Building Trust and Safety in Artificial Intelligence with Abstract Interpretation

Gagandeep Singh[1,2][0000−0002−9299−2961]

[1] University of Illinois at Urbana-Champaign (UIUC), USA
[2] VMware Research, USA ggnds@illinois.edu

1 Introduction

Deep neural networks (DNNs) are currently the dominant technology in artificial intelligence (AI) and have shown impressive performance in diverse applications including autonomous driving [9], medical diagnosis [2], and text generation [10]. However, their black-box construction [46] and vulnerability against environmental and adversarial noise [57, 30] have raised concerns about their safety, when deployed in the real world. Standard training [28] optimizes the model's accuracy but does not take into account desirable safety properties such as robustness [48], fairness [18], and monotonicity [49]. The standard practice of testing and interpreting DNN behavior on a finite set of unseen test inputs cannot guarantee safe and trustworthy DNN behavior on new inputs seen during deployment [59, 66]. This is because the DNN can misbehave if the inputs observed during deployment deviate even slightly from those in the test set [36, 20, 23].

To address these limitations, there is growing work on checking the safety of DNN models [5, 52, 51, 53, 44, 70, 58, 68, 3, 27, 50, 65, 31, 62, 11, 61, 25, 43, 17, 42, 45] and interpreting their behavior [7], on an infinite set of unseen inputs using formal certification. Testing and interpreting with formal methods provide a more reliable metric for measuring a model's safety than standard methods [12]. Formal methods can also be used during training [22, 38, 69, 74, 72, 40, 6] to guide the model to satisfy desirable safety and trustworthy properties.

DNN certification problem. The certification problem consists of two main components: (i) a trained DNN f, (ii) a property specification in the form of a tuple (ϕ, ψ) containing symbolic formulas ϕ and ψ. Here the formula ϕ is a precondition specifying the set of inputs on which, the DNN should not misbehave. The formula ψ is a postcondition that determines constraints that the DNN outputs $f(\phi)$ [26] or its gradients $f'(\phi)$ [33, 34] corresponding to the inputs in ϕ should satisfy, for its behaviors to be considered safe and trustworthy. A DNN certifier tries to check whether $f(\phi) \subseteq \psi$ (or $f'(\phi) \subseteq \psi$) holds. Both ϕ, ψ are typically specified as disjunctions of convex polyhedra. The property specifications are domain dependent and usually designed by DNN developers.

Local vs global properties. The precondition ϕ for local properties defines a local neighborhood around a sample input from the test set. For example, given a test image correctly classified as a car by a DNN, the popular local robustness property specifies that all images generated by rotating the original

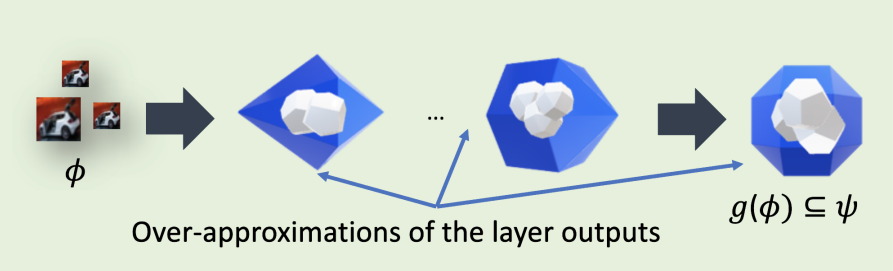

Fig. 1: Neural network certification with abstract interpretation involves computing an abstract element (in blue) containing the true network output $f(\phi)$ (in white) at each layer using the corresponding abstract transformer.

image within ± 5 degrees are also classified as a car [5]. In contrast, ϕ for global properties does not depend upon a test input. For domains where the input features have semantic meaning, e.g., air traffic collision avoidance systems [26] or security classifiers [13], global properties can be specified by defining a valid range for the input features expected in a real-world deployment. Certifying global properties yields stronger safety guarantees, however, they are difficult to formulate for popular domains, such as vision and NLP, where the individual features processed by the DNN have no semantic meaning. While certifying local properties is not ideal, the local certification results enable testing the safety of the model on an infinite set of unseen inputs, not possible with standard methods.

2 Certification for Testing Model Safety

DNN certification can be seen as an instance of program verification (DNNs can be written as programs) making it undecidable. State-of-the-art certifiers are therefore incomplete in general. These certifiers can be formulated using the elegant framework of abstract interpretation [15]. While abstract interpretation-based certifiers can certify both local and global properties, for the remainder of this paper, we focus on the certification of local properties as they are more common in real-world applications. Figure 1 shows the high-level idea behind DNN certification with abstract interpretation. Here, the certifier is parameterized by the choice of an abstract domain. The certifier first computes an abstract element $\alpha(\phi) \supseteq \phi$ that includes the input region ϕ. Next, the analyzer symbolically propagates $\alpha(\phi)$ through the different layers of the network. At each layer, the analyzer computes an abstract element (in blue) overapproximating the exact layer output (in white) corresponding to ϕ. The element is computed by applying an abstract transformer that approximates the effect of the operations (e.g., ReLU, affine) applied at the layer. Propagation through all the layers yields an abstract element $g(\alpha(\phi)) \supseteq f(\phi)$ at the output layer. Next, the certifier checks if $g(\alpha(\phi)) \subseteq \psi$ holds for the bigger region $g(\alpha(\phi))$. If the answer is yes, then $f(\phi) \subseteq \psi$ also holds for the smaller region $f(\phi)$. Because of the overapproxima-

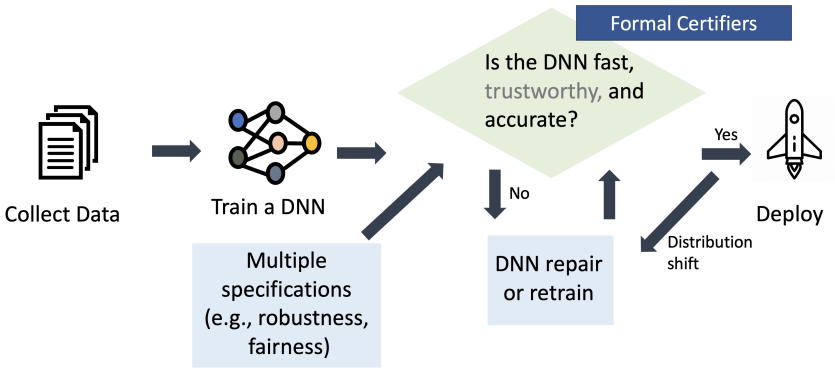

Fig. 2: Development pipeline for building fast, accurate, and trustworthy DNNs. Certification is used for testing model trustworthiness (green diamond).

tion, it can be the case that $g(\alpha(\phi)) \subseteq \psi$ does not hold while $f(\phi) \subseteq \psi$ holds. To reduce the amount of overapproximation, refinements [62, 53, 41, 47, 63, 68, 71] can be applied.

To obtain an effective certifier, it is essential to design an abstract domain and corresponding abstract transformers such that $g(\alpha(\phi))$ is as close as possible to the true output $f(\phi)$ while g can also be computed in a reasonable amount of time for practical networks. The classical domains, such as Polyhedra [16, 55] and Octagons [37, 54], used for analyzing programs are not well suited for DNN certification. This is because the DNNs have a different structure compared to traditional programs. For example, DNNs have a large number of non-linear assignments but typically do not have infinite loops. For efficient certification, new abstract domains and transformers tailored for DNN certification have been developed. Examples include DeepPoly [52], DeepZ [51], Star sets [58], and DeepJ [33]. These custom solutions can scale to realistic DNNs with upto a million neurons [39], or more than 100 layers [68], certifying diverse safety properties in different real-world applications including autonomous driving [72], job-scheduling [68], data center management [12], and financial modeling [32].

Incremental certification. By leveraging formal certification to check DNN safety and trust, the development pipeline shown in Figure 2 can be employed [61] to obtain fast, accurate, and trustworthy DNNs. First, a DNN is trained to maximize its test accuracy. Next, a domain expert designs a set of safety specifications (e.g., robustness, fairness) defining the expected network behavior in different real-world scenarios. If the model satisfies the desired specifications, then the DNN is considered fit for deployment. Otherwise, it is iteratively repaired (e.g., by fine-tuning [1] or LP-solving [56]) till we obtain a fast, accurate, and trustworthy DNN. We note that repair is preferred over retraining as it is cheaper. However, if a repair is not possible, then the DNN is retrained from scratch. After deployment, the DNN is monitored to check for distribution shifts, gener-

ating inputs not covered by the specifications. If a distribution shift is detected, then new specifications are designed, and the model is repaired or retrained.

Domain experts usually design a large number of local properties (around 10-100K). Therefore, the certifier needs to be run several thousand times on the same DNN. Further, as shown in Figure 2, the model repair is applied, before or after deployment, in case the DNN does not satisfy the desired specifications. The certifier is needed again to check the safety of the repaired model. Existing certifiers do not *scale* in such a deployment setting: they can precisely certify individual specifications in a few seconds or minutes, however, the certification of a large and diverse set of specifications on a single DNN can take multiple days to years or the certifier can run out of memory. Given multiple DNNs are generated due to repair or retraining, it makes using existing certifiers for safe and trustworthy development infeasible. The inefficiency is because the certifier needs to be run from scratch for every new pair of specifications and DNNs. A straightforward approach to overcome this limitation is to run the certifier on several machines. However, such an approach is not sustainable due to its huge environmental cost [67, 8]. Further, in many cases, large computational resources are not available. For example, to preserve privacy, reduce latency, and increase battery lifetime, DNNs are increasingly employed on edge devices with limited computational power [64, 14]. Therefore, for sustainable, democratic, and trustworthy DNN development, it is essential to develop new general approaches for incremental certification to improve the certifier scalability, when certifying multiple specifications and networks.

The main challenge in developing incremental certifiers is determining information that (i) can be reused across multiple specifications and DNNs to improve scalability, and (ii) is efficient to compute and store. Recent works [19, 61] have developed general mechanisms to enable incremental certification by reusing proofs across multiple specifications and DNNs. These methods can be plugged into state-of-the-art certifiers based on abstract interpretation [52, 51] to improve their scalability inside the development pipeline of Figure 2. [19] introduced the concept of *proof sharing* across multiple specifications on the same DNN. Proof sharing is based on the key insight that it is possible to construct a small number of abstract elements as proof templates at an intermediate DNN layer, that capture the intermediate proofs of a large number of specifications. To certify a new specification, we run the certifier partially till the layer at which the templates are available. If the intermediate proof is subsumed by an existing template, then the specification is proved without running the certifier till the end, saving time and memory. The work of [61] introduced the concept of *proof transfer* across similar networks obtained after incremental changes to an original network (e.g., after fine-tuning [1]). The key insight behind this concept is that it is possible to efficiently transfer the proof templates generated on the original network to multiple similar networks, such that the transformed templates capture the proofs of a large number of specifications on similar networks. The transferred templates can improve certifier precision and scalability when certifying multiple specifications on similar networks. [60] considers incre-

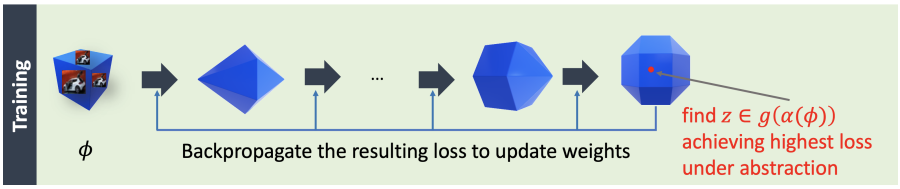

Fig. 3: Certified training involves computing the point $z \in g(\alpha(\phi))$ where the robust loss if maximum. The resulting loss is backpropagated through the certifier code to update the model parameters.

mental certification for certifiers combining abstract interpretation with branch and bound (BaB) [11] and uses the trace of BaB as proof templates to improve certification speed across multiple similar DNNs.

3 Certification for Training Safe DNNs

DNNs trained to only maximize accuracy with standard training [28] are often unsafe [36]. Next, we describe how certifiers can be leveraged during training to obtain safe DNNs. While the description here applies to different safety properties, we focus on robustness as it is the most common property for safe training considered in the literature. Robust training involves defining a robust loss function $\mathcal{L}_R$ for each point $x \in \phi$ with the property that $\mathcal{L}_R$ at x is ≤ 0 iff in the DNN output $z = f(x)$, the score z_c for the correct class c is higher than all other classes z_i, i.e., $z_c > z_i$. The DNN is robust iff $\mathcal{L}_R \leq 0$ for all $x \in \phi$. The DNN parameters can be updated during training to minimize the maximum value of $\mathcal{L}_R$. This min-max formulation makes robust training a harder optimization problem than standard training. Computing the worst-case robust loss exactly requires computing $f(\phi)$ which is infeasible. Therefore an approximation of $\mathcal{L}_R$ is computed in practice. Adversarial training methods [36] compute a lower bound on the worst-case robust loss by heuristically computing a point $x \in \phi$ at which the robust loss is high. x is then added to the training dataset. On the other hand, certified training [72, 38, 69, 74, 21, 6] methods compute an upper bound on the worst-case robust loss using abstract interpretation-based DNN certifiers. Figure 3 shows the high-level idea behind certified training which leverages the output $g(\alpha(\phi))$ computed by the DNN certifier. Here one computes $z \in g(\alpha(\phi))$ where the robust loss is maximum and then updates the model with respect to the resulting loss value. State-of-the-art certified training methods employ differentiable certifiers [38, 51], which makes the computation of the worst-case robust loss differentiable. As a result, the parameter updates are performed by differentiating through the certifier code directly.

Since certified training computes an upper bound on the worst-case robust loss when this loss is ≤ 0, the actual loss is also ≤ 0. This is not the case with the lower bound computed by adversarial training. As a result, DNNs trained with certified training achieve higher robustness guarantees than those trained

with adversarial training [38]. They are also easier to certify than those trained with adversarial and standard training. Even imprecise abstract domains such as intervals give precise certification results for DNNs trained with certified training. The work of [4] theoretically shows the existence of two DNNs f, f' such that (i) they have the same accuracy, and (ii) interval analysis achieves the same certification results on f' as a more precise certifier on f.

Training with only the robust loss deteriorates model accuracy, therefore in practice, robust loss is combined with standard accuracy loss during training using custom mechanisms [21]. While one would expect that training with precise certifiers yields more accurate and robust DNNs than imprecise ones, as they reduce the approximation error in computing the robust loss, in practice, the highly imprecise interval domain performs the best for certified training. This is because the optimization problem for training becomes harder with more complex abstract domains [24]. Most certified training methods target robustness with respect to norm-based changes to pixel intensities in images. Even with all the progress in this direction, DNNs trained with state-of-the-art certified training methods [40, 6, 74] suffer significant loss of accuracy on popular datasets such as CIFAR10 [29]. There have been conflicting hypotheses in the literature about whether accuracy conflicts with norm-based robustness [59] or not [73]. The work of [72] is the first to build a certified training method for challenging geometric robustness by developing a fast geometric certifier that can be efficiently parallelized on GPUs. Interestingly, the work shows that it is possible to achieve both high accuracy and robustness on the autonomous driving dataset [9]. Therefore, in certain practical scenarios, both high accuracy and safety may be achievable.

4 Certification for Interpreting DNNs

Abstract interpretation-based DNN certifiers [52, 51, 70] generate high-dimensional abstract elements at different layers capturing complex relationships between neurons and DNN inputs to prove DNN safety. However, the individual neurons and inputs in the DNN do not have any semantic meaning, unlike the variables in programs, therefore it is not clear whether the safety proofs are based on any meaningful features learned by the DNN. If the DNN is proven to be safe but the proof is based on meaningless features not aligned with human intuition, then the DNN behavior cannot be considered trustworthy. While there has been a lot of work on interpreting black-box DNNs, standard methods [46, 66] can only explain the DNN behavior on individual inputs and cannot interpret the complex invariants encoded by the abstract elements capturing DNN behavior on an infinite set of inputs. The main challenge in interpreting DNN proofs is in mapping the complex abstract elements to human understandable interpretations.

The work of [7] is the first to develop a method for interpreting robustness proofs computed by DNN certifiers. The method can interpret proofs computed by different certifiers. It builds upon the novel concept of *proof features* that are computed by projecting the high-dimensional abstract elements onto individual neurons. The proof features can be analyzed independently by generating the

corresponding interpretations. Since certain proof features can be more important for the proof than others, a priority function over the proof features that signify the importance of each individual proof feature in the complete proof is defined. The method extracts a set of proof features by retaining only the more important parts of the proof that preserve the property.

A comparison of proof interpretations for DNNs trained with standard and robust training methods [36, 74, 6] on the popular MNIST [35] and CIFAR10 datasets [29] shows that the proof features corresponding to the standard networks rely on meaningless input features while the proofs of adversarially trained DNNs [36] filter out some of these spurious features. In contrast, the networks trained with certifiable training [74] produce proofs that do not rely on any spurious features but they also miss out on some meaningful features. Proofs for training methods that combine both empirical and certified robustness [6] not only preserve meaningful features but also selectively filter out spurious ones. These observations are empirically shown to be not contingent on any specific DNN certifier. These insights suggest that DNNs can satisfy safety properties but their behavior can still be untrustworthy.

References

1. Agrawal, P., Girshick, R.B., Malik, J.: Analyzing the performance of multilayer neural networks for object recognition. In: ECCV (7). Lecture Notes in Computer Science, vol. 8695, pp. 329–344. Springer (2014)
2. Amato, F., López, A., Peña-Méndez, E.M., Vaňhara, P., Hampl, A., Havel, J.: Artificial neural networks in medical diagnosis. Journal of Applied Biomedicine **11**(2) (2013)
3. Anderson, G., Pailoor, S., Dillig, I., Chaudhuri, S.: Optimization and abstraction: A synergistic approach for analyzing neural network robustness. In: Proc. Programming Language Design and Implementation (PLDI). p. 731–744 (2019)
4. Baader, M., Mirman, M., Vechev, M.: Universal approximation with certified networks. In: International Conference on Learning Representations (2020)
5. Balunovic, M., Baader, M., Singh, G., Gehr, T., Vechev, M.: Certifying geometric robustness of neural networks. In: Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019)
6. Balunovic, M., Vechev, M.T.: Adversarial training and provable defenses: Bridging the gap. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020)
7. Banerjee, D., Singh, A., Singh, G.: Interpreting robustness proofs of deep neural networks (2023)
8. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: Elish, M.C., Isaac, W., Zemel, R.S. (eds.) FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021. pp. 610–623. ACM (2021)
9. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)

10. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: Proc. Advances in Neural Information Processing Systems (NeurIPS) (2020)
11. Bunel, R., Lu, J., Turkaslan, I., Kohli, P., Torr, P., Mudigonda, P.: Branch and bound for piecewise linear neural network verification. Journal of Machine Learning Research **21**(2020) (2020)
12. Chakravarthy, A., Narodytska, N., Rathis, A., Vilcu, M., Sharif, M., Singh, G.: Property-driven evaluation of rl-controllers in self-driving datacenters. In: Workshop on Challenges in Deploying and Monitoring Machine Learning Systems (DMML) (2022)
13. Chen, Y., Wang, S., Qin, Y., Liao, X., Jana, S., Wagner, D.A.: Learning security classifiers with verified global robustness properties. In: Proc. Conference on Computer and Communications Security (CCS). pp. 477–494. ACM (2021)
14. Chugh, U., Mitra, A., Deshwal, A., Swaroop, N.P., Saluja, A., Lee, S., Song, J.: An automated approach to accelerate dnns on edge devices. In: ISCAS. pp. 1–5. IEEE (2021)
15. Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977. pp. 238–252. ACM (1977)
16. Cousot, P., Halbwachs, N.: Automatic discovery of linear restraints among variables of a program. In: Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages, Tucson, Arizona, USA, January 1978. pp. 84–96. ACM Press (1978)
17. Dathathri, S., Dvijotham, K., Kurakin, A., Raghunathan, A., Uesato, J., Bunel, R., Shankar, S., Steinhardt, J., Goodfellow, I.J., Liang, P., Kohli, P.: Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020)
18. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.S.: Fairness through awareness. In: Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012. pp. 214–226. ACM (2012)
19. Fischer, M., Sprecher, C., Dimitrov, D.I., Singh, G., Vechev, M.T.: Shared certificates for neural network verification. In: Computer Aided Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13371, pp. 127–148. Springer (2022)
20. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
21. Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T.A., Kohli, P.: On the effectiveness of interval bound propagation for training verifiably robust models. CoRR **abs/1810.12715** (2018)
22. Gowal, S., Dvijotham, K.D., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., Kohli, P.: Scalable verified training for provably robust image classification. In: Proc. IEEE/CVF International Conference on Computer Vision (ICCV). pp. 4842–4851 (2019)

23. Heo, J., Joo, S., Moon, T.: Fooling neural network interpretations via adversarial model manipulation. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 2921–2932 (2019)
24. Jovanovic, N., Balunovic, M., Baader, M., Vechev, M.T.: On the paradox of certified training. Trans. Mach. Learn. Res. **2022** (2022)
25. Kabaha, A., Drachsler-Cohen, D.: Boosting robustness verification of semantic feature neighborhoods. In: Static Analysis - 29th International Symposium, SAS 2022, Auckland, New Zealand, December 5-7, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13790, pp. 299–324. Springer (2022)
26. Katz, G., Barrett, C., Dill, D., Julian, K., Kochenderfer, M.: Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In: Proc. 29th Int. Conf. on Computer Aided Verification (CAV). pp. 97–117 (2017)
27. Katz, G., Huang, D., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., Dill, D., Kochenderfer, M., Barrett, C.: The Marabou Framework for Verification and Analysis of Deep Neural Networks, pp. 443–452 (07 2019)
28. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
29. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
30. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: ICLR (Workshop). OpenReview.net (2017)
31. Lan, J., Zheng, Y., Lomuscio, A.: Tight neural network verification via semidefinite relaxations and linear reformulations. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022,. pp. 7272–7280. AAAI Press (2022)
32. Laurel, J., Qian, S.B., Singh, G., Misailovic, S.: Synthesizing precise static analyzers for automatic differentiation. Proc. ACM Program. Lang. (OOPSLA2) (2023)
33. Laurel, J., Yang, R., Singh, G., Misailovic, S.: A dual number abstraction for static analysis of clarke jacobians. Proc. ACM Program. Lang. **6**(POPL), 1–30 (2022)
34. Laurel, J., Yang, R., Ugare, S., Nagel, R., Singh, G., Misailovic, S.: A general construction for abstract interpretation of higher-order automatic differentiation. Proc. ACM Program. Lang. **6**(OOPSLA2), 1007–1035 (2022)
35. LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E., Jackel, L.D.: Handwritten digit recognition with a back-propagation network. In: NIPS. pp. 396–404 (1989)
36. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
37. Miné, A.: The octagon abstract domain. High. Order Symb. Comput. **19**(1), 31–100 (2006)
38. Mirman, M., Gehr, T., Vechev, M.: Differentiable abstract interpretation for provably robust neural networks. In: Proc. International Conference on Machine Learning (ICML). pp. 3578–3586 (2018)
39. Müller, C., Serre, F., Singh, G., Püschel, M., Vechev, M.T.: Scaling polyhedral neural network verification on gpus. In: Proceedings of Machine Learning and Systems 2021, MLSys 2021, virtual, April 5-9, 2021. mlsys.org (2021)
40. Müller, M.N., Eckert, F., Fischer, M., Vechev, M.T.: Certified training: Small boxes are all you need. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023)
41. Müller, M.N., Makarchuk, G., Singh, G., Püschel, M., Vechev, M.: Precise multi-neuron abstractions for neural network certification. arXiv preprint arXiv:2103.03638 (2021)

42. Munakata, S., Urban, C., Yokoyama, H., Yamamoto, K., Munakata, K.: Verifying attention robustness of deep neural networks against semantic perturbations. In: NASA Formal Methods - 15th International Symposium, NFM 2023, Houston, TX, USA, May 16-18, 2023, Proceedings. Lecture Notes in Computer Science, vol. 13903, pp. 37–61. Springer (2023)
43. Palma, A.D., Behl, H.S., Bunel, R., Torr, P.H.S., Kumar, M.P.: Scaling the convex barrier with active sets. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021)
44. Paulsen, B., Wang, J., Wang, C.: Reludiff: Differential verification of deep neural networks. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering. p. 714–726. ICSE '20 (2020)
45. Ranzato, F., Urban, C., Zanella, M.: Fairness-aware training of decision trees by abstract interpretation. In: CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021. pp. 1508–1517. ACM (2021)
46. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should I trust you?": Explaining the predictions of any classifier. In: Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1135–1144. ACM (2016)
47. Ryou, W., Chen, J., Balunovic, M., Singh, G., Dan, A., Vechev, M.: Scalable polyhedral verification of recurrent neural networks. In: International Conference on Computer Aided Verification. pp. 225–248. Springer (2021)
48. Shafique, M., Naseer, M., Theocharides, T., Kyrkou, C., Mutlu, O., Orosa, L., Choi, J.: Robust machine learning systems: Challenges,current trends, perspectives, and the road ahead. IEEE Design Test **37**(2), 30–57 (2020)
49. Sill, J.: Monotonic networks. In: Advances in Neural Information Processing Systems 10, [NIPS Conference, Denver, Colorado, USA, 1997]. pp. 661–667. The MIT Press (1997)
50. Singh, G., Ganvir, R., Püschel, M., Vechev, M.: Beyond the single neuron convex barrier for neural network certification. In: Advances in Neural Information Processing Systems (2019)
51. Singh, G., Gehr, T., Mirman, M., Püschel, M., Vechev, M.: Fast and effective robustness certification. Advances in Neural Information Processing Systems **31** (2018)
52. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. Proceedings of the ACM on Programming Languages **3**(POPL) (2019)
53. Singh, G., Gehr, T., Püschel, M., Vechev, M.: Robustness certification with refinement. In: International Conference on Learning Representations (2019)
54. Singh, G., Püschel, M., Vechev, M.T.: Making numerical program analysis fast. In: Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation, Portland, OR, USA, June 15-17, 2015. pp. 303–313. ACM (2015)
55. Singh, G., Püschel, M., Vechev, M.T.: Fast polyhedra abstract domain. In: Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017. pp. 46–59. ACM (2017)
56. Sotoudeh, M., Thakur, A.V.: Provable repair of deep neural networks. In: PLDI '21: 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, Virtual Event, Canada, June 20-25, 2021. pp. 588–603. ACM (2021)
57. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. In: ICLR (Poster) (2014)

58. Tran, H.D., Manzanas Lopez, D., Musau, P., Yang, X., Nguyen, L.V., Xiang, W., Johnson, T.T.: Star-based reachability analysis of deep neural networks. In: Formal Methods – The Next 30 Years. pp. 670–686. Springer International Publishing, Cham (2019)
59. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A.: Robustness may be at odds with accuracy. In: proc. International Conference on Learning Representations, ICLR. OpenReview.net (2019)
60. Ugare, S., Banerjee, D., Misailovic, S., Singh, G.: Incremental verification of neural networks. Proc. ACM Program. Lang. **7**(PLDI) (jun 2023)
61. Ugare, S., Singh, G., Misailovic, S.: Proof transfer for fast certification of multiple approximate neural networks. Proc. ACM Program. Lang. **6**(OOPSLA1), 1–29 (2022)
62. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Efficient formal safety analysis of neural networks. In: Advances in Neural Information Processing Systems (2018)
63. Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C.J., Kolter, J.Z.: Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. arXiv preprint arXiv:2103.06624 (2021)
64. Wang, X., Hersche, M., Tömekce, B., Kaya, B., Magno, M., Benini, L.: An accurate eegnet-based motor-imagery brain-computer interface for low-power edge computing. In: IEEE International Symposium on Medical Measurements and Applications, (MeMeA). pp. 1–6. IEEE (2020)
65. Wong, E., Kolter, J.Z.: Provable defenses against adversarial examples via the convex outer adversarial polytope. In: Proc. International Conference on Machine Learning, ICML. Proceedings of Machine Learning Research, vol. 80, pp. 5283–5292. PMLR (2018)
66. Wong, E., Santurkar, S., Madry, A.: Leveraging sparse linear layers for debuggable deep networks. In: Proceedings of the 38th International Conference on Machine Learning, ICML. Proceedings of Machine Learning Research, vol. 139, pp. 11205–11216. PMLR (2021)
67. Wu, C., Raghavendra, R., Gupta, U., Acun, B., Ardalani, N., Maeng, K., Chang, G., Behram, F.A., Huang, J., Bai, C., Gschwind, M., Gupta, A., Ott, M., Melnikov, A., Candido, S., Brooks, D., Chauhan, G., Lee, B., Lee, H.S., Akyildiz, B., Balandat, M., Spisak, J., Jain, R., Rabbat, M., Hazelwood, K.: Sustainable AI: environmental implications, challenges and opportunities. In: MLSys. mlsys.org (2022)
68. Wu, H., Barrett, C., Sharif, M., Narodytska, N., Singh, G.: Scalable verification of gnn-based job schedulers. Proc. ACM Program. Lang. **6**(OOPSLA2) (oct 2022)
69. Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K.W., Huang, M., Kailkhura, B., Lin, X., Hsieh, C.J.: Automatic perturbation analysis for scalable certified robustness and beyond. In: Proc. Neural Information Processing Systems (NeurIPS). pp. 1129–1141 (2020)
70. Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., Hsieh, C.J.: Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In: International Conference on Learning Representations (2021)
71. Yang, P., Li, R., Li, J., Huang, C., Wang, J., Sun, J., Xue, B., Zhang, L.: Improving neural network verification through spurious region guided refinement. In: Tools and Algorithms for the Construction and Analysis of Systems TACAS. Lecture Notes in Computer Science, vol. 12651, pp. 389–408. Springer (2021)

72. Yang, R., Laurel, J., Misailovic, S., Singh, G.: Provable defense against geometric transformations. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023)
73. Yang, Y., Rashtchian, C., Zhang, H., Salakhutdinov, R., Chaudhuri, K.: A closer look at accuracy vs. robustness. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS) (2020)
74. Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., Hsieh, C.J.: Towards stable and efficient training of verifiably robust neural networks. In: Proc. International Conference on Learning Representations (ICLR) (2020)