Machine Learning with Verifiable Guarantees

Thomas Flinkow and Rosemary Monahan

Department of Computer Science
Maynooth University

VerifAI Workshop
9^{th} March 2026

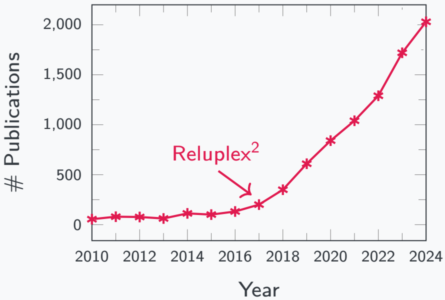
Why Verify Neural Networks?

Figure: Stickers placed on a stop sign cause a neural network to misclassify it as *Speed Limit 45 mph*.[1]

[1] K. Eykholt et al. (2018). 'Robust Physical-World Attacks on Deep Learning Visual Classification'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. DOI: 10.1109/CVPR.2018.00175.

How to Verify Neural Networks?

Paper titles containing "formal verification" and "neural network" over time



Community Efforts

VNN-COMP International Verification of Neural Networks Competition (https://sites.google.com/view/vnn2025)

VNN-LIB International Standard for the Verification of Neural Networks (https://www.vnnlib.org)

[2] G. Katz et al. (2017). 'Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks'. In: *Computer Aided Verification*. DOI: 10.1007/978-3-319-63387-9_5.

Training to Satisfy Constraints

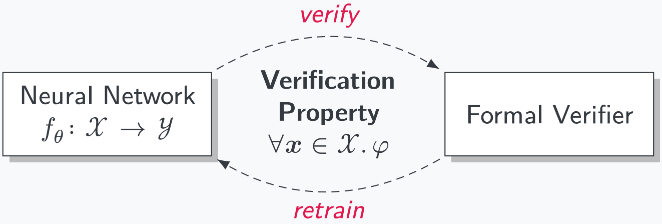


Figure: The continuous verification cycle.

Training Process:

1. translate φ into real-valued, differentiable loss $[\![\varphi]\!]$,
2. find a counterexample in $\mathcal{X}$ (using gradient descent) that violates φ, and
3. add that counterexample to the training data.

Example

- **Constraint:** $\forall \boldsymbol{x} \in \mathcal{X}.\,(5 \leq f(\boldsymbol{x}) \wedge f(\boldsymbol{x}) \leq 10)$
- **Translate:** e.g. using DL2[3]:
  $[\![a \leq b]\!]_{\mathsf{DL2}} = \max(0, [\![a]\!]_{\mathsf{DL2}} - [\![b]\!]_{\mathsf{DL2}})$ and
  $[\![a \wedge b]\!]_{\mathsf{DL2}} = [\![a]\!]_{\mathsf{DL2}} + [\![b]\!]_{\mathsf{DL2}}$

$$[\![5 \leq f(\boldsymbol{x}) \wedge f(\boldsymbol{x}) \leq 10]\!]_{\mathsf{DL2}}$$
$$[\![5 \leq f(\boldsymbol{x})]\!] + [\![f(\boldsymbol{x}) \leq 10]\!]_{\mathsf{DL2}}$$
$$\max(0, 5 - f(\boldsymbol{x})) + \max(0, f(\boldsymbol{x}) - 10)$$

- **Find a counterexample:**

$$x^* = \max_{\boldsymbol{x}' \in \mathcal{X}} [\![5 \leq f(\boldsymbol{x}) \wedge f(\boldsymbol{x}) \leq 10]\!]_{\mathsf{DL2}}$$

- **Use counterexample in training.**

[3] M. Fischer et al. (2019). 'DL2: Training and Querying Neural Networks with Logic'. In: *Proceedings of the 36th International Conference on Machine Learning*.

Previous Work: Comparing Existing Differentiable Logics[4]

Logic	**Domain**	$[\![T]\!]$	$[\![F]\!]$	$[\![\neg x]\!]$	$[\![x \wedge y]\!]$	$[\![x \vee y]\!]$
DL2	$[0, \infty)$	0	∞	undefined	$x + y$	xy
Gödel	$[0, 1]$	1	0	$1 - x$	$\min(x, y)$	$\max(x, y)$

Research Question

How do existing differentiable logics (DLs) compare in terms of:

- gradients (learning behaviour)?
- logical consistency?
- establishing formal guarantees?

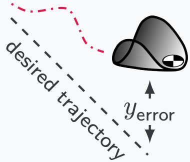
Findings

Training with *any* DL yields significantly improved *empirical* constraint satisfaction, but fails to provide (strong) formal guarantees.

[4] T. Flinkow, B. A. Pearlmutter et al. (2025). 'Comparing Differentiable Logics for Learning with Logical Constraints'. In: *Science of Computer Programming*. DOI: 10.1016/j.scico.2025.103280.

Previous Work: Alsomitra Drone Controller[5]

(a) An *Alsomitra macrocarpa* seed.



(b) The desired linear trajectory.

Constraint: *If the drone is above and close to the line, pitching down quickly and moving fast, the network should make it pitch up.*

Logic	RMSE	CAcc (%)	CSec (%)
Baseline	3.61×10^{-4}	0.00	0.00
DL2	1.23×10^{-3}	100.00	95.31
Fuzzy logic	1.16×10^{-3}	100.00	92.19

[5] C. Kessler et al. (2026). 'Neural Network Verification for Gliding Drone Control: A Case Study'. In: *AI Verification*. DOI: 10.1007/978-3-031-99991-8_9.

Recent Work (in Progress): Strong(er) Formal Guarantees

Issues

- DL2 $[\![-]\!]_{\mathsf{DL2}} \colon \Phi \to [0, \infty)$ loss is 0 if the constraint is satisfied.
- Consider the simple constraint $f(\boldsymbol{x}) \leq 5$.
- $[\![a \leq b]\!]_{\mathsf{DL2}} = \max(0, a - b)$ means gradients vanish once $a \leq b$.
- **Not *finding* a counterexample does not mean there is none!**

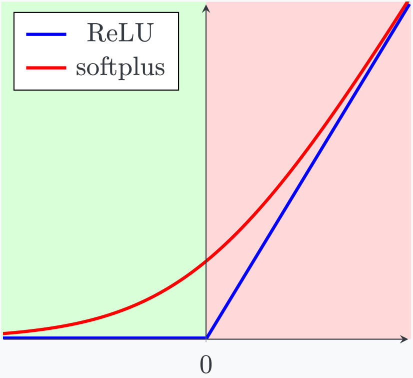
Ideas for a new differentiable logic

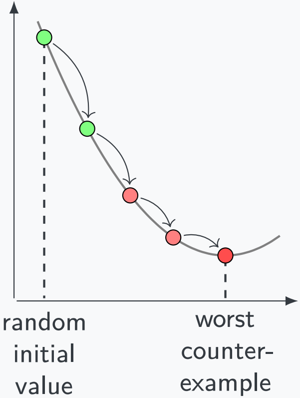
- Never let gradients vanish (i.e. make it *possible* to always find counterexamples)!
- Smooth connectives $[\![\wedge^s]\!]_{\mathsf{Ours}}$ and $[\![\vee^s]\!]_{\mathsf{Ours}}$ that approximate $\min$ and $\max$ as $s \to \infty$.

Non-vanishing Gradients Everywhere

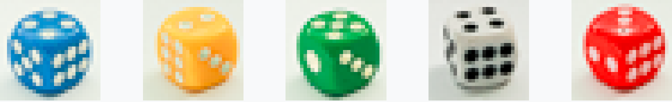
Idea: even when a constraint is satisfied, provide a small gradient.

$$[\![a \leq b]\!]_{\mathsf{DL2}} = \mathrm{ReLU}(a-b) \qquad [\![a \leq b]\!]_{\mathsf{Ours}} = \mathrm{softplus}(a-b)$$





Experimental Results



Constraint: *Predictions must be physically possible: the network should not predict two faces that are on opposite sides of the die.*

Logic	PAcc (%)	CAcc (%)	CSec (%)	Verified Accuracy (%)[a]		
				$\epsilon = 4/255$	$\epsilon = 8/255$	$\epsilon = 16/255$
Baseline	85.8	97.1	5.9	20.6	$0.0^{(1)}$	0.0
DL2	82.6	98.5	29.4	$51.5^{(1)}$	$16.2^{(2)}$	$1.5^{(1)}$
Ours	78.9	100.0	100.0	100.0	100.0	$91.2^{(5)}$

[a] Superscript $^{(k)}$ indicates the inputs that timed out and remain unknown.

Summary & Conclusion

1. Motivation

Correct-by-construction ML by specification-driven training.

2. Current Work

A differentiable logic with stronger formal guarantees.

3. Future Work

Stronger guarantees and more expressive specifications.

Thomas Flinkow

Department of Computer Science
Maynooth University

Email: thomas.flinkow@mu.ie
https://www.cs.nuim.ie/~tflinkow/

Thank you! Any questions?

1. T. Flinkow, B. A. Pearlmutter et al. (2025). 'Comparing Differentiable Logics for Learning with Logical Constraints'. In: *Science of Computer Programming* 244, p. 103280. ISSN: 0167-6423. DOI: 10.1016/j.scico.2025.103280
2. T. Flinkow, M. Casadio et al. (2025). *A General Framework for Property-Driven Machine Learning*. DOI: 10.48550/arXiv.2505.00466. arXiv: 2505.00466 [cs]

This work was supported by Taighde Éireann - Research Ireland under grant number 20/FFP-P/8853 and by the ADAPT Research Ireland Centre for AI Driven Digital Content Technology at Maynooth University under grant number 13/RC/2106_P2.