Special Meeting 2

Neural Network Verification by Abstract Interpretation

Yi-Nung Tsao
yi-nung.tsao@uni.lu

Boss:
Grégoire Danoy, Pierre Talbot

June 20, 2025

Outline

- Progress Review
- Neural network verification
- Backward-Forward Analysis (BFA)
- Next steps

Progress Review

Paper submission journey

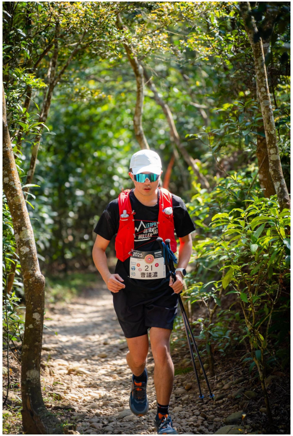
- 31st Jan, 2025: First paper submission to International Conference on Computer Aided Verification (CAV) 2025.
- 28th Mar, 2025, Rejected by CAV 2025. 😢

> @A1 Mar 28
>
> After careful deliberations, the reviewers agree that although the approach proposed in this paper is interesting and deals with an important and timely problem, the experimental evaluation is not yet mature enough to allow the paper to be accepted at this time. We encourage the authors to improve this point and resubmit the paper.

- 11th May, 2025: Second submission attempt of the same paper to International Static Analysis Symposium (SAS) 2025.
- 16th - 18th June, 2025: Author response from SAS 2025. 😵‍💫

Trail running journey

Feb - Mar, 2025: Trail running training in Taiwan. 🏃



Neural Network Verification

Neural networks are widely used in many applications

- Public Safety and Security
- Image and Video Recognition
- Medical Diagnosis
- ...

But, neural networks are vulnerable to adversarial examples

An **adversarial example** is a correctly classified input with small noise that causes the neural networks to produce an incorrect result despite the modifed input appearing normal to humans.

stop sign
Confidence: 0.9153

Adversarial perturbation

flowerpot
Confidence: 0.8374

To ensure the reliability of neural networks

A **neural network** consists of *an input layer, multiple hidden layers, and an output layer* where each layer is made up of several neurons.

Definition: Layer

Let n_ℓ be the number of neurons in the layer ℓ. Then a layer function $N_\ell \colon \mathbb{R}^{n_\ell} \to \mathbb{R}^{n_{\ell+1}}$ is defined as follows:

$$N_\ell(\boldsymbol{x}) \triangleq \sigma(\boldsymbol{W}_\ell \boldsymbol{x} + \boldsymbol{b}_\ell),$$

where σ is an activation function.

Definition: Neural Network

Let $\boldsymbol{L}$ be the index set of layers. A neural network is a function $N \colon \mathbb{R}^{d_{in}} \to \mathbb{R}^{d_{out}}$ defined as follows:

$$N \triangleq N_{|\boldsymbol{L}|} \circ N_{|\boldsymbol{L}|-1} \circ \ldots \circ N_1,$$

where d_{in} and d_{out} is the dimension of the input and output layer, respectively.

To ensure the reliability of neural networks

Definition: Preconditions

The preconditions in the input layer are defined by the set $\Phi(\boldsymbol{x}_0, \epsilon) \triangleq \{\boldsymbol{x} \in \mathbb{R}^{d_{in}} \mid p(\boldsymbol{x}, \boldsymbol{x}_0) \leq 0\}$, where $p \colon \mathbb{R}^{d_{in}} \times \mathbb{R}^{d_{in}} \to \mathbb{R}$ is a function defining a perturbation and $\epsilon \in \mathbb{R}$ is the maximum perturbation.

Origin Image

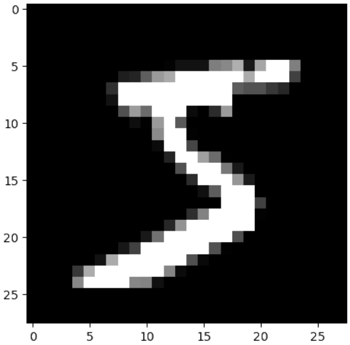
L infinity

Rotation

To ensure the reliability of neural networks

Definition: Postconditions

Let $y_i = N(\boldsymbol{x}_0)_i$ be the output value of the neuron i in the output layer. The postconditions in the output layer are defined by the set of predicates
$\boldsymbol{\Psi}^{|\boldsymbol{L}|} \triangleq \{\sum_{i=1}^{n_{|\boldsymbol{L}|}} w_{ji} y_i + b_j \geq 0 \mid \forall j \in \{1, \ldots, n_{|\boldsymbol{L}|-1}\} b_j \in \mathbb{R}$ and $\forall i \in \{1, \ldots, n_{|\boldsymbol{L}|}\} w_{ji} \in \mathbb{R}\}$.

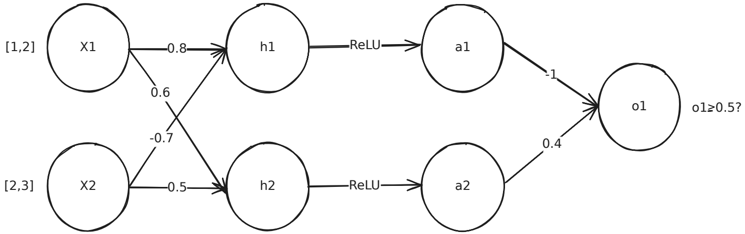


To ensure the reliability of neural networks

By *neural network* N, *preconditions* $\mathbf{\Phi}(\boldsymbol{x}_0, \epsilon)$, *and postconditions* $\mathbf{\Psi}^{|\boldsymbol{L}|}$,
we can formulate the neural network verification problem as:

$$\forall \boldsymbol{x} \in \mathbf{\Phi}(\boldsymbol{x}_0, \epsilon), N(\boldsymbol{x}) \models \bigwedge \mathbf{\Psi}^{|\boldsymbol{L}|}, \text{ where } \boldsymbol{x}_0 \text{ is the input vector.} \tag{1}$$

Example - neural network verification



- **Neural network:** the given direct acyclic graph in above.
- **Preconditions:** $1 \leq x_1 \leq 2 \wedge 2 \leq x_2 \leq 3$.
- **Postconditions:** $o_1 \geq 0.5$.

How to verify neural networks?

There are two directions to verify neural networks:

1. **Incomplete methods**:
   Overapproximation methods, most of them are based on **Abstract Interpretation**.
2. **Complete methods**:
   Exhaustive search methods such as branch and bound algorithm.

Both directions are providing a **soundness** which means that there is no *false positive*.

What is abstract interpretation?

Abstract interpretation is a **sound** and **incomplete** framework for analyzing programs by overapproximating (abstract domain) the program semantics (concrete domain).

Example

x is a real number, its possible value is between 1 and 2.
We would like to know if $x + 5 \leq 7$.

We cannot enumerate all the possible values by computing $x + 5$. (infinity many values)
Instead, we can use *interval abstract domain* to represent x as an interval $[1, 2]$.
Then, we can use interval arithmetic to compute the overapproximated bounds of $x + 5$:

$$x + 5 \in [1 + 5, 2 + 5] = [6, 7].$$

By checking the upper bound of the interval, we can conclude that $x + 5 \leq 7$ is **true**. □

What is abstract interpretation?

Abstract interpretation (AI) is

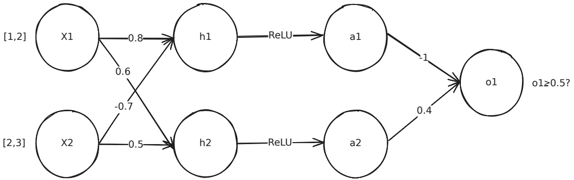
- **Sound**: there is no *false positive* result.
- **Incomplete**: there may be *false negative* results.

In other words, if the verification result obtained by AI is **true** then it must be **true** in the concrete domain.

But if the verification result obtained by AI is **false** then it may be **true** or **false** in the concrete domain.

An example for verifying neural networks by abstract interpretation

Example (Concrete Domain)



$x_1, x_2, h_1, h_2, o_1 \in \mathbb{R}, a_1, a_2 \in \mathbb{R}^+ \cup \{0\}$

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

$h_1 = 0.8x_1 - 0.7x_2, h_2 = 0.6x_1 + 0.5x_2$

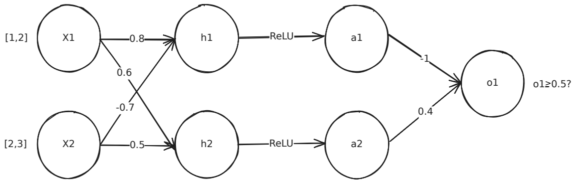
$a_1 = \max(0, h_1), a_2 = \max(0, h_2)$

$o_1 = -a_1 + 0.4a_2$

An example for verifying neural networks by abstract interpretation

We define an interval abstract domain for each neuron in the network.

Example (Interval Abstract Domain)



By interval arithmetic, we can compute the interval bounds for each neuron in the network.

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

$h_1 = 0.8 \times [1, 2] - 0.7 \times [2, 3] = [-1.3, 0.2], h_2 = 0.6 \times [1, 2] + 0.5 \times [2, 3] = [1.6, 2.7]$

$a_1 = [\max(0, -1.3), \max(0, 0.2)] = [0, 0.2], a_2 = [\max(0, 1.6), \max(0, 2.7)] = [1.6, 2.7]$

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

Research gap

The limitations of existing overapproximation methods are:

- Most of overapproximation methods only consider the **forward propagation** of the bounds.
- In some cases, the accumulated imprecision across layers causes the overapproximation at the output layer to become too coarse to verify the postconditions.

Research gap

The limitations of existing overapproximation methods are:

- Most of overapproximation methods only consider the **forward propagation** of the bounds.
- In some cases, the accumulated imprecision across layers causes the overapproximation at the output layer to become too coarse to verify the postconditions.

How to improve the precision of AI-based incomplete methods?

Research gap

The limitations of existing overapproximation methods are:

- Most of overapproximation methods only consider the **forward propagation** of the bounds.
- In some cases, the accumulated imprecision across layers causes the overapproximation at the output layer to become too coarse to verify the postconditions.

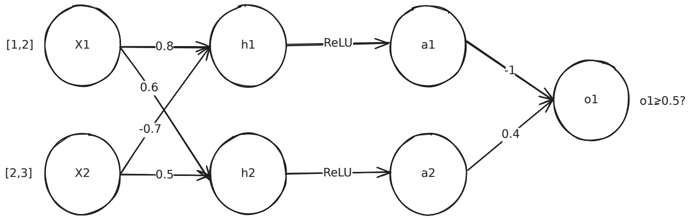
How to improve the precision of AI-based incomplete methods?

We propose a novel method, **Backward-Forward Analysis (BFA)**.

- **Backward analysis:** generate postconditions for each layer in the networks.
- **Forward analysis:**
  1. **Bound checking:** check if the verification process can be terminated ealier.
  2. **Overaproximation:** propagate the overapproximation to the next layer.

Backward-Forward Analysis (BFA)

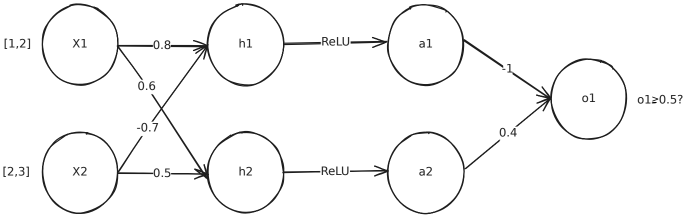
BFA - Backward Analysis



Given the postcondition $o_1 - 0.5 \geq 0$ and network constraint $o_1 = -a_1 + 0.4a_2$, we derive a new postcondition by replacing o_1 with $-a_1 + 0.4a_2$:

$$-a_1 + 0.4a_2 - 0.5 \geq 0$$

BFA - Backward Analysis



- $o_1 - 0.5 \geq 0 \wedge o_1 = -a_1 + 0.4a_2 \Leftrightarrow -a_1 + 0.4a_2 - 0.5 \geq 0$
- $a_1 = \max(0, h_1) \wedge a_2 = \max(0, h_2)$
- $-\max(0, h_1) + 0.4\max(0, h_2) - 0.5 \geq 0$
- $-h_1 + 0.4h_2 - 0.5 \geq 0 \Rightarrow -\max(0, h_1) + 0.4\max(0, h_2) - 0.5 \geq 0$ (Lemma 1)
- $h_1 = 0.8x_1 - 0.7x_2 \wedge h_2 = 0.6x_1 + 0.5x_2$
- $-(0.8x_1 - 0.7x_2) + 04(0.6x_1 + 0.5x_2) - 0.5 \geq 0$
- $-0.56x_1 + 0.9x_2 - 0.5 \geq 0$

Backward Analysis - overapproximating ReLU activation function

Lemma 1:

$\sum_i w_i y_i + b \geq 0 \Rightarrow \sum_i w_i \max(0, y_i) + b \geq 0$, where $\forall i, w_i \geq 0 \vee y_i \geq 0$, and $\forall i, w_i, y_i, b \in \mathbb{R}$.

Proof:

Given that $\forall i, w_i \geq 0 \vee y_i \geq 0$, then both w_i and y_i cannot be negative at the same time. Therefore, $\forall i, w_i \max(0, y_i) + b \geq w_i y_i + b$, and thus $\sum_i w_i \max(0, y_i) + b \geq \sum_i w_i y_i + b$. By transitivity, we conclude that if $\sum_i w_i y_i + b \geq 0$, then $\sum_i w_i \max(0, y_i) + b \geq 0$. $\square$

In backward analysis, we assume the disjunctive condition in Lemma 1 is always true.

$$-h_1 + 0.4h_2 - 0.5 \geq 0 \Rightarrow -\max(0, h_1) + 0.4\max(0, h_2) - 0.5 \geq 0$$

Forward Analysis - bound checking

To ensure the disjunctive condition in Lemma 1 is always true, we develop a **bound checking** method in forward analysis.

Lemma 1:

$\sum_i w_i y_i + b \geq 0 \Rightarrow \sum_i w_i \max(0, y_i) + b \geq 0$, where $\boldsymbol{\forall i, w_i \geq 0 \vee y_i \geq 0}$,
and $\forall i, w_i, y_i, b \in \mathbb{R}$.

If the disjunctive condition in Lemma 1 is violated, then the lower bound relationship is not existed.

Example

Given the postcondition $-h_1 + 0.4h_2 - 0.5 \geq 0$.
If $h_1 = -0.6 \wedge h_2 = 0$, then $-h_1 + 0.4h_2 = 0.6 - 0.5 = 0.1 \geq 0$,
but $-\max(0, h_1) + 0.4\max(0, h_2) - 0.5 = -0.5 \not\geq 0.1$. $\square$

Forward Analysis - bound checking

Bound Checking Principle

If the coefficient and the variables' value are negative, then repalce it with 0 such that Lemma 1 is satisfied.

Example

Given the postcondition $-h_1 + 0.4h_2 - 0.5 \geq 0$.
If $h_1 = -0.6 \wedge h_2 = 0$ then $-h_1$ will be replaced by 0. Thus, we have:

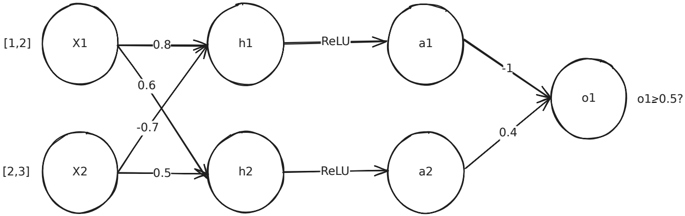
$$0 + 0.4h_2 - 0.5 = -0.5.$$

This result is identical with

$$-\max(0, h_1) + 0.4\max(0, h_2) - 0.5 = -0.5.$$

□

Example



Given the preconditions $(1 \leq x_1 \leq 2 \wedge 2 \leq x_2 \leq 3)$ and a postcondition $(-0.56x_1 + 0.9x_2 - 0.5 \geq 0)$ in the input layer, we apply the bound checking method:

$$\begin{aligned} -0.56x_1 + 0.9x_2 - 0.5 &= -0.56 \times [1, 2] + 0.9 \times [2, 3] - [0.5, 0.5] \\ &= [-1.12, -0.56] + [1.8, 2.7] - [0.5, 0.5] \\ &= [0.18, 1.64] 🥳 \quad \square \end{aligned}$$

Overview of BFA

BFA is a **sound** and **incomplete** verification method.

1. **Backward analysis:**
   We generate postconditions for each layer by replacing the variables and removing the activation functions in the previous layer.

2. **Forward analysis:**
   a. Applying **bound checking** to check if the overapproximation is satisfied in current layer.
   b. Overapproximating the bounds of each neuron in next layer by any existing forward propagation methods.

Experimental environment

All experiments are run on a 2.6GHz 64 cores processor AMD Epyc ROME 7H12 CPU with 256 GB memory. We evaluated BFA on MNIST and CIFAR-10 datasets.

The preconditions are defined by L_∞-norm perturbation, $\Phi(\boldsymbol{x}_0, \epsilon) = \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \leq \epsilon\}$, for the first 100 images from both datasets.

We compared BFA with 3 well-known bound propagation methods:

- DeepPoly/CROWN (DeepPoly)
- Symbolic Linear Relaxation (SLR)
- Interval Bound Propagation (IBP)

Experimental results - verification rate

How many images can be verified as safe?

Table 1: Verification rate for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on the first 100 images from MNIST testing dataset.

Networks	Perturbation	DeepPoly		SLR		IBP	
		Base	BFA	Base	BFA	Base	BFA
3×50	$\epsilon \leq 0.01$	0.98	**0.99**	0.96	0.96	0.36	**0.51**
	$0.01 < \epsilon \leq 0.02$	0.76	0.76	0.59	**0.60**	0.02	**0.05**
	$\epsilon > 0.02$	0.33	0.33	0.20	0.20	0.00	0.00
3×100	$\epsilon \leq 0.01$	0.96	**0.97**	0.89	**0.92**	0.04	**0.43**
	$0.01 < \epsilon \leq 0.02$	0.64	**0.64**	0.30	**0.31**	0.00	**0.01**
	$\epsilon > 0.02$	0.12	0.12	0.01	0.01	0.00	0.00
6×100	$\epsilon \leq 0.01$	0.96	**0.98**	0.22	**0.42**	0.00	**0.09**
	$0.01 < \epsilon \leq 0.02$	0.52	**0.53**	0.00	0.00	0.00	0.00
	$\epsilon > 0.02$	0.11	0.11	0.00	0.00	0.00	0.00
6×200	$\epsilon \leq 0.01$	0.86	**0.86**	0.01	**0.08**	0.00	**0.05**
	$0.01 < \epsilon \leq 0.02$	0.15	0.15	0.00	**0.01**	0.00	**0.01**
	$\epsilon > 0.02$	0.01	0.01	0.00	0.00	0.00	0.00
9×100	$\epsilon \leq 0.01$	0.93	**0.94**	0.01	**0.07**	0.00	0.00
	$0.01 < \epsilon \leq 0.02$	0.51	**0.53**	0.00	0.00	0.00	0.00
	$\epsilon > 0.02$	0.13	0.13	0.00	0.00	0.00	0.00
9×200	$\epsilon \leq 0.01$	0.81	0.81	0.00	**0.08**	0.00	**0.07**
	$0.01 < \epsilon \leq 0.02$	0.18	**0.18**	0.00	**0.04**	0.00	0.00
	$\epsilon > 0.02$	0.03	0.03	0.00	**0.01**	0.00	0.00

Table 2: Verification rate for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on CIFAR10 dataset.

Networks	Perturbation	DeepPoly		SLR		IBP	
		Base	BFA	Base	BFA	Base	BFA
4×100	$\epsilon \leq 0.0004$	0.75	**0.83**	0.65	0.65	0.00	**0.14**
	$0.0004 < \epsilon \leq 0.0008$	0.72	0.72	0.25	**0.36**	0.00	**0.07**
	$\epsilon > 0.0008$	0.47	0.47	0.04	**0.14**	0.00	**0.04**
6×100	$\epsilon \leq 0.0004$	0.74	**0.80**	0.03	**0.21**	0.00	0.00
	$0.0004 < \epsilon \leq 0.0008$	0.33	**0.47**	0.00	**0.09**	0.00	0.00
	$\epsilon > 0.0008$	0.12	0.12	0.00	0.00	0.00	0.00
7×1024	$\epsilon \leq 0.0004$	1.00	1.00	0.00	**0.43**	0.00	**0.23**
	$0.0004 < \epsilon \leq 0.0008$	0.85	**1.00**	0.00	**0.12**	0.00	**0.12**
	$\epsilon > 0.0008$	0.54	**0.96**	0.00	0.00	0.00	0.00
9×200	$\epsilon \leq 0.0004$	0.88	**1.00**	0.00	**0.38**	0.00	0.00
	$0.0004 < \epsilon \leq 0.0008$	0.75	**1.00**	0.00	0.00	0.00	0.00
	$\epsilon > 0.0008$	0.63	0.63	0.00	0.00	0.00	0.00

Experimental results - submatched rate

How many images can be verified as safe in intermediate layers?

Table 3: Submatched rate summary for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on MNIST dataset.

Networks	Perturbation Range	DeepPolyBFA	SLRBFA	IBPBFA
Small Networks	$\epsilon \leq 0.01$	**0.40**	**0.39**	**0.65**
	$0.01 < \epsilon \leq 0.02$	**0.04**	**0.04**	**0.38**
	$\epsilon > 0.02$	0.00	0.00	0.00
Medium Networks	$\epsilon \leq 0.01$	**0.82**	**0.99**	**1.00**
	$0.01 < \epsilon \leq 0.02$	**0.38**	**0.25**	0.00
	$\epsilon > 0.02$	**0.02**	0.00	0.00
Deep Networks	$\epsilon \leq 0.01$	**0.59**	**0.73**	**0.50**
	$0.01 < \epsilon \leq 0.02$	**0.54**	**0.27**	0.00
	$\epsilon > 0.02$	**0.22**	**0.17**	0.00

Table 4: Submatched rate summary for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on CIFAR10 dataset.

Networks	Perturbation Range	DeepPolyBFA	SLRBFA	IBPBFA
4×100	$\epsilon \leq 0.0004$	**0.35**	**0.40**	**1.00**
	$0.0004 < \epsilon \leq 0.0008$	**0.29**	**0.50**	**1.00**
	$\epsilon > 0.0008$	**0.23**	**0.84**	**1.00**
6×100	$\epsilon \leq 0.0004$	**0.29**	**0.25**	0.00
	$0.0004 < \epsilon \leq 0.0008$	**0.40**	**0.25**	0.00
	$\epsilon > 0.0008$	0.00	0.00	0.00
7×1024	$\epsilon \leq 0.0004$	**1.00**	**1.00**	**1.00**
	$0.0004 < \epsilon \leq 0.0008$	**1.00**	**0.50**	**0.50**
	$\epsilon > 0.0008$	**1.00**	0.00	0.00
9×200	$\epsilon \leq 0.0004$	**1.00**	**1.00**	0.00
	$0.0004 < \epsilon \leq 0.0008$	**0.88**	0.00	0.00
	$\epsilon > 0.0008$	0.00	0.00	0.00

Small Networks: 3 hidden layers, Medium Networks: 6 hidden layers, Large Networks: 9 hidden layers.

Experimental results - average computational time

Table 5: Average runtime (in seconds) for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on the first 100 images from MNIST testing dataset.

Networks	DeepPoly		SLR		IBP	
	Base	BFA	Base	BFA	Base	BFA
3×50	25.13	25.86	25.22	26.67	23.67	24.73
3×100	51.22	53.54	52.03	55.21	49.94	52.80
6×100	75.65	86.92	76.01	90.14	72.61	86.65
6×200	208.94	242.48	218.25	240.69	202.10	237.65
9×100	97.56	114.45	94.38	118.30	94.06	114.86
9×200	280.47	341.30	279.58	342.85	270.07	335.89

Table 6: Average runtime (in seconds) for L_∞-norm-based perturbations by BFA against DeepPoly, SLR, and IBP on CIFAR10 dataset.

Networks	DeepPoly		SLR		IBP	
	Base	BFA	Base	BFA	Base	BFA
4×100	27.49	29.40	30.52	30.94	26.03	27.38
6×100	36.24	37.57	39.38	41.95	33.61	36.66
7×1024	620.41	641.89	661.11	700.64	586.55	631.43
9×200	23.16	23.70	24.86	26.42	21.56	23.34

Feedback from CAV 2025

We compared BFA with DeepZ, which is proposed in 2018, only.

- We were trying to use other recent methods, but we found some issues to execute them.
- The implementation is not elegant, thus the computational time is extremely expensive.
- Since the expensive computational time, we cannot significantly improve the verification performance.

Therefore, the reviewers suggested us to try to use other recent methods such as `DeepPoly`, `alpha-beta crown`, `NeuralSAT` etc.

Also, they mentioned that it would be nice to have a result showing the effectiveness of BFA. (submatched rate)

Feedback from SAS 2025

We still lack of strong experiments to convince the reviewers that BFA is an effective method.
But ...

Feedback from SAS 2025

We still lack of strong experiments to convince the reviewers that BFA is an effective method.
But ...

Thanks for the reviewers' comments and questions, we have additional insights from our work!
Not sure if we will be accepted, but we will keep improving our work! 💪 😎

Next Steps

Extended works

- Change a proper name of our paper.
- Try to have more insights from our work and demonstrate it.
- Fix out of memory issue when verifying full testing dataset.
- Try to exectue `alpha-beta-crown` and fix the issue from them.
- ...

Future Plane

2025, so far, it is not perfect for me, but I'll ...

- Work hard in this summer to try to make at least one breakthrough.
- Read textbooks to have solid knowledge in my research area.
- Teach in Lattice Theory course in Winter semester 2025.

Besides research ...

- 12th July, Lënster Trail 15 km and 316 D+ with Pierre.
- 6th Sep, Escher Kulturlaf 10 km.
- 19th Oct, Amsterdam Marathon 2025.

Break my limitations in running! 🏃 🏆

Thank you for your attention!