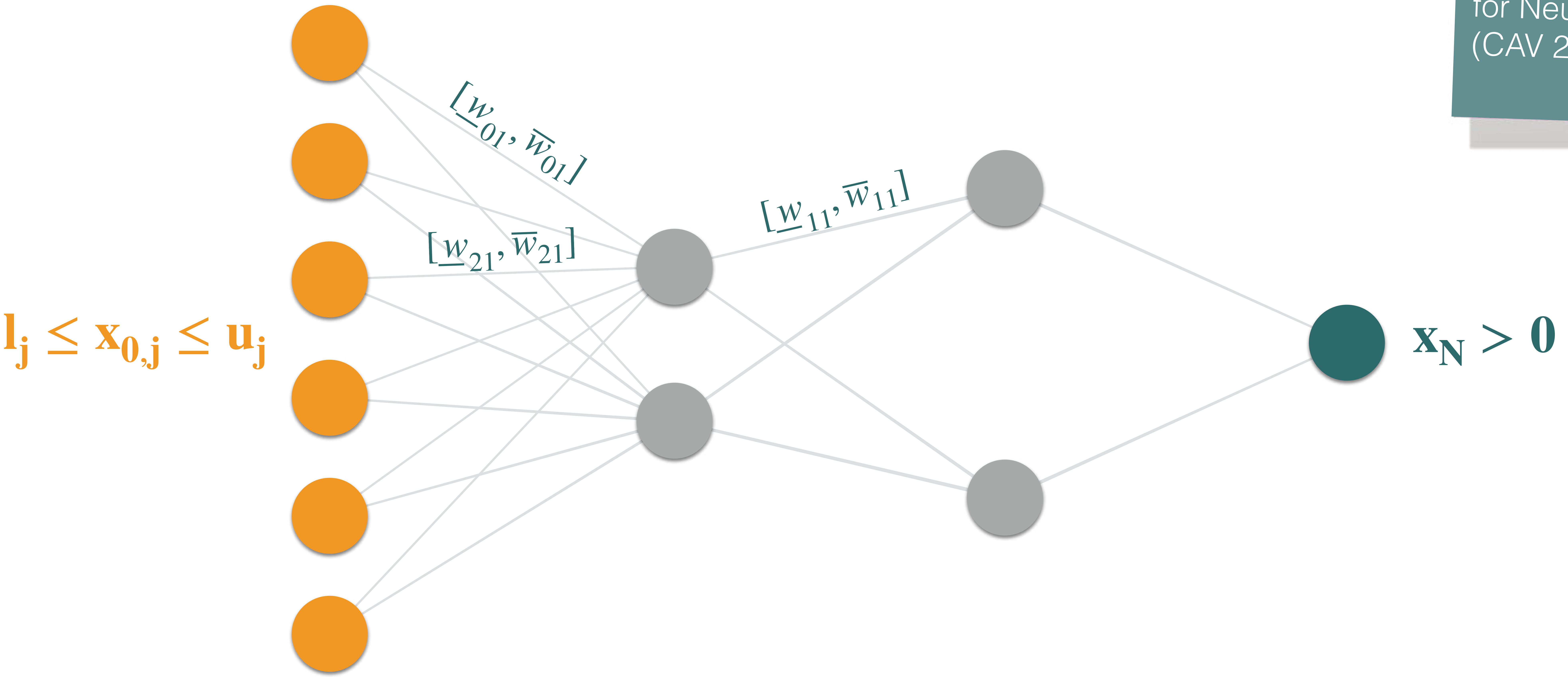
# Interval Neural Networks

## Abstraction-Based Method

merge neuro
according to parti
replace weigh

**Related Work**

Y. Y. Elboher et al. - An Abstraction-Based Framework for Neural Network Verification (CAV 2020)




P. Prabhakar and Z. R. Afza - Abstraction based Output Range Analysis for Neural Networks (NeurIPS 2019)

# Other Incomplete Methods

- W. Xiang, H.-D. Tran, and T. T. Johnson - Output Reachable Set Estimation and Verification for Multi-Layer Neural Networks (2018)
  **an approach combining simulation and linear programming**

- K. Dvijotham, R. Stanforth, S. Gowal, T. Mann, and P. Kohli - A Dual Approach to Scalable Verification of Deep Networks (UAI 2018)
  **an approach based on duality for verifying neural networks**

- E. Wong and Z. Kolter - Provable Defenses Against Adversarial Examples via the Convex Outer Adversarial Polytope (ICML 2018)
  A. Raghunathan, J. Steinhardt, and P. Liang - Certified Defenses against Adversarial Examples (ICML 2018)
  T.-W. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon. Towards Fast Computation of Certified Robustness for ReLU Networks (ICML 2018)
  H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel - Efficient Neural Network Robustness Certification with General Activation Functions (NeurIPS 2018)
  **approaches for finding a lower bound on robustness to adversarial perturbations**

# Other Incomplete Methods

- A. Boopathy, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel - CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks (AAAI 2019)
  **approach focusing on convolutional neural networks**

- C.-Y. Ko, Z. Lyu, T.-W. Weng, L. Daniel, N. Wong, and D. Lin - POPQORN: Quantifying Robustness of Recurrent Neural Networks (ICML 2019)
  H. Zhang, M. Shinn, A. Gupta, A. Gurfinkel, N. Le, and N. Narodytska - Verification of Recurrent Neural Networks for Cognitive Tasks via Reachability Analysis (ECAI 2020)
  **approaches focusing on recurrent neural networks**

- D. Gopinath, H. Converse, C. S. Pasareanu, and A. Taly - Property Inference for Deep Neural Networks (ASE 2019)
  **an approach for inferring safety properties of neural networks**

# Complete Methods

## ADVANTAGES

- **sound** and **complete**

## DISADVANTAGES

- soundness not typically guaranteed with respect to **floating-point arithmetic**
- **do not scale** to **large models**
- **often** limited **to certain model** architectures

# Incomplete Methods

## ADVANTAGES

- **able to scale** to **large models**
- sound often also with respect to **floating-point arithmetic**
- **less limited to certain model architectures**

## DISADVANTAGES

- suffer from **false positives**

QUESTIONS?