Imperial College London

computational optimisation group

EPSRC
Engineering and Physical Sciences Research Council

Partition-based formulations for mixed-integer optimization of trained ReLU neural networks

Calvin Tsay, Jan Kronqvist, Alexander Thebelt, & Ruth Misener

Funding EPSRC EP/P016871/1 & EP/T001577/1

Friday 18th June, 2021

Papers Kronqvist, Misener, Tsay, *CPAIOR*, 2021; *arXiv:2101.12708*
Tsay, Kronqvist, Thebelt, Misener, *arXiv:2102.04373*, 2021

Team members

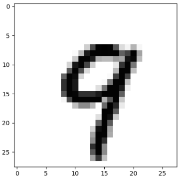
Calvin Tsay
Imperial

Jan Kronqvist
KTH

Alexander Thebelt
Imperial

Ruth Misener
Imperial

Optimization problems over trained neural networks



Given . . . .

Trained NN

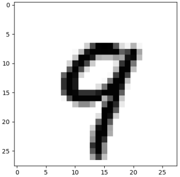
Image $\bar{\boldsymbol{x}}$

Label $j = 9$

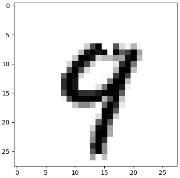
Adversary? $k = 4$

- **Verification** *[Feasibility]* Is there an adversary labeled k within a given perturbation ℓ_1 or ℓ_∞?
- **Optimal adversary** What image within perturbation ℓ_1 or ℓ_∞ maximizes the prediction difference?
- **Minimally distorted adversary** [Croce and Hein, 2020] Smallest perturbation ℓ_1 or ℓ_∞ over which the NN predicts label k?
- **Lossless compression** [Serra et al., 2020] Can I safely remove NN nodes or layers?

Optimization problems over trained neural networks



$\downarrow \ell_1$



$k = 4,$

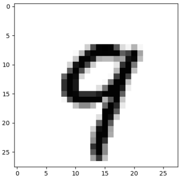
$\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_1 = 4$

Given . . . .

Trained NN	
Image	$\bar{\boldsymbol{x}}$
Label	$j = 9$
Adversary?	$k = 4$

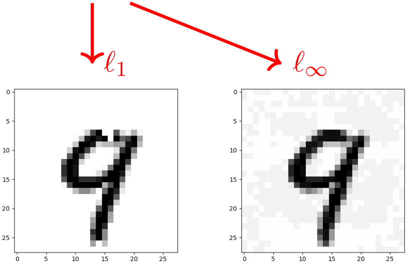
- **Verification** *[Feasibility]* Is there an adversary labeled k within a given perturbation ℓ_1 or ℓ_∞?
- **Optimal adversary** What image within perturbation ℓ_1 or ℓ_∞ maximizes the prediction difference?
- **Minimally distorted adversary** [Croce and Hein, 2020] Smallest perturbation ℓ_1 or ℓ_∞ over which the NN predicts label k?
- **Lossless compression** [Serra et al., 2020] Can I safely remove NN nodes or layers?

Optimization problems over trained neural networks



Given . . . .

Trained NN	
Image	$\bar{\boldsymbol{x}}$
Label	$j = 9$
Adversary?	$k = 4$



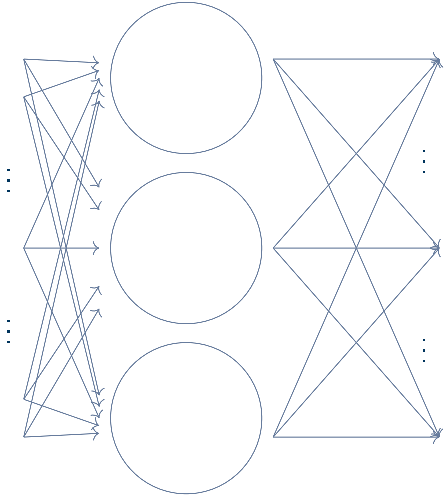
$k = 4,$ $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_1 = 4$

$k = 4,$ $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_\infty = 0.05$

- **Verification** *[Feasibility]* Is there an adversary labeled k within a given perturbation ℓ_1 or ℓ_∞?
- **Optimal adversary** What image within perturbation ℓ_1 or ℓ_∞ maximizes the prediction difference?
- **Minimally distorted adversary** [Croce and Hein, 2020] Smallest perturbation ℓ_1 or ℓ_∞ over which the NN predicts label k?
- **Lossless compression** [Serra et al., 2020] Can I safely remove NN nodes or layers?

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



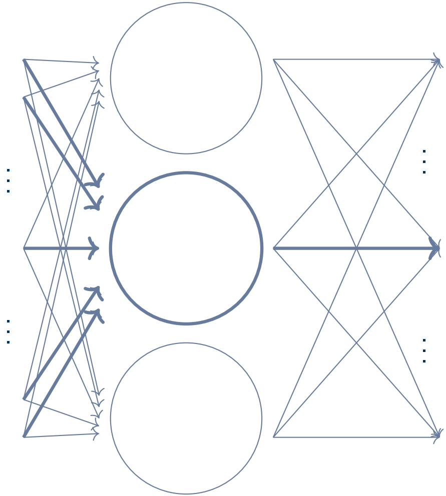
$$y \geq (\boldsymbol{w}^T \boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T \boldsymbol{x} + b) - (1 - \sigma) LB^0$$
$$0 \leq y \leq \sigma UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients $LB^0, UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T \boldsymbol{x} + b) \in [LB^0, UB^0]$

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



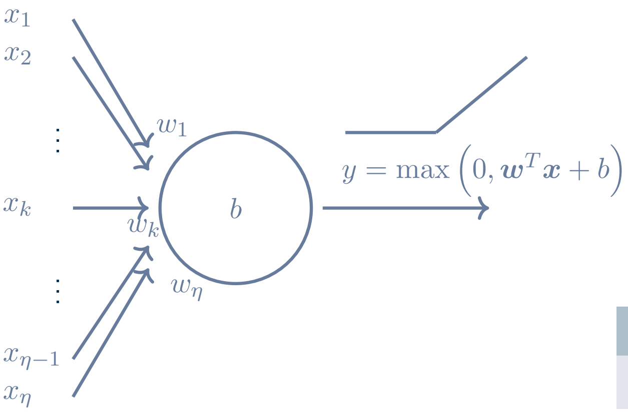
$$y \geq (\boldsymbol{w}^T \boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T \boldsymbol{x} + b) - (1 - \sigma) LB^0$$
$$0 \leq y \leq \sigma UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients $LB^0, UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T \boldsymbol{x} + b) \in [LB^0, UB^0]$

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



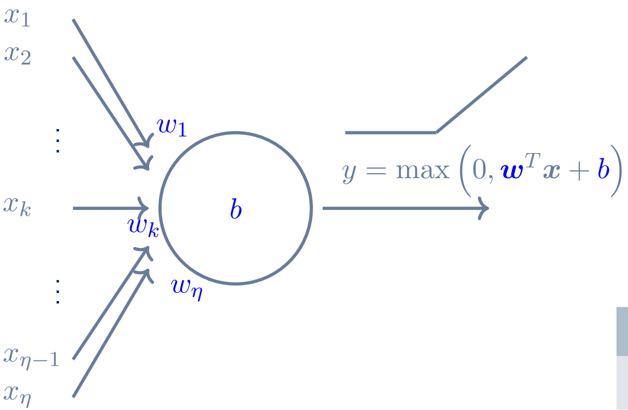
$$y \geq (\boldsymbol{w}^T\boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T\boldsymbol{x} + b) - (1-\sigma)LB^0$$
$$0 \leq y \leq \sigma\, UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients $LB^0, UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T\boldsymbol{x} + b) \in [LB^0, UB^0]$

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



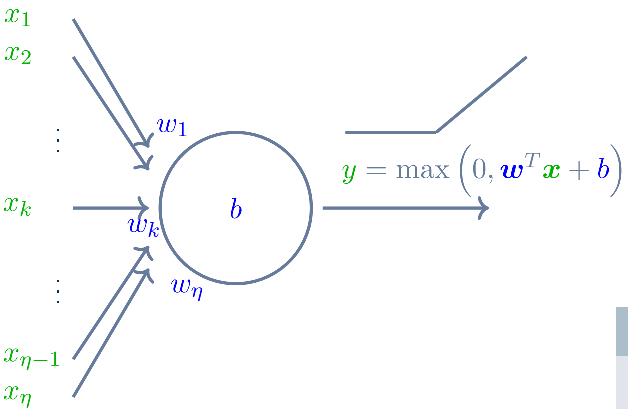
$$y \geq (\boldsymbol{w}^T\boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T\boldsymbol{x} + b) - (1-\sigma)LB^0$$
$$0 \leq y \leq \sigma\, UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients LB^0, $UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T\boldsymbol{x} + b) \in [LB^0, UB^0]$

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



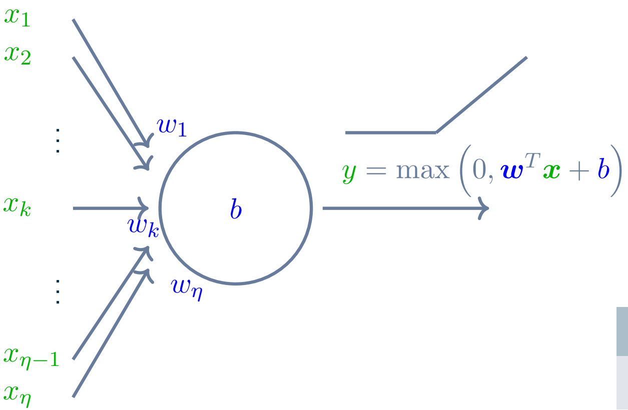
$$y \geq (\boldsymbol{w}^T\boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T\boldsymbol{x} + b) - (1 - \sigma)LB^0$$
$$0 \leq y \leq \sigma\, UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients LB^0, $UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T\boldsymbol{x} + b) \in [LB^0, UB^0]$

Big-M formulation of a learned ReLU neural network

Lomuscio and Maganti [2017], Fischetti and Jo [2018]



$$y \geq (\boldsymbol{w}^T\boldsymbol{x} + b)$$
$$y \leq (\boldsymbol{w}^T\boldsymbol{x} + b) - (1 - \sigma)LB^0$$
$$0 \leq y \leq \sigma\, UB^0$$
$$\sigma \in \{0, 1\}$$

Big-M coefficients $LB^0, UB^0 \in \mathbb{R}$

$(\boldsymbol{w}^T\boldsymbol{x} + b) \in [LB^0, UB^0]$

Why do we want to improve big-M?

State-of-the-art verification tools rely on big-M

MIPVerify [Tjeng et al., 2017] • NSVerify [Akintunde et al., 2018]

Applications using big-M

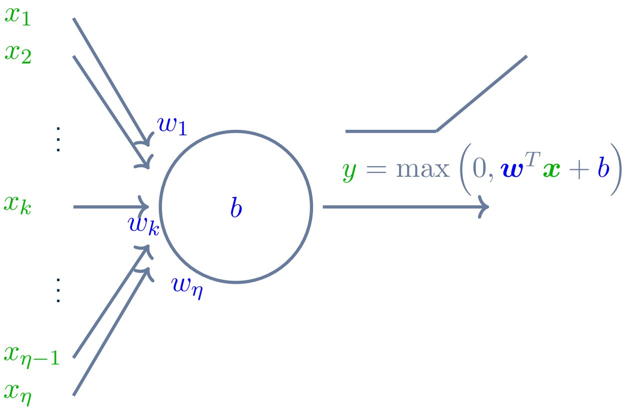
Optimize NN [Dutta et al., 2018, Lomuscio and Maganti, 2017, Wu et al., 2020, Grimstad and Andersson, 2019] • Get perturbation bounds [Cheng et al., 2017] • Compress NN [Serra et al., 2020] • Count regions [Serra et al., 2018] • Find adversarial examples [Fischetti and Jo, 2018]

Develop alternatives to dynamic cut generation

Anderson et al. [2020] develop exponentially many cuts from a node's convex hull. This method does not scale well to large NNs [De Palma et al., 2021]. Botoeva et al. [2020] find callback frequencies $<0.025\%$ balance computational burden and model tightening.

Idea: Partition-based formulation of a learned ReLU neural network

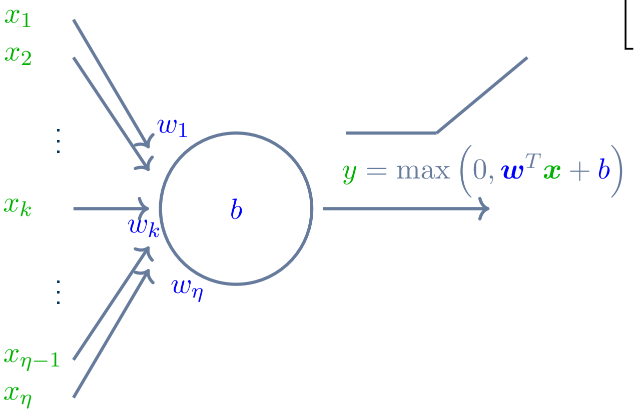
Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



Idea: Partition-based formulation of a learned ReLU neural network

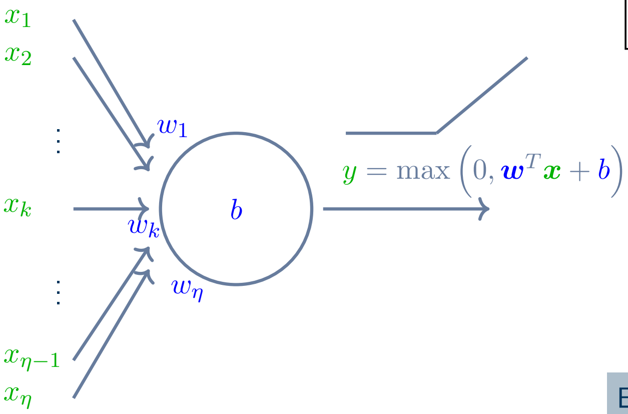
Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]

$$\begin{bmatrix} y = 0 \\ \boldsymbol{w}^T\boldsymbol{x} + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = \boldsymbol{w}^T\boldsymbol{x} + b \\ \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \end{bmatrix}$$



Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



$$\begin{bmatrix} y = 0 \\ \boldsymbol{w}^T\boldsymbol{x} + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = \boldsymbol{w}^T\boldsymbol{x} + b \\ \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \end{bmatrix}$$

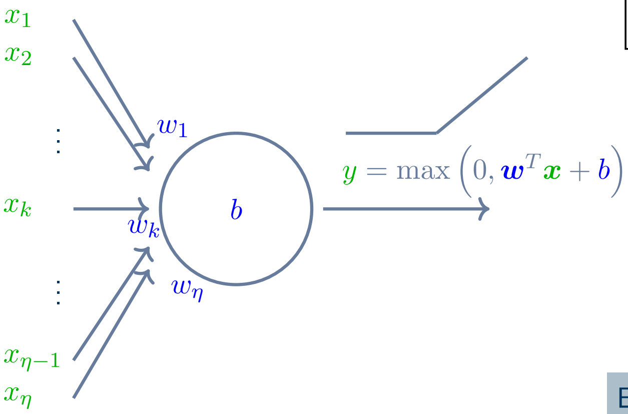
$$\begin{aligned}
\boldsymbol{w}^T\boldsymbol{x} &= z^a + z^b \\
z^a + \sigma b &\leq 0 \\
z^b + (1-\sigma)b &\geq 0 \\
y &= z^b + (1-\sigma)b \\
\sigma LB^a \leq\ & z^a \leq \sigma UB^a \\
(1-\sigma)LB^b \leq\ & z^b \leq (1-\sigma)UB^b
\end{aligned}$$

Extra: 1 variable z^b, 4 constraints

Equivalent to the big-M formulation

Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



$$\begin{bmatrix} y = 0 \\ \boldsymbol{w}^T\boldsymbol{x} + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = \boldsymbol{w}^T\boldsymbol{x} + b \\ \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \end{bmatrix}$$

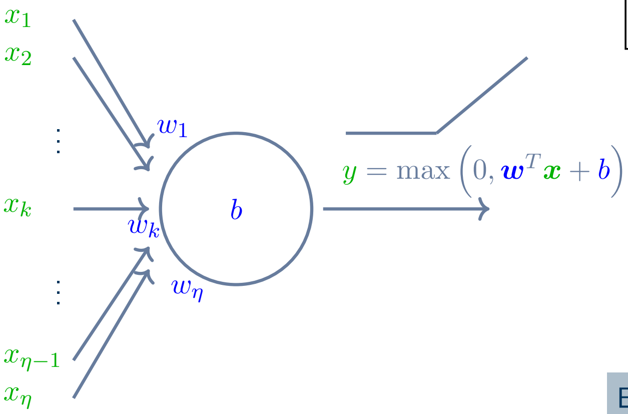
$$\boldsymbol{w}^T\boldsymbol{x} = z^a + z^b$$
$$z^a + \sigma b \leq 0$$
$$z^b + (1-\sigma)b \geq 0$$
$$y = z^b + (1-\sigma)b$$
$$\sigma LB^a \leq z^a \leq \sigma UB^a$$
$$(1-\sigma)LB^b \leq z^b \leq (1-\sigma)UB^b$$

Extra: 1 variable z^b, 4 constraints

Equivalent to the big-M formulation

Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



$$\begin{bmatrix} y = 0 \\ \boldsymbol{w}^T\boldsymbol{x} + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = \boldsymbol{w}^T\boldsymbol{x} + b \\ \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \end{bmatrix}$$

$$\boldsymbol{w}^T\boldsymbol{x} = z^a + z^b$$

$$z^a + \sigma b \leq 0$$

$$z^b + (1-\sigma) b \geq 0$$

$$y = z^b + (1-\sigma) b$$

$$\sigma LB^a \leq z^a \leq \sigma UB^a$$

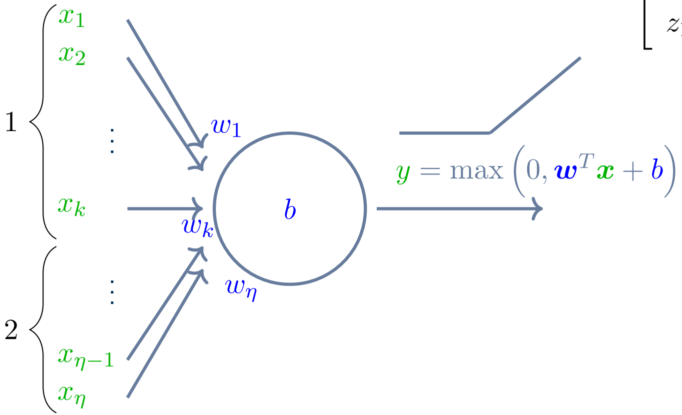
$$(1-\sigma) LB^b \leq z^b \leq (1-\sigma) UB^b$$

Extra: 1 variable z^b, 4 constraints

Equivalent to the big-M formulation

Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



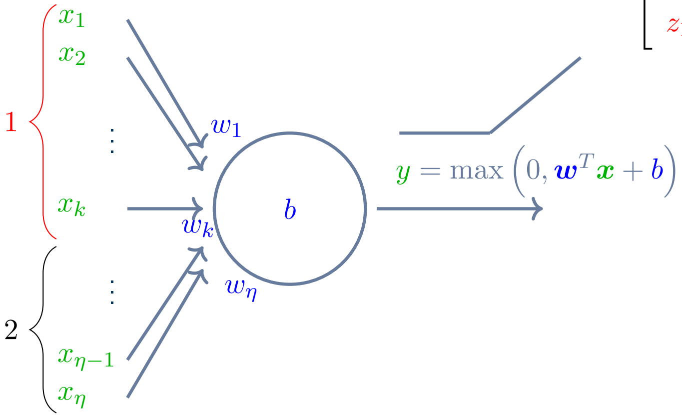
$$\begin{bmatrix} y = 0 \\ z_1 + z_2 + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = z_1 + z_2 + b \\ z_1 + z_2 + b \geq 0 \end{bmatrix}$$

$$\sum_{i=1}^{k} w_i x_i = z_1^a + z_1^b$$

$$\sum_{i=k+1}^{\eta} w_i x_i = z_2^a + z_2^b$$

Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



$$\begin{bmatrix} y = 0 \\ z_1 + z_2 + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = z_1 + z_2 + b \\ z_1 + z_2 + b \geq 0 \end{bmatrix}$$

$$\sum_{i=1}^{k} w_i x_i = z_1^a + z_1^b$$

$$\sum_{i=k+1}^{\eta} w_i x_i = z_2^a + z_2^b$$

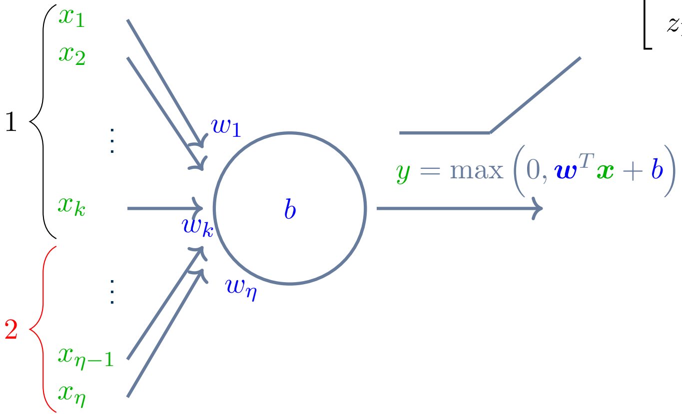
Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]

$$\begin{bmatrix} y = 0 \\ z_1 + z_2 + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = z_1 + z_2 + b \\ z_1 + z_2 + b \geq 0 \end{bmatrix}$$

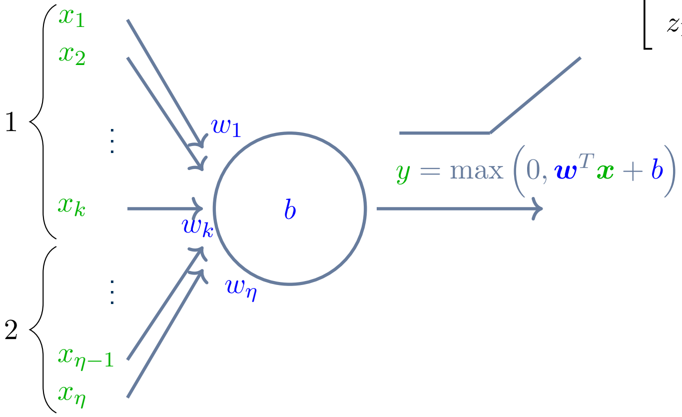
$$\sum_{i=1}^{k} w_i x_i = z_1^a + z_1^b$$

$$\sum_{i=k+1}^{\eta} w_i x_i = z_2^a + z_2^b$$



Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]



$$\begin{bmatrix} y = 0 \\ z_1 + z_2 + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = z_1 + z_2 + b \\ z_1 + z_2 + b \geq 0 \end{bmatrix}$$

$$\sum_{i=1}^{k} w_i x_i = z_1^a + z_1^b$$

$$\sum_{i=k+1}^{\eta} w_i x_i = z_2^a + z_2^b$$

$$z_1^a + z_2^a + \sigma b \leq 0$$

$$z_1^b + z_2^b + (1-\sigma) b \geq 0$$

$$y = z_1^b + z_2^b + (1-\sigma) b$$

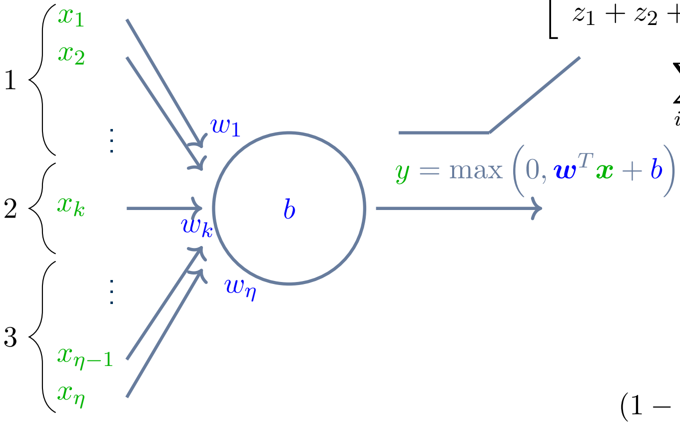
$$\sigma LB_n^a \leq z_n^a \leq \sigma UB_n^a, \forall n = 1,2$$

$$(1-\sigma) LB_n^b \leq z_n^b \leq (1-\sigma) UB_n^b, \forall n = 1,2$$

Idea: Partition-based formulation of a learned ReLU neural network

Use disjunctive programming [Balas, 2018] to develop intermediate relaxations [Kronqvist et al., 2021]

$$\begin{bmatrix} y = 0 \\ z_1 + z_2 + z_3 + b \leq 0 \end{bmatrix} \vee \begin{bmatrix} y = z_1 + z_2 + z_3 + b \\ z_1 + z_2 + z_3 + b \geq 0 \end{bmatrix}$$



$$\sum_{i=1}^{k} w_i x_i = z_n^a + z_n^b, \quad \forall n = 1, 2, 3$$

$$z_1^a + z_2^a + z_3^a + \sigma b \leq 0$$

$$z_1^b + z_2^b + z_3^b + (1 - \sigma) b \geq 0$$

$$y = z_1^b + z_2^b + z_3^b + (1 - \sigma) b$$

$$\sigma LB_n^a \leq z_n^a \leq \sigma UB_n^a, \forall n = 1, 2, 3$$

$$(1 - \sigma) LB_n^b \leq z_n^b \leq (1 - \sigma) UB_n^b, \forall n = 1, 2, 3$$

Proposed formulation & observations

Tsay et al. [2021]

$$\sum_n \left(\sum_{i \in \mathbb{S}_n} w_i x_i - z_n^b \right) + \sigma b \leq 0$$

$$\sum_n z_n^b + (1-\sigma) b \geq 0$$

$$y = \sum_n z_n^b + (1-\sigma) b, \quad \sigma \in \{0, 1\}$$

$$\sigma LB_n^a \leq \sum_{i \in \mathbb{S}_n} w_i x_i - z_n^b \leq \sigma UB_n^a, \forall n = 1, \ldots, N$$

$$(1-\sigma) LB_n^b \leq z_n^b \leq (1-\sigma) UB_n^b, \forall n = 1, \ldots, N$$

Differences from big-M

Partition parameter N • Choose subsets $\mathbb{S}_1 \cup \ldots \cup \mathbb{S}_N = \{1, \ldots, \eta\}$; $\mathbb{S}_n \cap \mathbb{S}_{n'} = \emptyset$ $\forall n \neq n'$ • N new auxiliary variables z_n^b after eliminating z_n^a • $4N$ new constraints

Observations

$N = 1$ eqv. to big-M • $N = \eta$ eqv. to convex hull • Eqv. non-lifted formulation has 2^N constraints • Can have hierarchy of relaxations with increasing tightness/size

Tighten bounds LB_n^a, UB_n^a, LB_n^b, UB_n^b

Interval arithmetic? Optimization-based bounds tightening (OBBT)? [Tjeng et al., 2017, Dvijotham et al., 2018]

Proposed formulation & observations

Tsay et al. [2021]

$$\sum_n \left(\sum_{i \in \mathbb{S}_n} w_i x_i - z_n^b \right) + \sigma b \leq 0$$

$$\sum_n z_n^b + (1-\sigma) b \geq 0$$

$$y = \sum_n z_n^b + (1-\sigma) b, \quad \sigma \in \{0,1\}$$

$$\sigma LB_n^a \leq \sum_{i \in \mathbb{S}_n} w_i x_i - z_n^b \leq \sigma UB_n^a, \forall n = 1, \ldots, N$$

$$(1-\sigma) LB_n^b \leq z_n^b \leq (1-\sigma) UB_n^b, \forall n = 1, \ldots, N$$

Differences from big-M

Partition parameter N • Choose subsets $\mathbb{S}_1 \cup \ldots \cup \mathbb{S}_N = \{1, \ldots, \eta\}$; $\mathbb{S}_n \cap \mathbb{S}_{n'} = \emptyset$ $\forall n \neq n'$ • N new auxiliary variables z_n^b after eliminating z_n^a • $4N$ new constraints

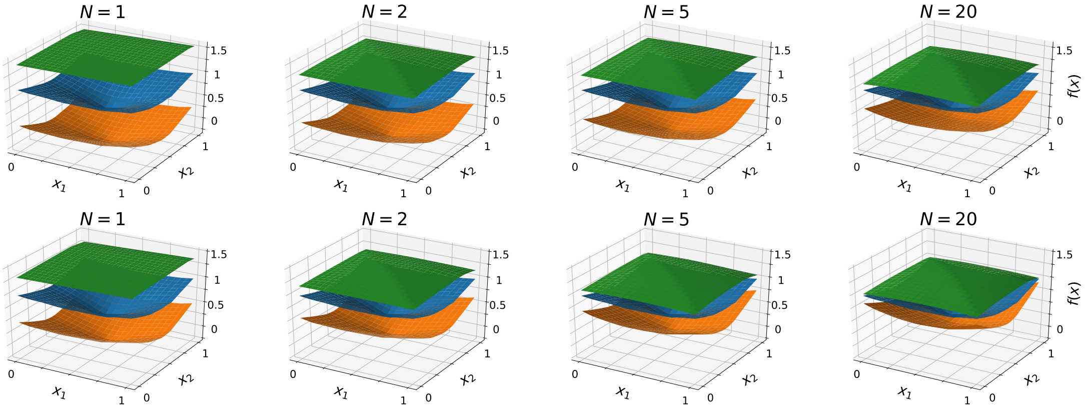
Observations

$N = 1$ eqv. to big-M • $N = \eta$ eqv. to convex hull • Eqv. non-lifted formulation has 2^N constraints • Can have hierarchy of relaxations with increasing tightness/size

Tighten bounds LB_n^a, UB_n^a, LB_n^b, UB_n^b

Interval arithmetic? Optimization-based bounds tightening (OBBT)? [Tjeng et al., 2017, Dvijotham et al., 2018]

Increasing relaxation tightness with increasing N

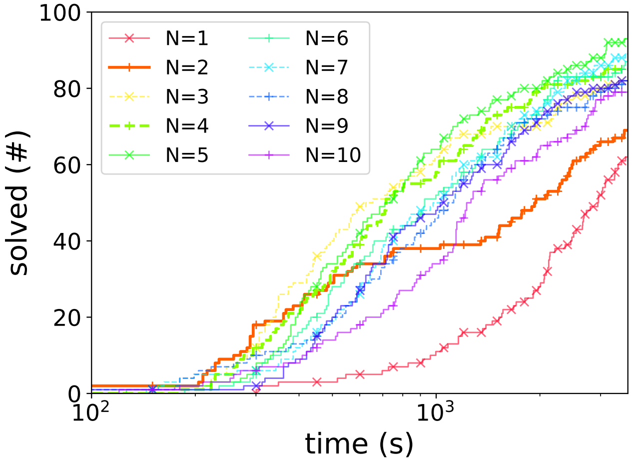


Example: two-input (x_1, x_2) NN trained on scaled Ackley function

$N = 1$ and $N = 20$ equiv. to big-M and convex hull, respectively. The z_n^b bounds were obtained using interval arithmetic (top row) and optimization-based bounds tightening (bottom row).

Importance of parameter N: Number solved vs. run time $\left(\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_1 = 5\right)$

Optimal adversary CIFAR-10; CNN with $n_{\text{Layers}} = 2$ & $n_{\text{Hidden}} = 100$; Each line averages 100 examples $\bar{\boldsymbol{x}}$



Observations

- $N = 1$ (equivalent to big-M) performs worst;
- $N = 2$ good for easy problems;
- Intermediate N balances model size vs. tightness;
- Performance declines $N \geq 7$

Effect of input partitioning choice

Consider $\boldsymbol{w} = [1, 1, 100, 100]$ and $x_i \in [0, 1], i = 1, ..., 4$

$$\begin{bmatrix} x_1 \leq \sigma 1 \\ x_2 + 100x_3 + 100x_4 \leq \sigma 201 \end{bmatrix} \text{or} \begin{bmatrix} x_1 + 100x_3 \leq \sigma 101 \\ x_2 + 100x_4 \leq \sigma 101 \end{bmatrix} \text{or} \begin{bmatrix} x_1 + x_2 \leq \sigma 2 \\ x_3 + x_4 \leq \sigma 2 \end{bmatrix}$$

Proposed partitioning: Want similar weights $\boldsymbol{w}$

- **Equal size** $|\mathbb{S}_1| = |\mathbb{S}_2| = ... = |\mathbb{S}_N|$ with weights in each partition as similar as possible
- **Equal range** $\underset{i \in \mathbb{S}_1}{\mathrm{range}}(w_i) = ... = \underset{i \in \mathbb{S}_N}{\mathrm{range}}(w_i)$, reduce outliers effect w/ 2 extra bins, $N \geq 3$

Weak partitioning: Dissimilar weights

- **Random partitions** Assign indices $\{1, ..., \eta\}$ randomly to partitions $\mathbb{S}_1, ..., \mathbb{S}_N$
- **Uneven magnitudes** Weights "dealt" in a snake-draft order by decreasing magnitude.

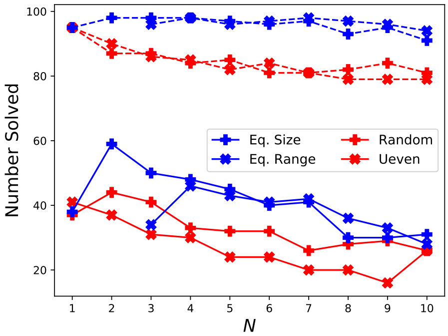
Experimental conditions [Tsay et al., 2021]

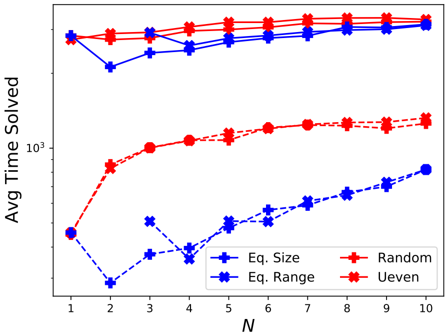
- 3.2 GHz Intel Core i7-8700 CPU (12 threads), 16 GB memory,
- Models implemented & solved: Gurobi v 9.1 [Gurobi Optimization, LLC, 2020],
- LP: Dual simplex, `cuts` 1, `TimeLimit` $3600s$, default termination criteria, `MIPFocus` 3,
- Trained NNs on MNIST [LeCun et al., 2010] & CIFAR-10 [Krizhevsky et al., 2009], Dense models: $n_{\text{Layers}} \times n_{\text{Hidden}}$ hidden & 10 output nodes. CNN2 from 'ConvSmall' [1]: {`Conv2D`(16, (4,4), (2,2)), `Conv2D`(32, (4,4), (2,2)), `Dense`(100), `Dense`(10)}. CNN1 halves channels in each layer: {`Conv2D`(8, (4,4), (2,2)), `Conv2D`(16, (4,4), (2,2)), `Dense`(100), `Dense`(10)}. CNN1/CNN2: 1,852/3,604 (MNIST) & 2,476/4,852 (CIFAR-10) nodes,
- NNs implemented in PyTorch [Paszke et al., 2019] using standard training (no regularization or methods to improve robustness).

[1] Based on ERAN dataset (`https://github.com/eth-sri/eran`)

Importance of partitioning strategy: Number Solved & Average Time Solved

Optimal adversary MNIST; CNN with $n_{\text{Layers}} = 2$ & $n_{\text{Hidden}} = 100$; Each line averages 100 runs; Max time 3600 s





Our partitioning strategies solve more problems faster than random & uneven partitions

Optimal adversary examples

Number solved (in 3600s) and average solve times for big-M vs equal-size partitions. Average times computed for problems solved by all 3 formulations

Dataset	Model	ϵ_1	Big-M		2 Partitions		4 Partitions	
			solved(#)	av.t (s)	solved(#)	av.t (s)	solved(#)	av.t (s)
MNIST	2×50	5	100	57.6	100	**42.7**	100	83.9
	2×50	10	97	431.8	98	**270.5**	98	398.4
	2×100	2.5	92	525.2	100	**285.1**	94	553.9
	2×100	5	32	1473.7	59	**494.6**	48	988.9
	CNN1*	0.25	68	1099.7	86	**618.8**	87	840.0
	CNN1*	0.5	2	2293.2	16	**1076.0**	11	2161.2
CIFAR-10	2×100	5	62	1982.3	69	1083.4	85	**752.8**
	2×100	10	23	2319.0	28	1320.2	34	**1318.1**

*OBBT performed on all NN nodes

Verification examples

Number solved (in 3600s) and average solve times for big-M vs equal-size partitions. Average times computed for problems solved by all 3 formulations. OBBT performed for all problems.

Dataset	Model	ϵ_∞	Big-M		2 Partitions		4 Partitions	
			solved(#)	av.t (s)	solved(#)	av.t (s)	solved(#)	av.t (s)
MNIST	CNN1	0.050	82	198.5	92	**27.3**	90	52.4
	CNN1	0.075	30	632.5	52	**139.6**	42	281.6
	CNN2	0.075	21	667.1	36	**160.7**	31	306.0
	CNN2	0.100	1	505.3	5	**134.7**	5	246.3
CIFAR-10	CNN1	0.007	99	100.6	100	**25.9**	99	45.4
	CNN1	0.010	98	85.1	100	**21.1**	100	37.5
	CNN2	0.010	40	2246.7	72	**859.9**	35	2449.4

ℓ_1-minimally distorted adversary examples

Number solved (in 3600s) and average solve times for big-M vs equal-size partitions. Average times and ϵ_1 are computed for problems solved by all 3 formulations. OBBT was performed for all problems.

Dataset	Model	$\mathrm{avg}(\epsilon_1)$	Big-M		2 Partitions		4 Partitions	
			solved(#)	av.t (s)	solved(#)	av.t (s)	solved(#)	av.t (s)
MNIST	2×50	6.51	52	675.0	93	**150.9**	89	166.6
	2×75	4.41	16	547.3	37	**310.5**	31	424.0
	2×100	2.73	7	710.8	13	**572.9**	10	777.9

Generalizing from NN: How to take steps *between* big-M & convex hull?

MI(NL)P with disjunctions

$$\bigvee_{l \in \mathcal{D}} \left[g_k(\boldsymbol{x}) \leq b_k \quad \forall k \in \mathcal{C}_l \right]$$

$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^{\eta},$$

$\mathcal{D}$: disjunct indices, $\mathcal{C}_l$: constraint indices, $\mathcal{X}$: convex compact set

Generalizing from NN: How to take steps *between* big-M & convex hull?

MI(NL)P with disjunctions

$$\bigvee_{l \in \mathcal{D}} \left[g_k(\boldsymbol{x}) \le b_k \quad \forall k \in \mathcal{C}_l \right]$$

$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^\eta,$$

$\mathcal{D}$: disjunct indices, $\mathcal{C}_l$: constraint indices, $\mathcal{X}$: convex compact set

Assumptions

1. Functions $g_k : \mathbb{R}^\eta \to \mathbb{R}$ convex additively separable: $g_k(\boldsymbol{x}) = \sum_{i=1}^{\eta} h_{ik}(x_i)$ where $h_{ik} : \mathbb{R} \to \mathbb{R}$ are convex functions, and each disjunct is non-empty on $\mathcal{X}$.
2. Functions g_k are bounded over $\mathcal{X}$,
3. Far fewer constraints than # variables in each disjunction: $|\mathcal{C}_l| << \eta$.

Generalizing from NN: How to take steps *between* big-M & convex hull?

MI(NL)P with disjunctions

$$\bigvee_{l \in \mathcal{D}} \left[g_k(\boldsymbol{x}) \le b_k \quad \forall k \in \mathcal{C}_l \right]$$

$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^\eta,$$

$\mathcal{D}$: disjunct indices, $\mathcal{C}_l$: constraint indices, $\mathcal{X}$: convex compact set

Applications?

Clustering [Papageorgiou and Trespalacios, 2018, Sağlam et al., 2006] • **Mixed-integer classification** [Liittschwager and Wang, 1978, Rubin, 1997] • **Coverage optimization** [Huang and Tseng, 2005] • **P_ball** [Kronqvist and Misener, 2020] • **ReLU NN**

Assumptions

1. Functions $g_k : \mathbb{R}^\eta \to \mathbb{R}$ convex additively separable: $g_k(\boldsymbol{x}) = \sum_{i=1}^{\eta} h_{ik}(x_i)$ where $h_{ik} : \mathbb{R} \to \mathbb{R}$ are convex functions, and each disjunct is non-empty on $\mathcal{X}$.
2. Functions g_k are bounded over $\mathcal{X}$,
3. Far fewer constraints than # variables in each disjunction: $|\mathcal{C}_l| << \eta$.

N-split representation of original disjunction [Kronqvist et al., 2021]

Lift with $N \times |\mathcal{D}|$ new variables, Relax splitted constraints to global constraints, Feasible x space doesn't change

$$\bigvee_{l \in \mathcal{D}} \left[\sum_{i=1}^{\eta} h_{il}(x_i) \leq b_l \right]$$
$$\boldsymbol{x} \in \mathcal{X}$$

N-split representation of original disjunction [Kronqvist et al., 2021]

Lift with $N \times |\mathcal{D}|$ new variables, Relax splitted constraints to global constraints, Feasible x space doesn't change

$$\bigvee_{l \in \mathcal{D}} \left[\sum_{i=1}^{\eta} h_{il}(x_i) \le b_l \right] \qquad \longrightarrow \qquad \bigvee_{l \in \mathcal{D}} \begin{bmatrix} \sum_{i \in \mathcal{I}_1} h_{il}(x_i) \le \alpha_1^l \\ \vdots \\ \sum_{i \in \mathcal{I}_N} h_{il}(x_i) \le \alpha_N^l \\ \sum_{s=1}^{N} \alpha_s^l \le b_l \\ \underline{\alpha}_s^l \le \alpha_s^l \le \bar{\alpha}_s^l \\ \forall s \in \{1, \dots, N\} \end{bmatrix}$$

$$\boldsymbol{x} \in \mathcal{X}$$

$$\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\alpha}^l \in \mathbb{R}^N \ \forall \ l \in \mathcal{D},$$

$$\underline{\alpha}_s^l := \min_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i),$$

$$\bar{\alpha}_s^l := \max_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i).$$

N-split representation of original disjunction [Kronqvist et al., 2021]

Lift with $N \times |\mathcal{D}|$ new variables, Relax splitted constraints to global constraints, Feasible $\boldsymbol{x}$ space doesn't change

$$\bigvee_{l \in \mathcal{D}} \left[\sum_{i=1}^{\eta} h_{il}(x_i) \le b_l \right] \quad \boldsymbol{x} \in \mathcal{X}$$

$$\longrightarrow$$

$$\bigvee_{l \in \mathcal{D}} \begin{bmatrix} \sum_{i \in \mathcal{I}_1} h_{il}(x_i) \le \alpha_1^l \\ \vdots \\ \sum_{i \in \mathcal{I}_N} h_{il}(x_i) \le \alpha_N^l \\ \sum_{s=1}^{N} \alpha_s^l \le b_l \\ \underline{\alpha}_s^l \le \alpha_s^l \le \bar{\alpha}_s^l \\ \qquad \forall s \in \{1, \ldots, N\} \end{bmatrix}$$

$$\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\alpha}^l \in \mathbb{R}^N \ \forall\, l \in \mathcal{D},$$

$$\underline{\alpha}_s^l := \min_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i),$$

$$\bar{\alpha}_s^l := \max_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i).$$

$$\longrightarrow$$

$$\bigvee_{l \in \mathcal{D}} \begin{bmatrix} \sum_{s=1}^{N} \alpha_s^l \le b_l \\ \underline{\alpha}_s^l \le \alpha_s^l \le \bar{\alpha}_s^l \\ \qquad \forall s \in \{1, \ldots, N\} \end{bmatrix}$$

$$\sum_{i \in \mathcal{I}_s} h_{il}(x_i) \le \alpha_s^l$$

$$\forall s \in \{1, \ldots, N\},\ l \in \mathcal{D},$$

$$\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\alpha}^l \in \mathbb{R}^N \ \forall\, l \in \mathcal{D},$$

$$\underline{\alpha}_s^l := \min_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i),$$

$$\bar{\alpha}_s^l := \max_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{I}_s} h_{il}(x_i).$$

N-split formulation

Represent the convex hull of the N-split disjunction using extended formulation [Balas, 1998]

$$\alpha_s^l = \sum_{d \in \mathcal{D}} \nu_d^{\alpha_s^l} \qquad \forall\ s \in \{1, \dots, N\},\ l \in \mathcal{D}$$

$$\sum_{s=1}^{N} \nu_l^{\alpha_s^l} \leq b_l \lambda_l \qquad \forall\ l \in \mathcal{D}$$

$$\underline{\alpha}_s^l \lambda_d \leq \nu_d^{\alpha_s^l} \leq \bar{\alpha}_s^l \lambda_d \qquad \forall\ s \in \{1, \dots, N\},\ l, d \in \mathcal{D}$$

$$\sum_{i \in \mathcal{I}_s} h_{il}(x_i) \leq \alpha_s^l \qquad \forall\ s \in \{1, \dots, N\}, l \in \mathcal{D}$$

$$\sum_{l \in \mathcal{D}} \lambda_l = 1, \quad \boldsymbol{\lambda} \in \{0, 1\}^{|\mathcal{D}|}$$

$$\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\alpha}^l \in \mathbb{R}^N,\ \boldsymbol{\nu}^{\alpha_s^l} \in \mathbb{R}^N \qquad \forall\ s \in \{1, \dots, N\},\ l \in \mathcal{D}$$

Properties

- Feasible $\boldsymbol{x}$ space stays same,
- 1-split equiv. big-M,
- If h affine, η-split equiv. convex hull,
- If h affine, can get big-M to convex hull relaxation hierarchy.

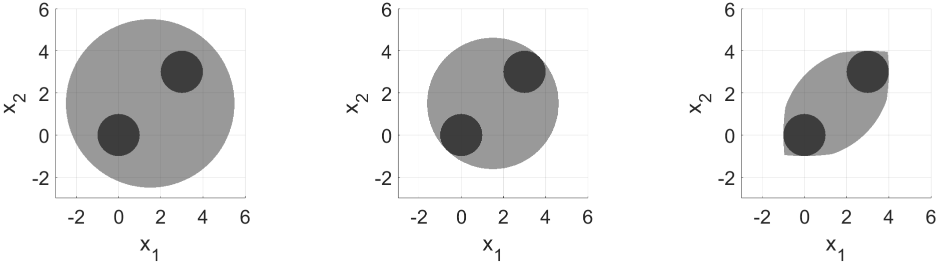
More in Kronqvist et al. [2021]

N-splits also may be useful with nonlinear functions

Illustrative example

$$\left[\sum_{i=1}^{4} x_i^2 \leq 1\right] \vee \left[\sum_{i=1}^{4} (3 - x_i)^2 \leq 1\right]$$

$$\boldsymbol{x} \in \mathbb{R}^4$$



1-split/big-M ($\{x_1, x_2, x_3, x_4\}$)

2-split ($\{x_1, x_2\}, \{x_3, x_4\}$)

4-split ($\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}$)

instance		big-M	2-split	4-split	8-split	16-split	32-split	convex hull
Cluster_g1	time	>1800	81.0	13.9	2.9	**1.7**	3.5	42.0
	nodes	>8998	2946	1096	256	98	91	**73**
Cluster_g2	time	>1800	106.3	7.7	4.3	**2.1**	4.5	40.6
	nodes	>10431	1736	481	217	104	86	**77**
Cluster_g3	time	>1800	>1800	870.6	**407.2**	597.5	NA	>1800
	nodes	>28906	>40820	19307	**14923**	16806		>7797
P_ball_1	time	403.0	235.4	285.1	**18.5**	NA	NA	42.2
	nodes	29493	7919	5518	2202			**1437**
P_ball_2	time	>1800	483.6	326.6	41.6	30.6	NA	**28.2**
	nodes	>19622	13602	5871	3921	1261		**531**
P_ball_3	time	>1800	>1800	>1800	149.3	91.1	**78.7**	114.0
	nodes	>7537	>6035	>6708	7042	3572	631	**554**
		big-M	14-split	28-split	56-split	196-split	392-split	convex hull
Cluster_m1	time	>1800	>1800	129.5	76.8	**32.0**	33.2	313.3
	nodes	>10680	>9651	2926	1462	524	**195**	228
Cluster_m2	time	>1800	1116.5	156.1	**27.1**	97.0	54.2	1260.1
	nodes	>4867	6220	1915	805	2752	1155	**131**
Cluster_m3	time	>1800	>1800	429.5	60.0	23.2	**19.8**	>1800
	nodes	>4419	>4197	3095	1502	741	**397**	>93

Between steps: Partition-based formulations

Summary

- New relaxations of disjunctions intermediate to big-M & convex hull, form a hierarchy under additional assumptions,
- Introduce parameter N (# splits) and aggregation choice (how to group variables), parameter choices robust across many instances,
- ReLU NN: Solve 25% more problems in 1 hr, average 2.2X speedup for solved problems,
- Promising computational results for other classes of problems.

Want to know more?

- **Papers** `https://arxiv.org/abs/2102.04373` • `https://arxiv.org/abs/2101.12708`
- **Twitter** @CogImperial • @CalvinTsay • @JanKronqvist • @AThebelt • @RuthMisener
- **YouTube** `https://www.youtube.com/channel/UCXRdjQRm9XfZj2c4XW1xpzg`

References I

Michael Akintunde, Alessio Lomuscio, Lalit Maganti, and Edoardo Pirovano. Reachability analysis for neural agent-environment systems. In *KR*, pages 184–193, 2018.

Ross Anderson, Joey Huchette, Will Ma, Christian Tjandraatmadja, and Juan Pablo Vielma. Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming*, pages 1–37, 2020.

Egon Balas. Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, 89(1-3):3–44, 1998.

Egon Balas. *Disjunctive Programming*. Springer International Publishing, 2018. doi: 10.1007/978-3-030-00148-3.

Elena Botoeva, Panagiotis Kouvaros, Jan Kronqvist, Alessio Lomuscio, and Ruth Misener. Efficient verification of ReLU-based neural networks via dependency analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3291–3299, 2020.

References II

Chih-Hong Cheng, Georg Nührenberg, and Harald Ruess. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 251–268. Springer, 2017.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020.

Alessandro De Palma, Harkirat Singh Behl, Rudy Bunel, Philip HS Torr, and M Pawan Kumar. Scaling the convex barrier with active sets. *arXiv preprint arXiv:2101.05844*, 2021.

Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods Symposium*, pages 121–138. Springer, 2018.

References III

Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *UAI*, volume 1, page 3, 2018.

Matteo Fischetti and Jason Jo. Deep neural networks and mixed integer linear optimization. *Constraints*, 23(3):296–309, 2018.

Bjarne Grimstad and Henrik Andersson. ReLU networks as surrogate models in mixed-integer linear programs. *Computers & Chemical Engineering*, 131:106580, 2019.

Gurobi Optimization, LLC. Gurobi optimizer reference manual, 2020. URL `http://www.gurobi.com`.

Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. *Mobile networks and Applications*, 10(4):519–528, 2005.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

References IV

Jan Kronqvist and Ruth Misener. A disjunctive cut strengthening technique for convex MINLP. *Optimization and Engineering*, pages 1–31, 2020.

Jan Kronqvist, Ruth Misener, and Calvin Tsay. Between steps: Intermediate relaxations between big-M and convex hull formulations. *arXiv preprint arXiv:2101.12708*, 2021.

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

JM Liittschwager and C Wang. Integer programming solution of a classification problem. *Management Science*, 24(14):1515–1525, 1978.

Alessio Lomuscio and Lalit Maganti. An approach to reachability analysis for feed-forward ReLU neural networks. *arXiv preprint arXiv:1706.07351*, 2017.

Dimitri J Papageorgiou and Francisco Trespalacios. Pseudo basic steps: bound improvement guarantees from Lagrangian decomposition in convex disjunctive programming. *EURO Journal on Computational Optimization*, 6(1):55–83, 2018.

References V

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

Paul A Rubin. Solving mixed integer classification problems by decomposition. *Annals of Operations Research*, 74:51–64, 1997.

Burcu Sağlam, F Sibel Salman, Serpil Sayın, and Metin Türkay. A mixed-integer programming approach to the clustering problem with an application in customer segmentation. *European Journal of Operational Research*, 173(3):866–879, 2006.

Thiago Serra, Christian Tjandraatmadja, and Srikumar Ramalingam. Bounding and counting linear regions of deep neural networks. In *International Conference on Machine Learning*, pages 4558–4566. PMLR, 2018.

References VI

Thiago Serra, Abhinav Kumar, and Srikumar Ramalingam. Lossless compression of deep neural networks. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 417–430. Springer, 2020.

Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Calvin Tsay, Jan Kronqvist, Alexander Thebelt, and Ruth Misener. Partition-based formulations for mixed-integer optimization of trained relu neural networks. *arXiv preprint arXiv:2102.04373*, 2021.

Ga Wu, Buser Say, and Scott Sanner. Scalable planning with deep neural network learned transition models. *Journal of Artificial Intelligence Research*, 68:571–606, 2020.