Revisiting Differential Verification: Equivalence Verification with Confidence

Samuel Teuber(✉), Philipp Kern, Marvin Janzen, and Bernhard Beckert

Karlsruhe Institute of Technology, Karlsruhe, Germany
teuber@kit.edu

Abstract. When validated neural networks (NNs) are pruned (and retrained) before deployment, it is desirable to prove that the new NN behaves *equivalently* to the (original) reference NN. To this end, our paper revisits the idea of *differential verification* which performs reasoning on differences between NNs: On the one hand, our paper proposes a novel abstract domain for differential verification admitting more efficient reasoning about equivalence. On the other hand, we investigate empirically and theoretically which equivalence properties are (not) efficiently solved using differential reasoning. Based on the gained insights, and following a recent line of work on confidence-based verification, we propose a novel equivalence property that is amenable to Differential Verification while providing guarantees for *large parts of the input space* instead of small-scale guarantees constructed w.r.t. predetermined input points. To this end, we propose an improved approach for (approximately) constraining an NN's softmax confidence. We implement our approach in a new tool called *VeryDiff* and perform an extensive evaluation on numerous old and new benchmark families, including new pruned NNs for particle jet classification in the context of CERN's LHC where we observe median speedups $> 300\times$ over the State-of-the-Art verifier α, β-CROWN.

Keywords: Neural Network Verification · Equivalence Verification · Differential Verification · Confidence-Based Verification · Zonotopes.

1 Introduction

Specifying what an NN is supposed to do is a difficult problem, that is at most partially solved. One class of specifications that is comparatively easy to formalize are equivalence properties: Given an "old" reference NN f_1, we aim to prove that a "new" NN f_2 behaves in some way equivalently. For example ε equivalence [17,22] requires that the numerical outputs of f_1 and f_2 for the same input point differ by at most ε or Top-1 equivalence [17] requires that the two NNs' classifications match. Known applications of equivalence verification are verification after retraining or pruning [33], student-teacher training [17,28], analysis of sensitivity to NN-based preprocessing steps [21] and construction of quantized NNs [19]. Several publications [9,14,17,22,23,28,33] have proposed methods for

A. Gurfinkel and M. Heule (Eds.): TACAS 2025, LNCS 15697, pp. 257–278, 2025.
https://doi.org/10.1007/978-3-031-90653-4_13

the verification of equivalence properties (sometimes calling it "approximate conformance" [14]). While it is known that equivalence verification w.r.t. the ε equivalence (Definition 2) property is coNP-complete [28], the complexity-theoretic status of Top-1 equivalence verification (Definition 3) remained unclear so far.

Although the field of equivalence verification has already seen significant improvements in scalability, it has also been critiqued for its shortcomings in practical applicability [20]. On the one hand, further speedups are necessary to make the tooling applicable to realistic case studies. On the other hand, besides greater efficiency, we also require equivalence properties that hold more globally: "Classic" equivalence properties have often only been evaluated w.r.t. small ε-balls around individual datapoints. In this work, we take steps in both directions of greater applicability. First, we extend the well-known and efficient Zonotope abstract domain to differential reasoning and demonstrate its superior performance over pre-existing equivalence verification approaches by coupling *Differential Zonotopes* with a suitable refinement heuristic and generator compression techniques. Secondly, we propose a confidence-based classification equivalence property that can be verifiable on larger parts of an input space by constraining the input space w.r.t. the reference NN's confidence in its outputs.

Contribution. This work encompasses multiple theoretical and practical contributions to the field of equivalence verification:

(C1) We prove that deciding if two ReLU NNs are Top-1 equivalent is a coNP-complete decision problem, i.e. it is as hard as ε-equivalence verification [28] or the classic NN verification problem [15, 24].
(C2) We propose *Differential Zonotopes*: An abstract domain that allows the usage of the differential verification methodology w.r.t. the Zonotope abstract domain by propagating a Zonotope bounding the *difference* between two NNs in lock-step with a reachability analysis for the individual NNs.
(C3) We implement the proposed approach in a new tool and evaluate its efficiency. For ε equivalence we achieve median speedups >10 for 8 of 9 comparisons (4.5 in the other case).
(C4) For Top-1 equivalence we demonstrate empirically that Differential Zonotopes do not aid verification. We provide a theoretical intuition for this observation and demonstrate this is a fundamental limitation of Differential Verification in general – independently of the chosen abstract domain.
(C5) Based on these insights, we propose a new confidence-based equivalence property for classification NN which is 1. verifiable on larger parts of the input space of NNs; 2. amenable to differential verification. Furthermore, we propose a simpler *and* more precise linear approximation of the softmax function in comparison to prior work [2]. In additional experiments, we demonstrate that our tool can certify 327% more benchmark queries than α, β-CROWN for confidence-based equivalence.

Related Work. Prior work on Zonotope-based NN verification [10, 27] verified non-relational properties w.r.t. a single NN. We extend this work by providing a

methodology that allows reasoning about *differences* between NNs. Equivalence properties, can in principle, be analyzed using classical NN verification techniques such as α, β-CROWN [25,35,37–39] for a single NN by building "product-networks" (similar to product-programs in classical program verification [6]), but early work on NN equivalence verification demonstrated that this approach is inefficient due to the accumulation of overapproximation errors in the two independent NNs [22]. While this view was recently challenged by [14], Section 7 conclusively demonstrates that tailored verification tools still outperform State-of-the-Art "classical" NN verification tools for NNs with similar weight structures.

Prior work suggested using Star-Sets for equivalence verification without analyzing weight differences and heavily relied on LP solving [28]. Prior work on differential verification [22,23] did not verify the equivalence of classifications and also fell short of using the Zonotope abstract domain. In this work, we argue that analyzing classification-based equivalence properties is particularly important, because numerical bounds alone (e.g. the ones provided by prior differential verification [22, 23]) *cannot* guarantee equivalence in *outcome*. Section 7 compares to equivalence verifiers [17, 23, 28]. In another line of work, QEBVerif [40] proposes a sound and complete analysis technique tailored to quantized NNs which is not directly applicable to other kinds of NNs studied in our evaluation.

Another line of research analyzes relational properties w.r.t. multiple runs of a *single* NN. All listed works do not verify equivalence. For example, Banerjee *et al.* [5] propose an abstract domain for relational properties, but assume that all executions happen on the same NN. This makes their approach incompatible with our benchmarks which require the analysis of multiple *different* NN. Another incomplete relational verifier [4] also assumes executions on a single NN and requires tailored relaxations not available for equivalence properties. Encoding relational properties via product NNs has also been explored by Athavale *et al.* [2]. We compare against Marabou (which they used) and we prove that our approximation of softmax, though simpler, is always more precise.

Overview. Section 2 introduces the necessary background on NN verification via Zonotopes, equivalence verification and confidence based NN verification. Section 3 proves the coNP-completeness of Top-1 equivalence. Subsequently, we introduce *Differential Zonotopes* as an abstract domain for differential reasoning via Zonotopes (Section 4) and explain how Differential Zonotopes can be used to perform equivalence verification (Section 5). Section 6 explains why Top-1 equivalence does not benefit from differential reasoning in general and derives a new confidence-based equivalence property that may hold on large parts of the input space and can be verified more efficiently using differential verification. Finally, Section 7 provides an evaluation of our approach. The appendix of our paper can be found in the preprint [29].

2 Background

We deal with the verification of piece-wise linear, feed-forward neural networks (NNs). A NN with input dimension $I \in \mathbb{N}$ and output dimension $O \in \mathbb{N}$ and

$L \in \mathbb{N}$ layers can be summarized as a function $f : \mathbb{R}^I \to \mathbb{R}^O$ which maps input vectors $\mathbf{x}^{(0)} \in \mathbb{R}^I$ to output vectors $\mathbf{x}^{(L)} \in \mathbb{R}^O$. In more detail, each layer of the NN consists of an affine transformation $\tilde{\mathbf{x}}^{(i)} = W^{(i)}\mathbf{x}^{(i-1)} + b^{(i)}$ (for a matrix $W^{(i)}$ and a vector $b^{(i)}$) followed by the application of a non-linear function $\mathbf{x}^{(i)} = h^{(i)}\left(\tilde{\mathbf{x}}^{(i)}\right)$. Many feed-forward architectures can be compiled into this format [26]. We focus on the case of NNs with ReLU activations, i.e. $h^{(i)}\left(\tilde{\mathbf{x}}^{(i)}\right) = \mathrm{ReLU}\left(\tilde{\mathbf{x}}^{(i)}\right) = \max\left(0, \tilde{\mathbf{x}}^{(i)}\right)$ for all $1 \leq i \leq L$. $f^{(i)}(\mathbf{x})$ is the computation of the NN's first i layers. We uniformly denote vectors in bold ($\mathbf{v}$) and matrices in capital letters (M) and Affine Forms/Zonotopes are denoted as $\mathfrak{z}/\mathcal{Z}$.

NN Verification. A well-known primitive in the literature on NN verification are *Zonotopes* [10,12,27]: An abstract domain that allows the efficient propagation of an interval box over the input space through (piece-wise) affine systems:

Definition 1 (Zonotope). *A* Zonotope *with input dimension n and output dimension m is a collection of m* Affine Forms *of the structure*

$$\mathbf{g}\epsilon + c \qquad\qquad (\epsilon \in [-1,1]^n, \mathbf{g} \in \mathbb{R}^n, c \in \mathbb{R})$$

We denote a single Affine Form as a tuple $(\mathbf{g}, c)$. Given an Affine Form $\mathfrak{z} = (\mathbf{g}, c)$ and a vector $\mathbf{v} \in [-1,1]^d$ ($d \leq n$) we denote by $\mathfrak{z}(\mathbf{v})$ the Affine Form (or value for $d = n$) where the first d values of ϵ are fixed to $\mathbf{v}$, i.e. to $(\mathbf{g}_{d+1:n}, \mathbf{g}_{1:d}\mathbf{v} + c)$. For $\mathfrak{z} = (\mathbf{g}, c)$, we denote the set of points described by $\mathfrak{z}$ as $\langle\mathfrak{z}\rangle = \{\mathfrak{z}(\epsilon) \mid \epsilon \in [-1,1]^n\}$.

Via $\mathfrak{z}(\mathbf{v})$ we denote the values reachable given the (input) vector $\mathbf{v}$. To improve clarity, some transformations applied to Zonotopes will be described for the 1-dimensional case, i.e. to a single Affine Form. Nonetheless, a major advantage of Zonotopes lies in their Matrix representation: m Affine Forms are then a matrix $G \in \mathbb{R}^{n \times m}$ and a vector $\mathbf{c} \in \mathbb{R}^m$ (then denoted as $\mathcal{Z} = (G, \mathbf{c})$). A component of $\mathbf{g}$/a column of G is called a *generator*. Given a Zonotope $\mathcal{Z}$ we denote its i-th Affine Form (represented by G's i-th row and $\mathbf{c}$'s i-th component) as $\mathcal{Z}_i$. Similar to the affine forms, we define $\langle\mathcal{Z}\rangle = \{\mathbf{x} \in \mathbb{R}^m | \epsilon \in [-1,1]^n, \mathbf{x}_i = \mathcal{Z}_i(\epsilon)\}$. Zonotopes are a good fit for analyzing (piece-wise) linear NN as they are closed under affine transformations [3]:

Proposition 1 (Affine Zonotope Transformation [12]). *For some Zonotope $\mathcal{Z} = (G, \mathbf{c})$ and an affine transformation $h(\mathbf{x}) = W\mathbf{x} + \mathbf{b}$, the Zonotope $\hat{\mathcal{Z}} = (WG, W\mathbf{c} + \mathbf{b})$ exactly describes the transformation h applied to the points $x \in \langle\mathcal{Z}\rangle$, i.e. for all $d \leq n$ and $\mathbf{v} \in [-1,1]^d$: $\{W\mathbf{x} + \mathbf{b} \mid \mathbf{x} \in \langle\mathcal{Z}(\mathbf{v})\rangle\} = \langle\hat{\mathcal{Z}}(\mathbf{v})\rangle$*

This proposition implies $\{W\mathbf{x} + \mathbf{b} \mid \mathbf{x} \in \langle\mathcal{Z}\rangle\} = \langle\hat{\mathcal{Z}}\rangle$, but it is even stronger as it also guarantees a linear map from an input ($\mathbf{v}$) to all reachable outputs ($\langle\hat{\mathcal{Z}}(\mathbf{v})\rangle$). Zonotopes also admit efficient computation of interval bounds for their outputs:

Proposition 2 (Zonotope Output Bounds [12]). *Consider some Affine Form $\mathfrak{z} = (\mathbf{g}, c)$ it holds for all $\mathbf{x} \in [-1,1]^n$ that:*

$$\mathbf{g}\mathbf{x} + c \in \left[\underline{\mathfrak{z}}, \overline{\mathfrak{z}}\right] \coloneqq \left[c - \sum_{i=0}^{n} |\mathbf{g}_i|, c + \sum_{i=0}^{n} |\mathbf{g}_i|\right]$$

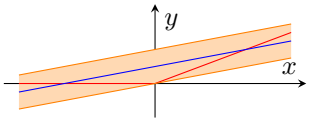


Fig. 1: ReLU approximation

Zonotopes cannot exactly represent the application of ReLU, but we can approximate the effect by distinguishing three cases: 1. The upper-bound $\overline{\hat{z}}$ is negative and thus $\mathrm{ReLU}(x) = 0$ for $x \in \langle \hat{z} \rangle$; 2. The lower-bound $\underline{\hat{z}}$ is positive and thus $\mathrm{ReLU}(x) = x$ for $x \in \langle \hat{z} \rangle$; 3. The ReLU-node is *instable* and its output is thus piece-wise linear. The first two cases can be represented as an affine transformation and the third case requires an approximation (see Figure 1 for intuition): *Interpolation* between $\mathrm{ReLU}(x) = 0$ and $\mathrm{ReLU}(x) = x$ (see the blue line in Figure 1 and $\lambda \mathbf{g}$ in Proposition 3) yields a representation, that we can turn into a sound overapproximation (meaning all possible output values of $\mathrm{ReLU}(x)$ are contained). This is achieved by adding a new generator that appropriately bounds the error of the interpolated function (see orange lines in Figure 1 and $\frac{1}{2}\lambda \underline{\hat{z}}$ in Proposition 3). This result is summarized as follows:

Proposition 3 (ReLU **Zonotope Transformation [27]).** *Consider some Affine Form* $\hat{z} = (\mathbf{g}, c)$. *Define a new Affine Form* $\hat{\hat{z}} = (\hat{\mathbf{g}}, \hat{c})$ *such that:*

$$\begin{array}{lll} \hat{\mathbf{g}} = 0 \in \mathbb{R}^n & \hat{c} = 0 & \text{if } \overline{\hat{z}} < 0 \\ \hat{\mathbf{g}} = \mathbf{g} \in \mathbb{R}^n & \hat{c} = c & \text{if } \underline{\hat{z}} > 0 \\ \hat{\mathbf{g}} = \begin{pmatrix} \lambda \mathbf{g} \\ \frac{1}{2}\lambda \underline{\hat{z}} \end{pmatrix} \in \mathbb{R}^{n+1} & \hat{c} = c - \frac{1}{2}\lambda \underline{\hat{z}} & \text{else} \end{array}$$

for $\lambda = \frac{\overline{\hat{z}}}{(\overline{\hat{z}} - \underline{\hat{z}})}$. *Then* $\hat{\hat{z}}$ *guarantees for all* $d \leq n$ *and* $\mathbf{v} \in [-1, 1]^d$ *that:*

$$\{\mathrm{ReLU}(x) | x \in \langle \hat{z}(\mathbf{v}) \rangle\} \subseteq \langle \hat{\hat{z}}(\mathbf{v}) \rangle$$

NN verification via Zonotopes typically proceeds as follows: An input set described as Zonotope is propagated through the NN using the transformers from Propositions 1 and 3. This yields an overapproximation of the NN's behavior. Depending on the verification property, one can either check the property by computing the Zonotope's bounds (Proposition 2) or by solving a linear program. If a property cannot be established, the problem is refined by either splitting the input space, w.r.t. its dimensions (input-splitting, e.g. [27]) or w.r.t. a particular neuron to eliminate the ReLU's nonlinearity (neuron-splitting, e.g. [3]).

Equivalence Verification. To show that two NNs f_1, f_2 behave equivalently, we can, for example, verify that the NNs' outputs are equal up to some ε:

Definition 2 (ε **Equivalence [17,22]).** *Given two NNs* $f_1, f_2 : \mathbb{R}^I \to \mathbb{R}^O$ *and an input set* $X \subseteq \mathbb{R}^I$ *we say* f_1 *and* f_2 *are* ε equivalent *w.r.t. a p-norm iff for all* $\mathbf{x} \in X$ *it holds that* $\|f_1(\mathbf{x}) - f_2(\mathbf{x})\|_p < \varepsilon$

Deciding ε equivalence is coNP-complete [28]. Another line of work proposes verification of Top-1 equivalence which is important for classification NNs [17]:

Definition 3 (Top-1 Equivalence [17]). *Given two NNs* $f_1, f_2 : \mathbb{R}^I \to \mathbb{R}^O$ *and an input set* $X \subseteq \mathbb{R}^I$*,* f_1 *and* f_2 *are* Top-1 equivalent *iff for all* $\mathbf{x} \in X$ *we have* $\operatorname{argmax}_i f_{1,i}(\mathbf{x}) = \operatorname{argmax}_i f_{2,i}(\mathbf{x})$*. More formally, Top-1 equivalence guarantees that for all* $k \in [1, O], \mathbf{x} \in X$ *if for all* $j \neq k$ *it holds that* $f_{1,k}(\mathbf{x}) \geq f_{1,j}(\mathbf{x})$ *then it also holds that* $f_{2,k}(\mathbf{x}) \geq f_{2,j}(\mathbf{x})$ *for all* $j \neq k$*.*

For ε equivalence, prior work by Paulsen *et al.* [22,23] introduced *differential verification*: For two NNs of equal depth (i.e. $L_1 = L_2$), equivalence can be verified more effectively by reasoning about weight differences. To this end, Paulsen *et al.* used symbolic intervals [30,31] not only for bounding values of a single NN, but to bound the *difference* of values between two NNs f_1, f_2, i.e. at any layer $1 \leq i \leq L$ we compute two linear symbolic bound functions $l_\Delta^{(i)}(\mathbf{x}), u_\Delta^{(i)}(\mathbf{x})$ such that $l_\Delta^{(i)}(\mathbf{x}) \leq f_1^{(i)}(\mathbf{x}) - f_2^{(i)}(\mathbf{x}) \leq u_\Delta^{(i)}(\mathbf{x})$. Differential bounds are computed by propagating symbolic intervals through two NNs in lock-step. This enables the computation of bounds on the difference at every layer.

Confidence Based Verification Many classification NNs provide a confidence for their classification by using the Softmax function $\operatorname{softmax}_i(x) = \frac{e^{x_i}}{\sum_{i=1}^{n} x_i}$. A recent line of work [2] proposes to use the confidence values classically provided by classification NNs as a starting point for verification. For example, Athavale et al. [2] propose a global, confidence-based robustness property which stipulates that inputs classified with high confidence must be robust to noise (i.e. we require the same classification for some bounded perturbation of the input). While this introduces reliance on the NN's confidence, it enables *global* specifications verified on the full input space and this limitation can be addressed by an orthogonal direction of research that aims at training calibrated NNs [1,13], i.e. NNs that correctly estimate the confidence of their predictions.

3 Complexity of Top-1 Verification

Prior work showed that the classic NN verification problem for a single ReLU-NN is an NP-complete problem [15,24]. Similarly, the problem of finding a violation for ε equivalence is NP-complete for ReLU-NNs as the single-NN verification problem can be reduced to this setting [28]. We now show, that finding a violation of Top-1 equivalence (Top-1-Net-Equiv) is also NP-complete implying that proving absence of counterexamples is coNP-complete (proof see [29, Sec. A.1]):

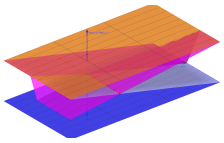
Theorem 1 (Top-1-Net-Equiv is coNP-complete). *Let* $X \subseteq \mathbb{R}^I$ *be some polytope over the input space of two* ReLU*-NNs* f_1, f_2*. Deciding whether there exists* $\mathbf{x} \in X$ *and a* $k \in [1, O]$ *s.t.* $(f_1(\mathbf{x}))_k \geq (f_1(\mathbf{x}))_i$ *for all* $i \in [1, O]$ *but for some* $j \in [1, O]$ *it holds that* $(f_2(\mathbf{x}))_k < (f_2(\mathbf{x}))_j$ *is NP-complete.*

Proof Sketch. Our proof differs from the proof for ε equivalence in the reduced problem which is the "classical" NN verification problem [15,24] for ε equivalence verification. To apply a similar proof technique to Top-1 equivalence, we require an NN verification instance with only *strict* inequality constraints in the input

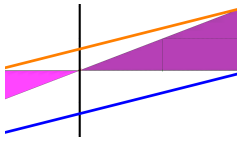
and output. Therefore, we first prove the NP-completeness of verifying *strict* inequality constraints (Definition 8 in [29, Sec. A.1]) by adapting prior proofs [15,24]. Then, we can reduce this problem to Top-1 equivalence verification. The reduction works by constructing an instance of Top-1-Net-Equiv that has a Top-1 violation iff the violating input also satisfies the constraints of the original NN verification problem. □

4 Equivalence Analysis via Zonotopes

Given two NNs $f_1, f_2 : \mathbb{R}^I \to \mathbb{R}^O$ with L layers each, we follow the basic principle of differential verification, i.e. we bound the difference $f_1^{(i)} - f_2^{(i)}$ at every layer $1 \leq i \leq L$. For our presentation, we assume that all layers of the NN have the same width, i.e. the same number of nodes. In practice, this limitation can be lifted by enriching the thinner layer with zero rows [33]. Our approach to differential verification is summarized in Algorithm 1: First and foremost, we perform a classic reachability analysis via Zonotopes for the two NNs f_1, f_2. To this end, we propagate a given input Zonotope $\mathcal{Z}_{\text{in}}$ through both NNs resulting in output Zonotopes $\mathcal{Z}', \mathcal{Z}''$. This part of the analysis uses the well-known Zonotope transformers described in Propositions 1 and 3. The individual reachability analysis is complemented with the computation of the *Differential Zonotope* $\mathcal{Z}^\Delta$ which is initialized with 0 (meaning the NN's inputs are initially equal) and computed in lockstep to the computation of $\mathcal{Z}'$ and $\mathcal{Z}''$: At every layer, we overapproximate the maximal deviation between the two NNs. Using the transformers described in the remainder of this Section, we can prove that the Differential Zonotope always overapproximates the difference between the two NNs (proof see [29, Sec. A.2]):



(a) Bounds on difference of two instable neurons.



(b) Projection of Figure 2a w.r.t. neuron difference.

Fig. 2: Visualization for the construction of a new Affine Form via ReLU$_\Delta$

Theorem 2 (Soundness). *Let f_1, f_2 be two feed-forward* ReLU*-NNs, $\mathcal{Z}_{in}$ some Zonotope mapping n generators to I dimensions and $\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^\Delta$ the output of* Reach$_\Delta$ $(f_1, f_2, \mathcal{Z}_{in})$*. The following statements hold for all $\mathbf{v} \in [-1, 1]^n$:*

1. $f_1(\mathcal{Z}_{in}(\mathbf{v})) \in \langle \mathcal{Z}'(\mathbf{v}) \rangle$ *and* $f_2(\mathcal{Z}_{in}(\mathbf{v})) \in \langle \mathcal{Z}''(\mathbf{v}) \rangle$
2. $\big(f_1(\mathcal{Z}_{in}(\mathbf{v})) - f_2(\mathcal{Z}_{in}(\mathbf{v})) \big) \in \langle \mathcal{Z}^\Delta(\mathbf{v}) \rangle$

For the descriptions of transformations, we again focus on a single Affine Form. In this section, we denote the representation of an Affine Form as $\mathfrak{z} = (\mathbf{e}, \mathbf{a}, c)$ where $\mathbf{e}$ are the n original (*e*xact) generators present in an Affine Form of $(\mathcal{Z}_{\text{in}})_i$ and $\mathbf{a}$ are the p (*a*pproximate) generators added via ReLU transformations. For Differential Affine Forms we split the approximate generators into $\mathbf{a}'^\Delta, \mathbf{a}''^\Delta$ and $\mathbf{a}^\Delta$, distinguishing their origin ($\mathfrak{z}', \mathfrak{z}''$ or generators added to $\mathfrak{z}^\Delta$ directly). This

Algorithm 1 Verification with Differential Zonotopes

Input: NNs $g_1, g_2 : \mathbb{R}^I \to \mathbb{R}^O$ with L layers, Input-Zonotope $Z_{\text{in}} = (G_{\text{in}}, b_{\text{in}})$
Output: Reachable Zonotopes $\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^\Delta$

```
procedure REACH_Δ(g_1, g_2, Z_in)
    Z' ← Z_in
    Z'' ← copy(Z_in)
    Z^Δ ← (0,0) ∈ R^{I×I} × R^I                ▷ Initialize Differential Zonotope with 0
    for l ∈ [1, L] do
        if layer l is affine then
            Z^Δ ← AFFINE_Δ(Z^Δ, Z', Z'', W_{1/2}^(l), b_{1/2}^(l))
                                                ▷ See Lemma 1
            Z' ← AFFINE(Z', W_1^(l), b_1^(l))
                                                ▷ See Transformation in Proposition 1
            Z'' ← AFFINE(Z'', W_2^(l), b_2^(l))
                                                ▷ See Transformation in Proposition 1
        else                                    ▷ Transformation for ReLU Layer
            Ẑ', Ẑ'' ← RELU(Z'), RELU(Z'')
                                                ▷ See Transformation in Proposition 3
            Z^Δ ← RELU_Δ(Z^Δ, Z', Z'', Ẑ', Ẑ'')
                                                ▷ See Lemma 2
            Z', Z'' ← Ẑ', Ẑ''
    return Z', Z'', Z^Δ
```

yields the Affine Form $\mathfrak{Z}^\Delta = \left(\mathbf{e}^\Delta, \mathbf{a}'^\Delta, \mathbf{a}''^\Delta, \mathbf{a}^\Delta, c^\Delta\right)$. We assume that the corresponding vectors have equal dimensions (i.e. $\mathbf{e}'$,$\mathbf{e}''$ and $\mathbf{e}^\Delta$ all have dimension n_1; $\mathbf{a}'$ and $\mathbf{a}'^\Delta$ have dimension n_2 and $\mathbf{a}''$ as well as $\mathbf{a}''^\Delta$ have dimension n_3). The points described by two Affine Forms and a Differential Affine Form are then described via common generator values $\epsilon_\mathbf{1}, \epsilon_\mathbf{2}, \epsilon_\mathbf{3}, \epsilon_\mathbf{4}$ across all three Affine Forms. This definition naturally generalizes to the multidimensional (Zonotope) case by fixing all ϵ values across dimensions and Zonotopes. We also denote the points contained in a tuple $\left(\mathfrak{Z}', \mathfrak{Z}'', \mathfrak{Z}^\Delta\right)$ using $\langle\cdot\rangle$:

Definition 4 (Points Contained by Differential Affine Form). *Given two Affine Forms $\mathfrak{Z}' = (\mathbf{e}', \mathbf{a}', c'), \mathfrak{Z}'' = (\mathbf{e}'', \mathbf{a}'', c'')$ and a Differential Affine Form $\mathfrak{Z}^\Delta = \left(\mathbf{e}^\Delta, \mathbf{a}'^\Delta, \mathbf{a}''^\Delta, \mathbf{a}^\Delta, c^\Delta\right)$ with resp. a matching number of columns, we define the set of points described by $(\mathfrak{Z}', \mathfrak{Z}'', \mathfrak{Z}^\Delta)$ where n_1, n_2, n_3, n_4 are resp. the number of columns in $\mathbf{e}^\Delta, \mathbf{a}'^\Delta, \mathbf{a}''^\Delta, \mathbf{a}^\Delta$ and $\overline{n} = n_1 + n_2 + n_3 + n_4$:*

$$
\begin{aligned}
&\langle(\mathfrak{Z}', \mathfrak{Z}'', \mathfrak{Z}^\Delta)\rangle = \\
&\quad \Big\{ (x, y) \;\Big|\; \exists\, \epsilon_\mathbf{1} \in [-1,1]^{n_1},\ \epsilon_\mathbf{2} \in [-1,1]^{n_2},\ \epsilon_\mathbf{3} \in [-1,1]^{n_3},\ \epsilon_\mathbf{4} \in [-1,1]^{n_4} \\
&\qquad\qquad x = \mathbf{e}'\epsilon_\mathbf{1} + \mathbf{a}'\epsilon_\mathbf{2} + c' \ \wedge \\
&\qquad\qquad y = \mathbf{e}''\epsilon_\mathbf{1} + \mathbf{a}''\epsilon_\mathbf{3} + c'' \ \wedge \\
&\qquad\qquad (x - y) = \mathbf{e}^\Delta\epsilon_\mathbf{1} + \mathbf{a}'^\Delta\epsilon_\mathbf{2} + \mathbf{a}''^\Delta\epsilon_\mathbf{3} + \mathbf{a}^\Delta\epsilon_\mathbf{4} + c^\Delta \Big\}
\end{aligned}
$$

Omitting the computation of $\mathcal{Z}^{\Delta}$, REACH_{Δ} corresponds to computing reachable outputs of f_1, f_2 w.r.t. a common input Zonotope $\mathcal{Z}_{\text{in}}$. This algorithm implicitly computes the *naive* Differential Affine Form $\mathfrak{z}' - \mathfrak{z}''$, i.e. the affine form $\mathfrak{z}^{\Delta} = (\mathbf{e}' - \mathbf{e}'', \mathbf{a}', -\mathbf{a}'', 0, c' - c'')$.

Affine Transformations. For affine transformations, we construct a transformation based on the insights from Differential Symbolic Bounds [22] that exactly two quantities determine the output difference: First, the difference accumulated so far (represented by the current Differential Affine Form $\mathfrak{z}_{\Delta}$ and scaled by the layer's affine transformation); second, the difference between the affine transformations in the current layer. Given two affine transformations $w_1^{(l)}x + b_1^{(l)}$ and $w_2^{(l)}x + b_2^{(l)}$ and Affine Forms $\mathfrak{z}' = (\mathbf{e}', \mathbf{a}', c'), \mathfrak{z}'' = (\mathbf{e}'', \mathbf{a}'', c''), \mathfrak{z}^{\Delta} = \left(\mathbf{e}^{\Delta}, \mathbf{a}'^{\Delta}, \mathbf{a}''^{\Delta}, \mathbf{a}^{\Delta}, c^{\Delta}\right)$ we propose AFFINE_{Δ} which returns a new Affine Form $\hat{\mathfrak{z}}^{\Delta} = \left(\hat{\mathbf{e}}^{\Delta}, \hat{\mathbf{a}}'^{\Delta}, \hat{\mathbf{a}}''^{\Delta}, \hat{\mathbf{a}}^{\Delta}, \hat{c}^{\Delta}\right)$ with:

$$\begin{aligned} \hat{\mathbf{e}}^{\Delta} &= w_1^{(l)}\mathbf{e}^{\Delta} + \left(w_1^{(l)} - w_2^{(l)}\right)\mathbf{e}'' \quad & \hat{\mathbf{a}}'^{\Delta} &= w_1^{(l)}\mathbf{a}'^{\Delta} \\ \hat{\mathbf{a}}''^{\Delta} &= w_1^{(l)}\mathbf{a}''^{\Delta} + \left(w_1^{(l)} - w_2^{(l)}\right)\mathbf{a}'' \quad & \hat{\mathbf{a}}^{\Delta} &= w_1^{(l)}\mathbf{a}^{\Delta} \\ \hat{c}^{\Delta} &= w_1^{(l)}c^{\Delta} + \left(w_1^{(l)} - w_2^{(l)}\right)c'' + \left(b_1^{(l)} - b_2^{(l)}\right) \end{aligned} \tag{1}$$

We scale prior differences by $w_1^{(l)}$ and add the new difference $(w_1^{(l)} - w_2^{(l)})$ to $\hat{\mathbf{e}}^{\Delta}$, $\hat{\mathbf{a}}''$ (scaled by the reachable values of f_2, i.e. by $\mathfrak{z}''$) and $\hat{c}^{\Delta}$ (and resp. $(b_1^{(l)} - b_2^{(l)})$). The transformation is sound (proof in [29, Sec. A.2] generalizes to Zonotopes):

Lemma 1 (Soundness of AFFINE_{Δ}). *For affine transformations $\alpha_1(x) = w_1^{(l)}x + b_1^{(l)}$ and $\alpha_2(x) = w_2^{(l)}x + b_2^{(l)}$ and Affine Forms $\mathfrak{z}' = (\mathbf{e}, \mathbf{a}', c'), \mathfrak{z}'' = (\mathbf{e}'', \mathbf{a}'', \mathbf{c}''), \mathfrak{z}^{\Delta} = \left(\mathbf{e}^{\Delta}, \mathbf{a}^{\Delta}, \mathbf{a}^{\Delta}, \mathbf{a}^{\Delta}, c^{\Delta}\right)$ the transformation Affine_{Δ} is sound, i.e. if $\hat{\mathfrak{z}}^{\Delta}$ is the result of AFFINE_{Δ}, then for all $d \leq n$ and $\mathbf{v} \in [-1, 1]^d$ with $(x, y) \in \langle(\mathfrak{z}'(\mathbf{v}), \mathfrak{z}''(\mathbf{v}), \mathfrak{z}^{\Delta}(\mathbf{v}))\rangle$ and $\hat{\mathfrak{z}}', \hat{\mathfrak{z}}''$ the outputs of AFFINE (Proposition 1) we get:*

$$(\alpha_1(x), \alpha_2(y)) \in \langle(\tilde{z}_1(v), \tilde{z}_2(v), \tilde{z}_{\Delta}(v))\rangle$$

ReLU *Transformations.* Since ReLU is piece-wise linear, we cannot hope to construct an exact transformation for this case. However, by carefully distinguishing the possible cases (e.g. both nodes are solely negative, one is negative one is instable, etc.) we provide linear representations for 6 out of the 9 cases (both nodes stable or a negative and an instable node) while overapproximating the other three. Given $\mathfrak{z}' = (\mathbf{e}', \mathbf{a}', c'), \mathfrak{z}'' = (\mathbf{e}'', \mathbf{a}'', c'')$ and a Differential Affine Form $\mathfrak{z}^{\Delta} = \left(\mathbf{e}^{\Delta}, \mathbf{a}'^{\Delta}, \mathbf{a}''^{\Delta}, \mathbf{a}^{\Delta}, c^{\Delta}\right)$ as well as $\hat{\mathfrak{z}}', \hat{\mathfrak{z}}''$ (the result of applying ReLU in the individual NNs), we compute the result of the ReLU_{Δ} transformation, using the case distinctions from Table 1. The transformation is sound (proof see [29, Sec. A.2]):

Lemma 2 (Soundness of ReLU_{Δ}). *Consider Affine Form $\mathfrak{z}' = (\mathbf{e}, \mathbf{a}', c'), \mathfrak{z}'' = (\mathbf{e}'', \mathbf{a}'', \mathbf{c}'')$ and a Differential Affine Form $\mathfrak{z}^{\Delta} = \left(\mathbf{e}^{\Delta}, \mathbf{a}'^{\Delta}, \mathbf{a}''^{\Delta}, \mathbf{a}^{\Delta}, c^{\Delta}\right)$. Let $\hat{\mathfrak{z}}'/\hat{\mathfrak{z}}''$ be the result of the* RELU *transformation on $\mathfrak{z}'/\mathfrak{z}''$ (see Proposition 3).*

Table 1: Case Distinction for the construction of a new Affine Form $\text{ReLU}_\Delta\left(\hat{\mathfrak{z}}^\Delta, \mathfrak{z}', \mathfrak{z}'', \hat{\mathfrak{z}}', \hat{\mathfrak{z}}''\right) = \hat{\mathfrak{z}}^\Delta = \left(\hat{\mathbf{e}}^\Delta, \hat{\mathbf{a}}'^\Delta, \hat{\mathbf{a}}''^\Delta, \hat{\mathbf{a}}^\Delta, \hat{c}^\Delta\right)$. Additions of vectors with different lengths signify addition for the components in the common-length prefix. We distinguish the nodes' different phases by positive (+), negative (–), and instable (∼). The variables λ, μ and ν are initialized as follows: $\lambda' = \frac{-\underline{\mathfrak{z}}'}{\overline{\mathfrak{z}}' - \underline{\mathfrak{z}}'}$, $\mu' = 0.5 * \lambda' \overline{\mathfrak{z}}'$ (accordingly for λ'', μ'') and $\lambda^\Delta = \text{clamp}\left(\frac{\overline{\mathfrak{z}}^\Delta}{\overline{\mathfrak{z}}^\Delta - \underline{\mathfrak{z}}^\Delta}, 0, 1\right)$, $\mu^\Delta = 0.5 * \max\left(-\underline{\mathfrak{z}}^\Delta, \overline{\mathfrak{z}}^\Delta\right)$, $\nu^\Delta = \lambda^\Delta * \max\left(0, -\underline{\mathfrak{z}}^\Delta\right)$

Neurons		**New Affine Form** $\hat{\mathfrak{z}}^\Delta$				
f_1	f_2	$\hat{\mathbf{e}}$	$\hat{\mathbf{a}}'^\Delta$	$\hat{\mathbf{a}}''^\Delta$	$\hat{\mathbf{a}}^\Delta$	$\hat{c}$
Both nodes stable						
–	–	0	0	0	0	0
–	+	$-\mathbf{e}''$	0	$-\mathbf{a}''$	0	$-c''$
+	–	$\mathbf{e}'$	$\mathbf{a}'$	0	0	c'
+	+	$\mathbf{e}^\Delta$	$\mathbf{a}'^\Delta$	$\mathbf{a}''^\Delta$	$\mathbf{a}^\Delta$	c^Δ
Negative + Instable						
∼	–	$\hat{\mathbf{e}}'$	$\hat{\mathbf{a}}'$	0	0	$\hat{c}'$
–	∼	$-\hat{\mathbf{e}}''$	0	$-\hat{\mathbf{a}}''$	0	$-\hat{c}''$
Positive + Instable						
∼	+	$\mathbf{e}^\Delta - \lambda'\mathbf{e}'$	$\mathbf{a}'^\Delta - \lambda'\mathbf{a}'$	$\mathbf{a}''^\Delta$	$((\mathbf{a}^\Delta)^T \mid \mu')^T$	$c^\Delta - \lambda' c' + \mu'$
+	∼	$\mathbf{e}^\Delta + \lambda''\mathbf{e}''$	$\mathbf{a}'^\Delta$	$\mathbf{a}''^\Delta + \lambda''\mathbf{a}''$	$((\mathbf{a}^\Delta)^T \mid \mu'')^T$	$c^\Delta + \lambda'' c'' - \mu''$
All Instable						
∼	∼	$\lambda^\Delta \mathbf{e}^\Delta$	$\lambda^\Delta \mathbf{a}'^\Delta$	$\lambda^\Delta \mathbf{a}''^\Delta$	$\left(\lambda^\Delta (\mathbf{a}^\Delta)^T \mid \mu^\Delta\right)^T$	$\lambda^\Delta c^\Delta + \nu^\Delta - \mu^\Delta$

Define $\hat{\mathfrak{z}}^\Delta = \left(\hat{\mathbf{e}}^\Delta, \hat{\mathbf{a}}^\Delta, \hat{\mathbf{a}}^\Delta, \hat{\mathbf{a}}^\Delta, c^\Delta\right)$ *such that it matches the case distinctions in Table 1. Then* $\hat{\mathfrak{z}}^\Delta$ *overapproximates the behavior of* ReLU*, i.e. for all* $d \leq n$ *and* $\mathbf{v} \in [-1, 1]^d$ *with* $(x, y) \in \langle(\mathfrak{z}'(\mathbf{v}), \mathfrak{z}''(\mathbf{v}), \mathfrak{z}^\Delta(\mathbf{v}))\rangle$ *we get that:*

$$(\text{ReLU}(x), \text{ReLU}(y)) \in \langle(\hat{\mathfrak{z}}'(\mathbf{v}), \hat{\mathfrak{z}}''(\mathbf{v}), \hat{\mathfrak{z}}^\Delta(\mathbf{v}))\rangle$$

Unfortunately, reasoning about the difference of two ReLUs is not particularly intuitive. The first 6 cases in Table 1 follow from substituting Node 1 and Node 2 values. In the other cases (*Positive + Instable* and *All Instable*), the difference is *not* linear in the input or output Zonotopes. Thus, we append an additional generator to the Differential Zonotope, i.e. to $\mathbf{a}^\Delta$. The approximation for the case of two instable neurons is plotted in Figure 2: The bounding planes ensure that 0 is always reachable and that the prior difference is within the bound.

5 Verification of Equivalence Properties

To verify equivalence with Reach_Δ, we proceed as follows: We propagate a Zonotope $\mathcal{Z}_{\text{in}}$ through the two NNs f_1, f_2 as explained above. Subsequently, we check a condition on the outputs $\left(\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^\Delta\right)$ that implies the desired equivalence property. If this check fails, we refine the Zonotope by splitting the input space. For the naive case, we compute $\mathcal{Z}^\Delta$ as explained in Section 4.

ε equivalence. Checking ε equivalence w.r.t. $\mathcal{Z}^{\Delta}$ works similarly to the approach for symbolic interval-based differential verification [22, 23]: We compute $\mathcal{Z}^{\Delta}$'s interval bounds and check for an absolute bound $> \varepsilon$. If all bounds are smaller, we have proven ε equivalence for $\langle \mathcal{Z}_{\text{in}} \rangle$ (see also Lemma 7 in [29, Sec. A.2]).

Top-1 Equivalence. Since Top-1 equivalence considers the order of outputs, it cannot be read off from $\mathcal{Z}^{\Delta}$'s bounds directly. We frame the property as an LP optimization problem in Definition 5. The LP has an optimal solution ≤ 0 if no $\mathbf{x} \in \langle \mathcal{Z}_{\text{in}} \rangle$ classified as k in f_1 is classified as j in f_2. Formally, the LP computes an upper bound for $(f_2(\mathbf{x}))_j - (f_2(\mathbf{x}))_k$ (maximized expression) under the condition that $(f_1(\mathbf{x}))_k$ is the maximum of $f_1(\mathbf{x})$ (first constraint) and under the additional condition that $\mathcal{Z}^{\Delta}$ bounds the difference between f_1 and f_2, resp. the difference between the reachable points in $\mathcal{Z}'$ and $\mathcal{Z}''$ (second constraint; see Definition 9 in [29, Sec. A.2]). The first constraint ensures a gap t between the largest and second largest output of f_1. In this section, we only consider $t = 0$.

Definition 5 (Top-1 Violation LP). *Given $\mathcal{Z}' = (G', \mathbf{c}') = (E', A', \mathbf{c}')$, $\mathcal{Z}'' = (E'', A'', \mathbf{c}'')$, $\mathcal{Z}^{\Delta} = \left(G^{\Delta}, \mathbf{c}^{\Delta}\right)$, a constant $t \geq 0$ and $k, j \in [1, O]$ with $k \neq j$ the* Top-1 Violation LP *is defined below. E' and E'' have n_1 columns, A' has n_2 columns, A'' has n_3 columns and A^{Δ} has n_4 columns. $\mathbf{x}$ contains generators for these matrices in order and has dimension $\bar{n} = n_1 + n_2 + n_3 + n_4$.*

$$
\begin{aligned}
\max_{\mathbf{x} \in [-1,1]^{\bar{n}}} \; & \left(E''\mathbf{x}_{1:n_1} + A''\mathbf{x}_{(n_1+n_2+1):(n_1+n_2+n_3)} + \mathbf{c}\right)_j \\
& - \left(E''\mathbf{x}_{1:n_1} + A''\mathbf{x}_{(n_1+n_2+1):(n_1+n_2+n_3)} + \mathbf{c}\right)_k \\
\textit{s.t.} \; & \left(G'\mathbf{x}_{1:(n_1+n_2)} + \mathbf{c}'\right)_l + t \leq \sum_{i=1}^{n_1+n_2} (G')_{k,i}\, x_i + (\mathbf{c}')_k \;\; \textit{for } l \neq k \\
& \mathcal{Z}' = \mathcal{Z}'' + \mathcal{Z}^{\Delta}
\end{aligned}
$$

If the LP's maximum is positive, we check whether the generated counterexample is spurious. Verification requires $\mathcal{O}\left(O^2\right)$ LP optimizations. In practice, we reuse the same constraint formulation for each k and optimize over all possible $j \neq k$ admitting warm starts. Our approach is sound (see proof in [29, Sec. A.2]):

Lemma 3 (Soundness for Top-1). *Consider $\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^{\Delta}$ provided by* REACH_{Δ} *w.r.t. Z_{in}: If for all $k, j \in [1, O]$ $(k \neq j)$ the Top-1 Violation LP with $t = 0$ has a maximum ≤ 0, then f_1, f_2 satisfy Top-1 equivalence w.r.t. inputs in $\langle Z_{in} \rangle$.*

Input Space Refinement. If verification fails, we split the input space in half and solve the verification problems separately. To this end, we use a heuristic to estimate the influence of splits along different input dimensions Splitting can improve the bounds in two ways: Either the reduced input range directly reduces the computed output bounds, or the reduced range reduces the number of instable neurons and hence reduces the over-approximation error w.r.t. output bounds. Our heuristic works similar to forward-mode gradient computation

estimating the influence of input dimensions on output bounds. For an analysis with n generators in $\mathcal{Z}_{\text{in}}$ and m generators in $\mathcal{Z}'/\mathcal{Z}''$ our heuristic requires two matrices $(n \times m)$ and two additional matrix multiplication per layer. For details see Appendix [29, Sec. B]; we leave a fine-grained analysis of refinement strategies to future work.

Generator Compression. To increase performance we analyze Z_{in}. In case an input dimension has range 0, we eliminate the generator in Z_{in}. This optimization speeds up equivalence verification for, e.g., targeted pixel perturbations [22].

Counterexample Generation. To generate counterexamples we check the Zonotope's center point for property violation. Additionally, we check the input points maximizing property violation according to our Zonotope approximation.

Completeness. As we only employ axis-aligned input-splitting, we cannot provide a completeness guarantee. However, our evaluation (Section 7) demonstrates, that this approach outperforms complete State-of-the-Art solvers.

6 Equivalence Verification for Classification NNs

Top-1 equivalence is particularly useful when verifying the equivalence of classification NNs. Indeed, there are examples of classification NNs which are ε-equivalent, but not Top-1 equivalent w.r.t. some input region (see also Appendix [29, Sec. C.3]). This underlines the importance of choosing the right equivalence property. Unfortunately, as we empirically show in the Appendix [29, Sec. C.2], classic Top-1 equivalence does not benefit from Differential Verification. Moreover, prior work on equivalence verification only provides guarantees for small parts of the input space, e.g. by proving Top-1 equivalence for ϵ-balls around given data points. As pointed out in orthogonal work [11], ϵ-balls around data points are not necessarily a semantically useful specification. Moreover, proving Top-1 equivalence on large parts of the input space is typically impossible, because pruning NNs invariably *will* change their behavior. This raises two questions: 1. Why does Top-1 equivalence not benefit from Differential Verification? 2. What equivalence property for classification NNs is verifiable on large parts of the input space while it can benefit from Differential Verification? We will answer these questions in order.

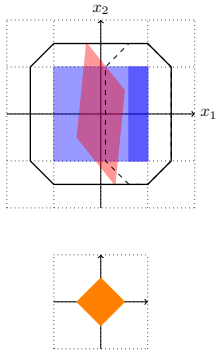


Fig. 3: Differential Zonotopes for Top-1 equivalence

Ineffectiveness for Top-1 equivalence. Our initial intuition would have been that the tighter bounds in $\mathcal{Z}^{\Delta}$ should also aid the verification of Top-1 equivalence. To refute this intuition, consider the sketch in Figure 3: The light blue area

represents a Zonotope for the output space reachable via f_1 (i.e. $\mathcal{Z}'$) and the red area describes a Zonotope for the output space reachable via f_2 (i.e. $\mathcal{Z}''$). The orange Zonotope describes $\mathcal{Z}^{\Delta}$, i.e. it is a bound for the difference between the two outputs. Depending on the nature of $\mathcal{Z}'$'s and $\mathcal{Z}^{\Delta}$'s generators (i.e. if they are shared or not), the output region for $\mathcal{Z}' - \mathcal{Z}^{\Delta}$ could reach as far as the solid black line. Thus, while $\mathcal{Z}^{\Delta}$ limits the difference for individual input points, it does not necessarily provide effective bounds for the reachable values of f_2 resp. $\mathcal{Z}''$. However, if via an LP formulation, the reachable values from $\mathcal{Z}'$ were restrained to $x_1 \geq 0.6$ (dark blue area on the right), then adding $\mathcal{Z}^{\Delta}$ would yield the black dashed region meaningfully constrains the behavior of f_2 beyond the constraints by $\mathcal{Z}''$ (red area). We could then prove that f_2's first dimension (x_1) is positive. Notably, our constraint on f_1 (≥ 0.6) is *stricter* than our constraint on f_2 (≥ 0) making the Differential Zonotope useful even if it has reachable values in negative x_1 direction. Top-1 equivalence imposes *equal* constraints on the difference between two dimensions in both NNs. Thus, if the difference has a negative bound in $\mathcal{Z}^{\Delta}$ (a likely outcome), Differential Verification, independently of the considered abstract domain, *cannot* help for Top-1 equivalence: While the concrete regions would look different for other abstract domains, the outcome would be the same given a negative differential bound for x_1.

Confidence-Based Equivalence We propose the notion of δ-Top-1 equivalence. Our key idea is to integrate the confidence values (computed by softmax) into the verified property. In contrast to the arbitrary threshold $x_1 \geq 0.6$, constraints based on confidence values have intuitive meaning:

Definition 6 (δ-Top-1 equivalence). *Given two NNs $f_1, f_2 : \mathbb{R}^I \to \mathbb{R}^O$, $\delta \in \left[\frac{1}{2}, 1\right)$ and an input region $Y \subseteq \mathbb{R}^I$, f_2 is* δ-Top-1 equivalent *w.r.t. f_1 iff f_1, f_2 are Top-1 equivalent w.r.t. $X_{f_1,Y}(\delta) = \{\mathbf{x} \in Y \mid \exists i \in [1, O] \;\; \text{softmax}_i(f_1(\mathbf{x})) \geq \delta\}$.*

We assume that f_1, f_2 are ReLU NNs with one softmax function after the last layer. Verifying δ-Top-1 equivalence achieves multiple objectives: First, even for larger Y (e.g. intervals over standard deviations for normalized inputs), we may provide guarantees for a suitable δ. Secondly, we rely on confidence estimates of a component we already trust: The reference NN. Finally, from a technical perspective, constraining the confidence level of f_1 to $\geq \delta$ while "only" requiring the same classification in f_2 achieves the asymmetry necessary for exploiting Differential Verification. Unfortunately, deciding δ-Top-1 equivalence is a coNP-hard decision problem (proof in [29, Sec. A.3]):

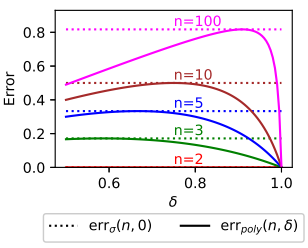


Fig. 4: Approximation Errors w.r.t. confidence δ: Ours (solid) and Athavale *et al.* [2] (dashed).

Corollary 1 (Complexity of δ-Top-1). *Let $Y \subseteq \mathbb{R}^I$ be a polytope, f_1, f_2 be two* ReLU-softmax-*NNs, $\frac{1}{2} \leq \delta < 1$. Deciding whether there exists a $\mathbf{y} \in Y$ and $k \in [1, O]$ s.t. $(f_1(x))_k \geq \delta$ but $\exists j \in [1, O] \;\; (f_2(\mathbf{x}))_k < (f_2(\mathbf{x}))_j$ is NP-hard.*

Due to the usage of softmax, the previous NP-membership argument does not apply in this case. To verify δ-Top-1 equivalence, we part from prior work on confidence-based verification [2] and propose the following approximation of all vectors $\mathbf{z}$ for which output i has a confidence $\geq \delta$ (proof in [29, Sec. A.3]):

Lemma 4 (Linear approximation of softmax**).** *For $\delta \in [1/2, 1)$ the following set relationship holds:*

$$\{\mathbf{z} \in \mathbb{R}^n \mid \operatorname{softmax}(\mathbf{z})_i \geq \delta\} \subseteq \left\{ \mathbf{z} \in \mathbb{R}^n \;\middle|\; \bigwedge_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{z}_i - \mathbf{z}_j \geq \ln\left(\frac{\delta}{1-\delta}\right) \right\} =: P_n(\delta)$$

We can then prove δ-Top-1 equivalence for $\delta \geq \frac{1}{2}$ by reusing the Top-1 Violation LP with t chosen appropriately (proof in [29, Sec. A.3]):

Corollary 2 (Soundness for δ-Top-1). *Given $\mathcal{Z}', \mathcal{Z}'', \mathcal{Z}^{\Delta}$ from* $\textsc{Reach}_{\Delta}$ *w.r.t. Z_{in}, $\frac{1}{2} \leq \delta < 1$. If for all $k, j \in [1, O]$ $(k \neq j)$ the Top-1 Violation LPs with $t = \ln\left(\frac{\delta}{1-\delta}\right)$ have maxima ≤ 0, then f_2 is δ-Top-1 equivalence w.r.t. f_1 on $\langle Z_{in} \rangle$.*

In comparison to the softmax approximation by Athavale *et al.* [2], we approximate via *one* polytope in the output space (in contrast to a 35-segment piece-wise linear approximation). Additionally, our approximation is parametrized in the confidence threshold δ while their approximation is uniform across confidence values. We now analyze the precision of the two approximations. Given a desired confidence level δ, we consider its error the maximal deviation below δ still encompassed by the approximation. For Athavale *et al.* [2] this maximal error is given as a function $\operatorname{err}_{\sigma}(n, \upsilon)$ in the input dimension n and the sigmoid approximation error υ (see Definition 10 in [29, Sec. A.3] or [2, Thm. 1]). For our approximation, we want to derive the error via the minimal confidence value that is still part of $P_n(\delta)$, i.e. as $\operatorname{err}_{\text{poly}}(n, \delta) = \delta - \min\{\max_i \operatorname{softmax}(\mathbf{z})_i \mid \mathbf{z} \in P_n(\delta)\}$. We can derive the following properties for our approximation error $\operatorname{err}_{\text{poly}}(n, \delta)$ in relation to the approximation error $\operatorname{err}_{\sigma}(n, \upsilon)$ incurred by Athavale *et al.* [2] (proof in [29, Sec. A.3]):

Lemma 5 (Maximal Error for our softmax **approximation).** *Consider $n \geq 2$ and $\delta \geq \frac{1}{2}$, then:*

1. $\operatorname{err}_{poly}(n, \delta) = \delta - \delta / (\delta(2-n) + n - 1)$ *and* $\operatorname{err}_{poly}(2, \delta) = 0$
2. *For all $\upsilon > 0$ we get* $\operatorname{err}_{\sigma}(n, \upsilon) > \operatorname{err}_{\sigma}(n, 0) = \max_{\delta \in [\frac{1}{2}, 1]} \operatorname{err}_{poly}(n, \delta)$
3. $\lim_{\delta \to 1} \operatorname{err}_{poly}(n, \delta) = 0$

Note, that while $\operatorname{err}_{\text{poly}}(n, 1)$ is well defined for δ the necessary bound $\ln\left(\frac{\delta}{1-\delta}\right)$ is not, i.e. we can only check for $\delta < 1$. The observations described in Lemma 5 are also observable in Figure 4: By parameterizing the approximation in the confidence threshold δ we achieve significant precision gains over prior work – independent of output dimensionality (n) and in particular as we approach $\delta = 1$.

7 Evaluation

We implemented Differential Zonotope verification in a new tool[1] called *VeryDiff* in Julia [7]. First, we analyze the efficiency of Differential Zonotopes compared to our naive approach for verifying ε or (δ-)Top-1 equivalence. We also compare the performance of VeryDiff to the previous State-of-the-Art and demonstrate significant performance improvements for ε and δ-Top-1 equivalence across all benchmark families. Our Appendix [29, Sec. C] contains an extended evaluation.

Experimental Setup. We compare six different tools or configurations on old and new benchmark families. A detailed summary of baselines, benchmark families and NN architectures can be found in the Appendix [29, Sec. C.1]. For ε and Top-1 equivalence, we evaluate on preexisting and new ACAS and MNIST NNs (airborne collision avoidance and handwritten digit recognition) where the second NN is generated via pruning (and possibly further training). For MNIST, we evaluate w.r.t. input regions generated by Paulsen *et al.* [23] which prove equivalence on L_∞ bounded perturbations of images (L_∞ Properties) or targeted pixel perturbations (Pixel Attacks) For δ-Top-1 equivalence, we introduce a new NN verification benchmark for particle jet classification at CERN's Large Hadron Collider (LHC) [8]. We analyze equivalence w.r.t. pruned and further trained NN. NNs in this context come with strict real-time requirements making pruned NNs highly desirable [8]. We verify equivalence for boxes defined via standard deviations over the normalized input space.

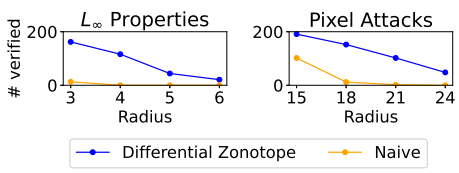


Fig. 5: MNIST benchmark queries for which VeryDiff with(out) Differential Zonotopes proves equivalence.

Where Differential Verification helps. We compared VeryDiff with Differential Zonotopes activated to the naive computation without Differential Zonotopes. A summary of all results can be found in Table 6 [29, Sec. C.2]. We see significant improvements for ε equivalence (1430% more instances certified for the MNIST (VeriPrune) benchmark family). Figure 5 breaks down the verified ε equivalence queries with/without Differential Zonotopes by input radius across all MNIST benchmark queries and shows that Differential Verification helps to verify larger input regions. On the other hand, differential analysis slows down the verification of Top-1 equivalence verification. We provide a complementary theoretical analysis to this observation in Section 6 and posit this is a fundamental limitation of Differential Verification and not specific to our implementation. In contrast, certification of confidence-based equivalence can profit from Differential Verification across two benchmark families. While we see diminishing speedups as we push the confidence threshold δ closer to 1 (implicitly reducing the input space with guarantees), realistic thresholds $\delta > 0.5$ (e.g. 0.9 instead

[1] https://github.com/samysweb/VeryDiff-Artifact/

Table 2: Verification results for ε and δ-Top-1 equivalence: Speedups for commonly solved instances; improvements reported w.r.t. the best other tool. (*α, β-CROWN for MNIST spends 80% of its time on neuron-bound refinement. Speedups may be exaggerated while increases in solved instances are accurate).

Benchmark		Variant	Equiv.		Counterex.		Speedup	
							Median	Max
Standard ε eq.	ACAS	VeryDiff (ours)	**150**	(+24.0%)	**153**	(+2.0%)	—	—
		NNEquiv	37		142		37.3	8091.2
		MILPEquiv	16		3		7224.8	36297.0
		Marabou	110		109		141.3	10070.5
		α, β-CROWN	121		150		15.4	1954.1
	MNIST (VeriPrune)	VeryDiff (ours)	**352**	(+101.1%)	62	(-43.6%)	—	—
		NNEquiv	0		103		12.6	166.2
		Marabou	10		24		183.9	1390.8
		α, β-CROWN*	175		**110**		(516.9)	(4220.8)
NeuroDiff ε eq.	ACAS	VeryDiff (ours)	**169**	(+39.7%)	**161**	(+103.8%)	—	—
		NeuroDiff	121		79		43.1	16134.7
	MNIST (VeriPrune)	VeryDiff (ours)	**457**	(+11.5%)	**242**	(+404.1%)	—	—
		NeuroDiff	410		48		4.5	1086.8
δ-Top-1	LHC	VeryDiff (ours)	**77**	(327.8%)	—		—	—
		α, β-CROWN	18		—		324.5	11274.3

of $1 - 10^{-7}$) profit *most* from differential verification (up to speedups of 677 over the naive technique on commonly solved queries for LHC w.r.t. $\delta = 0.99$). We found queries where ε and Top-1 equivalence results differ – underlining the importance of choosing equivalence properties w.r.t. the task at hand.

***ε* Equivalence.** We compare our tool with other ε equivalence verification techniques from the literature and summarize the results in Table 2. Across both benchmarks from prior literature (ACAS and MNIST), we significantly outperform equivalence-specific verifiers (NNEquiv [28], MILPEquiv [17], NeuroDiff [23]) as well as general NN verification techniques (α, β-CROWN [18,25, 32,35–39], Marabou [16,34]) for certification of equivalence. α, β-CROWN outperforms VeryDiff in the search for counterexamples. We suspect this is due to its adversarial attack techniques [38]. Due to incompatible differences in the property checked by NeuroDiff's implementation [29, Sec. C.1], we performed a separate comparison where VeryDiff outperforms NeuroDiff on the same property.

δ-Top-1 Equivalence. Differential Verification significantly improves upon the generic State-of-the-Art NN verifier α, β-CROWN (see Table 2). We concede that α, β-CROWN's attack techniques outperform VeryDiff's counterexample generation (see ε equivalence). Hence, our evaluation focuses on equivalence certification. The objective for δ-Top-1 equivalence is to provide guarantees for low values δ. α, β-CROWN was only able to provide guarantees for 10 of the NNs in the benchmark set. In each case, VeryDiff was able to prove equivalence for lower (i.e. better) or equal δ values. For the 3 NNs where the provided guarantees of α, β-CROWN and VeryDiff matched, both tools only verified equivalence for $\delta =$

$1 - 10^{-7}$, i.e. they provided an extremely limited guarantee. This underlines that VeryDiff is a significant step forward in the verification of δ-Top-1 equivalence.

Limitations Larger weight differences between NNs and accumulating ReLU approximations may decrease speedups achievable via Differential Verification (see discussion in Appendix [29, Sec. C.4]). Nonetheless, VeryDiff outperforms alternative verifiers – often by orders of magnitude. Another limitation for confidence-based NN verification is the possibility of satisfied for high confidence thresholds (for mitigation via calibration [1, 13] see Athavale *et al.* [2]). However, for equivalence verification, we consider the reference NN f_1 (incl. its confidence) trustworthy.

8 Conclusion

We introduced Differential Zonotopes as an abstract domain for NN equivalence verification. Our extensive evaluation shows that we outperform the Differential Verification tool NeuroDiff [22, 23]) as well as State-of-the-Art NN verifiers. Moreover, our paper provides insights into the circumstances where differential reasoning does (not) aid verification. As discussed in Sections 6 and 7, whether Differential Verification helps is not always straightforward for specifications involving classification. We believe that confidence-based equivalence is the way forward to scale equivalence verification beyond tiny input regions such as ϵ-balls criticized in the literature [11]. Finally, we introduced a simpler approximation for softmax that is provably tighter than prior work [2]. This approximation is also of independent interest as it allows more precise verification of softmax NNs using simple linear constraints.

Future Work. We see potential in extending generic NN verifiers (e.g. Marabou's network level reasoner [34]) with differential abstract domains to improve their reasoning capabilities for relational properties. Beyond equivalence, we hope to explore the applicability of Differential Verification techniques to other properties such as robustness or fairness.

Acknowledgements. This work was supported by funding from the pilot program Core-Informatics of the Helmholtz Association (HGF). Additional financial support was provided by the program "Invest BW" from the Baden-Württemberg Ministry of Economic Affairs, Labour and Tourism.

References

1. Ao, S., Rueger, S., Siddharthan, A.: Two sides of miscalibration: Identifying over and under-confidence prediction for network calibration. In: Evans, R.J., Shpitser, I. (eds.) Uncertainty in Artificial Intelligence, UAI 2023, July 31 - 4 August 2023, Pittsburgh, PA, USA. Proceedings of Machine Learning Research, vol. 216, pp. 77–87. PMLR (2023), https://proceedings.mlr.press/v216/ao23a.html
2. Athavale, A., Bartocci, E., Christakis, M., Maffei, M., Nickovic, D., Weissenbacher, G.: Verifying global two-safety properties in neural networks with confidence. In: Gurfinkel, A., Ganesh, V. (eds.) Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part II. LNCS, vol. 14682, pp. 329–351. Springer (2024). https://doi.org/10.1007/978-3-031-65630-9_17
3. Bak, S., Tran, H., Hobbs, K., Johnson, T.T.: Improved geometric path enumeration for verifying ReLU neural networks. In: Lahiri, S.K., Wang, C. (eds.) Computer Aided Verification - 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part I. LNCS, vol. 12224, pp. 66–96. Springer (2020). https://doi.org/10.1007/978-3-030-53288-8_4
4. Banerjee, D., Singh, G.: Relational DNN verification with cross executional bound refinement. In: Forty-first International Conference on Machine Learning (2024), https://openreview.net/forum?id=HOG80Yk4Gw
5. Banerjee, D., Xu, C., Singh, G.: Input-relational verification of deep neural networks. Proc. ACM Program. Lang. **8**(PLDI), 1–27 (2024). https://doi.org/10.1145/3656377
6. Barthe, G., Crespo, J.M., Kunz, C.: Relational verification using product programs. In: Butler, M.J., Schulte, W. (eds.) FM 2011: Formal Methods - 17th International Symposium on Formal Methods, Limerick, Ireland, June 20-24, 2011. Proceedings. LNCS, vol. 6664, pp. 200–214. Springer (2011). https://doi.org/10.1007/978-3-642-21437-0_17
7. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A fresh approach to numerical computing. SIAM Rev. **59**(1), 65–98 (2017). https://doi.org/10.1137/141000671, https://doi.org/10.1137/141000671
8. Duarte, J., Han, S., Harris, P., Jindariani, S., Kreinar, E., Kreis, B., Ngadiuba, J., Pierini, M., Rivera, R., Tran, N., Wu, Z.: Fast inference of deep neural networks in fpgas for particle physics. Journal of Instrumentation **13**(07), P07027 (jul 2018). https://doi.org/10.1088/1748-0221/13/07/P07027
9. Eleftheriadis, C., Kekatos, N., Katsaros, P., Tripakis, S.: On neural network equivalence checking using SMT solvers. In: Bogomolov, S., Parker, D. (eds.) Formal Modeling and Analysis of Timed Systems - 20th International Conference, FORMATS 2022, Warsaw, Poland, September 13-15, 2022, Proceedings. LNCS, vol. 13465, pp. 237–257. Springer (2022). https://doi.org/10.1007/978-3-031-15839-1_14
10. Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., Vechev, M.T.: AI2: safety and robustness certification of neural networks with abstract interpretation. In: 2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA. pp. 3–18. IEEE Computer Society (2018). https://doi.org/10.1109/SP.2018.00058
11. Geng, C., Le, N., Xu, X., Wang, Z., Gurfinkel, A., Si, X.: Towards reliable neural specifications. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) International Conference on Machine Learning, ICML 2023,

23-29 July 2023, Honolulu, Hawaii, USA. Proceedings of Machine Learning Research, vol. 202, pp. 11196–11212. PMLR (2023), https://proceedings.mlr.press/v202/geng23a.html
12. Ghorbal, K., Goubault, E., Putot, S.: The zonotope abstract domain taylor1+. In: Bouajjani, A., Maler, O. (eds.) Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5643, pp. 627–633. Springer (2009). https://doi.org/10.1007/978-3-642-02658-4_47, https://doi.org/10.1007/978-3-642-02658-4_47
13. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 1321–1330. PMLR (2017), http://proceedings.mlr.press/v70/guo17a.html
14. Habeeb, P., Prabhakar, P.: Approximate conformance verification of deep neural networks. In: Benz, N., Gopinath, D., Shi, N. (eds.) NASA Formal Methods - 16th International Symposium, NFM 2024, Moffett Field, CA, USA, June 4-6, 2024, Proceedings. LNCS, vol. 14627, pp. 223–238. Springer (2024). https://doi.org/10.1007/978-3-031-60698-4_13
15. Katz, G., Barrett, C.W., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: An efficient SMT solver for verifying deep neural networks. In: Majumdar, R., Kuncak, V. (eds.) Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I. LNCS, vol. 10426, pp. 97–117. Springer (2017). https://doi.org/10.1007/978-3-319-63387-9_5
16. Katz, G., Huang, D.A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljic, A., Dill, D.L., Kochenderfer, M.J., Barrett, C.W.: The Marabou framework for verification and analysis of deep neural networks. In: Dillig, I., Tasiran, S. (eds.) Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I. LNCS, vol. 11561, pp. 443–452. Springer (2019). https://doi.org/10.1007/978-3-030-25540-4_26
17. Kleine Büning, M., Kern, P., Sinz, C.: Verifying equivalence properties of neural networks with ReLU activation functions. In: Simonis, H. (ed.) Principles and Practice of Constraint Programming - 26th International Conference, CP 2020, Louvain-la-Neuve, Belgium, September 7-11, 2020, Proceedings. LNCS, vol. 12333, pp. 868–884. Springer (2020). https://doi.org/10.1007/978-3-030-58475-7_50
18. Kotha, S., Brix, C., Kolter, J.Z., Dvijotham, K., Zhang, H.: Provably bounding neural network preimages. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023 (2023), http://papers.nips.cc/paper_files/paper/2023/hash/fe061ec0ae03c5cf5b5323a2b9121bfd-Abstract-Conference.html
19. Matos, J.B.P., Filho, E.B.d.L., Bessa, I., Manino, E., Song, X., Cordeiro, L.C.: Counterexample guided neural network quantization refinement. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems pp. 1–1 (2023). https://doi.org/10.1109/TCAD.2023.3335313
20. Munakata, S., Tokumoto, S., Yamamoto, K., Munakata, K.: Towards formal repair and verification of industry-scale deep neural networks. In: 45th IEEE/ACM International Conference on Software Engineering: ICSE 2023 Com-

panion Proceedings, Melbourne, Australia, May 14-20, 2023. pp. 360–364. IEEE (2023). https://doi.org/10.1109/ICSE-COMPANION58688.2023.00103, https://doi.org/10.1109/ICSE-Companion58688.2023.00103
21. Narodytska, N., Kasiviswanathan, S.P., Ryzhyk, L., Sagiv, M., Walsh, T.: Verifying properties of binarized deep neural networks. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18, New Orleans, Louisiana, USA, February 2-7, 2018. pp. 6615–6624. AAAI Press (2018). https://doi.org/10.1609/AAAI.V32I1.12206
22. Paulsen, B., Wang, J., Wang, C.: Reludiff: differential verification of deep neural networks. In: Rothermel, G., Bae, D. (eds.) ICSE '20: 42nd International Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020. pp. 714–726. ACM (2020). https://doi.org/10.1145/3377811.3380337
23. Paulsen, B., Wang, J., Wang, J., Wang, C.: NeuroDiff: scalable differential verification of neural networks using fine-grained approximation. In: 35th IEEE/ACM International Conference on Automated Software Engineering, ASE 2020, Melbourne, Australia, September 21-25, 2020. pp. 784–796. IEEE (2020). https://doi.org/10.1145/3324884.3416560
24. Sälzer, M., Lange, M.: Reachability is np-complete even for the simplest neural networks. In: Bell, P.C., Totzke, P., Potapov, I. (eds.) Reachability Problems - 15th International Conference, RP 2021, Liverpool, UK, October 25-27, 2021, Proceedings. LNCS, vol. 13035, pp. 149–164. Springer (2021). https://doi.org/10.1007/978-3-030-89716-1_10
25. Shi, Z., Jin, Q., Kolter, Z., Jana, S., Hsieh, C., Zhang, H.: Neural network verification with branch-and-bound for general nonlinearities. CoRR **abs/2405.21063** (2024). https://doi.org/10.48550/ARXIV.2405.21063, https://doi.org/10.48550/arXiv.2405.21063
26. Shriver, D., Elbaum, S.G., Dwyer, M.B.: DNNV: A framework for deep neural network verification. In: Silva, A., Leino, K.R.M. (eds.) Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part I. LNCS, vol. 12759, pp. 137–150. Springer (2021). https://doi.org/10.1007/978-3-030-81685-8_6
27. Singh, G., Gehr, T., Mirman, M., Püschel, M., Vechev, M.T.: Fast and effective robustness certification. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 10825–10836 (2018), https://proceedings.neurips.cc/paper/2018/hash/f2f446980d8e971ef3da97af089481c3-Abstract.html
28. Teuber, S., Büning, M.K., Kern, P., Sinz, C.: Geometric path enumeration for equivalence verification of neural networks. In: 33rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2021, Washington, DC, USA, November 1-3, 2021. pp. 200–208. IEEE (2021). https://doi.org/10.1109/ICTAI52525.2021.00035
29. Teuber, S., Kern, P., Janzen, M., Beckert, B.: Revisiting differential verification: Equivalence verification with confidence. CoRR **abs/2410.20207** (2024). https://doi.org/10.48550/ARXIV.2410.20207
30. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Efficient formal safety analysis of neural networks. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31: Annual Conference on Neural Informa-

tion Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 6369–6379 (2018), https://proceedings.neurips.cc/paper/2018/hash/2ecd2bd94734e5dd392d8678bc64cdab-Abstract.html
31. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Formal security analysis of neural networks using symbolic intervals. In: Enck, W., Felt, A.P. (eds.) 27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018. pp. 1599–1614. USENIX Association (2018), https://www.usenix.org/conference/usenixsecurity18/presentation/wang-shiqi
32. Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C., Kolter, J.Z.: Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. CoRR **abs/2103.06624** (2021), https://arxiv.org/abs/2103.06624
33. Wang, W., Wang, K., Cheng, Z., Yang, Y.: Veriprune: Equivalence verification of node pruned neural network. Neurocomputing **577**, 127347 (2024). https://doi.org/https://doi.org/10.1016/j.neucom.2024.127347, https://www.sciencedirect.com/science/article/pii/S0925231224001188
34. Wu, H., Isac, O., Zeljic, A., Tagomori, T., Daggitt, M.L., Kokke, W., Refaeli, I., Amir, G., Julian, K., Bassan, S., Huang, P., Lahav, O., Wu, M., Zhang, M., Komendantskaya, E., Katz, G., Barrett, C.W.: Marabou 2.0: A versatile formal analyzer of neural networks. In: Gurfinkel, A., Ganesh, V. (eds.) Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part II. LNCS, vol. 14682, pp. 249–264. Springer (2024). https://doi.org/10.1007/978-3-031-65630-9_13
35. Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K., Huang, M., Kailkhura, B., Lin, X., Hsieh, C.: Automatic perturbation analysis for scalable certified robustness and beyond. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020), https://proceedings.neurips.cc/paper/2020/hash/0cbc5671ae26f67871cb914d81ef8fc1-Abstract.html
36. Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., Hsieh, C.: Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021 (2021)
37. Zhang, H., Wang, S., Xu, K., Li, L., Li, B., Jana, S., Hsieh, C., Kolter, J.Z.: General cutting planes for bound-propagation-based neural network verification. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022 (2022), http://papers.nips.cc/paper_files/paper/2022/hash/0b06c8673ebb453e5e468f7743d8f54e-Abstract-Conference.html
38. Zhang, H., Wang, S., Xu, K., Wang, Y., Jana, S., Hsieh, C., Kolter, J.Z.: A branch and bound framework for stronger adversarial attacks of ReLU networks. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., Sabato, S. (eds.) International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA. Proceedings of Machine Learning Research, vol. 162, pp. 26591–26604. PMLR (2022), https://proceedings.mlr.press/v162/zhang22ae.html
39. Zhang, H., Weng, T., Chen, P., Hsieh, C., Daniel, L.: Efficient neural network robustness certification with general activation functions. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances

in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 4944–4953 (2018), https://proceedings.neurips.cc/paper/2018/hash/d04863f100d59b3eb688a11f95b0ae60-Abstract.html

40. Zhang, Y., Song, F., Sun, J.: QEBVerif: Quantization error bound verification of neural networks. In: Enea, C., Lal, A. (eds.) Computer Aided Verification - 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part II. LNCS, vol. 13965, pp. 413–437. Springer (2023). https://doi.org/10.1007/978-3-031-37703-7_20