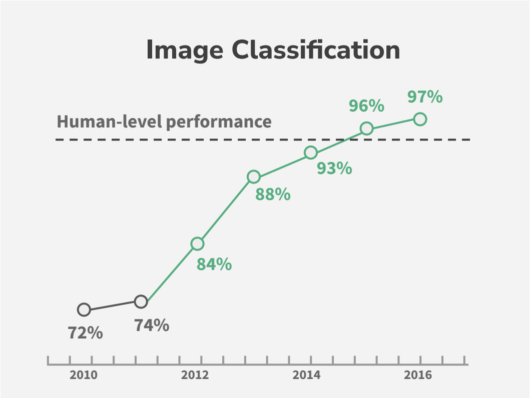
PRIMA: General and Precise Neural Network Certification via Scalable Convex Hull Approximations





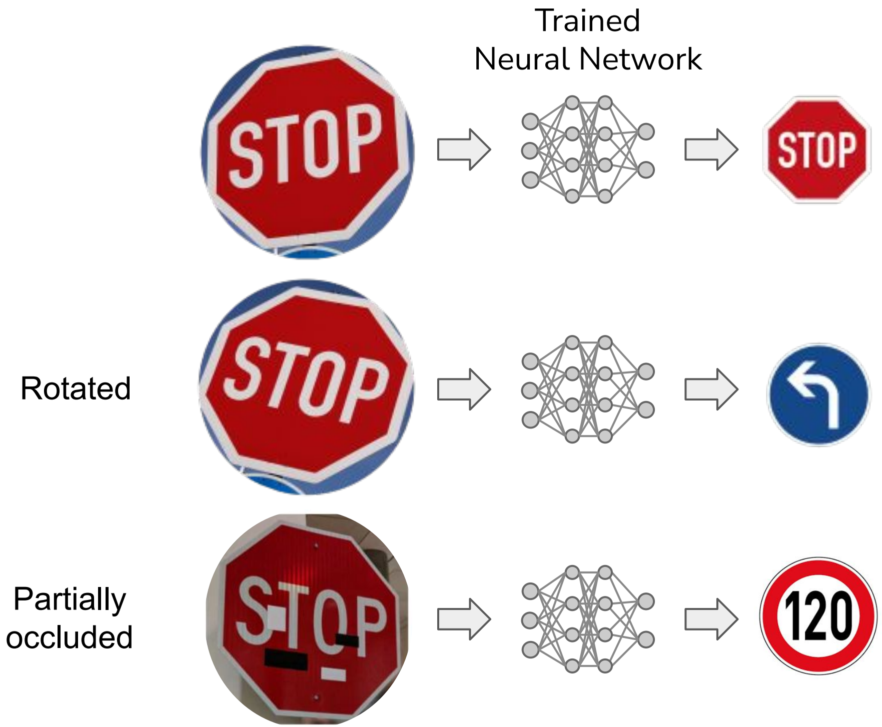
Mark Niklas Müller   Gleb Makarchuk   Gagandeep Singh   Markus Püschel   Martin Vechev

ETH zürich   UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN   SRILAB



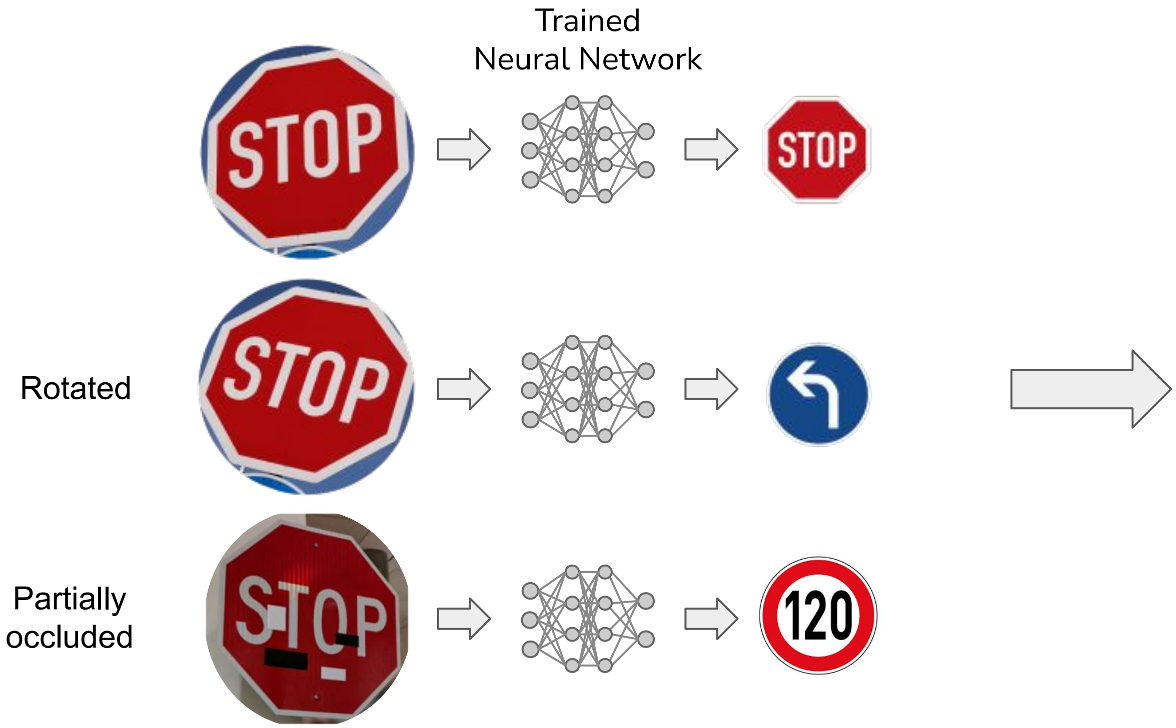
Standard accuracy exceeding human performance

Neural Networks Lack Trustworthiness

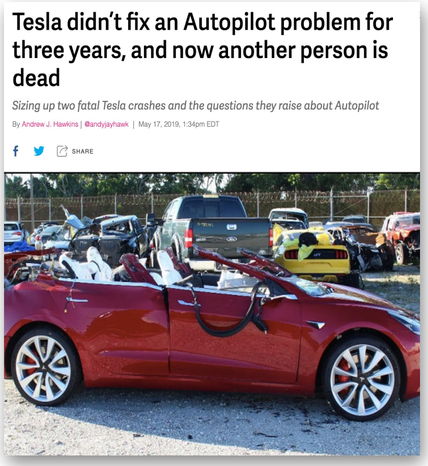


Fischer, Marc, Maximilian Baader, and Martin Vechev. "Certified defense to image transformations via randomized smoothing." [CoRR 2019]
Eykholt, Kevin, et al. "Robust physical-world attacks on deep learning visual classification." [IEEE 2018]

Neural Networks Lack Trustworthiness



Car fatalities



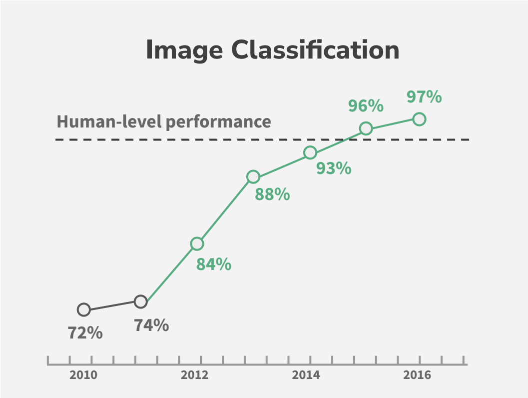
Tesla didn't fix an Autopilot problem for three years, and now another person is dead

Sizing up two fatal Tesla crashes and the questions they raise about Autopilot

By Andrew J. Hawkins | @andyjayhawk | May 17, 2019, 1:34pm EDT

SHARE

Fischer, Marc, Maximilian Baader, and Martin Vechev. "Certified defense to image transformations via randomized smoothing." [CoRR 2019]
Eykholt, Kevin, et al. "Robust physical-world attacks on deep learning visual classification." [IEEE 2018]

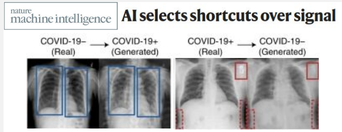


Car fatalities

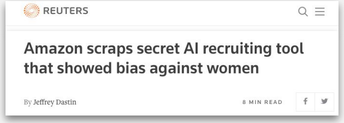

Tesla didn't fix an Autopilot problem for three years, and now another person is dead

Sizing up two fatal Tesla crashes and the questions they raise about Autopilot

Misdiagnosed patients



Unfair models


REUTERS

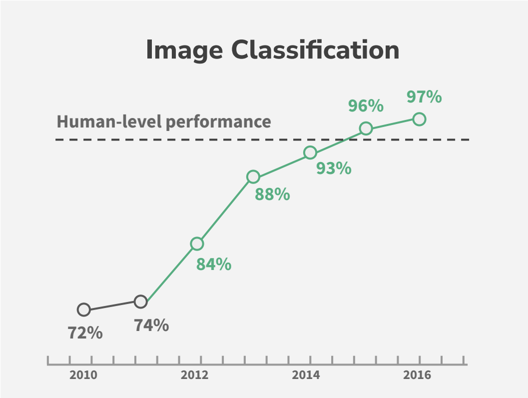
Amazon scraps secret AI recruiting tool that showed bias against women

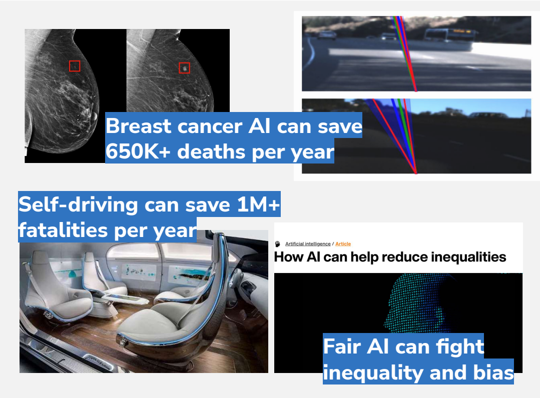
By Jeffrey Dastin

8 MIN READ

Standard accuracy exceeding human performance

Standard accuracy is not enough for real-world AI



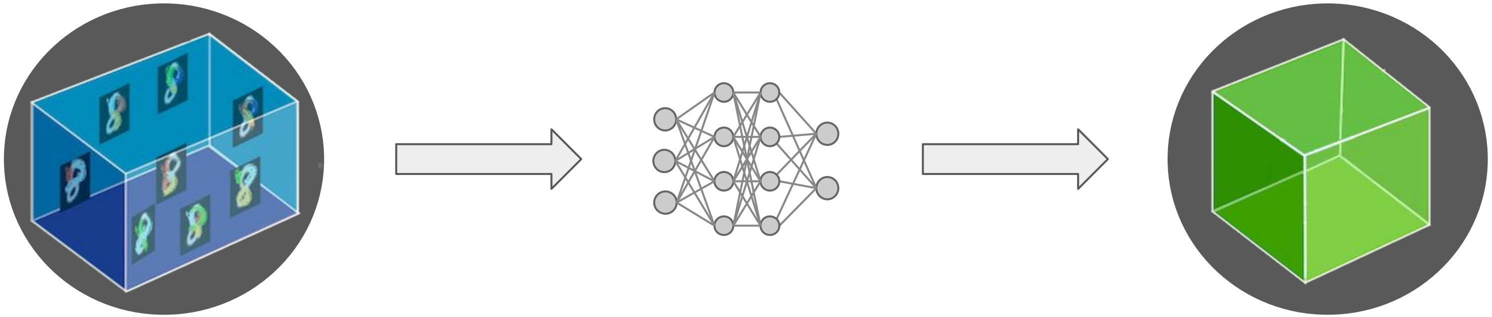


Standard accuracy exceeding human performance

Standard accuracy is not enough for real-world AI

We need certifiably trustworthy AI

Problem Statement: Neural Network Verification



Precondition over
Network Inputs

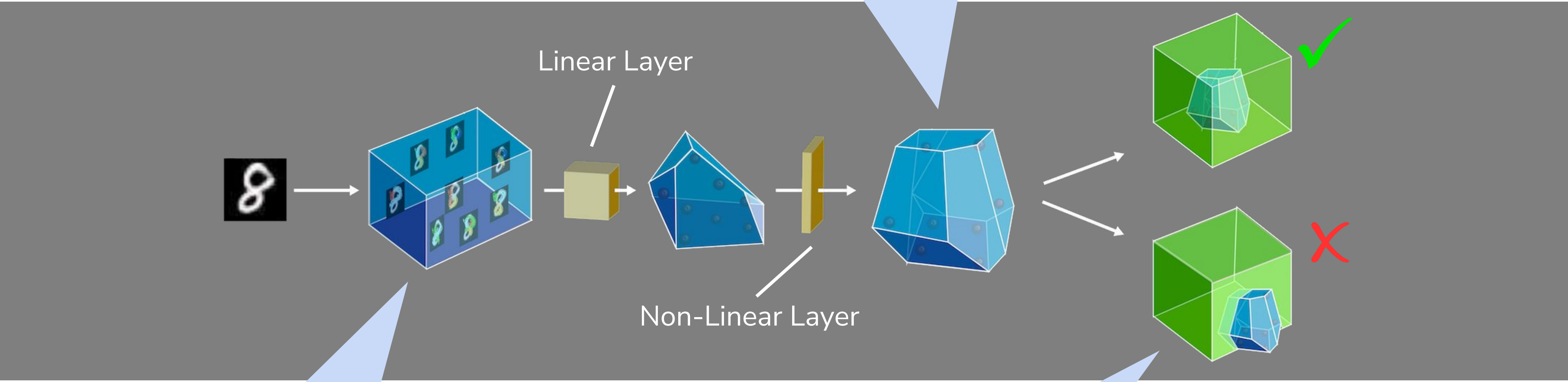
Trained
Neural Network

Postcondition over
Network Outputs

Prove that postcondition holds for all inputs satisfying the precondition

Neural Network Verification via Abstract Interpretation

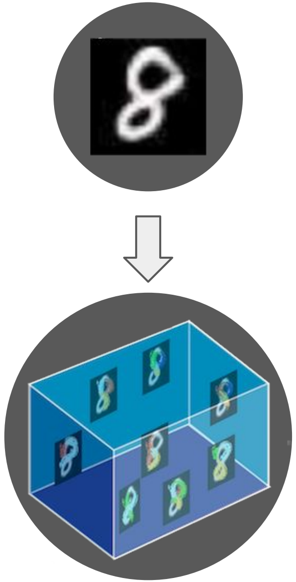
Convex over-approximation of network output



Precondition encoded in convex abstraction

Reason over postcondition

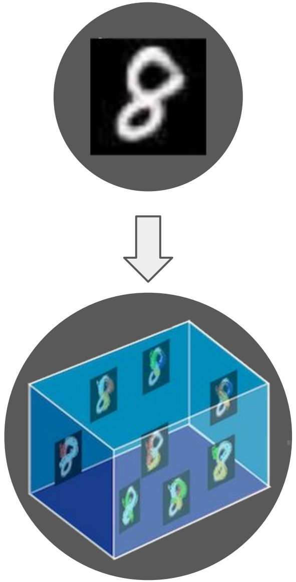
Example Specification



Robustness to ℓ_∞-norm bounded perturbations:

$$y = \arg\max_i h(\mathbf{x})_i = \arg\max_i h(\mathbf{x}')_i, \quad \forall \mathbf{x}' \in B^\epsilon(\mathbf{x})$$

Example Specification



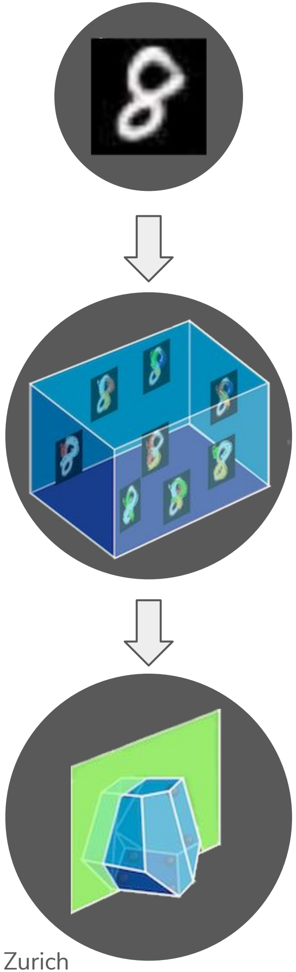
Robustness to ℓ_∞-norm bounded perturbations:

$$y = \arg\max_i h(\mathbf{x})_i = \arg\max_i h(\mathbf{x}')_i, \quad \forall \mathbf{x}' \in B^\epsilon(\mathbf{x})$$

Equivalently:

$$\min_{x' \in B^\epsilon(\mathbf{x})} h(\mathbf{x}')_y - h(\mathbf{x}')_i > 0, \quad \forall i \neq y$$

Example Specification



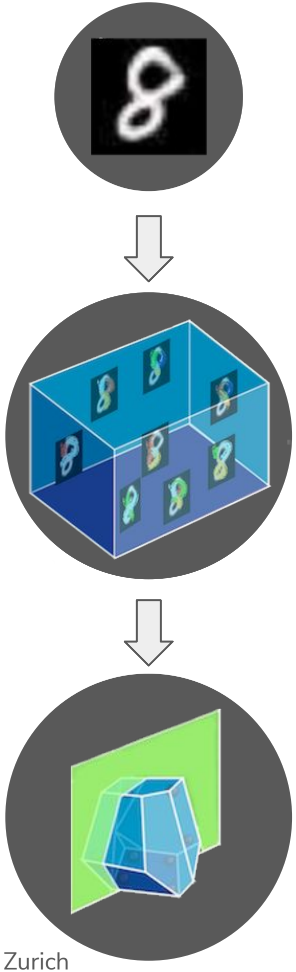
Robustness to ℓ_∞-norm bounded perturbations:

$$y = \arg\max_i h(\mathbf{x})_i = \arg\max_i h(\mathbf{x}')_i, \quad \forall \mathbf{x}' \in B^\epsilon(\mathbf{x})$$

Equivalently:

$$\underline{h(\mathbf{x}')_y - h(\mathbf{x}')_i} > 0, \quad \forall i \neq y$$

Example Specification



Robustness to ℓ_∞-norm bounded perturbations:

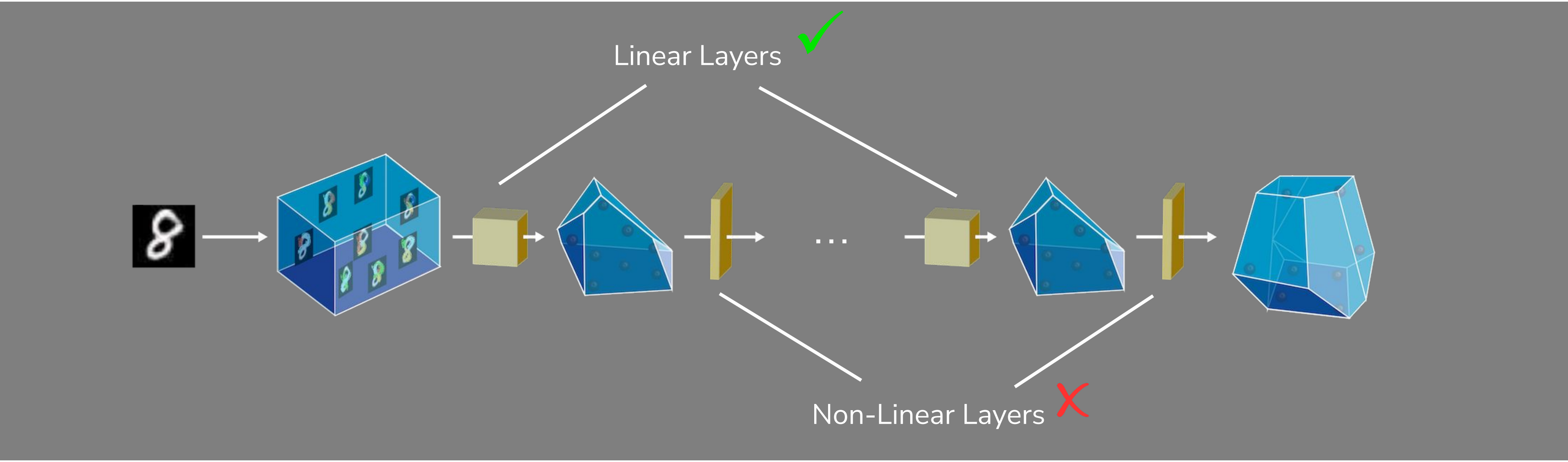
$$y = \arg\max_i h(\mathbf{x})_i = \arg\max_i h(\mathbf{x}')_i, \quad \forall \mathbf{x}' \in B^\epsilon(\mathbf{x})$$

Equivalently:

$$\underline{h(\mathbf{x}')_y - h(\mathbf{x}')_i} > 0, \quad \forall i \neq y$$

Encode polyhedral abstraction as LP

Main Challenge



Main Challenge: Abstracting Non-Linear Activations

The Neural Network Verification Race

Reluplex [Katz et al. ICCAV 2017]

IBP [Gowal et al. CoRR2018]

AI2 [Gehr et. al IEEE S&P 2018]

RefinePoly [Singh et al. ICLR 2018]

SDP-cert [Raghunathan et al. NeurIPS 2018]

DeepZ [Singh et al. NeurIPS 2018]

CROWN [Zhang et al. NeurIPS 2018]

DeepPoly [Singh et al. POPL 2019]

MIPVerify [Tjeng et al. ICLR 2019]

k-Poly [Singh et al. NeurIPS 2019]

hBox [Mirman et al. CoRR 2019]

Marabou [Katz et al. ICCAV 2019]

BaB [Bunel JMLR 2020]

OptC2V [Tjandraatmadja et al. NeurIPS 2020]

GNN BaB [Jaeckle arXiv 2021]

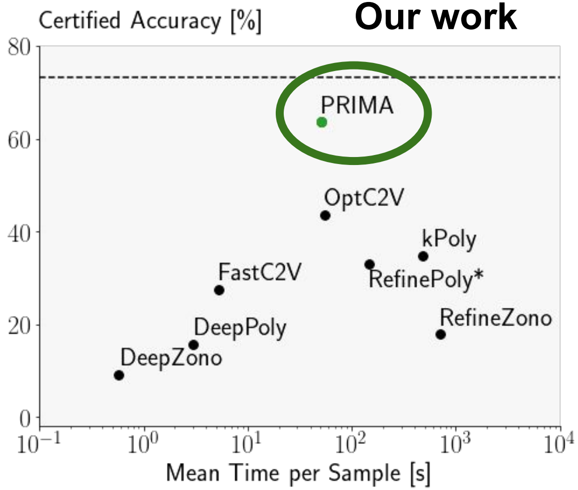
Fast and Complete [Xu et al. ICLR2021]

Beta-Crown [Wang et al. NeurIPS 2021]

DeepSplit [Henricksen IJCAI2021]

~200 papers / year

Our work



PRIMA: Precise abstractions at reasonable runtime

Abstracting Non-Linearities

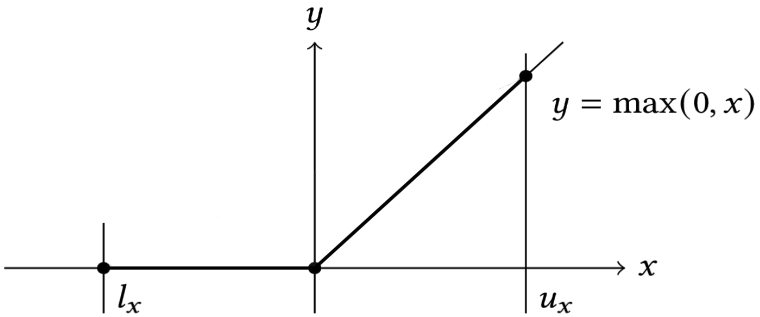
1. Key Contribution: Efficient Multi-Neuron Abstractions

2. Key Contribution: Approximate Convex Hull Computation

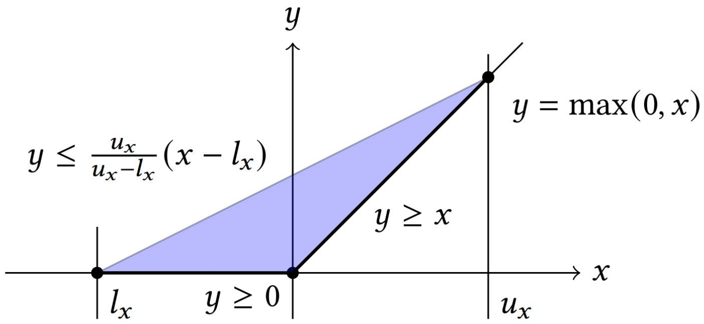
Empirical Evaluation

Abstracting Non-Linear Activations

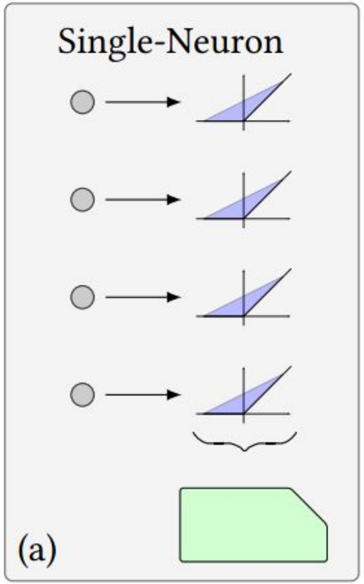
ReLU Activation



Convex Abstraction



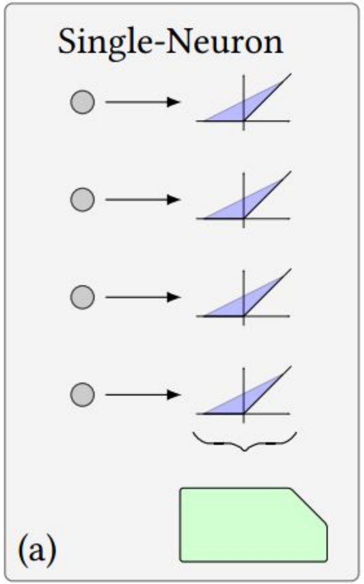
Abstracting Non-Linear Activations



Triangle
Relaxation

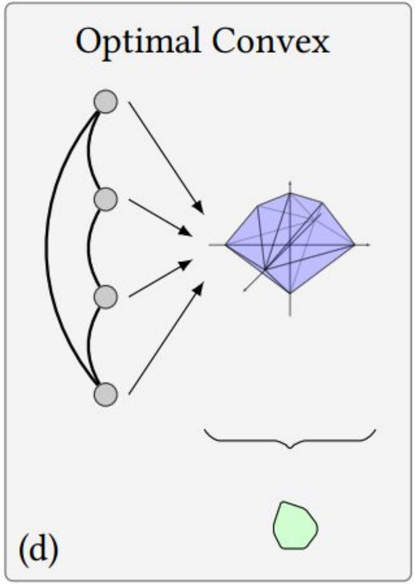
Single Neuron
Convex Barrier
[Salman et al. 2019]

Abstracting Non-Linear Activations

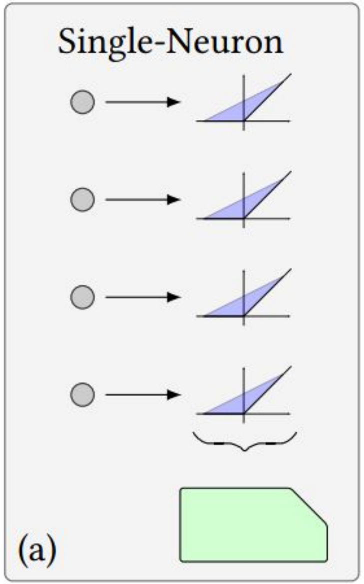


Triangle Relaxation

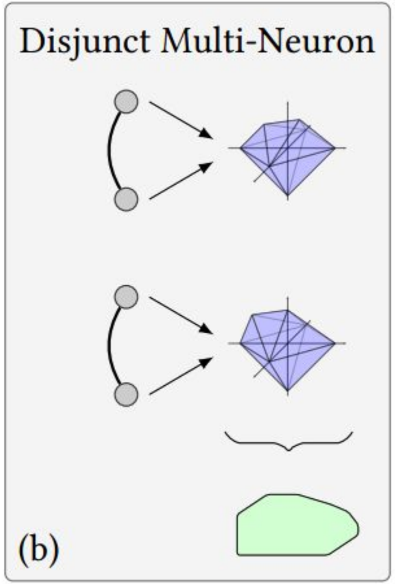
Exponential complexity in number of neurons
⇒ Intractable



Abstracting Non-Linear Activations

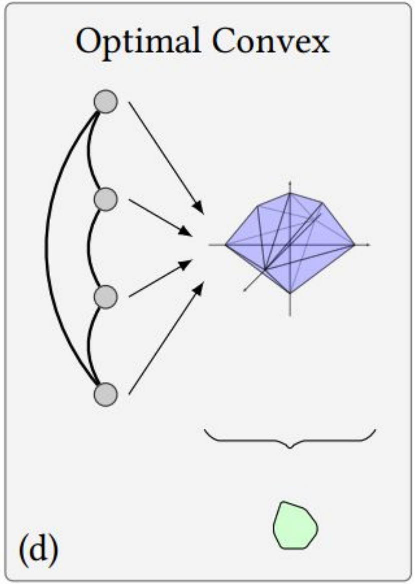


Triangle
Relaxation

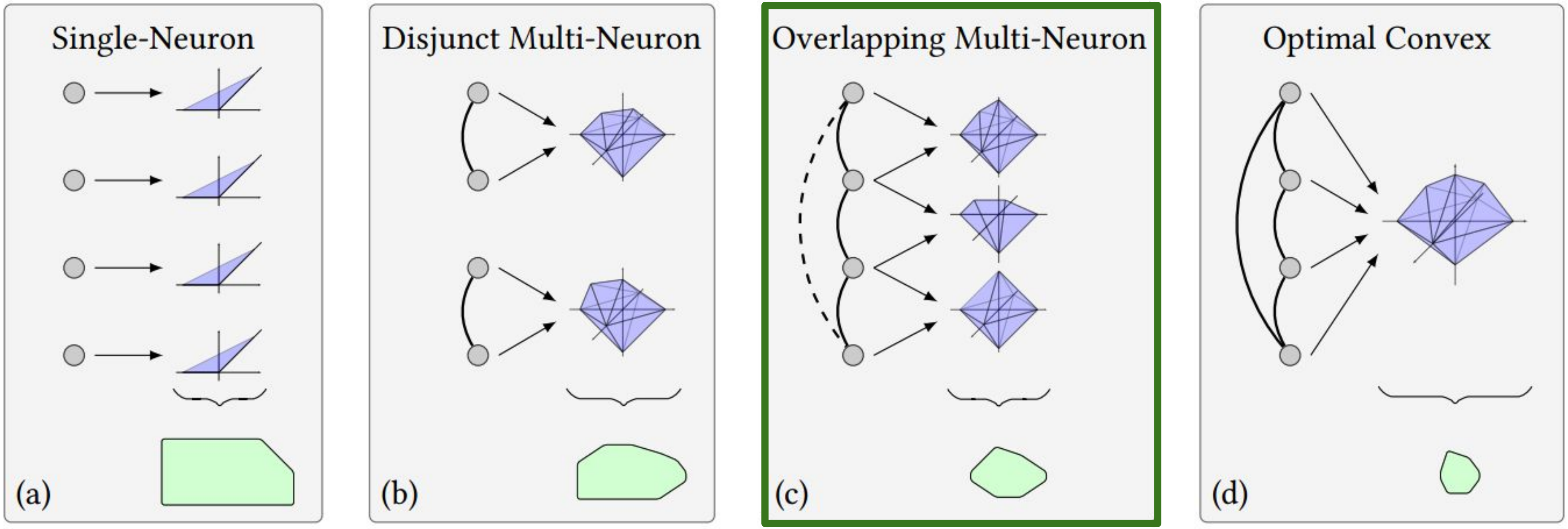


k-Poly
[Singh et al. 2019]

Expensive convex
hull computation
⇒ Few groups



Abstracting Non-Linear Activations



Triangle Relaxation

k-Poly [Singh et al. 2019]

PRIMA (this work)

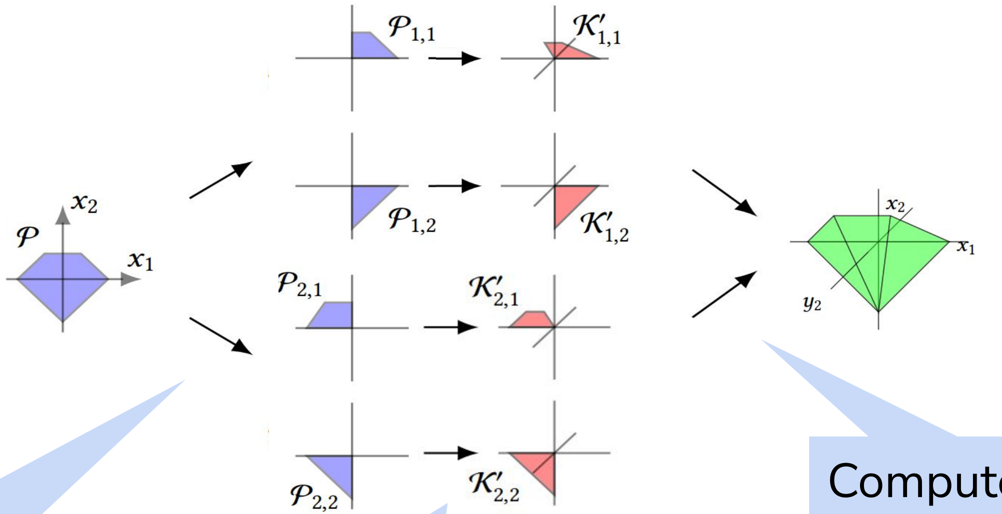
Abstracting Non-Linearities

1. Key Contribution: Efficient Multi-Neuron Abstractions

2. Key Contribution: Approximate Convex Hull Computation

Empirical Evaluation

Abstracting Multiple Neurons Jointly

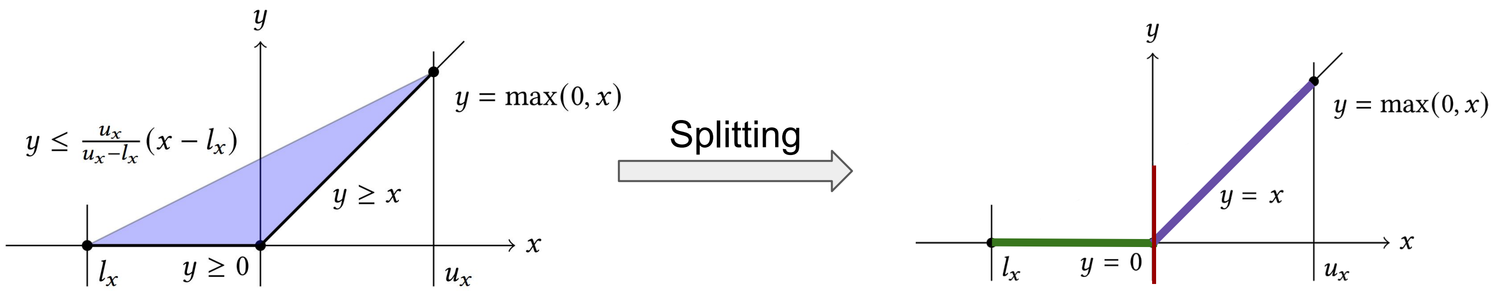


Partition input-space into smaller regions

Bound tightly on these smaller regions

Compute the convex hull

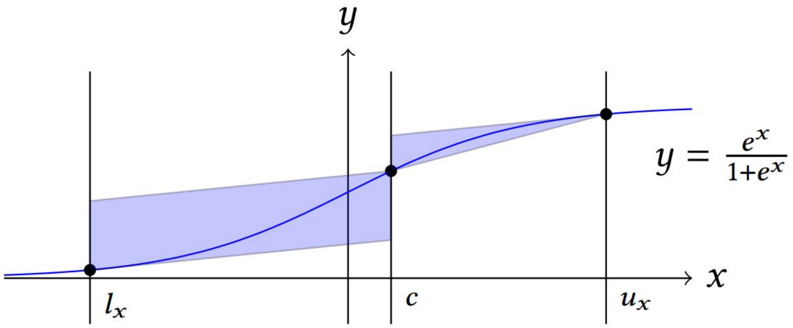
Multi-Neuron Abstractions for ReLU



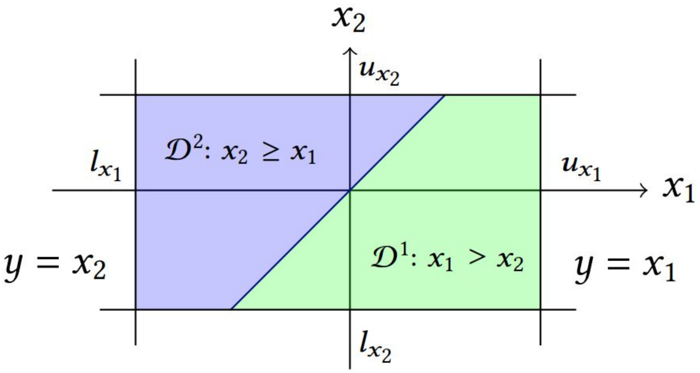
Multi-Neuron Abstractions Beyond ReLU

- Split into arbitrary regions
- Bound tightly on these regions

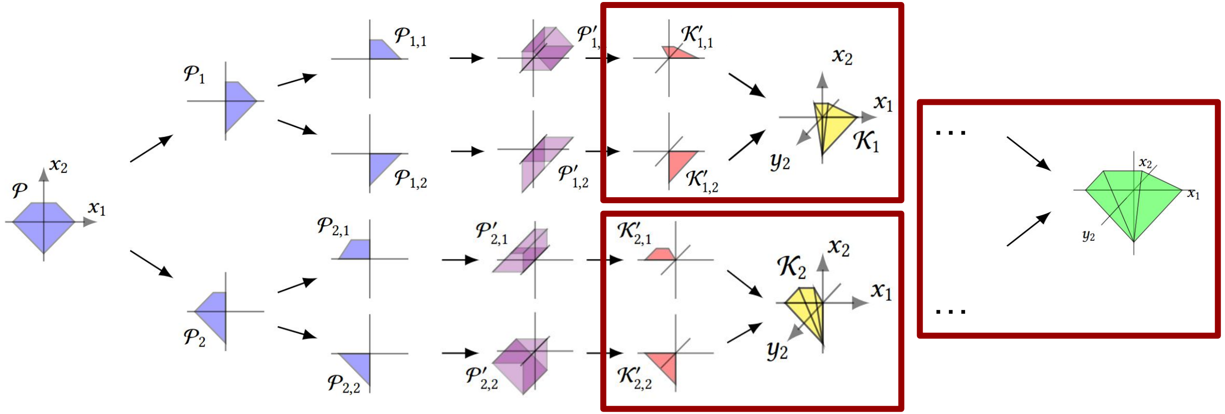
Sigmoid



MaxPool $y = \max(x_1, x_2)$



The Split Bound Lift Method (SBLM)



	# convex hulls (general)	for groups of k = 3 neurons
k-ReLU [Singh et al. 2019]	2^k-1 (2k)-dimensional	7 x 6d
SBLM (ours)	2^{k-i} (k+i)-dimensional for i ∈ [1, k]	4 x 4d, 2 x 5d, 1 x 6d

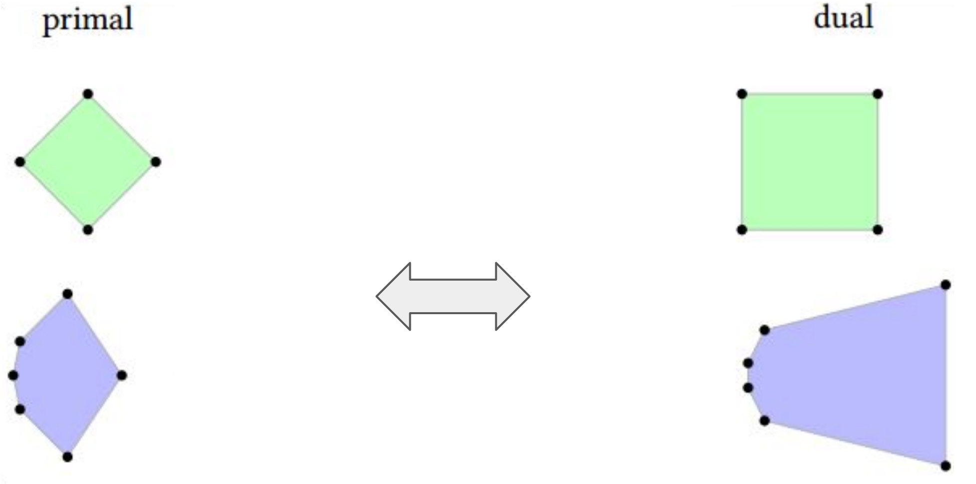
Abstracting Non-Linearities

1. Key Contribution: Efficient Multi-Neuron Abstractions

2. Key Contribution: Approximate Convex Hull Computation

Empirical Evaluation

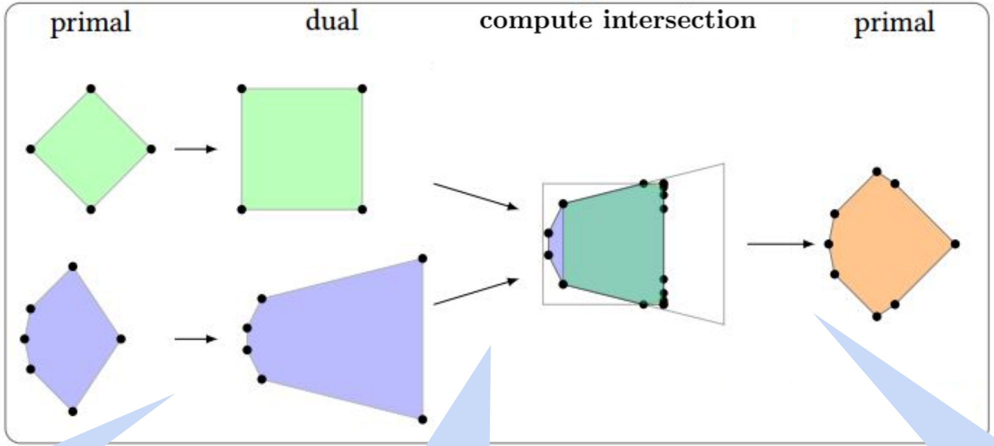
Duals of Polyhedra



Over-Approximate
Convex Hull

Under-Approximate
Intersection

Convex Hull Computation



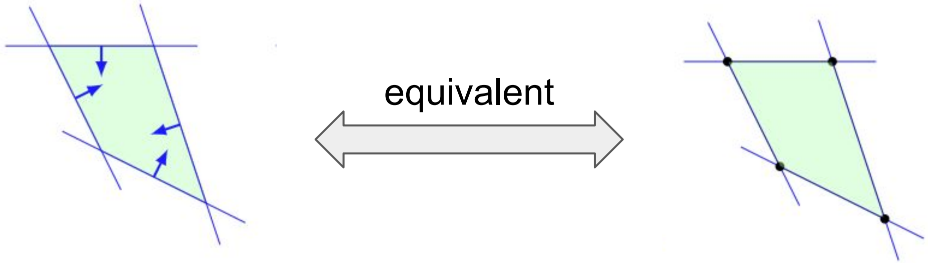
Translate into dual space

Compute under-approximation of intersection

Translate back into primal space

Polyhedra Representation

Double Description

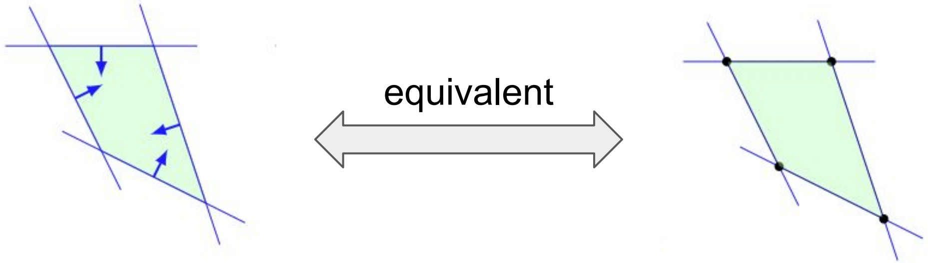


Supporting halfspaces
⇒ H-representation

Generating vertices
⇒ V-representation

Polyhedra Representation

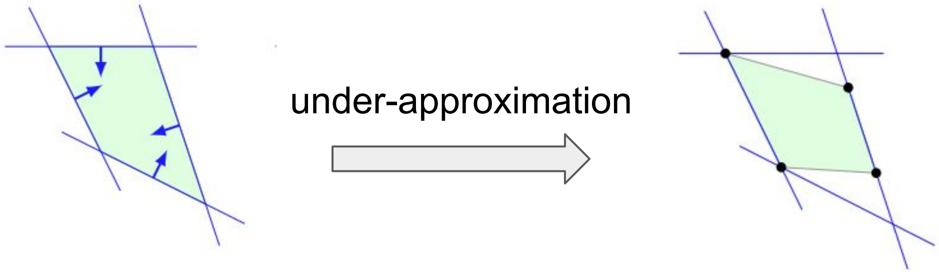
Double Description



Supporting halfspaces
$\Rightarrow$ H-representation

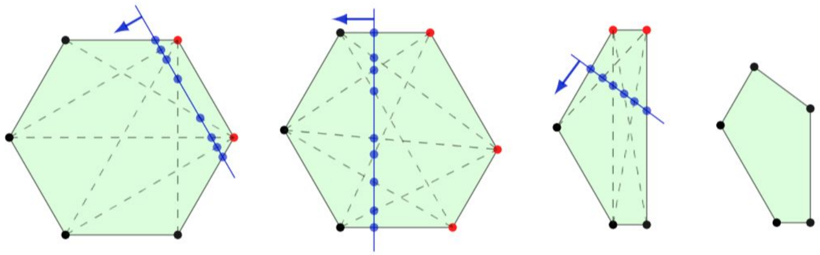
Generating vertices
$\Rightarrow$ V-representation

Partial Double Description



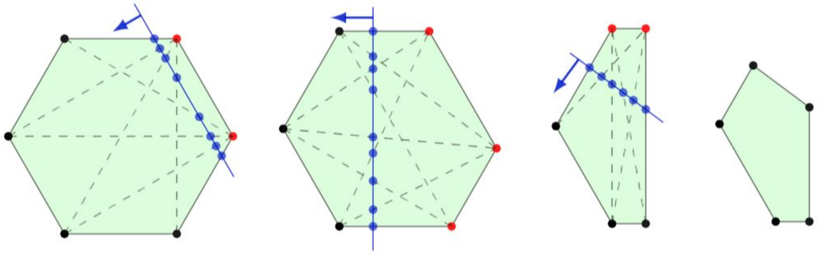
Batch Intersection

Sequential intersection (DDM)

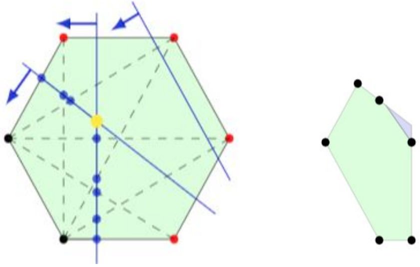


Batch Intersection

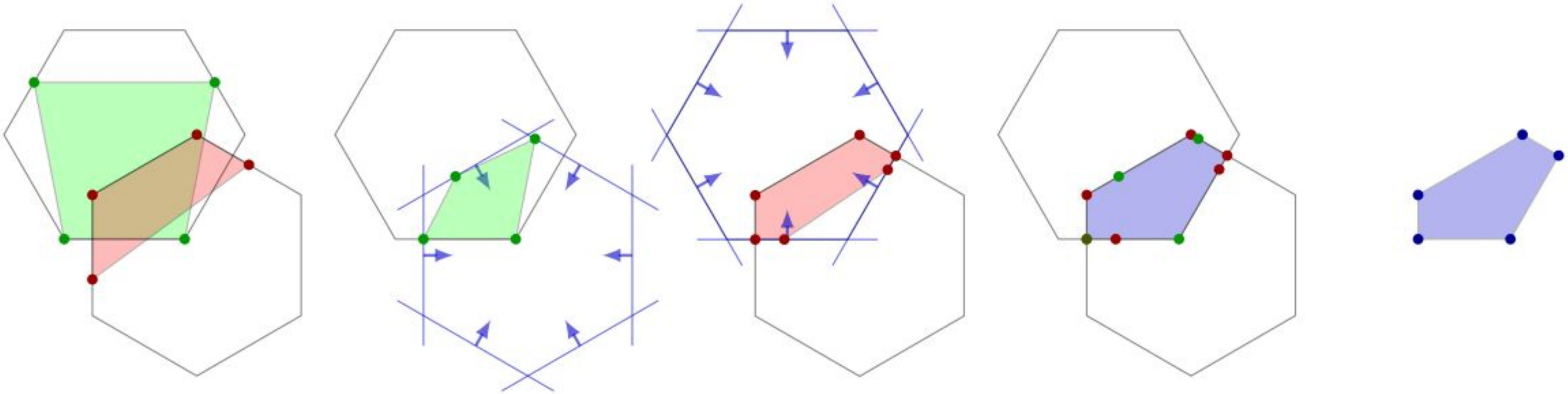
Sequential intersection (DDM)



Batch intersection (PDDM)

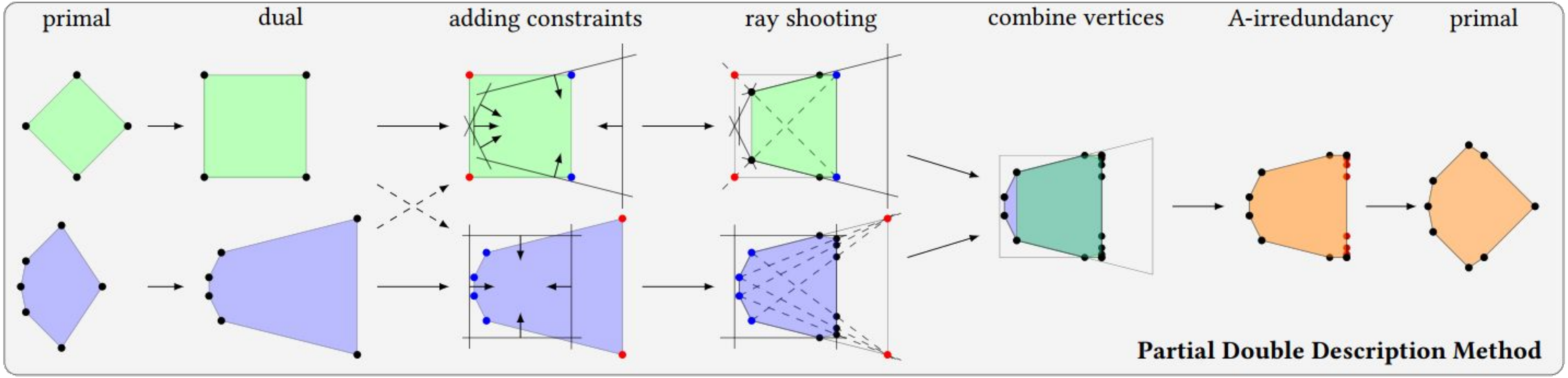


Boosting Precision



Approximate Input Polytopes | First Intersection | Second Intersection | Union | Exact Intersection

The Partial Double Description Method (PDDM)

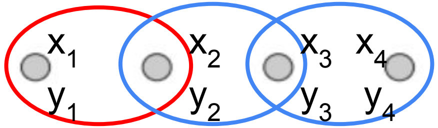


PDDM : A general method for over-approximate convex hull

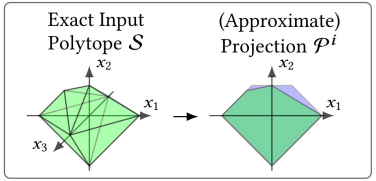
- Exact for small (≤ 3) dimensions
- Complexity: $\mathcal{O}(n_v \cdot n_a^4 + n_a^2 \log(n_a^2))$ instead of $\mathcal{O}(n_v \log(n_v) + n_v^{\lfloor d/2 \rfloor})$

Integrating Multi-Neuron Constraints in Certification

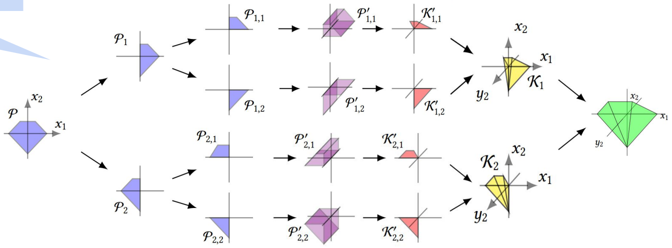
Decompose activations into overlapping groups



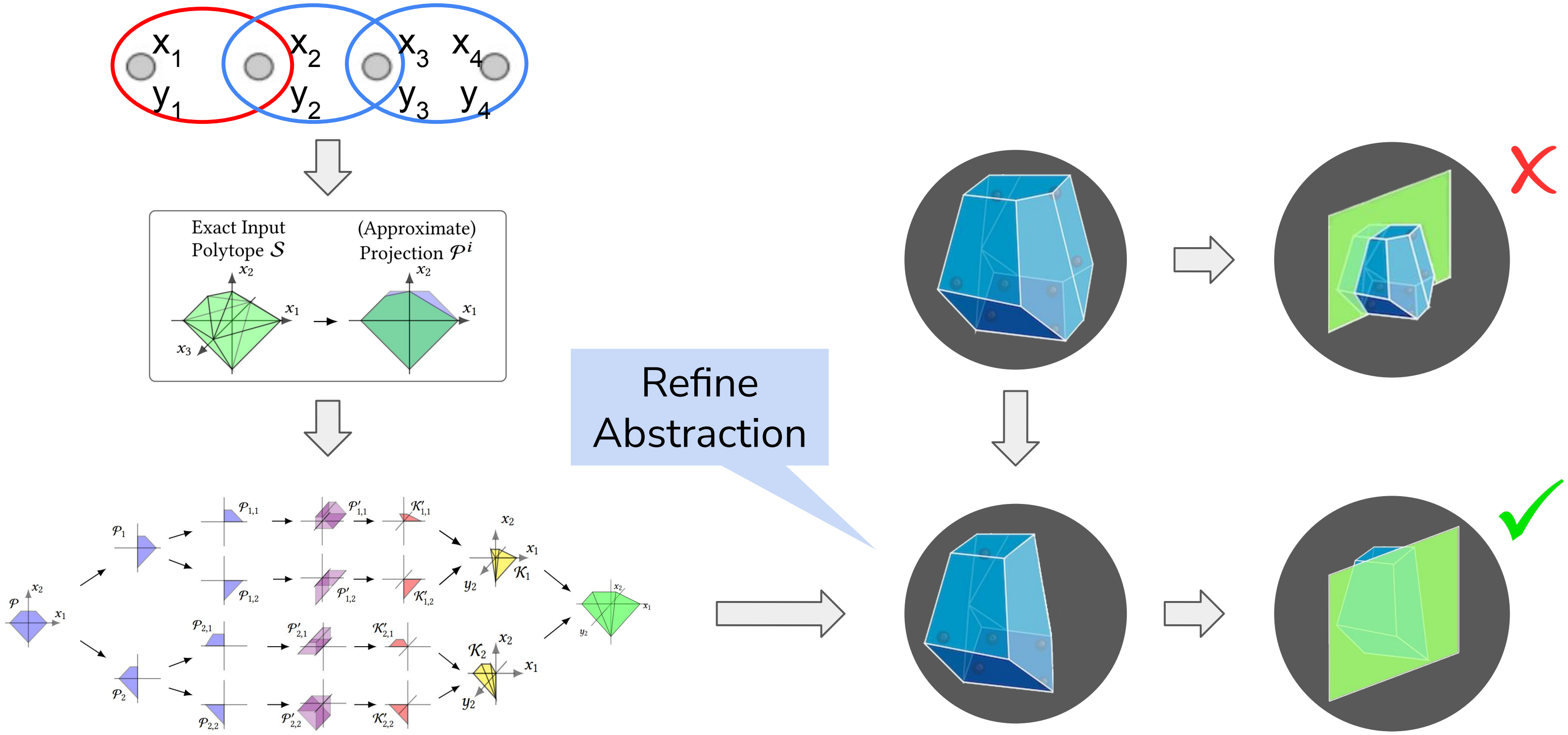
Over-approximate input space per group



Compute Multi-Neuron Abstraction



Integrating Multi-Neuron Constraints in Certification



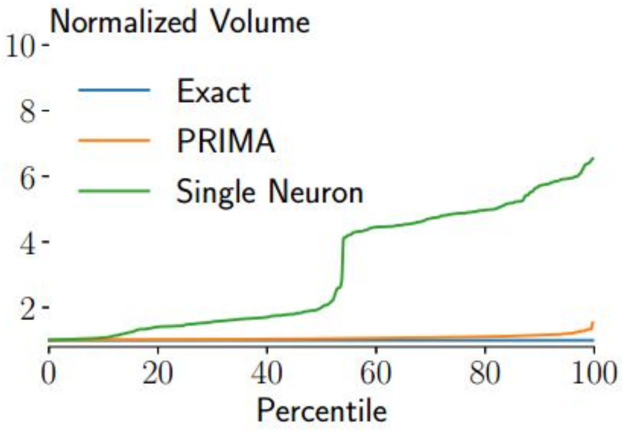
Abstracting Non-Linearities

1. Key Contribution: Efficient Multi-Neuron Abstractions

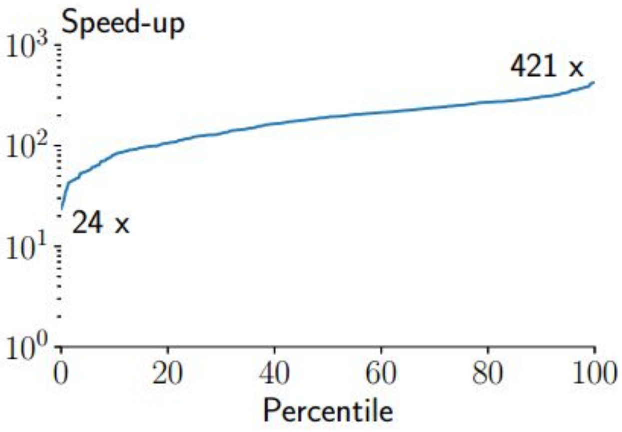
2. Key Contribution: Approximate Convex Hull Computation

Empirical Evaluation

Empirical Performance of Approximate Convex Hull



Precise
only 5% volume increase



Fast
200x speed-up

Verification Performance

Dataset	Model	Training	# Neurons	# Layers	Accuracy	ϵ	n_s	kPoly		OptC2V †		Prima (ours)		# Upper Bound
								# Ver	Time	# Ver	Time	# Ver	Time	
MNIST	5 × 100	NOR	510	5	960	0.026	100	441	307	429	137	**510**	159	842
	8 × 100	NOR	810	8	947	0.026	100	369	171	384	759	**428**	301	820
	5 × 200	NOR	1 010	5	972	0.015	50	574	187	601	403	**690**	224	901
	8 × 200	NOR	1 610	8	950	0.015	50	506	464	528	3451	**612**	395	911
	`ConvSmall`	NOR	3 604	3	980	0.120	100	347	477	436	55	**640**	51	733
	`ConvBig`	DiffAI	48 064	6	929	0.300	100	736	40	771	102	**775**	5.5	790
CIFAR10	`ConvSmall`	PGD	4 852	3	630	2/255	100	399	86	398	105	**458**	16	481
	`ConvBig`	PGD	62 464	6	631	2/255	100	459	346	n/a†	n/a†	**482**	128	550
	`ResNet`	Wong	107 496	10	290	8/255	50	245	91	n/a†	n/a†	**248**	1.9	248

† The OptC2V [Tjandraatmadja et al. 2020] code has not been released; we report their runtimes and results where available.

Verification Performance

Dataset	Model	Training	# Neurons	# Layers	Accuracy	ϵ	n_s	KPoly		OptC2V †		PRIMA (ours)		# Upper Bound
								# Ver	Time	# Ver	Time	# Ver	Time	
MNIST	5 × 100	NOR	510	5	960	0.026	100	441	307	429	137	**510**	159	842
	8 × 100	NOR	810	8	947	0.026	100	369	171	384	759	**428**	301	820
	5 × 200	NOR	1 010	5	972	0.015	50	574	187	601	403	**690**	224	901
	8 × 200	NOR	1 610	8	950	0.015	50	506	464	528	3451	**612**	395	911
	`ConvSmall`	NOR	3 604	3	980	0.120	100	347	477	436	55	**640**	51	733
	`ConvBig`	DiffAI	48 064	6	929	0.300	100	736	40	771	102	**775**	5.5	790
CIFAR10	`ConvSmall`	PGD	4 852	3	630	2/255	100	399	86	398	105	**458**	16	481
	`ConvBig`	PGD	62 464	6	631	2/255	100	459	346	n/a†	n/a†	**482**	128	550
	`ResNet`	Wong	107 496	10	290	8/255	50	245	91	n/a†	n/a†	**248**	1.9	248

†The OptC2V [Tjandraatmadja et al. 2020] code has not been released; we report their runtimes and results where available.

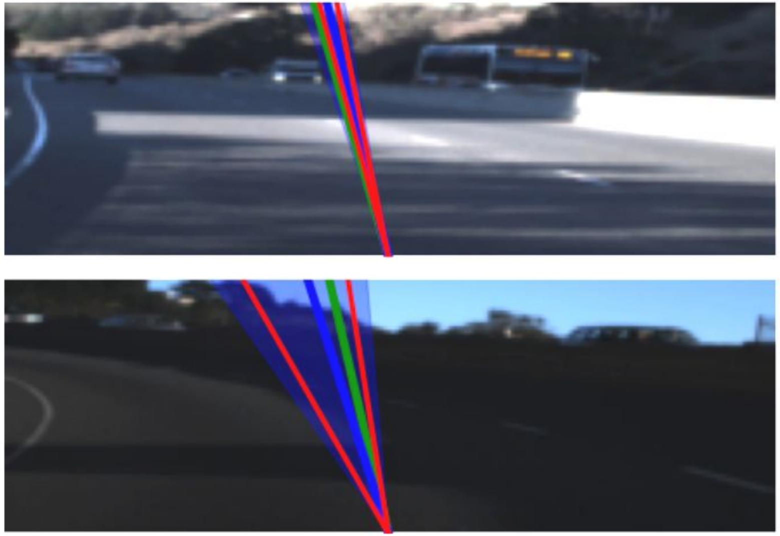
Verification Performance

Dataset	Model	Training	# Neurons	# Layers	Accuracy	ϵ	n_s	kPoly		OptC2V †		PRIMA (ours)		# Upper Bound
								# Ver	Time	# Ver	Time	# Ver	Time	
MNIST	5×100	NOR	510	5	960	0.026	100	441	307	429	137	**510**	159	842
	8×100	NOR	810	8	947	0.026	100	369	171	384	759	**428**	301	820
	5×200	NOR	1 010	5	972	0.015	50	574	187	601	403	**690**	224	901
	8×200	NOR	1 610	8	950	0.015	50	506	464	528	3451	**612**	395	911
	`ConvSmall`	NOR	3 604	3	980	0.120	100	347	477	436	55	**640**	51	733
	`ConvBig`	DiffAI	48 064	6	929	0.300	100	736	40	771	102	**775**	5.5	790
CIFAR10	`ConvSmall`	PGD	4 852	3	630	2/255	100	399	86	398	105	**458**	16	481
	`ConvBig`	PGD	62 464	6	631	2/255	100	459	346	n/a†	n/a†	**482**	128	550
	`ResNet`	Wong	107 496	10	290	8/255	50	245	91	n/a†	n/a†	**248**	1.9	248

† The OptC2V [Tjandraatmadja et al. 2020] code has not been released; we report their runtimes and results where available.

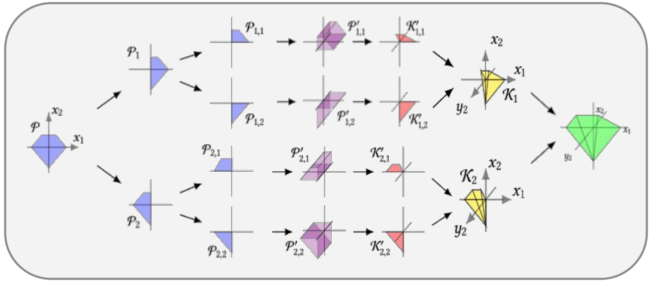
Application: Verification for Autonomous Driving

- Udacity autonomous driving dataset for **steering angle prediction**
- CNN architecture from NVIDIA
- 107k Neurons and 27M connections in 8 layers
- Input size (3 x 66 x 200)
- ℓ_∞-norm bounded perturbations with $\epsilon = 2/255$
- 260s per image

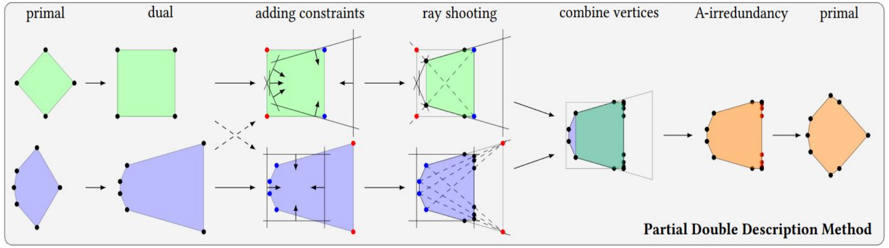


Green: target
Blue: certified region with PRIMA
Red: empirical bounds

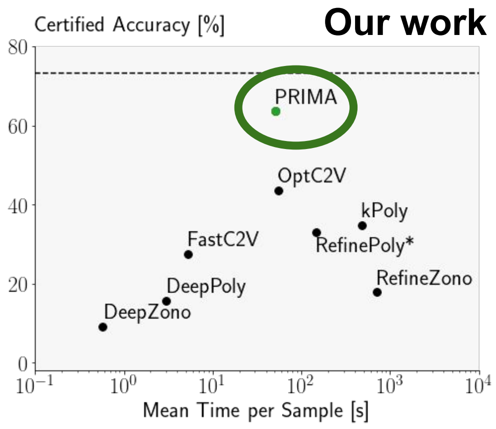
Summary



SBLM – General multi-neuron abstractions



PDDM – General convex hull approximation



Thank you for your attention!

Questions? – Contact us: mark.mueller@inf.ethz.ch

Code available on GitHub: https://github.com/eth-sri/eran