Differential Verification of Neural Networks: Theory and Applications

Samuel Teuber[0000−0001−7945−9110] and Philipp Kern[0000−0002−7618−7401]

Karlsruhe Institute of Technology, Karlsruhe Germany
{teuber, philipp.kern}@kit.edu

Abstract. Although techniques for the verification of neural networks (NNs) have seen significant progress in recent years, specifying meaningful properties remains non-trivial. One promising approach to the specification of neural networks are *relational properties* which do not specify the behavior of an NN in absolute terms, but impose constraints on related execution traces. We provide an overview on our recent advances in using *differential verification* in NN verification by leveraging the abstract domain of Zonotopes. This approach enables us to reason about changes in behavior locally, at the level of individual neurons of an NN. We discuss the fundamental ideas of our abstract domain and discuss current and future applications of the differential verification paradigm.

Keywords: Neural Network Verification · Differential Verification · Equivalence · Fully Homomorphic Encryption.

1 Introduction

While meaningfully specifying the desired behavior of neural networks (NNs) is generally challenging, one class of specifications that is comparatively easy to formalize are *relational* properties: Instead of specifying the NN's behavior in absolute terms, we specify how different execution traces of the NNs are supposed to relate to each other. Indeed, many specifications of NNs can be interpreted as relational specifications: Adversarial robustness relates the outputs of an NN for an (often fixed) input vector and its *perturbed* version. Fairness relates two executions of an NN where inputs only differ w.r.t. some *protected* attribute. Equivalence proves that executions of two NNs match (e.g. after pruning, additional training and/or quantization).

In this talk about prior work [7, 14], we provide an overview of our work on *differential verification* which proves relational properties of NNs by bounding the *difference* between two executions of an NN locally at the level of every neuron. We first provide a brief overview on *differential zonotopes* as an abstract domain for NN verification and then discuss its applications in NN verification.

2 Neural Network Verification and Zonotopes

We consider the verification of feed-forward NNs whose behavior can be summarized as a function $f : \mathbb{R}^I \to \mathbb{R}^O$ (input dimension I, output dimension O)

which computes its output through $L \in \mathbb{N}$ layers by iteratively mapping an input vector $\mathbf{x}^{(0)} \in \mathbb{R}^I$ to an output vector $\mathbf{x}^{(L)} \in \mathbb{R}^O$ via the rule $\mathbf{x}^{(i)} = \text{ReLU}\left(W^{(i)}\mathbf{x}^{(i-1)} + \mathbf{b}^{(i)}\right)$ (for a matrix $W^{(i)}$ and a vector $\mathbf{b}^{(i)}$). Analyzing an NN then amounts to dealing with the application of affine transformations $W^{(i)}\mathbf{x}^{(i-1)} + \mathbf{b}^{(i)}$ and the activation function $\text{ReLU}(\cdot)$. Despite this simplicity, verifying meaningful properties about NNs is typically NP-complete [6] (also for relational properties like equivalence [13, 14]).

Due to its balance between expressivity and efficiency, Zonotopes [4, 5, 10] are a popular abstract domain for the analysis of NNs. A zonotope consists of an n-dimensional unit box which is projected into an m-dimensional space via an affine transformation:

Definition 1 (Zonotope). *A* zonotope *with input and output dimension n and m, respectively, consists of a* generator matrix $G \in \mathbb{R}^{m \times n}$ *and a* center $\boldsymbol{c} \in \mathbb{R}^m$*, which we denote as $\mathcal{Z} = (G, \boldsymbol{c})$. A zonotope's concretization encompasses $\langle (G, \boldsymbol{c}) \rangle = \{\ G\boldsymbol{\epsilon} + \boldsymbol{c} \mid \boldsymbol{\epsilon} \in [-1, 1]^n\ \} \subseteq \mathbb{R}^m$. For concrete $\boldsymbol{\epsilon}$ we define the evaluation of the zonotope as $\mathcal{Z}(\boldsymbol{\epsilon}) = G\boldsymbol{\epsilon} + \boldsymbol{c}$.*

Zonotopes admit efficient, exact representation of affine transformations and can also be propagated through activation functions by using linear over-approximations of their output [10]. They can be used for verification by propagating a set of inputs represented as a zonotope through an individual NN f and then proving that the output bounds induced by the zonotope satisfy a given specification.

3 Differential Verification with Zonotopes

While Zonotopes have been used for the verification of *individual* NNs extensively in prior literature, we recently proposed their usage in relational NN verification [14]. To this end, we proposed to complement independent reachability analyses in individual NNs by the computation of a *differential zonotope* which *bounds* the difference between the reachable values in the independent NNs.

The differential zonotope is propagated in lock-step with the propagation of the individual zonotopes which allows *locally* bounding the difference between two execution traces of NNs at every neuron. Two independent zonotopes and a corresponding differential zonotope can then be concretized as follows:

Definition 2 (Concretization [14]). *For two individual zonotopes $\mathcal{Z}', \mathcal{Z}''$ and a differential zonotope $\mathcal{Z}^\Delta$ we define the concretization of $\left(\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^\Delta\right)$ as:*

$$\langle \left(\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^\Delta\right) \rangle = \left\{ (\boldsymbol{x}, \boldsymbol{y}) \;\middle|\; \begin{array}{ll} \exists \boldsymbol{\epsilon} \in [-1, 1]^n & \boldsymbol{x} = \mathcal{Z}'(\boldsymbol{\epsilon}) \wedge \boldsymbol{y} = \mathcal{Z}''(\boldsymbol{\epsilon}) \wedge \\ & (\boldsymbol{x} - \boldsymbol{y}) = \mathcal{Z}^\Delta(\boldsymbol{\epsilon}) \end{array} \right\}$$

To use this approach in practice, we proposed specialized abstract transformers which propagate $\mathcal{Z}'$ and $\mathcal{Z}''$ as well as the differential zonotope $\mathcal{Z}^\Delta$ through NNs f' and f'' together. Effectively, $\mathcal{Z}^\Delta$ represents local "coupling points" between the internal computations of f' and f''. This allows us to prove properties about the relation between $f'(\mathcal{Z}')$ and $f''(\mathcal{Z}'')$ much more effectively than by only propagating independent zonotopes or using self-composition.

4 Current Applications

We implemented the differential zonotope abstract domain in the tool VeryDiff [14]. Differential verification of NNs has originally been conceived to bound the numerical difference between an original NN and its float-truncated version [8, 9]. In this direction we can leverage it to prove that outputs of two NNs differ by at most ε, which can formally be expressed as a two-safety property:

Definition 3 (ε-equivalence [2, 8]). *Two NNs $f_1,\ f_2 : \mathbb{R}^I \to \mathbb{R}^O$ are called* ε-equivalent *on $X \subseteq \mathbb{R}^I$ iff $\|f_1(x) - f_2(x)\|_\infty \leq \varepsilon$.*

Beyond numerical equivalence we also used VeryDiff for other properties:

4.1 Confidence-Based Classification Equivalence

Unfortunately, differential verification is surprisingly inefficient in comparison to naive verification when proving Top-1 equivalence [2], i.e., in verifying the property that two NNs always output the same classification. However, differential verification can yield significant speedups in verification when we constrain the domain for which we verify classification equivalence to inputs for which the first NN is confident [14]. This enables us to provide *global* guarantees on the equivalence between an NN and its pruned/further trained version.

4.2 Error Bounds for Fully-Homomorphic Encryption

While originally conceived to analyze the impact of changes to the weight matrices of an NN, differential verification and the VeryDiff tool have since been extended to the analysis of changes in an NN's activation function. This analysis has interesting applications in fully-homomorphic encryption (FHE): While there exist reasonably efficient FHE schemes, such as CKKS [3], that enable us to run NN inference on untrusted hardware, these schemes require that the NN only uses addition and multiplication as elementary operations. However, many NNs leverage non-linear activation functions such as ReLU which *cannot* be expressed using addition and multiplication.

To circumvent this issue, modern NN workflows for FHE circuits approximate activation functions with polynomials. Here, differential verification enables us to compute bounds on the *difference* between the original NN and its polynomial approximation [7], essentially proving ε-equivalence between the original NN and its FHE circuit.

5 Outlook

Beyond equivalence verification and guaranteeing correctness for FHE NNs, we are exploring further applications in ongoing work: Differential verification can be leveraged to substitute a provably safe NN controller [11, 12] by a pruned variant and we are investigating its application for verifying global robustness [1]. Other potential applications include formally verifying fairness or monotonicity as well as quantitative (instead of qualitative) relational guarantees.

References

[1] Anagha Athavale et al. "Verifying Global Two-Safety Properties in Neural Networks with Confidence". In: *36th International Conference on Computer Aided Verification*. CAV 2024 (Montréal, QC, Canada, July 24–27, 2024). Ed. by Arie Gurfinkel and Vijay Ganesh. LNCS. Springer, 2024, pp. 329–351. DOI: 10.1007/978-3-031-65630-9_17.

[2] Marko Kleine Büning, Philipp Kern, and Carsten Sinz. "Verifying Equivalence Properties of Neural Networks with ReLU Activation Functions". In: *26th International Conference on Principles and Practice of Constraint Programming*. CP 2020 (Louvain-la-Neuve, Belgium, Sept. 7–11, 2020). Ed. by Helmut Simonis. LNCS. Springer, 2020, pp. 868–884. DOI: 10.1007/978-3-030-58475-7_50.

[3] Jung Hee Cheon et al. "Homomorphic Encryption for Arithmetic of Approximate Numbers". In: *23rd International Conference on the Theory and Applications of Cryptology and Information Security*. ASIACRYPT 2017 (Hong Kong, China, Dec. 3–7, 2017). Ed. by Tsuyoshi Takagi and Thomas Peyrin. LNCS. Springer, 2017, pp. 409–437. DOI: 10.1007/978-3-319-70694-8_15.

[4] Timon Gehr et al. "AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation". In: *IEEE Symposium on Security and Privacy*. SP 2018 (San Francisco, CA, USA, May 21–23, 2018). IEEE Computer Society, 2018, pp. 3–18. DOI: 10.1109/SP.2018.00058.

[5] Khalil Ghorbal, Eric Goubault, and Sylvie Putot. "The Zonotope Abstract Domain Taylor1+". In: *21st International Conference on Computer Aided Verification*. CAV 2009 (Grenoble, France, June 26–July 2, 2009). Ed. by Ahmed Bouajjani and Oded Maler. LNCS. Springer, 2009, pp. 627–633. DOI: 10.1007/978-3-642-02658-4_47.

[6] Guy Katz et al. "Reluplex: a calculus for reasoning about deep neural networks". In: *Formal Methods Syst. Des.* 60.1 (2022), pp. 87–116. DOI: 10.1007/S10703-021-00363-7.

[7] Philipp Kern, Edoardo Manino, and Carsten Sinz. "Certified Error Analysis of Homomorphically Encrypted Neural Networks". In: *Second International Symposium on AI Verification*. SAIV 2025 (Zagreb, Croatia, July 21–22, 2025). Ed. by Mirco Giacobbe and Anna Lukina. LNCS. Springer, 2025, pp. 156–179. DOI: 10.1007/978-3-031-99991-8_8.

[8] Brandon Paulsen, Jingbo Wang, and Chao Wang. "ReluDiff: differential verification of deep neural networks". In: *42nd International Conference on Software Engineering*. ICSE 2020 (Seoul, South Korea, June 27–July 19, 2020). Ed. by Gregg Rothermel and Doo-Hwan Bae. ACM, 2020, pp. 714–726. DOI: 10.1145/3377811.3380337.

[9] Brandon Paulsen et al. "NEURODIFF: Scalable Differential Verification of Neural Networks using Fine-Grained Approximation". In: *35th IEEE/ACM International Conference on Automated Software Engineering*. ASE 2020 (Melbourne, Autralia, Sept. 21–25, 2020). IEEE, 2020, pp. 784–796. DOI: 10.1145/3324884.3416560.

[10] Gagandeep Singh et al. "Fast and Effective Robustness Certification". In: *Annual Conference on Neural Information Processing Systems*. NeurIPS 2017 (Montréal, QC, Canada, Dec. 3–8, 2018). Ed. by Samy Bengio et al. 2018, pp. 10825–10836.

[11] Samuel Teuber, Debasmita Lohar, and Bernhard Beckert. "Of Good Demons and Bad Angels: Guaranteeing Safe Control under Finite Precision". In: *25th Conference on Formal Methods in Computer-Aided Design*. FMCAD 2025 (Menlo Park, CA, USA, Oct. 6–10, 2025). Ed. by Ahmed Irfan and Daniela Kaufmann. TU Wien Academic Press, 2025. DOI: 10.34727/2025/ISBN.978-3-85448-084-6_12.

[12] Samuel Teuber, Stefan Mitsch, and André Platzer. "Provably Safe Neural Network Controllers via Differential Dynamic Logic". In: *Annual Conference on Neural Information Processing Systems*. NeurIPS 2024 (Vancouver, BC, Canada, Dec. 10–15, 2024). Ed. by Amir Globersons et al. 2024.

[13] Samuel Teuber et al. "Geometric Path Enumeration for Equivalence Verification of Neural Networks". In: *33rd IEEE International Conference on Tools with Artificial Intelligence*. ICTAI 2021 (Washington, DC, USA, Nov. 1–3, 2020). IEEE, 2021, pp. 200–208. DOI: 10.1109/ICTAI52525.2021.00035.

[14] Samuel Teuber et al. "Revisiting Differential Verification: Equivalence Verification with Confidence". In: *31st International Conference Tools and Algorithms for the Construction and Analysis of Systems*. TACAS 2025 (Hamilton, ON, Canada, May 3–8, 2025). Ed. by Arie Gurfinkel and Marijn Heule. LNCS. Springer, 2025, pp. 257–278. DOI: 10.1007/978-3-031-90653-4_13.