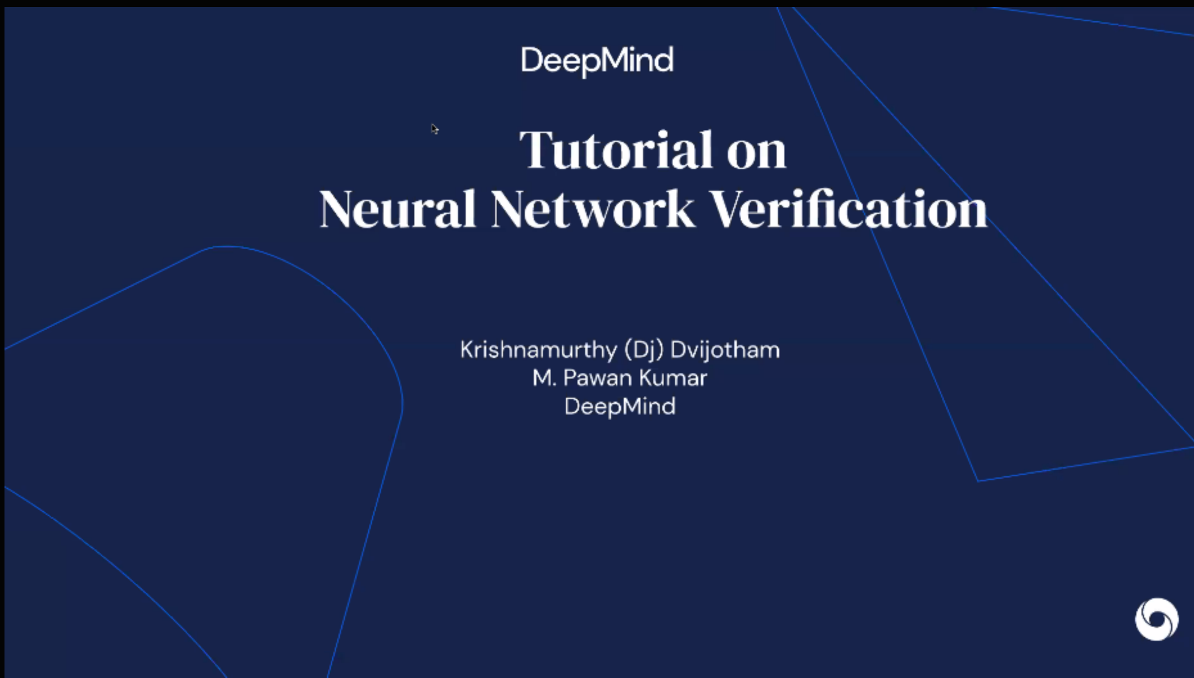
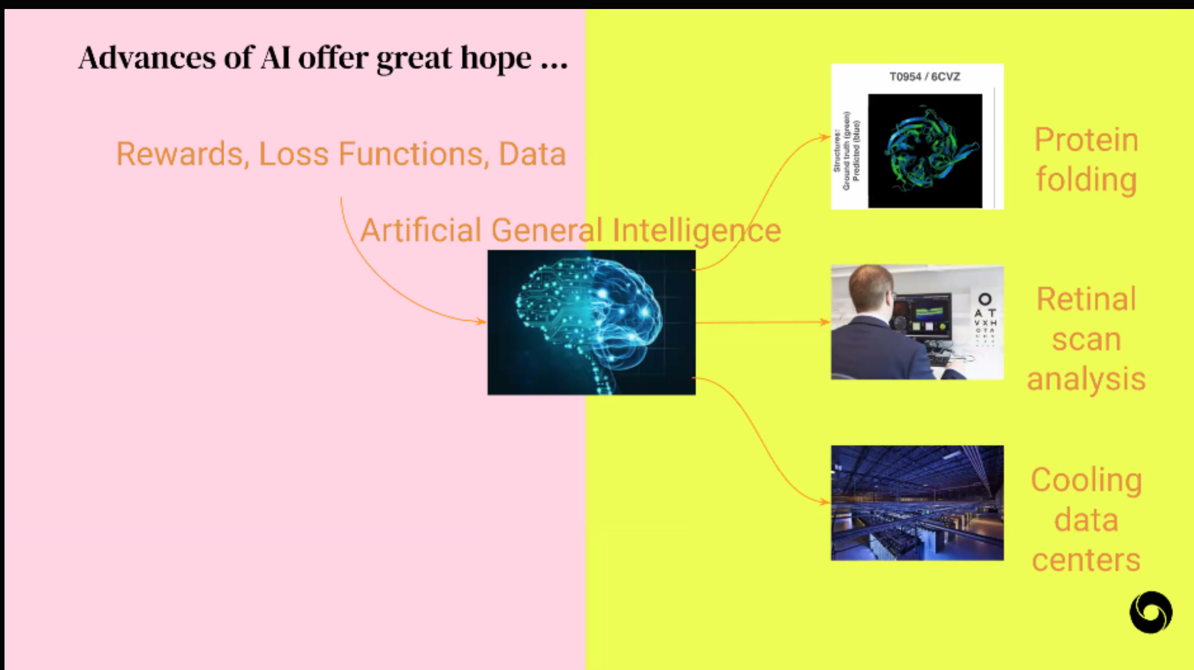
Day 11:

Speaker: Pawan Kumar, University of Oxford, UK

Krishnamurthy Dvijotham, DeepMind

Title: Neural Network Verification

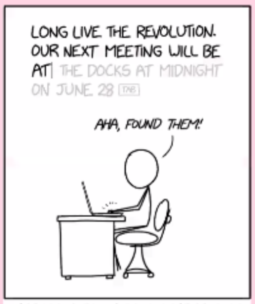




Failure modes of AI abound ....

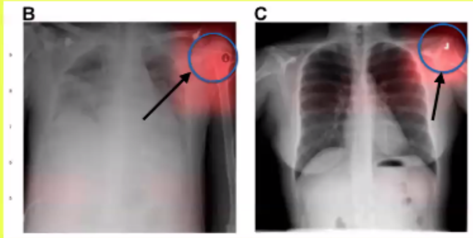
How is AI technology impacting the world today? And how can this go wrong?

Privacy Sensitive Data can be leaked *



WHEN YOU TRAIN PREDICTIVE MODELS ON INPUT FROM YOUR USERS, IT CAN LEAK INFORMATION IN UNEXPECTED WAYS.

Reliability + Privacy



Robustness to spurious correlations

* Markers in training data

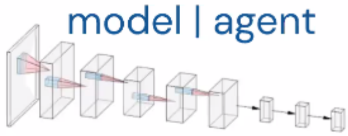
Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
94.0%	79.2%	100%	98.3%	20.8%
99.3%	65.5%	99.2%	94.0%	33.8%
88.0%	65.3%	99.7%	92.9%	34.4%

Fairness + Robustness

* Fairness Issues → Doesn't cover all demographies

The meta-problem

data | experience → model | agent



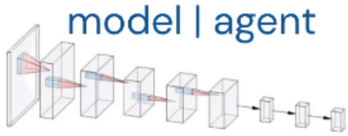
vanilla training

Biased
Limited
Sensitive

* Biased
* Non-robust
* Unsafe
* Non-private

The meta-solution

data | experience → model | agent



Biased
Limited
Sensitive

spec-consistent training

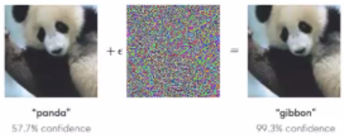
rules | specifications

Unbiased
Robust
Safe
Private
Calibrated

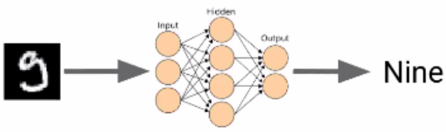
Formal specifications for ML models

- robustness to adversaries
- fairness and unbiasedness
- Physics-compliant (satisfies conservation of energy, conservation of momentum etc.)
- Uncertainty calibrated ...

Adversarial attacks on image classifiers



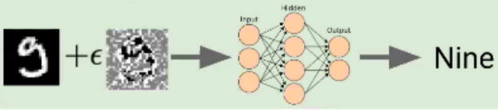
ε ↓
mostly empirical.



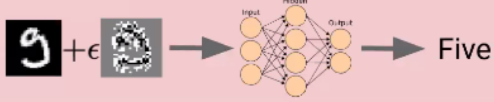
Specification: Output remains "Nine" for **ALL IMAGES** of the form

$\pm\epsilon$ $\|\;\;\| \leq 1$

Projected Gradient Attack



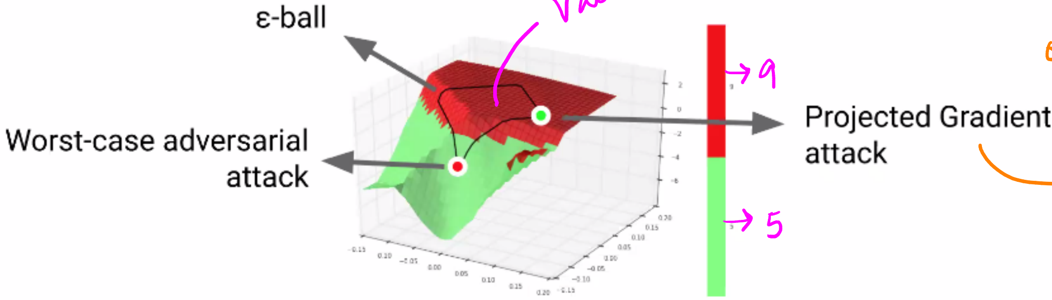
True worst case



Global Perturbations
(Very difficult to find)

Why PGD attack fails?

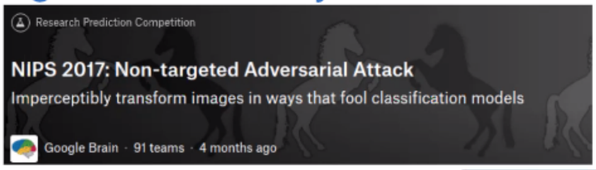
Valid Perturbation Stage



To converge to optimal solⁿ.
↑
Compute the gradient of the z-axis

Meta lesson: Finding failure modes of AI systems is difficult!

Defense strategies don’t really work



Evaluation of NIPS competition winners/published papers

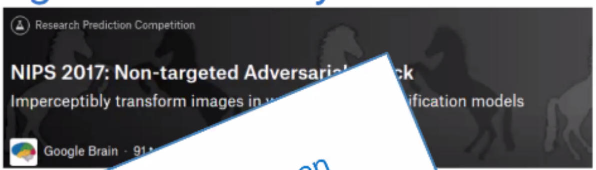
- Non-differentiable models (ICLR 2018)
- Generative-denoising (ICLR 2018)
- Denoising with semantic features (NIPS Competition winner)
- Constraining input gradients (ICML 2017)
- Stochasticity / Ensembling (ICLR 2018, NIPS 2nd place)

Defense Strategy	Standardized Evaluation	Strongest Adversary
CIFAR-10 (e = 8)		
Non-differentiability	43%	0%
Generative modeling	46%	10%
Adversarial Training	45%	45%
ImageNet (e = 2)		
Stochasticity	32%	1%
Denoising	61%	0%

Athalye et al. *Gradient obfuscation ...* ICML 2018

Uesato et al. *Dangers of weak attacks.* ICML 2018

Defense strategies don’t really work



Evaluation of NIPS competition winners/published papers

- Non-differentiable models (I
- Generative-denoisin
- Denoising with (NIPS Co
- Constraini
- Stochastici (ICLR 2018, I

Need for verification:
Provable guarantee that no adversarial attack can succeed

	...tion	Strongest Adversary
(e = 8)		
...ty	43%	0%
...ve modeling	46%	10%
Adversarial Training	45%	45%
ImageNet (e = 2)		
Stochasticity	32%	1%
Denoising	61%	0%

Athalye et al. *Gradient obfuscation ...* ICML 2018

Uesato et al. *Dangers of weak attacks.* ICML 2018

Hardness of verification in general

Verification by enumeration:

Discretize space of perturbations

 $\pm\epsilon$ 

(Perturbation size) $^{(\#Pixels)}$ - search space grows exponentially!

- Verifying 10% perturbation attack on MNIST takes $O(10^{1000})$ CPU-years
- NP-hard to find constant factor approx of optimal attack [Weng et al, 2018]

Hardness of verification in general

Verification by enumeration:

Discretize space of perturbations

(Perturbation size) [(#Pixels)] ... entially!

- Verifyi... on MNIST takes O(10^{1000}) CPU-years
- NP-ha... ctor approx of optimal attack [Weng et al, 2018]

Need for scalable verification:
Trade of scalability and completeness

Other specifications studied

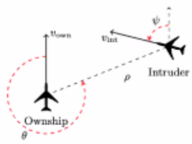
fails to give adverse results

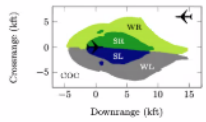
large Perturbations

Undersensitivity spec:
[Welbl et al, ICLR 2020]

Original Sample	**Premise:** A little boy in a blue shirt holding a toy. **Hypothesis:** A boy dressed in blue holds a toy. Entailment (86.4%)
Reduced Sample	**Premise:** A little boy in a blue shirt holding a toy. **Hypothesis:** A boy dressed in blue holds a toy. Entailment (91.9%)

Safe actions:
[Katz et al, CAV 2017]



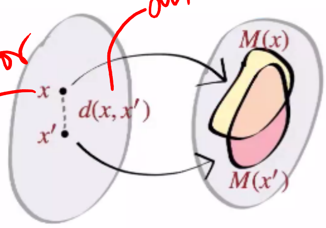


Other specifications studied

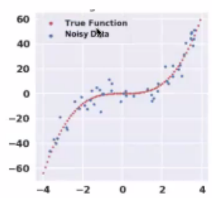
Individual fairness
[John et al, UAI 2020]

feature vector

distance



Probabilistic Safety
[Wicker et al, UAI 2019]



Neural Network Verification

Neural network f

Scalar output z = f($\mathbf{x}$)

E.g. in binary classification, z = s(y*;$\mathbf{x}$) – s(y;$\mathbf{x}$) for y ≠ y*

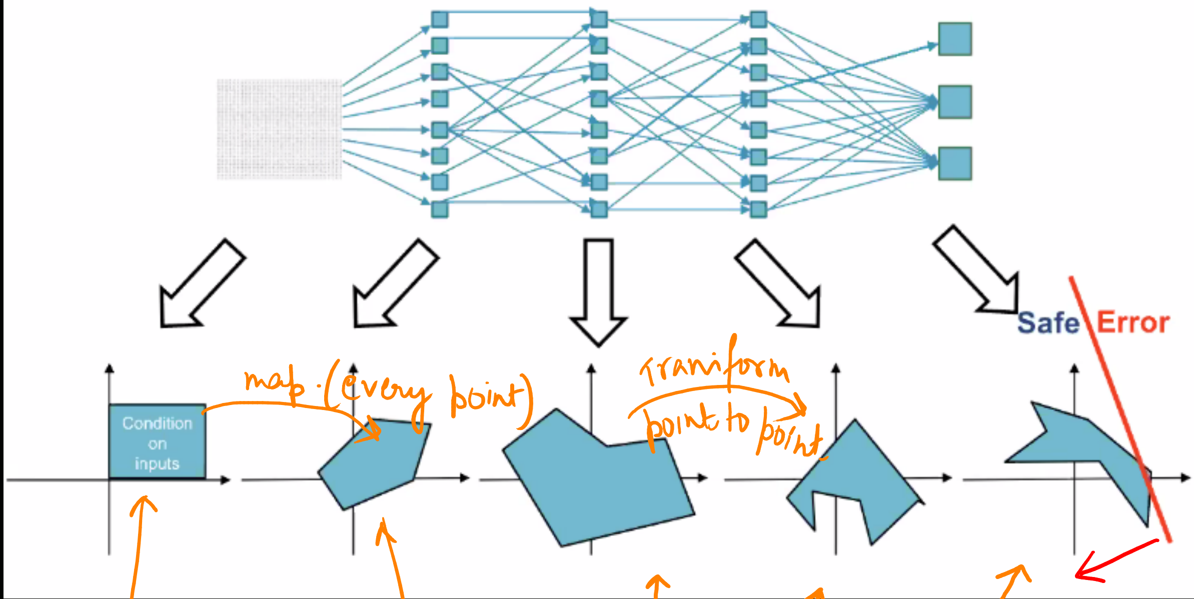
Property: f($\mathbf{x}$) > 0 for all $\mathbf{x} \in X$

Outline

- Incomplete Verification → Only if some cases verification will say false even if it's true
  - Overview
  - Example: Interval Bound Propagation
  - Example: Linear Programming Relaxation
- Complete Verification
  - Branch and Bound
  - Application to verification

Neural Network Verification

Is there an erroneous output?



Every possible i/p { i.e every point within the rectangle is a possible image for one example

(+ ε, -ε) etc.

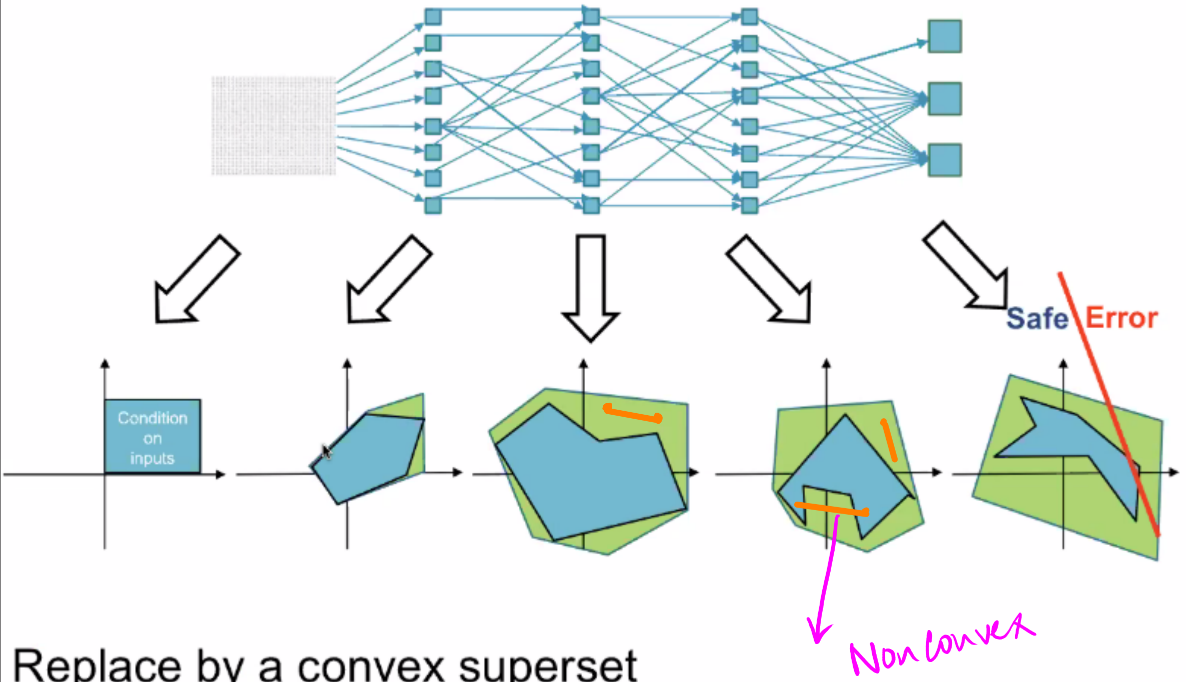
(or a possible image and its perturbations)

Transformed by some fun^n

Non-convexity makes the problem NP-hard

Incomplete Verification

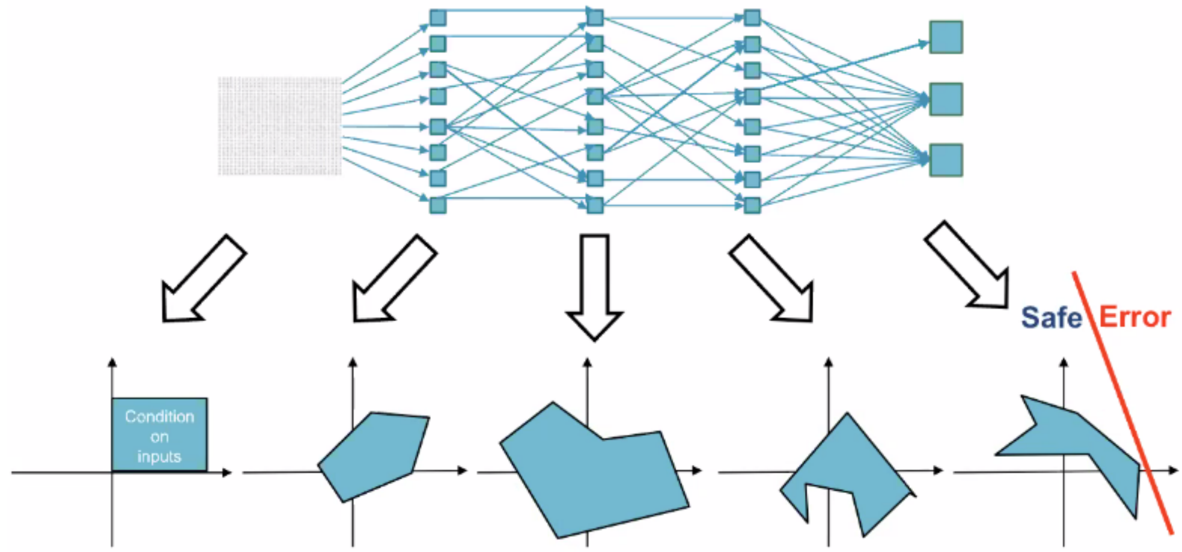
Is there an erroneous output?



Replace by a convex superset

Incomplete Verification

Is there an erroneous output?

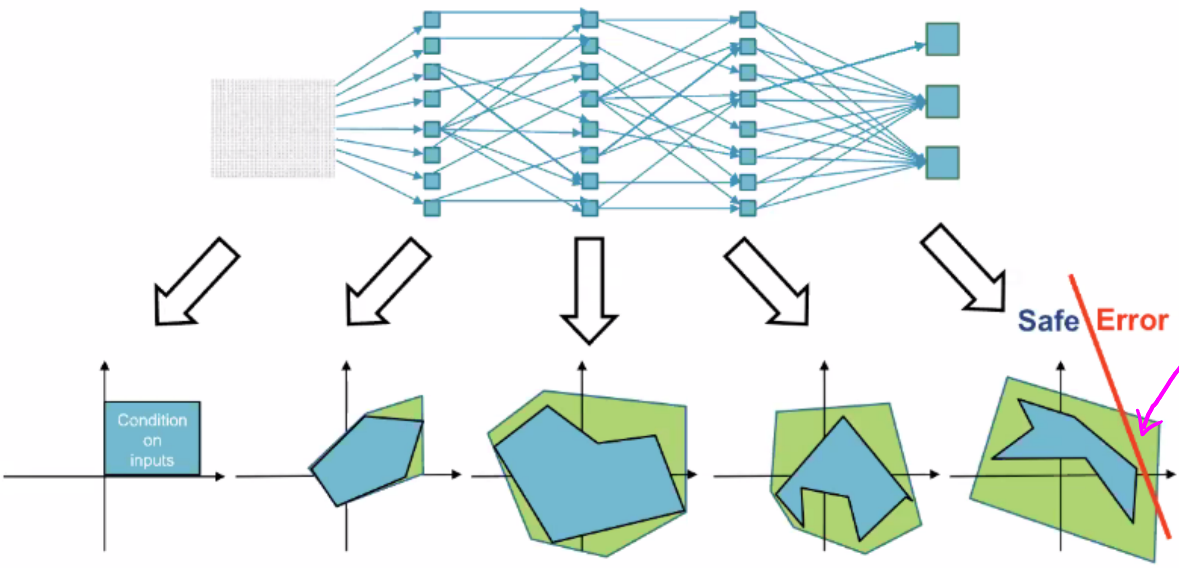


Say, non-convex set has no erroneous output

Incomplete Verification

N/W is robust but not always *

Is there an erroneous output?



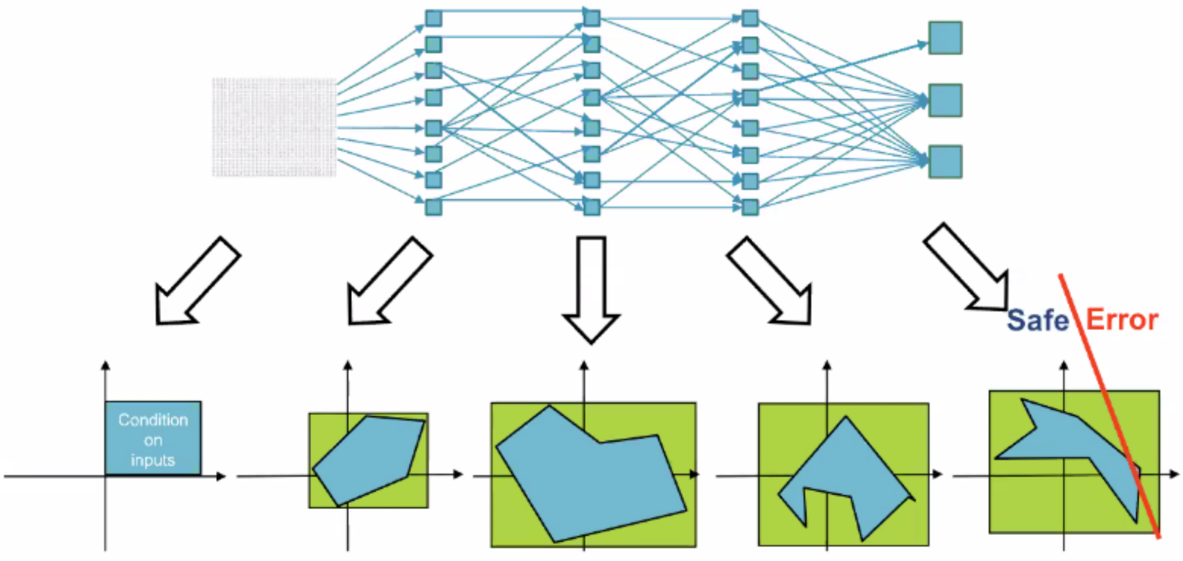
Convex superset might give incorrect answer

Incomplete Verification

- Useful in practice
- Verifiably robust training
- Key part of complete verification
- How do we construct convex superset?

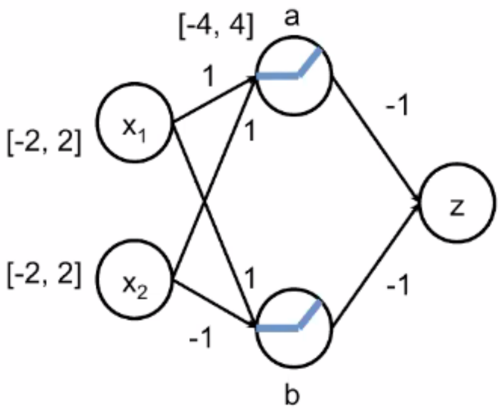
Inteval Bound Propagation

Is there an erroneous output?



Axis aligned convex superset

Example



$-2 \le x_1 \le 2$

$-2 \le x_2 \le 2$

$a_{in} = x_1 + x_2$

$a_{out} = \max\{a_{in}, 0\}$

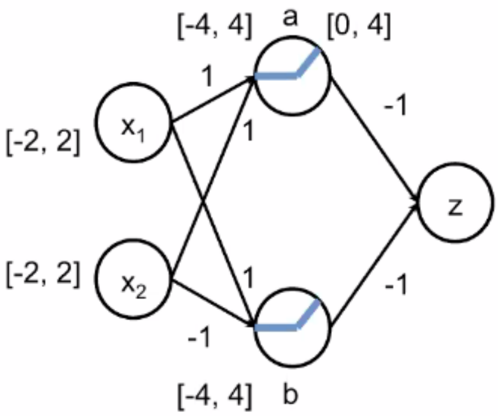
Minimum value of a_{in}? -4

Maximum value of a_{in}? 4

Minimum value of a_{out}? 0

Maximum value of a_{out}? 4

Example



$-2 \le x_1 \le 2$

$-2 \le x_2 \le 2$

$b_{in} = x_1 - x_2$

$b_{out} = \max\{b_{in}, 0\}$

Minimum value of b_{in}? -4

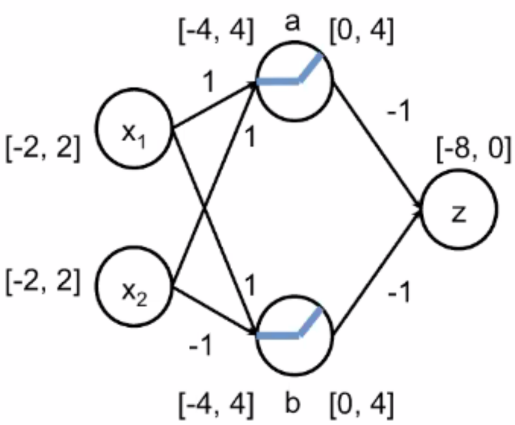
Maximum value of b_{in}? 4

Minimum value of b_{out}? 0

Maximum value of b_{out}? 4

* Deeper the n/w, more looser the interval

Example



$-2 \le x_1 \le 2$

$-2 \le x_2 \le 2$

$b_{in} = x_1 - x_2$

$b_{out} = \max\{b_{in}, 0\}$

$z = -a_{out} - b_{out}$

Minimum value of z? -8

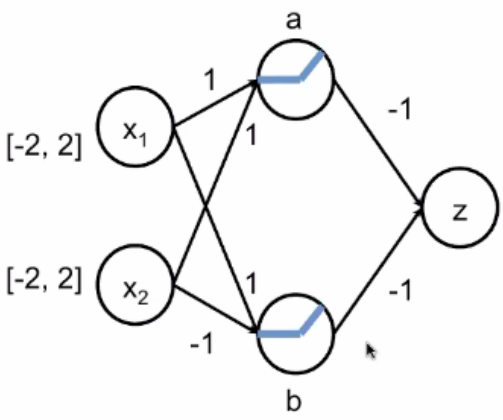
Maximum value of z? 0

→ output = [-8, 0] {No +ve o/p}

∴ Incomplete Verification

Linear Programming Relaxation

Example



$-2 \le x_1 \le 2$

$-2 \le x_2 \le 2$

$a_{in} = x_1 + x_2$

$b_{in} = x_1 - x_2$

$a_{out} = \max\{a_{in}, 0\}$

$b_{out} = \max\{b_{in}, 0\}$

$z = - a_{out} - b_{out}$

Example

min $\quad z$

Linear constraints s.t.

$\mathbf{-2 \le x_1 \le 2}$

$\mathbf{-2 \le x_2 \le 2}$

$\mathbf{a_{in} = x_1 + x_2}$

$\mathbf{b_{in} = x_1 - x_2}$

$a_{out} = \max\{a_{in}, 0\}$

$b_{out} = \max\{b_{in}, 0\}$

$\mathbf{z = - a_{out} - b_{out}}$

Example

$$\min \quad z$$

s.t. $-2 \leq x_1 \leq 2$

$-2 \leq x_2 \leq 2$

$a_{in} = x_1 + x_2$

$b_{in} = x_1 - x_2$

$a_{out} = \max\{a_{in}, 0\}$

$b_{out} = \max\{b_{in}, 0\}$

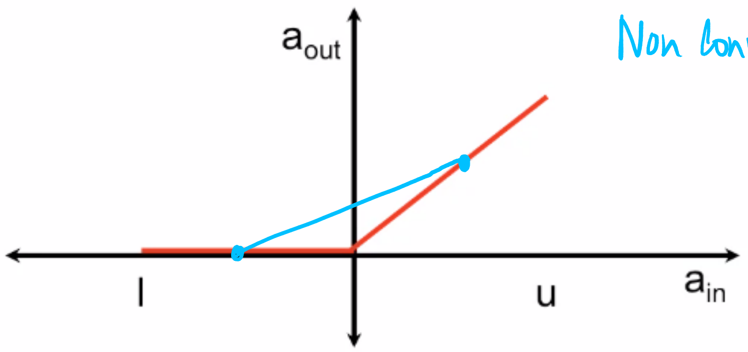
$z = -a_{out} - b_{out}$

Non-linear constraints

NP-hard problem

Relaxation

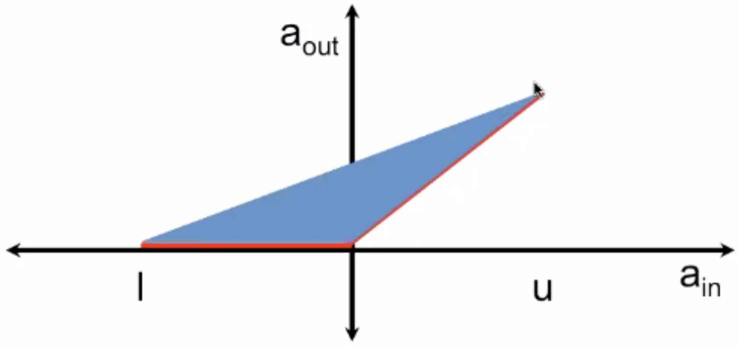
$a_{out} = \max\{a_{in}, 0\}$ $\qquad$ $a_{in} \in [l,u]$



Non Convex → RELU

Relaxation

$a_{out} = \max\{a_{in}, 0\}$ $\qquad a_{in} \in [l, u]$



Ehlers 2017

Replace with convex superset

Example

Linear Program

Several **"efficient"** solvers

$$
\begin{aligned}
\min \quad & z \\
\text{s.t.} \quad & -2 \le x_1 \le 2 \\
& -2 \le x_2 \le 2 \\
& a_{in} = x_1 + x_2 \\
& b_{in} = x_1 - x_2 \\
& a_{out} \ge 0,\ a_{out} \ge a_{in},\ a_{out} \le 0.5a_{in} + 2 \\
& b_{out} \ge 0,\ b_{out} \ge b_{in},\ b_{out} \le 0.5b_{in} + 2 \\
& z = -\ a_{out} - b_{out}
\end{aligned}
$$

Branch and Bound

- Unified framework for complete verification
- Different bounds and bounding algorithms
  - Bound propagation (e.g. β-CROWN)
  - Tight LP relaxations (e.g. disjunctive programming)
  - Efficient solvers (e.g. Stagewise, Active sets)
- Different branching
  - Hand-designed heuristics (e.g. BaBSR)
  - Learning based heuristics (e.g. NN Branching)

* Check Jax_verify $\rightarrow$ github.