Maynooth University
National University of Ireland Maynooth

Towards Correct-by-Construction Machine-Learnt Models

Thomas Flinkow, Barak A. Pearlmutter, and Rosemary Monahan

Department of Computer Science
Maynooth University

PhD Symposium @ iFM 2024
12th November 2024

Contents

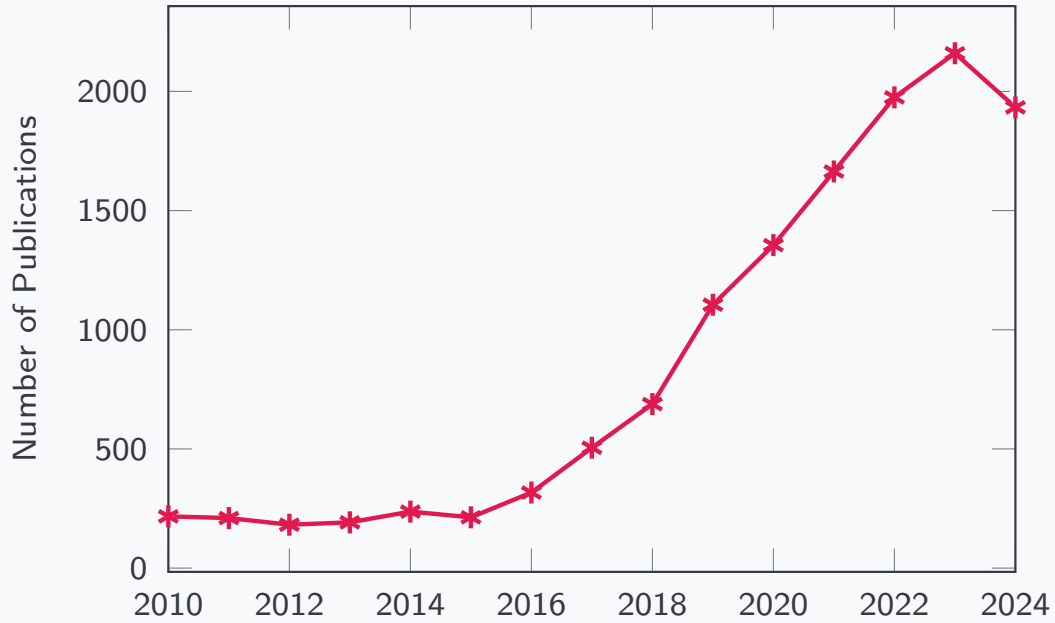
Introduction & Motivation

Motivation

Issue:
Neural networks fail to learn (safety) properties from data alone!

Formal Verification of Neural Networks

Scopus search results for 'neural network verification'



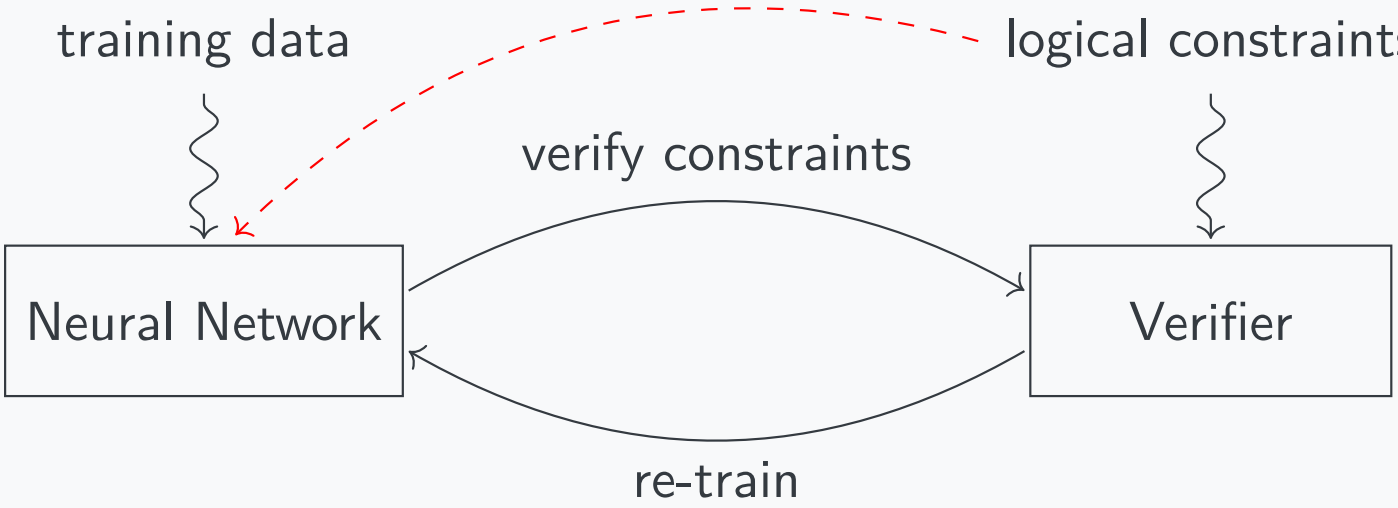
- **SMT Solvers:** Planet (Ehlers, 2017), Reluplex (Katz et al., 2017), Marabou (Katz et al., 2019), ...
- **MILP:** Branch-and-Bound (Bunel et al., 2018), MIPVerify (Tjeng et al., 2019), Sherlock (Dutta et al., 2019), ...
- **Abstract Interpretation:** AI2 (Gehr et al., 2018), DeepPoly (Singh et al., 2019), NNV (Tran et al., 2020), ...

Training with Logical Constraints

Task: train a neural network $\mathcal{N}$ to satisfy constraint ϕ.

Train: given data, labels, and loss function, iteratively update network weights.

Verify: afterwards, α, β-CROWN, Marabou, NNV, ERAN, ...



Note

Training with constraints does not guarantee their satisfaction!

Background

Property-driven Training with Differentiable Logics

Given data $\boldsymbol{x}_0$ and label $\boldsymbol{y}$, and constraint ϕ,
obtain optimal network weights $\boldsymbol{\theta}^+$ by

$$\boldsymbol{\theta}^+ = \arg\min_{\boldsymbol{\theta}} \ \alpha\mathcal{L}_{\mathsf{CE}}(\boldsymbol{x}_0, \boldsymbol{y}) + \beta\mathcal{L}_{\mathsf{C}}(\boldsymbol{x}_0, \boldsymbol{y}, \phi).$$

Optimisation handled by Gradient Descent:

$$\boldsymbol{\theta}_{i+1} := \boldsymbol{\theta}_i - \eta\nabla\mathcal{L}(\boldsymbol{\theta}_i)$$

Implications

Logical constraint ϕ must be differentiable!

DL2 (Fischer et al., 2019)

- mapping $[\![\cdot]\!]_{\text{DL2}} : \Phi \to [0, \infty)$,
- $[\![\phi]\!]_{\text{DL2}} = 0$ iff ϕ is satisfied,
- $[\![\phi]\!]_{\text{DL2}}$ is differentiable almost everywhere.

Recursive definition of loss translation:

$$
\begin{aligned}
[\![x \leq y]\!]_{\text{DL2}} &:= \max\{x - y, 0\} \\
[\![\phi \wedge \psi]\!]_{\text{DL2}} &:= [\![\phi]\!]_{\text{DL2}} + [\![\psi]\!]_{\text{DL2}} \\
[\![\phi \vee \psi]\!]_{\text{DL2}} &:= [\![\phi]\!]_{\text{DL2}} \cdot [\![\psi]\!]_{\text{DL2}}.
\end{aligned}
$$

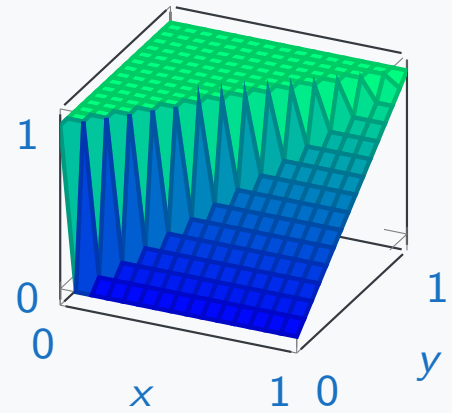
Fuzzy Logics (Ślusarz et al., 2023; van Krieken et al., 2022)

- logical system for reasoning with vagueness
- mapping $[\![\cdot]\!]_L : \Phi \to [0, 1]$, where $[\![\top]\!]_L = 1$ and $[\![\bot]\!]_L = 0$,
- operators happen to be differentiable almost everywhere

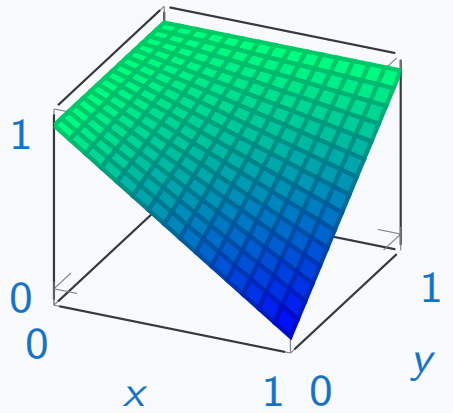
Logic	T-norm	S-norm	Implication
Gödel	$\min\{x, y\}$	$\max\{x, y\}$	$\begin{cases} 1, & \text{if } x < y, \\ y, & \text{else.} \end{cases}$
Kleene-Dienes			$S(N(x), y)$
Łukasiewicz	$\max\{0, x + y - 1\}$	$\min\{1, x + y\}$	
Reichenbach	xy	$x + y - xy$	
Goguen			$\begin{cases} 1, & \text{if } x < y, \\ y^x, & \text{else.} \end{cases}$

Derivatives: Modus Tollens Reasoning

$$[\![x \to y]\!]_{\mathrm{G}} = \begin{cases} 1, & \text{if } x \leq y \\ y, & \text{else} \end{cases}$$



$$[\![x \to y]\!]_{\mathrm{RC}} = 1 - x + xy$$



Example: 'If it rains, the ground will be wet.'

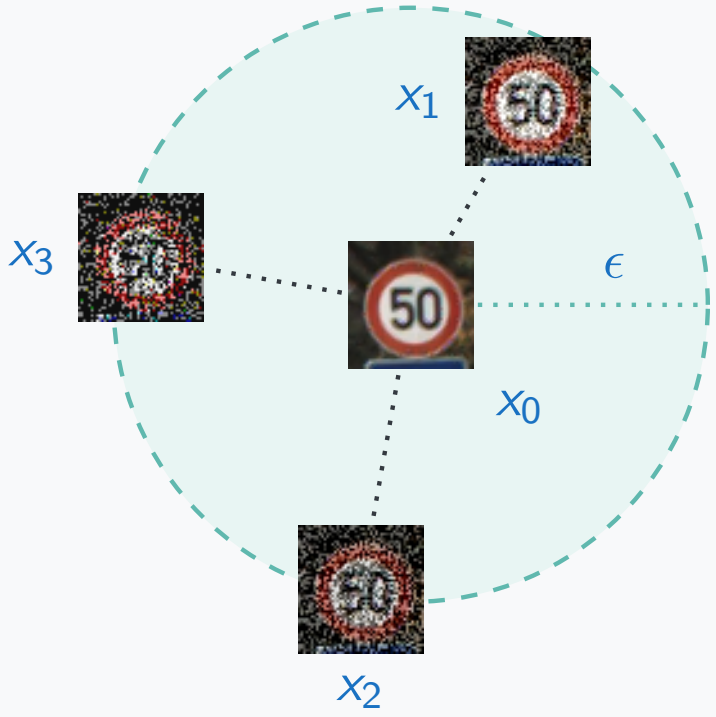
Let $[\![\text{it rains}]\!] = 0.1$ and $[\![\text{ground is wet}]\!] = 0$.

- Then, $[\![\text{it rains} \to \text{ground is wet}]\!]_{\mathrm{G}} = 0$. How to correct?
- With $\frac{\partial}{\partial x}[\![x \to y]\!]_{\mathrm{G}} = 0$ and $\frac{\partial}{\partial y}[\![x \to y]\!]_{\mathrm{G}} = 1$, we have no choice but to *incorrectly* insist that the ground must be wet.

Constraint Counterexamples with Adversarial Training

Insight from DL2 (Fischer et al., 2019)

Learning to satisfy $\forall x.\, x \vDash \phi$ by finding x^* such that $x^* \nvDash \phi$.



1. Approximate counterexample *outside* of training set using PGD:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \leq \epsilon}{\arg\max} \mathcal{L}_{\mathsf{C}}(\boldsymbol{x}_0, \boldsymbol{x}, \boldsymbol{y}, \phi)$$

2. Use this counterexample in training:

$$\boldsymbol{\theta}^+ = \underset{\boldsymbol{\theta}}{\arg\min} \quad \alpha \mathcal{L}_{\mathsf{CE}}(\boldsymbol{x}_0, \boldsymbol{y}) + \beta \mathcal{L}_{\mathsf{C}}(\boldsymbol{x}_0, \boldsymbol{x}^*, \boldsymbol{y}, \phi).$$

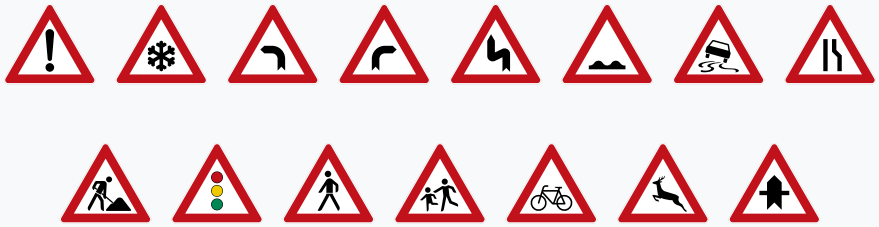
RQ1: Does training with constraints work in practice?

RQ2: Which logic translation works best?

Experimental Results

Group Constraint

(a) unique signs



(b) danger signs

(c) derestriction signs

(d) speed limit signs

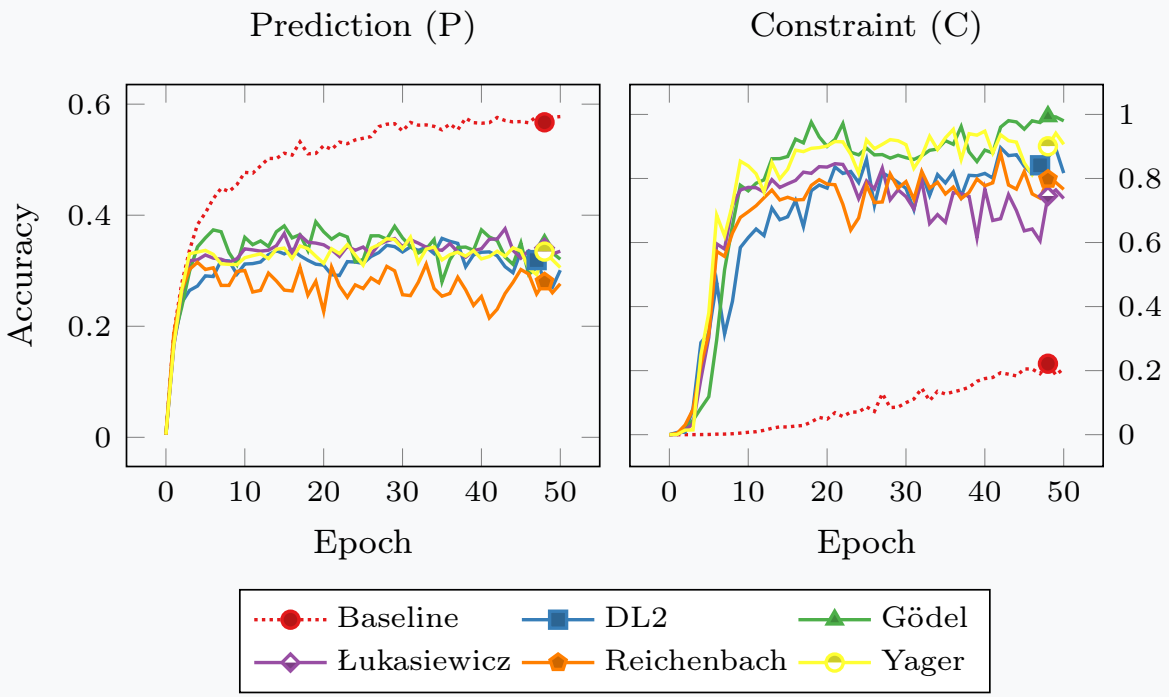
(e) other prohibitory signs

(f) mandatory signs

Definition

$$\forall \boldsymbol{x} \in ||\boldsymbol{x} - \boldsymbol{x}_0|| \leq \epsilon \quad \rightarrow \quad \left(\sum_{k \in \texttt{Group}} p_k \leq \delta \ \vee \ \sum_{k \in \texttt{Group}} p_k \geq 1 - \delta \right)$$

Group Constraint – Results



Logic	P	C
Baseline	56.73	22.15
DL2	31.92	84.12
Gödel	**34.76**	**99.39**
Łukasiewicz	34.13	74.54
Reichenbach	28.00	79.76
Yager	33.44	90.06

Observation

- Training with logical loss incurs a (considerable) penalty to prediction accuracy (Tsipras et al., 2018).
- Training with *any* logical loss leads to noticeably improved constraint accuracy. Differences between loss translations are not too significant.

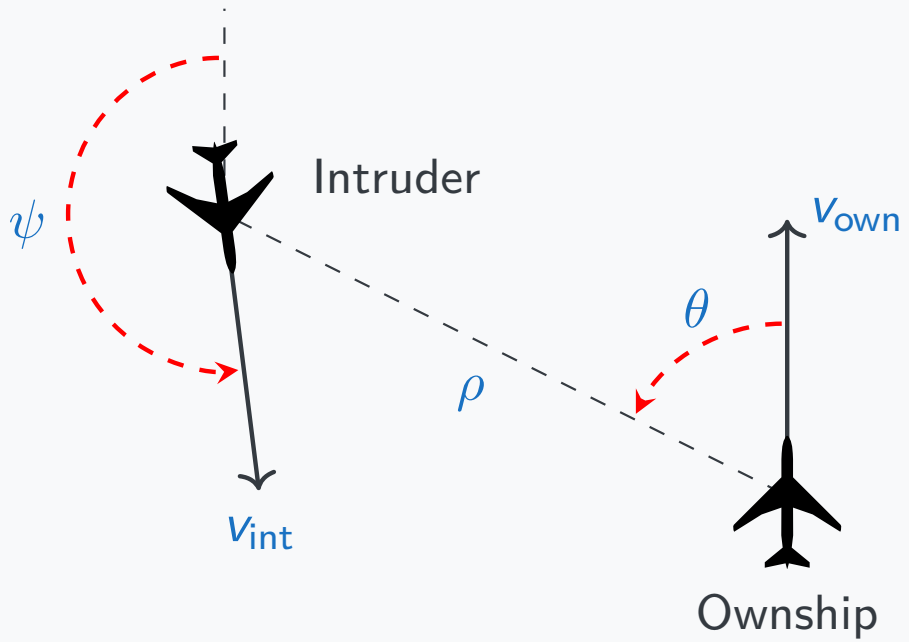
Future Work

Specifications for Machine Learning

- Many *complete* verifiers (can prove / disprove every property) but necessary to specify relevant regions of the input space.
- Often infeasible in practice (high-dimensional input spaces) $\rightarrow$ verification often limited to point-wise verification.

Example: Reluplex (Katz et al., 2017)

'If an intruder is *near* and *approaching from the left*, network should advise *strong right*'.



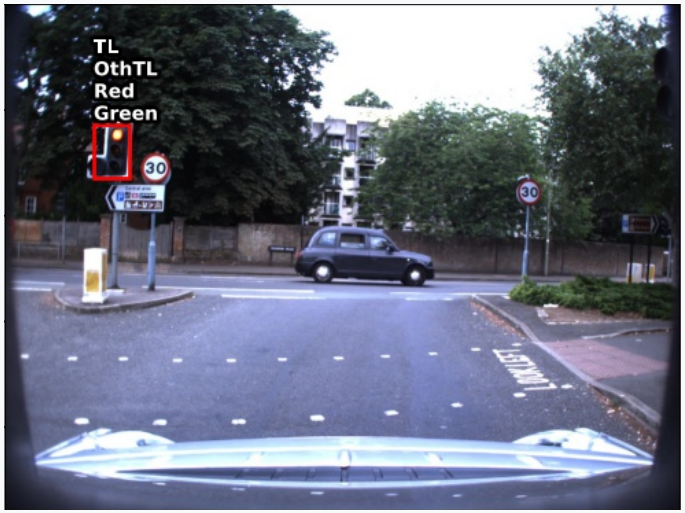
$$\rho \in [250\,\text{ft}, 400\,\text{ft}]$$
$$\theta \in [0.2\,\text{rad}, 0.4\,\text{rad}]$$
$$\psi \in [-\pi, -\pi + 0.005]$$
$$v_{\text{own}} \in [100\,\text{ft/s}, 400\,\text{ft/s}]$$
$$v_{\text{int}} \in [0\,\text{ft/s}, 400\,\text{ft/s}]$$

Expressive Logics

Example: ROAD-R Data Set (Giunchiglia et al., 2023)

Videos annotated with background knowledge (propositional logic).

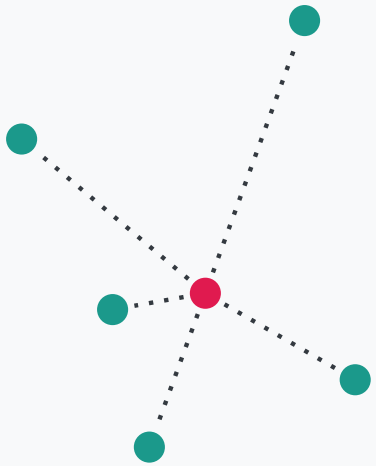
$$\{\neg \text{Ped}, \neg \text{Cyc}\} \cup \{\neg \text{Red}, \neg \text{Green}\} \cup \{\neg \text{Green}, \neg \text{Mov}\} \cup \ldots$$



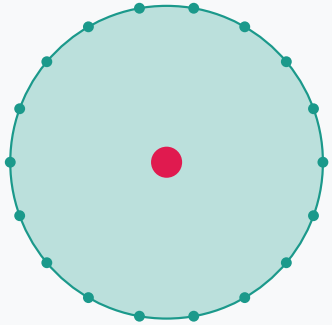
Is there a need for more expressive logics?
What about temporal or probabilistic logics?

Certified Training

Instead of *minimising* an *upper bound* on the loss, . . .



(a) Adversarial Training



(b) Certified Training

. . . *maximise* a *lower bound* on robustness!

Summary & Conclusion

1. Motivation

Use formal methods during training to get correct-by-construction models.

2. Approach

Integrate logical constraints into training using differentiable logics.

3. Experimental Results

Training with *any* differentiable logic works (but provides no guarantees).

4. Future Work

Specifications for ML in general, expressive logics (e.g. temporal, probabilistic), and certified training.

Thomas Flinkow

Department of Computer Science
Maynooth University

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

Maynooth University
National University of Ireland Maynooth

Thank you! Any questions?

References

Bunel, R. R., Turkaslan, I., Torr, P., Kohli, P., & Mudigonda, P. K. (2018).A Unified View of Piecewise Linear Neural Network Verification. *Advances in Neural Information Processing Systems*, *31*. Retrieved January 31, 2023, from https://proceedings.neurips.cc/paper/2018/hash/be53d253d6bc3258a8160556dda3e9b2-Abstract.html

Dutta, S., Chen, X., Jha, S., Sankaranarayanan, S., & Tiwari, A. (2019).Sherlock - A tool for verification of neural network feedback systems: Demo abstract. *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 262–263. https://doi.org/10.1145/3302504.3313351

References (cont.)

Ehlers, R. (2017). Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks. In D. D'Souza & K. Narayan Kumar (Eds.), *Automated Technology for Verification and Analysis* (pp. 269–286). Springer International Publishing. https://doi.org/10.1007/978-3-319-68167-2_19

Fischer, M., Balunovic, M., Drachsler-Cohen, D., Gehr, T., Zhang, C., & Vechev, M. (2019).DL2: Training and Querying Neural Networks with Logic. *Proceedings of the 36th International Conference on Machine Learning*, 1931–1941. Retrieved April 13, 2023, from https://proceedings.mlr.press/v97/fischer19a.html

Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., & Vechev, M. (2018).AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation. *2018 IEEE Symposium on Security and Privacy (SP)*, 3–18. https://doi.org/10.1109/sp.2018.00058

References (cont.)

Giunchiglia, E., Stoian, M. C., Khan, S., Cuzzolin, F., & Lukasiewicz, T. (2023).ROAD-R: The autonomous driving dataset with logical requirements. *Machine Learning*, *112*(9), 3261–3291. https://doi.org/10.1007/s10994-023-06322-z

Katz, G., Barrett, C., Dill, D. L., Julian, K., & Kochenderfer, M. J. (2017). Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In R. Majumdar & V. Kunčak (Eds.), *Computer Aided Verification* (pp. 97–117). Springer International Publishing. https://doi.org/10.1007/978-3-319-63387-9_5

Katz, G., Huang, D. A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., Dill, D. L., Kochenderfer, M. J., & Barrett, C. (2019). The Marabou Framework for Verification and Analysis of Deep Neural Networks. In I. Dillig & S. Tasiran (Eds.), *Computer Aided Verification* (pp. 443–452). Springer International Publishing. https://doi.org/10.1007/978-3-030-25540-4_26

References (cont.)

Singh, G., Gehr, T., Püschel, M., & Vechev, M. (2019).An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, *3*, 1–30. https://doi.org/10.1145/3290354

Ślusarz, N., Komendantskaya, E., Daggitt, M., Stewart, R., & Stark, K. (2023).Logic of Differentiable Logics: Towards a Uniform Semantics of DL. *EPiC Series in Computing*, *94*, 473–493. https://doi.org/10.29007/c1nt

Tjeng, V., Xiao, K., & Tedrake, R. (2019, February 17). *Evaluating Robustness of Neural Networks with Mixed Integer Programming*. arXiv: 1711.07356 [cs]. https://doi.org/10.48550/arXiv.1711.07356

References (cont.)

Tran, H.-D., Yang, X., Manzanas Lopez, D., Musau, P., Nguyen, L. V., Xiang, W., Bak, S., & Johnson, T. T. (2020). NNV: The Neural Network Verification Tool for Deep Neural Networks and Learning-Enabled Cyber-Physical Systems. In S. K. Lahiri & C. Wang (Eds.), *Computer Aided Verification* (pp. 3–17). Springer International Publishing. https://doi.org/10.1007/978-3-030-53288-8_1

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., & Madry, A. (2018).Robustness May Be at Odds with Accuracy. Retrieved September 2, 2023, from https://openreview.net/forum?id=SyxAb30cY7

van Krieken, E., Acar, E., & van Harmelen, F. (2022).Analyzing Differentiable Fuzzy Logic Operators. *Artificial Intelligence*, *302*, 103602. https://doi.org/10.1016/j.artint.2021.103602