The Fundamental Limits of Interval Arithmetic for Neural Networks

MATTHEW MIRMAN, ETH Zurich, Switzerland
MAXIMILIAN BAADER, ETH Zurich, Switzerland
MARTIN VECHEV, ETH Zurich, Switzerland

Interval analysis (or interval bound propagation, IBP) is a popular technique for verifying and training provably robust deep neural networks, a fundamental challenge in the area of reliable machine learning. However, despite substantial efforts, progress on addressing this key challenge has stagnated, calling into question whether interval arithmetic is a viable path forward.

In this paper we present two fundamental results on the limitations of interval arithmetic for analyzing neural networks. Our main impossibility theorem states that for any neural network classifying just three points, there is a valid specification over these points that interval analysis can not prove. Further, in the restricted case of one-hidden-layer neural networks we show a stronger impossibility result: given any radius $\alpha < 1$, there is a set of $O(\alpha^{-1})$ points with robust radius α, separated by distance 2, that no one-hidden-layer network can be proven to classify robustly via interval analysis.

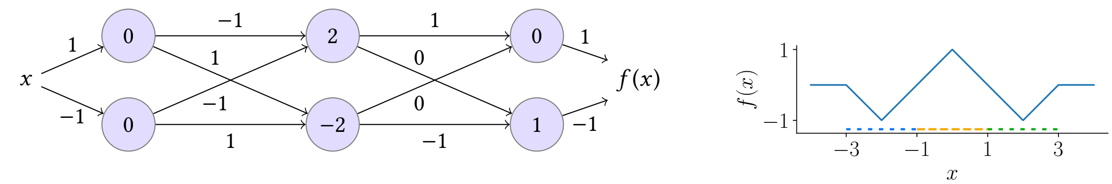
1 INTRODUCTION

As neural networks are increasingly used in safety critical environments, ensuring their behavior with *formal verification* has become a highly active research direction [LAL+19, HKR+20]. Because neural networks are often too large for complete verification methods, incomplete analysis techniques are frequently employed [GMT+18] – these can scale to larger models though may fail to prove a property that actually holds (as demonstrated in Fig. 1). Indeed, recent progress in constructing provable neural networks has been achieved thanks to leveraging incomplete methods, and particularly interval (box) bound propagation [MGV18]. However, while many improvements to *provable defenses* have been published [GDS+18, ZWC+18, ZCX+20, XTSM19, WSMK18, LSS21, BWL+21, SWZ+21, XZW+21] (most building on IBP), progress remains far from satisfactory: the state-of-the-art certified robust accuracy is roughly 60% on CIFAR10 [BV20], compared to state-of-the-art standard accuracies of above 95%. The stagnation of progress in constructing provably robust neural networks, and the importance of interval arithmetic to other areas of mathematics [Tuc02], has led to a fundamental question:

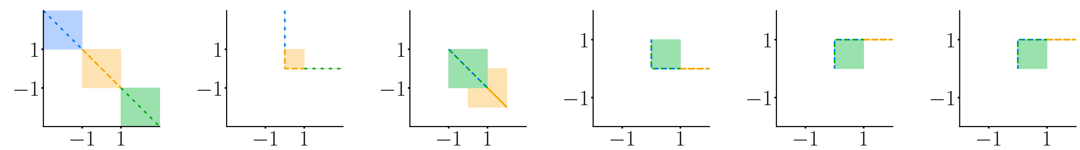
Do neural networks exist which can be efficiently (with interval analysis) proven correct?
(Fundamental Theoretical Question)

The first result addressing this question was investigated in [BMV20] which proved an analog to the universal approximation theorem [Cyb89, HSW89] for interval-analyzable networks. [WAPJ20] further showed that two hidden layer networks could also be interval-analyzable approximators. [Ano22] also demonstrated that training with interval propagation converges with high probability. While it is helpful to know that searching for networks which can be easily analyzed might not be futile, these results do not explain, and even contradict the provable training gap that is observed in practice. A preliminary negative result was shown in [WAPJ20]: verifying the robustness of arbitrary neural networks in general and thus translating arbitrary neural networks into interval-analyzable forms is NP-hard. In our work, we provide a strong negative answer, thus explaining the provable training gap: we demonstrate that non-trivial datasets can not be classified by interval-provable networks.

Authors' addresses: Matthew Mirman, matthew.mirman@inf.ethz.ch, ETH Zurich, Switzerland; Maximilian Baader, mbaader@inf.ethz.ch, ETH Zurich, Switzerland; Martin Vechev, martin.vechev@inf.ethz.ch, ETH Zurich, Ramistrasse 101, 8092, Zurich, Switzerland.



(a) Network. Encircled numbers are biases, and circles (■) are ReLU activations. (b) Network values $f(x)$ for $x \in [-4, 4]$.



(c) 1st affine layer (d) 1st ReLU layer (e) 2nd affine layer (f) 2nd ReLU layer (g) 3rd affine layer (h) 3rd ReLU layer

Fig. 1. An example of a neural network which is in fact robust, yet which interval arithmetic fails to *prove* is robust. The three intervals $L = [-3, -1]$ (■), $M = [-1, 1]$ (■) and $R = [1, 3]$ (■) are depicted using dashed lines in Fig. 1b. The interval propagation through the network is shown using the rectangles, the concrete values are shown by the dashed lines. Here, after the 2nd affine layer (Fig. 1e) the orange and green box overlap although the orange lines do not overlap, showing the loss of precision. The output Interval is $[-1, 1]$ for all three input intervals L, M and R.

Formally, given a neural network, or more generally any program, $f: \mathcal{X} \to \mathcal{Y}$, the goal of verification is to algorithmically prove that f maps an input specification, $S_I \subseteq \mathcal{X}$, to a subset of an output specification, $S_O \subseteq \mathcal{Y}$, where (S_I, S_O) is a member of a set $\mathcal{S} \subseteq \mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{Y})$, which we call the *specification task*. Interval analysis in particular replaces the basic operations of f with interval arithmetic [Moo66, HJVE01], producing a *sound* interval extension, $f^{\#}$: Intervals($\mathcal{X}$) $\to$ Intervals($\mathcal{Y}$), of f such that every element of S_I is mapped by f to an element of $f^{\#}(S_I)$. As representing and computing intervals is efficient, f is proven to meet the specification (S_I, S_O) by proxy of proving $f^{\#}(S_I) \subseteq S_O$. The specification tasks we consider are l_∞ *robust classifications*, meaning the input specifications are closed l_∞-balls (i.e., intervals), and output specifications are either $\mathbb{R}_{>0}$ or $\mathbb{R}_{<0}$. Formally, we say an l_∞-robustness classification $\mathcal{S}$ is *complete*, if for any specification $(S_I, S_O) \in \mathcal{S}$, there is an l_∞-ball, the *scheme*, $T_S \supseteq S_I$ such that for all l_∞-balls $B \subseteq T_S^\circ$, we have $(B, S_O) \in \mathcal{S}$.

Main contributions. In this paper, we present the first proofs capturing key limitations (incompleteness) of interval analysis for neural networks:

- **General interval impossibility (Corollary 5.12)**: It is impossible to construct a feed-forward ReLU-neural network of any shape (e.g., residual, convolutional, dense, fully-connected) that is *completely provably robust* (Definition 5.11) with interval analysis for a simple one-dimensional dataset with only three points.
- **One-layer strong interval impossibility (Theorem 4.6)**: Even when the requirement for complete provability is relaxed to regions that are distant from each other (α-interval provable with $\alpha < 1$ as in Definition 4.1), there are datasets with $O(\alpha^{-1})$ points that can not be provably robustly classified with one-hidden layer networks using interval analysis.
- **One-layer strong interval-agnostic possibility (Proposition 5.13)**: *completely-robust classifiers* can always be constructed with one-hidden layer networks, even if they are not necessarily provably robust using interval analysis. Together with Theorem 4.6 and Corollary 5.12 this implies that the restriction that a network be analyzable with interval-arithmetic is severely limiting.

2 PROBLEM MOTIVATION

Studying the robustness of artificial neural networks has become an important area of research, as neural networks are increasingly deployed in safety-critical applications such as self-driving cars [BTD+16]. [SZS+13] first demonstrated that neural networks classifying images can be fooled into misclassification by imperceptible pixel perturbations in an otherwise correctly classified image.

Many of these fooling techniques, known as adversarial attacks, have been developed [CW17, GSS15, KGB16, SYN15, CH19b, PMG16, WSK19]. To defend against these attack, methods hardening models have been proposed [PMW+16, TKP+17, WRK19, SHS20, BIL+16, CH20]. A particular line of research aims to provide *formal* guarantees (i.e., verify) that neural networks behave correctly [KBD+17, SGM+18, SGPV19, BWC+19, LTC19, WPW+18, BBS+19, ZAD21, LSR+21, CH19a, CAH18]. As complete verification of a neural network is NP-Hard [KBD+17], the majority of modern techniques is incomplete and are based on over-approximating the behavior of a network [GMT+18]. While incomplete methods can be highly efficient, it a correct classification of a network *might not be provably* correct, as is illustrated in Fig. 1. In fact, for naturally trained neural networks, only a small percentage of non-attackable input images are verifiable.

To improve verification rates, techniques to training networks that are amenable to verification [RSL18, MGV18, WK18, WSMK18] have been developed. While this has been a very active area of research, the state-of-the-art developed by [BV20], achieves a certified robust accuracy of 60.5% on CIFAR10 which is unsatisfactory compared to a state-of-the-art standard accuracy of above 95%.

The recent plateau of progress in closing this gap has raised concerns about whether there are theoretical limitations to neural network analysis [SYZ+19]. In this work, we provide fundamental limitations, which helps to explain the significant gab between certified robust accuracy, and standard accuracy. We focus on interval analysis, as some of the most successful and widely used methods have been based on it [MGV18, GDS+18].

3 BACKGROUND

In this section, we introduce the main concepts, and notation central to understanding our results.

3.1 General Notation

The main results in this work centers around the interval domain, which is technically the domain of axis-aligned bounding boxes, and thus we begin by describing notation related to such boxes.

Let $\mathcal{B}^d$, the set of closed, non-empty, axis-aligned boxes of dimension d (also known as l_∞ balls). We write the box with center $c \in \mathbb{R}^d$ and radius $r \in \mathbb{R}^d_{\geq 0}$ as $\mathbb{B}_r(c) := \{x \colon \forall i \in [d]. \exists \xi \in [-1, 1]. x_i = c_i + \xi r_i\}$. For a given box $B \in \mathcal{B}^d$ let $C(B)$ denote its center and $\mathcal{R}(B)$ denote its radius such that $B = \mathbb{B}_{\mathcal{R}(B)}(C(B))$. We also write $\mathbb{B}^\circ_\epsilon(x) := \{y \in \mathbb{R}^d \colon ||x - y||_\infty < \epsilon\}$ for $x \in \mathbb{R}^d$ and $\epsilon \in \mathbb{R}_{>0}$.

For a set Y, we write $Y|_i$ for the restriction of Y to the dimension i, or more formally, $Y|_i := \{y_i \colon y \in Y\}$. For any bounded and non-empty set C, the l_∞*-hull*, written $\mathcal{H}_\infty(C)$, is the smallest axis aligned box containing C. Formally, $\mathcal{H}_\infty(C) := \{x \colon \forall i \in [d]. \sup(C|_i) \leq x_i \leq \inf(C|_i)\}$.

If $f : A \to B$ and $S \subseteq A$ we write $f[S] := \{f(s) \colon s \in S\}$, but sometimes abuse notation and write $f(S) := f[S]$ to avoid clutter. Similarly, we also occasionally write $f^{-1} \circ g^{-1}$ even when f and g are non invertible to mean $(g \circ f)^{-1}$. For any set S we write $\mathcal{P}(S)$ to mean the powerset of S. For some positive natural $k \in \mathbb{N}$ we write $[k] := \{1, \ldots, k\}$.

3.2 Robustness and Interval Certification (IBP)

Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is some function (i.e., neural network). We say that this network assigns a label $l \in \{-1, 1\}$ to a point $x \in \mathbb{R}^d$ if $\operatorname{sign} f(x) = l$. In our case, we discuss l_∞-adversarial region specifications. In this case, we say that f is *ϵ-robust around x* with *label l* if $\forall x' \in \mathbb{B}_\epsilon(x).\ f(x') = l$.

The goal of robustness certification is to provide a guarantee that a neural network is robust at some point. However, robustness certification does not need to inform when a neural network is *not*-robust at a point. This leads to efficient methods in terms of *over-approximation*, originally described as *abstract-interpretation* [CC77] and applied to neural networks by [GMT+18].

Definition 3.1. Given the *concrete-domains* $\mathcal{D}$ and $\mathcal{G}$ we say the *concrete function* $f : \mathcal{D} \to \mathcal{G}$ *over-approximated* by the *transformed function (abstract transformer)* $f^\# : \tilde{\mathcal{D}} \to \tilde{\mathcal{G}}$ with *abstract-domains* $\tilde{\mathcal{D}} \subseteq \mathcal{P}(\mathcal{D})$ and $\tilde{\mathcal{G}} \subseteq \mathcal{P}(\mathcal{G})$ if it is *sound* if for any abstract set $\mathcal{S} \in \mathcal{D}$ we have $f(\mathcal{S}) \subseteq f^\#(\mathcal{S})$

The goal of such an over-approximation that it is possible to computationally represent and modify elements of *abstract-domains*, whereas it is not in general possible to do this for any subset of $\mathcal{D}$ (as it might very well be $\mathbb{R}$). We note our definitions are a slight departure from the traditional abstract interpretation literature. While typically, the abstract domain refers to the set of *representations* of subsets of the concrete domain, and the abstract transformer acts on these representations, we refer to the sets they represent themselves. As here we are concerned only with the question of the mathematical limitations of what is represented, and not the question of how to efficiently represent or perform computations (this is trivial for IBP), we can simplify our presentation dramatically by discussing only the represented sets and not the representations themselves.

To certify a function $f = g_n \circ \cdots \circ g_1$, where $g_i : \mathbb{R}^{d_i} \to \mathbb{R}^{d_{i+1}}$, is ϵ-robust at x using abstract interpretation, one may pick abstract-domains $\mathcal{D}_i \subseteq \mathcal{P}(\mathbb{R}^{d_i})$ and compute sound transformed functions, $g_1^\#, \ldots, g_n^\#$: If $g_i^\#$ and $g_{i+1}^\#$ over-approximates g_i and g_{i+1} then $g_i^\# \circ g_{i+1}^\#$ over-approximates $g_{i+1} \circ g_i$. Then, given one can show for $f^\# = g_n^\# \circ \cdots \circ g_1^\#$, that for some $\tilde{A} \in \mathcal{D}_1$ such that $\mathbb{B}_\epsilon(x) \subseteq \tilde{A}$, it is true that $\forall \tilde{y} \in f^\#(\tilde{A}).\ \operatorname{sign} \tilde{y} = f(x)$, then one will have also shown that $\forall x' \in \mathbb{B}_\epsilon(x).\ \operatorname{sign} f(x') = f(x)$.

For any function $f : A \to B$, we say that the *perfect transformation* of f is $f^{\mathcal{P}} : \mathcal{P}(A) \to \mathcal{P}(B)$ where for any set $S \subseteq A$ we have $f^{\mathcal{P}}(S) := f[S]$.

While the perfect transformation of f is just that, always perfect, it is important to note that over-approximation is typically *not precise*, meaning $f(S) \subsetneq f^\#(S)$. In this case, it is in fact possible to not prove the guarantee, such as robustness, even if that guarantee holds for f.

Interval Analysis. In this paper, we focus on the *Interval (or Box)-domain*, and in particular, Interval Bound Propagation (IBP) [GDS+18], which is also known as interval-analysis. In this case, $\mathcal{B}^d$ is used as the abstract domain, $\tilde{\mathcal{D}}$, when the concrete domain is $\mathbb{R}^d$. An interval, $B \in \mathcal{B}^d$, can either be represented as a center $c \in \mathbb{R}^d$ and radius $r \in \mathbb{R}^d_{\geq 0}$ as before, or as a lower-bound and upper bound, $\iota_l, \iota_u \in \mathbb{R}^d$ respectively such that for each dimension $j \in [d]$, we have $\iota_{l,j} \leq \iota_{u,j}$. The two representations are related as follows: $\iota_l = c - r$ and $\iota_u = c + r$, or $c = \frac{1}{2}(\iota_l + \iota_u)$ and $\iota_u = \frac{1}{2}(\iota_u - \iota_l)$.

Analyzing Neural Networks. The application of interval analysis to neural networks with ReLU-activations is straightforward. In this paper, we consider (feed forward) neural networks defined inductively as follows:

Definition 3.2. A σ-(neural) network with σ-activations, f, is any of the following forms:

- *Sequential Computation:* $f(x) = g_1(g_2(x))$ where g_1 and g_2 are also both σ-networks.
- *Relational Duplication:* $f(x) = (x, x)$.
- *Non-Relational Parallel Computation:* $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$ where g_1 and g_2 are also both σ-networks.
- *Constant:* $f(x) = \kappa$ for some constant $\kappa \in \mathbb{R}$.
- *Multiplication by a Constant:* $f(x) = \kappa \cdot x$ for some constant $\kappa \in \mathbb{R}_{\neq 0}$.
- *Activation:* $f(x) = \sigma(x)$.
- *Relational Addition:* $f(x_1, x_2) = x_1 + x_2$.

For the purposes of exploring its limits, we view IBP as method that implicitly constructs a transformed function which acts on intervals. We describe this transformed function inductively as well:

***Definition 3.3** (Interval Analysis).* The *interval transformation*, $f^\#$, of a ReLU-network f is as follows:

- *Sequential Abstraction:* If $f(x) = g_1(g_2(x))$ then $f^\#(B) \coloneqq g_1^\#(g_2^\#(B))$.
- *Relational Duplication:* If $f(x) = (x, x)$ then $f^\#(B) \coloneqq B \times B$.
- *Non-Relational Parallel Abstraction:* If $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$ then $f^\#(B) \coloneqq g_1^\#(B|_1) \times g_2^\#(B|_2)$.
- *Constant:* If $f(x) = \kappa$ for $\kappa \in \mathbb{R}$ where $m = n = 1$, then $f^\#(B) \coloneqq \{\kappa\}$.
- *Multiplication by a Constant:* If $f(x) = \kappa \cdot x$ for some constant $\kappa \in \mathbb{R}_{\neq 0}$ where $m = n = 1$, then $f^\#(B) \coloneqq \{\kappa \cdot x\}$.
- *Activation:* If $f(x) = \mathrm{ReLU}(x)$ (where $m = n = 1$), then $f^\#(B) \coloneqq \{\mathrm{ReLU}(x) : x \in B\}$.
- *Relational Addition:* If $f(x_1, x_2) = x_1 + x_2$ where $m = 2$ and $n = 1$, then $f^\#(B) \coloneqq \{a + b : a \in B|_1 \wedge b \in B|_2\}$.

Proposition 3.4. If f is a ReLU-network then the interval transformer, $f^\#$, over-approximates f.

4 LIMITS FOR SINGLE HIDDEN LAYER NETWORKS

In this section we present an upper-bound on the number of points that can be proven to be robustly classified with interval for a single-layer network. We do this by constructing a paradoxical dataset, which we call *flips*. We begin by formalizing this dataset, and the notion of robust and provably robust on this dataset.

Definition 4.1. We say:

- A *flip* is a point $\hat{\mathtt{x}}_i \coloneqq 2i$ with *label* $\hat{\mathtt{l}}_i \coloneqq (-1)^i$.
- $f : \mathbb{R} \to \mathbb{R}$ is a *classifier for k flips* if $\forall i \in [k].\ f(\hat{\mathtt{x}}_i) = \hat{\mathtt{l}}_i$.
- $F : \mathcal{B} \to \mathcal{P}(\mathbb{R})$ is an *α-classifier for k flips* if $\forall i \in [k].\ \forall y \in F([\hat{\mathtt{x}}_i - \alpha, \hat{\mathtt{x}}_i + \alpha]).\ \operatorname{sign} y = \hat{\mathtt{l}}_i \wedge F(\{\hat{\mathtt{x}}_i\}) = \{\hat{\mathtt{l}}_i\}$.
- If $f^{\mathcal{P}}$ (the perfect transformation) is an α-classifier for k flips, we say f is an *α-robust classifier* for *k flips.*
- If $f^\#$ (the interval transformation from Definition 3.3) is an α-classifier for k flips, then we say that f is a *provably α-robust classifier* for *k flips.*

We now specify the notion of single-layer network for which we demonstrate bounds:

Definition 4.2. A *single-layer σ-network*, $f : \mathbb{R} \to \mathbb{R}$, with *n-neurons* and *$\sigma$-activations* is a function with *pre-activation weights*, $N \in \mathbb{R}^n$, *pre-activation bias*, $b \in \mathbb{R}^n$, *post-activation weights*, $M \in \mathbb{R}^n$, and *post-activation bias*, $d \in \mathbb{R}$ (the weights and biases are known as *parameters*), such that $f(x) = M \cdot \sigma(Nx + b) + d$.

(a) ReLU($-x$) (b) $-$ReLU($-x$) (c) ReLU(x) (d) $-$ReLU(x)

Fig. 2. The orientations of neurons captured by $L+$, $L-$, $R+$, and $R-$ for qualifying the scope of imprecision-contributions.

We note that while the Definition 3.3 defines an ordering of addition, this definition does not. While concrete-addition is associative, abstract addition is not always. However, thankfully, for the interval transformation, it is, and the bounds we demonstrate apply to any ordering of the operations.

***Definition 4.3**.* Given a single-layer ReLU-network, f with n neurons and pre/post-activation weights N, M, the *imprecision-contributions* of f at x are:

$$\mathcal{A}_{D,S}(x) \coloneqq \sum_{N_i x + b_i \geq 0 \wedge M_i \in S \wedge N_i \in D} |M_i N_i|,$$
$$\text{and} \quad \mathcal{I}_{D,S}(x) \coloneqq \sum_{N_i x + b_i \geq 0 \wedge M_i \in S \wedge N_i \in D} M_i N_i$$

where D can be the set $L \coloneqq \mathbb{R}_{\leq 0}$, the set $R \coloneqq \mathbb{R}_{\geq 0}$ or $\mathbb{R}$ and S can be the set $+ \coloneqq \mathbb{R}_{\geq 0}$, the set $- \coloneqq \mathbb{R}_{\leq 0}$ or $\mathbb{R}$.

Intuitively, $L+, L-, R+, R-$ correspond to the orientations that a neuron can take, as visualized in Fig. 2. L (resp. R) results in contributions from neurons that activate as the argument x of the imprecision-contribution function decreases (resp. increases). Note that $f'(x) = \mathcal{I}_{\mathbb{R},\mathbb{R}}(x)$ if the derivative of f is defined at x.

***Lemma 4.4** (End-Neuron Imprecision-Bound).* For all $\kappa \geq 0$ and single-layer ReLU-networks, f, that classify k-flips for $k = \lceil \kappa \rceil + 5$, we have $\kappa < \max\{\mathcal{A}_{L,\mathbb{R}}(\hat{x}_1), \mathcal{A}_{R,\mathbb{R}}(\hat{x}_k)\}$.

Proof Overview. We prove this by induction on $c \coloneqq \lfloor \kappa \rfloor$, using two simultaneous inductive invariants:

$$c \leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_k) - \mathcal{A}_{L,+}(\hat{x}_k),$$
$$\text{and} \quad c \leq \mathcal{A}_{L,-}(\hat{x}_1) - \mathcal{A}_{R,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_{k-1}) - \mathcal{A}_{L,-}(\hat{x}_{k-1}).$$

This proof involves two key observations: (i) once imprecision-contribution in a direction has accumulated, it will only be larger for points further in that direction, (ii) one must measure not just the accumulated growth of the imprecision-contribution at the ends of the approximated data ($\hat{x}_1$ and $\hat{x}_k$) in the out-wards directed neurons, but the growth of the *relative* imprecision-contribution excluding contribution from in-wards directed neurons. We make these observations more precise in the full proof in the appendix.

(Full proof in Appendix A)

Before demonstrating our main result, we require one further lemma, used to find a specific data-point with enough accumulated imprecision contribution to cause a violation:

***Lemma 4.5** (Lower-Bound on Imprecision-Contribution).* For any $a \leq 1$ and $k \geq \lceil \frac{2}{a} \rceil + 5$ and single-layer ReLU-network, f, that classifies k flips, there is some point $j \in [k]$ such that

$$\hat{1}_j a^{-1}(f(\hat{x}_j + a) + f(\hat{x}_j - a)) < \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{x}_j) + \mathcal{A}_{R,\mathbb{R}}(\hat{x}_j - a) + \mathcal{A}_{L,\mathbb{R}}(\hat{x}_j + a).$$

Proof Overview. By using $c \coloneqq \frac{2}{a}$ we can apply Lemma 4.4 (with $\kappa = c$), to show bounds for either the left or right-most $\hat{x}$ (i.e., $j = 1$ or $j = k$). For this point, we use the knowledge that the function is continuous, piecewise differentiable, to find points $l \in [\hat{x}_j - a, \hat{x}_j]$ and $u \in [\hat{x}_j, \hat{x}_j + a]$ such that $\frac{f(\hat{x}_j) - f(\hat{x}_j - a)}{a} < f'(l)$ and

$f'(u) < \frac{f(\hat{x}_j+a)-f(\hat{x}_j)}{a}$ so we can use that

$$f'(l) - f'(u) = \mathcal{I}_{\mathbb{R},\mathbb{R}}(l) - \mathcal{I}_{\mathbb{R},\mathbb{R}}(u) \leq \mathcal{A}_{L,\mathbb{R}}(\hat{x}_j - a) - \mathcal{A}_{R,\mathbb{R}}(\hat{x}_j + a),$$

that $-c, c < \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{x}_j)$, and that

$$\hat{1}_j a^{-1}(f(\hat{x}_j + a) + f(\hat{x}_j - a)) = \frac{f(\hat{x}_j) - f(\hat{x}_j - a)}{a} - \frac{f(\hat{x}_j + a) - f(\hat{x}_j)}{a} - c$$

to produce the final upper bound.

(Full proof in Appendix A)

We are now ready to show our main theorem, an upper bound on the number of flips that can be provably robustly classified with a single layer network.

Theorem 4.6 *(Single-Layer Limit).* No single-layer ReLU-network can provably α-robustly classify $\lceil \frac{2}{\alpha} \rceil + 5$ or more flips for any $\alpha \in (0, 1]$.

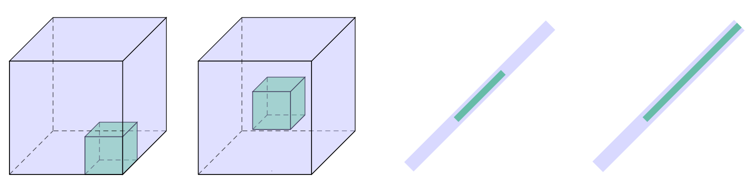
Proof Overview. The proof is by direct application of Lemma 4.5, and expansion of the definition of the derivative. We demonstrate that for the $\hat{x}$ found by Lemma 4.5, the center of the box must be strictly closer to 0 than it's radius.

(Full proof in Appendix A)

5 COMPLETELY INTERVAL PROVABLE CLASSIFIERS ARE IMPOSSIBLE

Here we show our main result, that no neural network can be completely provably robust with interval analysis for simple functions. We first introduce the necessary lemmas and machinery that allow us show a relationship between whether the network represents an invertible function, and where there is approximation error.

Counterintuitively, rather than being able to show that the transformed network in imprecise for a specific input box (i.e., that for a specific box B, we know $f[B] \subsetneq f^{\#}(B)$), we must, for any input box B containing non-invertible points on its surface, *find* an input box, A, that is a strict subsets of B (by a particular notion of strict defined below), such that $f(B) \subseteq f^{\#}(A)$. The fact that A is a very strict subset of B implies that interval analysis is imprecise enough on the network such that it can not be used to prove desired properties of B (such that f is completely robust for B). It is however crucial that A not be required to be *too* strict a subset of B. One might be tempted to find subsets of the topological interior of B. This however leads to significant technical issues: we need to have a notion of strict subset that applies even when some of the neurons in the network are unused (and zero). One can imagine the set representing the possible activations of those neurons as a lower dimensional surface embedded in a higher dimensional space, as in the case of Fig. 3(c) and (d). In this case, the interior of B would be empty, even though we might have identified a subset of it that induces imprecision.



(a) $A \not\sqsubset B$ (b) $A \sqsubset B$ (c) $A \sqsubset B$ (d) $A \not\sqsubset B$

Fig. 3. Visualization of the relative interior relation. The green (■) boxes are A and the purple (■) boxes are B.

5.1 The Relative Subset Relation

We begin by formalizing the intuitive concept from Fig. 3 using the notion of *relative interior*, and demonstrating some useful lemmas related to it. First, recall for a set $S \subseteq \mathbb{R}^d$ that the *affine hull* of S, written $\text{aff}(S)$ is the smallest linear-subspace of $\mathbb{R}^d$ that contains S.

Definition 5.1. We define *relative interior* as $\text{relint}(S) \coloneqq \{x \in S \colon \exists \epsilon > 0.\, \mathbb{B}^\circ_\epsilon(x) \cap \text{aff}(S) \subseteq S\}$

We note that if $S \in \mathcal{B}^d$, the set of closed, non-empty, axis-aligned boxes of dimension d, we can restate the relative interior as $\text{relint}(S) = \{x \in S \colon \forall i \in [d].\, x \in S|^\circ_i \cup \{C(S)_i\}\}$, where $S|^\circ_i$ is the interior of S's restriction to dimension i.

Definition 5.2 *(Relative Subset).* A is a *relative subset* of B, written $A \sqsubset B$, if and only if $A \subseteq \text{relint}(B)$.

We note again, that if $A, B \in \mathcal{B}^d$, we can rephrase $A \sqsubset B$ as follows: $A \subseteq B$ and for each dimension, i, where $B|^\circ_i$ isn't empty, $A|_i \subseteq B|^\circ_i$ or more concisely, $\forall i \in [d].\, (B|^\circ_i \neq \emptyset \implies A|_i \subseteq B|^\circ_i)$. In particular, in one dimension, for real intervals $[a, b]$ and $[b, c]$ we have $[a, b] \sqsubset [a', b']$ if and only if $a < a' \leq b' < b$ or $a = a' = b' = b$.

Let A, A', B, B', C be bounded and non-empty subsets of $\mathbb{R}^d$ in the following lemmas (the proofs of which can be found in Appendix B.1):

Lemma 5.3 *(Respects Projection).* $A \sqsubset B$ implies $A|_i \sqsubset B|_i$.

Lemma 5.4 *(Respects Cartesian Product).* $A \sqsubset B$ and $A' \sqsubset B'$ implies $A \times A' \sqsubset B \times B'$.

Lemma 5.5 *(Downward Union).* $A \sqsubset C$ and $B \sqsubset C$ implies $A \cup B \sqsubset C$.

Lemma 5.6 *(Downward Hull).* $C \in \mathcal{B}^d$ and $A \sqsubset C$ implies $\mathcal{H}_\infty(A) \sqsubset C$.

The following two trivial lemmas are trivial, and we frequently use them without mention:

Lemma 5.7 *(Singleton Reflexivity).* $\{a\} \sqsubset \{a\}$.

Lemma 5.8 *(Center-Singleton is Always a Relative Subset).* $A \in \mathcal{B}^d$ implies $\{C(A)\} \sqsubset A$.

It is important to note that some simple related properties counterintuitively do not always hold. Namely, if $A \sqsubset B$ and $B \subseteq C$ it is not always the case that $A \sqsubset C$. Furthermore, if $A \sqsubset B$ and $A' \sqsubset B'$ it is not always the case that $\mathcal{H}_\infty(A \cup A') \sqsubset \mathcal{H}_\infty(B \cup B')$.

5.2 Inversion With Respect to The Relative Subset Relation

Here we demonstrate that neural networks can loosely invert sets with respect to the relative subset relation. More formally, for any neural network f with ReLU-activations, one can essentially always find a strict subset, X' of the relative interior of a box X that the neural network maps to a superset of a specified subset Y of the relative interior of the l_∞-hull of $f(X)$.

Lemma 5.9 *(Concrete Relative Inversion).* Suppose f is a feed-forward network with ReLU-activations and $Y, X' \in \mathcal{B}^d$ and X is compact and non-empty. Then

$$Y \sqsubset \mathcal{H}_\infty(f(X)) \implies \exists X' \sqsubset \mathcal{H}_\infty(X).\, Y \subseteq f^\#(X').$$

Proof Overview. *(Full Proof in Appendix B.2)* The proof is by structural induction on the construction of f. We use the lemma itself as the induction hypothesis. Below we outline three key cases of the structural induction: sequential computations, relational parallel computations, and ReLU:

Case: $f = g \circ h$. *(Sequential Computation)*

Subproof. By definition, $Y \sqsubset \mathcal{H}_\infty(g \circ h(X))$. Thus, there is some $H \sqsubset \mathcal{H}_\infty(h(X))$ such that $Y \subseteq g^\#(H)$ by the induction hypothesis on g. Applying the induction hypothesis again with the network h, we get a set $X' \sqsubset \mathcal{H}_\infty(X)$ such that $H \subseteq h^\#(X')$. Thus, $Y \subseteq g^\#(H) \subseteq g^\# \circ h^\#(X') = f^\#(X')$. ◂

Case: $f(x) = (x, x)$. *(Relational Duplication)*

Subproof. In this case, we know $Y|_1 \sqsubset \mathcal{H}_\infty(X)$ and $Y|_2 \sqsubset \mathcal{H}_\infty(X)$ by Lemma 5.3. By Lemma 5.5, we know $X' \coloneqq \mathcal{H}_\infty(Y|_1 \cup Y|_2)$ is a relative subset of $\mathcal{H}_\infty(X)$, and also that $Y \subseteq X' \times X' = f^\#(X')$. ◂

Case: $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$. *(Non-Relational Parallel Computation)*

Subproof. We know $Y|_1 \sqsubset \mathcal{H}_\infty(g_1(X|_1))$ and $Y|_2 \sqsubset \mathcal{H}_\infty(g_1(X|_1))$, so we can apply the induction hypothesis twice to produce $L \sqsubset \mathcal{H}_\infty(X|_1)$ and $R \sqsubset \mathcal{H}_\infty(X|_2)$ such that $Y|_1 \subseteq g_1^\#(L)$ and $Y|_2 \subseteq g_1^\#(R)$. We choose $X' = L \times R$. By Lemma 5.4, we have $X' \sqsubset \mathcal{H}_\infty(X)$. Then $Y \subseteq g_1^\#(X'|_1) \times g_2^\#(X'|_2) = f^\#(X')$. ◂

Case: $f(x) = \mathrm{ReLU}(x)$. *(Activation)*

Subproof. We know here that $f : \mathbb{R} \to \mathbb{R}$ which simplifies the proof. In this case, l_∞-hull of $f(X)$ must either be a subset of $\mathbb{R}_{\geq 0}$ so either Y is the singleton set containing zero, or a subset of $\mathbb{R}_{>0}$. In the first case, we pick X' to be an easy-to-pick (the singleton set containing center as in Lemma 5.8) relative subset of the hull of X. Otherwise, we can pick Y itself, since $\mathrm{ReLU}(Y) = Y \subseteq \mathcal{H}_\infty(X)$. ◂

(Further Cases in Appendix B.2)

5.3 Impossibility for Non-Invertibility

We now prove our central result, that non-invertible neural networks necessarily induce approximation imprecision. Essentially, as visualized in Fig. 4, we show that there is a box, M which is a relative subset (i.e., usually very strict subset) of the l_∞-hull of any region for which the network is entirely not injective, such that analyzing the network with M includes any of the non-invertible points in the inferred approximation.

The key idea is that while there might be non-invertible points on the boundary of a region, M does not include those points, since it is a relative subset. Thus, we can use this theorem to infer areas where analyzing the network produces approximations that include points that aren't in the concrete, or true, set of possible network outputs.

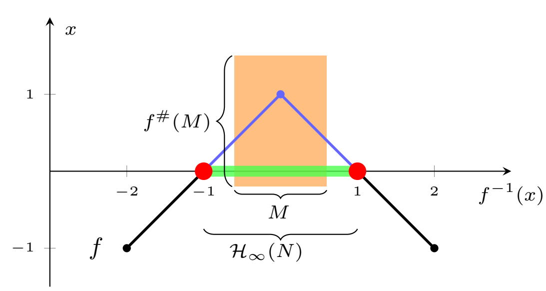


Fig. 4. A visualization of the claims of Theorem 5.10. The neural network, f is classifying three flips $\hat{x}_{-1}$, $\hat{x}_0$ and $\hat{x}_1$ (i.e., $D = \{(-2,-1), (0,1), (2,-1)\}$ as in Corollary 5.12). Here, $x = 0$ is the non-invertability and $f^{-1}(x) = \{-1, 1\}$. These two points mapping to x, which constitute $N \subseteq f^{-1}(x)$, are marked in red (■). The l_∞-hull, $\mathcal{H}_\infty(N)$, is marked in green (■). The perfect approximation of f on $\mathcal{H}_\infty(N)$ (i.e., $\{(v, f(v)) : v \in N\}$) is marked in blue (■). We can see that the interval approximation of the relative interior box $M \sqsubset \mathcal{H}_\infty(N)$ looses precision by looking at the orange region (■).

Theorem 5.10 *(Non-Invertibility Induces Interval Imprecision).* Suppose $f: \mathbb{R}^n \to \mathbb{R}^m$ is a feed-forward network with ReLU-activations and $x \in \mathbb{R}^m$ and $N \subseteq f^{-1}(x)$ is compact and non-empty. Then (assuming M is a box):

$$\exists M \sqsubset \mathcal{H}_\infty(N).\, x \in f^\#(M).$$

Proof Overview. *(Full Proof in Appendix B.2)* Again, the proof is by structural induction on the construction of f, using the lemma itself as the induction hypothesis. Below we outline three key cases: sequential computations, relational parallel computations, and addition:

Case: $f = g \circ h$ *(Sequential Computation)*

Subproof. We know that $N \subseteq h^{-1} \circ g^{-1}(x)$, and thus can use the induction hypothesis on h to produce $M' \sqsubset \mathcal{H}_\infty(h(N))$ such that $x \in g^\#(M')$. By Lemma 5.9, we know that there is some $M \sqsubset \mathcal{H}_\infty(N)$ such that $M' \subseteq h^\#(M)$ and thus that $x \in g^\# \circ h^\#(M)$. ◁

Case: $f(y) = (y, y)$. *(Relational Duplication)*

Subproof. We have $N \subseteq \{x_1\} \cap \{x_2\}$ so $N = \{x_1\}$. By singleton reflexivity, $N \sqsubset \mathcal{H}_\infty(N)$. Thus, $x \in f^\#(N)$. ◁

Case: $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$. *(Non-Relational Parallel Computation)*

Subproof. First we know $N|_1 \subseteq g_1^{-1}(x_1)$ and $N|_2 \subseteq g_2^{-1}(x_2)$ by projection and that $N|_1$ and $N|_2$ are still compact and non-empty. Thus, by the induction hypothesis twice we see that there are boxes $M_1 \sqsubset \mathcal{H}_\infty(N|_1)$ and $M_2 \sqsubset \mathcal{H}_\infty(N|_2)$ such that $x_1 \in g_1^\#(M_1)$ and $x_2 \in g_2^\#(M_2)$. Then $M_1 \times M_2 \sqsubset \mathcal{H}_\infty(N)$ by Lemma 5.4. Then $x_1 \in g_1^\#(M_1 \times M_2)$ and $x_2 \in g_2^\#(M_1 \times M_2)$ by soundness. Thus, there is some box $M \sqsubset \mathcal{H}_\infty(N)$ such that $x \in f^\#(M)$. ◁

Case: $f(y_1, y_2) = y_1 + y_2$. *(Addition)*

Subproof. Because $f: \mathbb{R}^2 \to \mathbb{R}$, we know $f^{-1}(x) = \{(a, x - a) : a \in \mathbb{R}\}$. We can pick $M = \{C(\mathcal{H}_\infty(N))\}$ and demonstrate that $M \sqsubset \mathcal{H}_\infty(N)$ and that $x \in f^\#(M)$. ◁

(Further Cases in Appendix B.2)

5.4 Implications and Corollaries

Here we demonstrate implications of this result, such as that no neural network can completely (as defined below), and provably robustly with interval, classify every dataset.

Definition 5.11. We say that a neural network $f: \mathbb{R}^d \to \mathbb{R}$ is a *completely ν-(provable) robust classifier* for points $x_1, \ldots, x_n \in \mathbb{R}^d$ and labels $l_1, \ldots, l_n \in \{-1, 1\}$ if given $\delta = \frac{1}{2} \min_{i \neq j} ||x_i - x_j||_\infty$ then $\forall \epsilon < \nu\delta.\, \forall i \in [d].\, f(\mathbb{B}_\epsilon(x_i))l_i > 0$. If it is *provable* then we also have that $\forall i \in [d].\, f^\#(\mathbb{B}_\epsilon(x_i))l_i > 0$.

We note that the specification task implicitly induced by this definition is complete as defined by the more general notion in the introduction.

Corollary 5.12 *(Completely Provable Classifiers are Impossible).* There is no feed forward ReLU-network that is a completely 1-provable classifier for the dataset $D = \{(-2, -1), (0, 1), (2, -1)\}$.

Proof. Suppose $f: \mathbb{R} \to \mathbb{R}$ is a completely 1-provable interpolator for this dataset. Then by continuity we know that $f(-1) = f(1) = 0$, and thus that $\{-1, 1\} \subseteq f^{-1}(0)$ (which is a compact non-empty set). Then by application of Theorem 5.10, there is some set $M \sqsubset \mathcal{H}_\infty(\{-1, 1\})$ such that $0 \in f^\#(M)$. Rephrased, this means that there are

$a, b \in (-1, 1)$ such that $0 \in f^{\#}([a, b])$. This contradicts with the definition of a completely 1-provable classifier however. ◻

***Proposition 5.13** (Single-Layer Completely Robust Classifiers Always Exist).* For *any* dataset of n points $x_i \in \mathbb{R}$ and labels $l_i \in \{-1, 1\}$ there is a one-hidden-layer ReLU-network that completely robustly (but not necessarily provably) classifies it.

Proof. We present the construction explicitly. Let $\delta \coloneqq \min\{|x_i - x_j| : i \neq j\}$ in

$$f(y) \coloneqq \sum_{i=1}^{n} l_i \left(\text{ReLU}\left[\frac{1}{\delta}(y - (x_i - \delta))\right] - \text{ReLU}\left[\frac{2}{\delta}(y - x_i)\right] + \text{ReLU}\left[\frac{1}{\delta}(y - (x_i + \delta))\right]\right)$$

One can check that this works by plugging in x_j, although a full proof is by induction. While this is not immediately of the form described for one-hidden layer networks, one can see easily how to algebraically convert this into that form. Because we only care about the robustness, and not interval provability of f, this is sufficient. ◻

6 DISCUSSION AND FUTURE WORK

While we limited the scope of our discussion to ReLU-activations, we note that our theorems extetend trivially to any monotone bounded activation. However, we observe that non-monotonic activations functions (such as absolute value) do not admit the same forms of theorems. Our preliminary experiments however indicate that substituting ReLU with these activation functions does not result in easier training or provability. This suggests that there are more general version of the the theorems presented here, in particular relating the difficulty of program synthesis with the relational expressiveness of the domain used to verify the specification.

7 CONCLUSION

In this paper, we proved two theorems that show limits in the expressiveness of interval provable neural networks. We showed that no ReLU-network can completely provably classify simple one-dimensional datasets containing only three points. This indicates a fundamental loss of precision whenever ReLU-networks are analyzed using interval arithmetic, which can not be regained, no matter the network. Further, we showed that a single hidden layer ReLU-network can not provably classify simple datasets even without the requirement for completeness, which is in stark contrast to classical universal approximation theorems, where a single hidden layer is sufficient. This shows that the approximative capabilities of interval provable networks are lower compared to standard neural networks.

ACKNOWLEDGMENTS

We thank Joseph Swernofsky for his helpful comments on earlier drafts of this work.

REFERENCES

[Ano22] Anonymous. On the convergence of certified robust training with interval bound propagation. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. under review.

[BBS+19] Mislav Balunovic, Maximilian Baader, Gagandeep Singh, Timon Gehr, and Martin Vechev. Certifying geometric robustness of neural networks. In *NeurIPS*, 2019.

[BIL+16] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya V. Nori, and Antonio Criminisi. Measuring neural net robustness with constraints. In *NeurIPS*, 2016.

[BMV20] Maximilian Baader, Matthew Mirman, and Martin Vechev. Universal approximation with certified networks. *ICLR*, 2020.

[BTD+16] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv preprint arxiv:1604.07316*, 2016.

[BV20] Mislav Balunovic and Martin Vechev. Adversarial training and provable defenses: Bridging the gap. In *ICLR*, 2020.

[BWC+19] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *AAAI*, volume 33, 2019.

[BWL+21] Akhilan Boopathy, Tsui-Wei Weng, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Luca Daniel. Fast training of provably robust neural networks by singleprop. *arXiv preprint arXiv:2102.01208*, 2021.

[CAH18] Francesco Croce, Maksym Andriushchenko, and Matthias Hein. Provable robustness of relu networks via maximization of linear regions. *arXiv preprint arXiv:1810.07481*, 2018.

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Symposium on Principles of Programming Languages (POPL)*, 1977.

[CH19a] Francesco Croce and Matthias Hein. Provable robustness against all adversarial l_p-perturbations for $p >= 1$. In *ICLR*, 2019.

[CH19b] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[CH20] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[CW17] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *Symposium on Security and Privacy (SP)*, 2017.

[Cyb89] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems (MCSS)*, 1989.

[GDS+18] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

[GMT+18] Timon Gehr, Matthew Mirman, Petar Tsankov, Dana Drachsler Cohen, Martin Vechev, and Swarat Chaudhuri. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *Symposium on Security and Privacy (SP)*, 2018.

[GSS15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[HJVE01] Timothy Hickey, Qun Ju, and Maarten H Van Emden. Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 48(5):1038–1068, 2001.

[HKR+20] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Computer Science Review*, 37:100270, 2020.

[HSW89] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 1989.

[KBD+17] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification (CAV)*, 2017.

[KGB16] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arxiv:1607.02533*, 2016.

[LAL+19] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, and Mykel J Kochenderfer. Algorithms for verifying deep neural networks. *arXiv preprint arXiv:1903.06758*, 2019.

[LSR+21] Wan-Yi Lin, Fatemeh Sheikholeslami, Leslie Rice, J Zico Kolter, et al. Certified robustness against physically-realizable patch attack via randomized cropping. In *ICLR*, 2021.

[LSS21] Chen Liu, Mathieu Salzmann, and Sabine Süsstrunk. Training provably robust models by polyhedral envelope regularization. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[LTC19] Chen Liu, Ryota Tomioka, and Volkan Cevher. On certifying non-uniform bound against adversarial attacks. In *ICML*, 2019.

[MGV18] Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.

[Moo66] Ramon E Moore. *Interval analysis*. Prentice-Hall Englewood Cliffs, NJ, 1966.

[PMG16] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[PMW+16] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 2016.

[RSL18] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *ICLR*, 2018.

[SGM+18] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *NeurIPS*, 2018.

[SGPV19] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. In *POPL*, 2019.

[SHS20] David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *ICML*, 2020.

[SWZ+21] Zhouxing Shi, Yihan Wang, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Fast certified robust training with short warmup. *Advances in Neural Information Processing Systems*, 2021.

[SYN15] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arxiv:1511.05432*, 2015.

[SYZ+19] Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robustness verification of neural networks. In *NeurIPS*, 2019.

[SZS+13] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[TKP+17] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[Tuc02] Warwick Tucker. A rigorous ode solver and smale's 14th problem. *Foundations of Computational Mathematics*, 2(1):53–117, 2002.

[WAPJ20] Zi Wang, Aws Albarghouthi, Gautam Prakriya, and Somesh Jha. Interval universal approximation for neural networks. In *POPL*, 2020.

[WK18] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. 2018.

[WPW+18] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *NeurIPS*, 2018.

[WRK19] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2019.

[WSK19] Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *ICML*, 2019.

[WSMK18] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

[XTSM19] Kai Xiao, Vincent Tjeng, Nur Muhammad Shafiullah, and Aleksander Madry. Training for faster adversarial robustness verification via inducing relu stability. In *International Conference on Learning Representations*, 2019.

[XZW+21] Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and Complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2021.

[ZAD21] Yuhao Zhang, Aws Albarghouthi, and Loris D'Antoni. Certified robustness to programmable transformations in lstms. In *EMNLP*, 2021.

[ZCX+20] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *ICLR*, 2020.

[ZWC+18] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems (NuerIPS)*, dec 2018.

A EXTENDED PROOFS FOR SINGLE HIDDEN LAYER NETWORK RESULTS

Here we restate the theorems and show the full proofs for the results in Section 4.

Lemma 4.4 *(End-Neuron Imprecision-Bound).* For all $\kappa \geq 0$ and single-layer ReLU-networks, f, that classify k-flips for $k = \lceil \kappa \rceil + 5$, we have $\kappa < \max\{\mathcal{A}_{L,\mathbb{R}}(\hat{x}_1), \mathcal{A}_{R,\mathbb{R}}(\hat{x}_k)\}$.

Proof of 4.4. We prove this by induction on c.

Induction Hypothesis: Given $c \in \mathbb{N} \cup \{0, -1, -2\}$ there is some even natural number $k \leq \text{ReLU}(c + 2) + 2$ such that for any single-layer ReLU-network, f that classifies k flips we have

$$
\begin{aligned}
c &\leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_k) - \mathcal{A}_{L,+}(\hat{x}_k), \\
\text{and } \ c &\leq \mathcal{A}_{L,-}(\hat{x}_1) - \mathcal{A}_{R,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_{k-1}) - \mathcal{A}_{L,-}(\hat{x}_{k-1}).
\end{aligned}
$$

Base Case: Suppose $c \leq 0$.
Subproof. Pick $k = 2$. Then

$$
\begin{aligned}
\mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_2) - \mathcal{A}_{L,+}(\hat{x}_2) = 0 \geq c, \quad \text{and} \\
\mathcal{A}_{L,-}(\hat{x}_1) - \mathcal{A}_{R,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_{2-1}) - \mathcal{A}_{L,-}(\hat{x}_{2-1}) = 0 \geq c.
\end{aligned}
$$
◁

Induction Step: Suppose $c > 0$, and the induction hypothesis holds for $c - 2$.
Subproof. Then there is some even natural $k' \leq \text{ReLU}(c - 2 + 2) + 2$ such that for any single-layer ReLU-network, f, that is a classifier for k' flips we have

$$
\begin{aligned}
c - 2 &\leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_{k'}) - \mathcal{A}_{L,+}(\hat{x}_{k'}), \\
\text{and } \ c - 2 &\leq \mathcal{A}_{L,-}(\hat{x}_1) - \mathcal{A}_{R,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_{k'-1}) - \mathcal{A}_{L,-}(\hat{x}_{k'-1}).
\end{aligned}
$$

Pick $k = k' + 2$. Then k is even, and $k \leq \text{ReLU}(c) + 4 \leq \text{ReLU}(c + 2) + 2$ since $c > 0$.
Let f be any single-layer ReLU-network that classifies k flips. Then f also classifies k' flips.
We only show the positive bound, that $c \leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_k) - \mathcal{A}_{L,+}(\hat{x}_k)$.
The proof for the negative bound is analogous.
There must be some point $l \in [\hat{x}_{k'}, \hat{x}_{k'+1}]$ such that $f'(l) \leq -1$ by the mean value theorem (since ReLU-networks are continuous) and because $f(\hat{x}_{k'}) = 1 = -f(\hat{x}_{k'+1})$.
Similarly, there must be some point $u \in [\hat{x}_{k'+1}, \hat{x}_k]$ such that $1 \leq f'(u)$. Thus,

$$
\begin{aligned}
2 &\leq f'(u) - f'(l) \\
&\leq \mathcal{I}_{L,+}(u) + \mathcal{I}_{L,-}(u) + \mathcal{I}_{R,+}(u) + \mathcal{I}_{R,-}(u) \\
&\quad - \mathcal{I}_{L,+}(l) - \mathcal{I}_{L,-}(l) - \mathcal{I}_{R,+}(l) - \mathcal{I}_{R,-}(l).
\end{aligned}
$$

We know $\mathcal{I}_{L,-}(u) - \mathcal{I}_{L,-}(l) \leq 0$ and $\mathcal{I}_{R,-}(u) - \mathcal{I}_{R,-}(l) \leq 0$ and $\mathcal{I}_{L,+}(t) = -\mathcal{A}_{L,+}(t)$ and $\mathcal{I}_{R,+}(t) = \mathcal{A}_{R,+}(t)$ for any t so

$$
\begin{aligned}
2 &\leq -\mathcal{A}_{L,+}(u) + \mathcal{A}_{R,+}(u) \\
&\quad + \mathcal{A}_{L,+}(l) - \mathcal{A}_{R,+}(l).
\end{aligned}
$$

We also know $\mathcal{A}_{L,S}(t)$ increases as t decreases and $\mathcal{A}_{R,S}(t)$ increases as t increases, so

$$\begin{aligned} 2 \leq &-\mathcal{A}_{L,+}(\hat{x}_k) + \mathcal{A}_{R,+}(\hat{x}_k) \\ &+ \mathcal{A}_{L,+}(\hat{x}_{k'}) - \mathcal{A}_{R,+}(\hat{x}_{k'}). \end{aligned}$$

By combining with the positive inductive bound, we get

$$\begin{aligned} (c-2) + 2 = c &\leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) + \mathcal{A}_{R,+}(\hat{x}_{k'}) - \mathcal{A}_{L,+}(\hat{x}_{k'}) \\ &- \mathcal{A}_{L,+}(\hat{x}_k) + \mathcal{A}_{R,+}(\hat{x}_k) + \mathcal{A}_{L,+}(\hat{x}_{k'}) - \mathcal{A}_{R,+}(\hat{x}_{k'}) \\ &\leq \mathcal{A}_{L,+}(\hat{x}_2) - \mathcal{A}_{R,+}(\hat{x}_2) - \mathcal{A}_{L,+}(\hat{x}_k) + \mathcal{A}_{R,+}(\hat{x}_k), \end{aligned}$$

which proves the positive bound of the induction hypothesis for c. ◃

Thus, by induction, we find that there is some $k \leq \lceil c \rceil + 5$ such that, after removing the negative terms and increasing by swapping $\hat{x}_2$ with $\hat{x}_1$ and $\hat{x}_{k-1}$ with $\hat{x}_k$:

$$\begin{aligned} c + 1 &\leq \mathcal{A}_{L,+}(\hat{x}_1) + \mathcal{A}_{R,+}(\hat{x}_k), \\ \text{and } \; c + 1 &\leq \mathcal{A}_{L,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_k). \end{aligned}$$

Summing these equations together gives us:

$$2c + 2 \leq \mathcal{A}_{L,+}(\hat{x}_1) + \mathcal{A}_{R,+}(\hat{x}_k) + \mathcal{A}_{L,-}(\hat{x}_1) + \mathcal{A}_{R,-}(\hat{x}_k) = 2\max\{\mathcal{A}_{L,\mathbb{R}}(\hat{x}_1), \mathcal{A}_{R,\mathbb{R}}(\hat{x}_k)\},$$

and thus that $c < \max\{\mathcal{A}_{L,\mathbb{R}}(\hat{x}_1), \mathcal{A}_{R,\mathbb{R}}(\hat{x}_k)$. □

Lemma 4.5 *(Lower-Bound on Imprecision-Contribution).* For any $a \leq 1$ and $k \geq \lceil \frac{2}{a} \rceil + 5$ and single-layer ReLU-network, f, that classifies k flips, there is some point $j \in [k]$ such that

$$\hat{1}_j a^{-1}(f(\hat{x}_j + a) + f(\hat{x}_j - a)) < \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{x}_j) + \mathcal{A}_{R,\mathbb{R}}(\hat{x}_j - a) + \mathcal{A}_{L,\mathbb{R}}(\hat{x}_j + a).$$

Proof of 4.5. Let $k \geq \lceil 2a^{-1} \rceil + 5$, and define $c := \frac{2}{a}$. Because $k \geq \lceil |c| \rceil + 5$, we can use Lemma 4.4 to get that $|c| < \max\{\mathcal{A}_{L,\mathbb{R}}(\hat{x}_1), \mathcal{A}_{R,\mathbb{R}}(\hat{x}_k)\}$.

For convenience, we define $\tilde{f}_L(x) = a^{-1}(f(x) - f(x-a))$ and $\tilde{f}_R(x) = a^{-1}(f(x+a) - f(x))$.

We only show the proof when $|c| < \mathcal{A}_{L,\mathbb{R}}(\hat{x}_1)$, the other case is analogous, but picking $j = k$.

In this case we know $0 < \mathcal{A}_{L,\mathbb{R}}(\hat{x}_1)$ and thus,

$$\hat{1}_1 a^{-1}(f(\hat{x}_1 + a) + f(\hat{x}_1 - a)) \leq \tilde{f}_L(\hat{x}_1) - \tilde{f}_R(\hat{x}_1) - c.$$

There must be a point $l \in [\hat{x}_1 - a, \hat{x}_1]$ such that $\tilde{f}_L(l) \leq f'(l)$ and a point $u \in [\hat{x}_1, \hat{x}_1 + a]$ such that $f'(u) \leq \tilde{f}_R(u)$. We can thus derive, in a manner similar to what is seen in Lemma 4.4:

$$\begin{aligned} \tilde{f}_L(\hat{x}_1) - \tilde{f}_R(\hat{x}_1) \leq f'(l) - f'(u) &\leq \mathcal{I}_{L,+}(l) + \mathcal{I}_{L,-}(l) + \mathcal{I}_{R,+}(l) + \mathcal{I}_{R,-}(l) \\ &- \mathcal{I}_{L,+}(u) - \mathcal{I}_{L,-}(u) - \mathcal{I}_{R,+}(u) - \mathcal{I}_{R,-}(u) \\ &\leq \mathcal{I}_{L,-}(l) - \mathcal{I}_{R,-}(u) \\ &\leq \mathcal{A}_{L,\mathbb{R}}(\hat{x}_1 - a) - \mathcal{A}_{R,\mathbb{R}}(\hat{x}_1 + a). \end{aligned}$$

$\tilde{f}_L(\hat{x}_1) - \tilde{f}_L(\hat{x}_1) - c < \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{x}_1) + \mathcal{A}_{L,\mathbb{R}}(\hat{x}_1 - a) + \mathcal{A}_{R,\mathbb{R}}(\hat{x}_1 + a)$ as $-c \leq |c| < \mathcal{A}_{L,\mathbb{R}}(\hat{x}_1) \leq \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{x}_1)$. □

Theorem 4.6 *(Single-Layer Limit).* No single-layer ReLU-network can provably α-robustly classify $\lceil \frac{2}{\alpha} \rceil + 5$ or more flips for any $\alpha \in (0, 1]$.

Proof of 4.6. Suppose $\alpha \in (0, 1]$, and assume, for the sake of contradiction, that f is a single-layer ReLU-network with weights N, M and biases b, d that provably α-robustly classifies $\lceil \frac{2}{\alpha} \rceil + 5$ flips.
We begin the proof by labeling the intermediate states of interval analysis for a point $\hat{\mathrm{x}}_j$ with interval radius α of the network f:

$$\begin{aligned} v^-_{\alpha,j} &\coloneqq \mathrm{ReLU}(N\hat{\mathrm{x}}_j + b - |N|\alpha) \\ v^+_{\alpha,j} &\coloneqq \mathrm{ReLU}(N\hat{\mathrm{x}}_j + b + |N|\alpha) \\ w_{\alpha,j} &\coloneqq v^+_{j,\alpha} - v^-_{j,\alpha} \\ c_{\alpha,j} &\coloneqq v^+_{j,\alpha} + v^-_{j,\alpha} \\ f^\#(\langle \hat{\mathrm{x}}_j, \alpha \rangle) &\coloneqq \langle \frac{1}{2} M c_{\alpha,j} + d, \frac{1}{2}|M| w_{\alpha,j} \rangle, \end{aligned}$$

where the notation $|X|$ means the point-wise absolute value (i.e., $|X|_i = |X_i|$).
Then our assumption for contradiction tells us that for any $j \in [k]$ we know that $(-1)^j (M c_{\alpha,j} + 2d) \geq |M| w_{\alpha,j}$.
Let j be such that Lemma 4.5 tells us has $\hat{1}_j \alpha^{-1}(f(\hat{\mathrm{x}}_j + \alpha) + f(\hat{\mathrm{x}}_j - \alpha)) < \mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{\mathrm{x}}_j) + \mathcal{A}_{R,\mathbb{R}}(\hat{\mathrm{x}}_j - \alpha) + \mathcal{A}_{L,\mathbb{R}}(\hat{\mathrm{x}}_j + \alpha)$.
We note that by expanding the definitions, sums, and meaning of absolute value, we can derive that $M c_{\alpha,j} + 2d = M(v^+_{\alpha,j} + v^+_{\alpha,j}) + 2d = f(\hat{\mathrm{x}}_j + \alpha) + f(\hat{\mathrm{x}}_j - \alpha)$. so our assumption for contradiction thus implies $|M| w_{\alpha,j} \leq (-1)^j (M c_{\alpha,j} + 2d) = (-1)^j (f(\hat{\mathrm{x}}_j + \alpha) + f(\hat{\mathrm{x}}_j - \alpha))$.
We perform the following deduction:

$$\begin{aligned} (-1)^j (f(\hat{\mathrm{x}}_j + \alpha) + f(\hat{\mathrm{x}}_j - \alpha)) &\geq \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq -\alpha |N_i|} |M_i| (N_i x + b_i + \alpha |N_i|) - |M| v^-_{\alpha,j} \\ &\geq \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq 0} |M_i| (N_i x + b_i + \alpha |N_i|) - |M| v^-_{\alpha,j} \\ &\geq |M| \,\mathrm{ReLU}(Nx + b) + \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq 0} \alpha |M_i N_i| \\ &- \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq \alpha |N_i|} |M_i| (N_i x + b_i - \alpha |N_i|) \\ &\geq \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq 0} \alpha |M_i N_i| + \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq \alpha |N_i|} \alpha |M_i N_i| \\ &\geq \alpha \left(\sum_{N_i \hat{\mathrm{x}}_j + b_i \geq 0} |M_i N_i| + \sum_{N_i \hat{\mathrm{x}}_j + b_i \geq \alpha |N_i|} |M_i N_i| \right) \\ &\geq \alpha (\mathcal{A}_{\mathbb{R},\mathbb{R}}(\hat{\mathrm{x}}_j) + \mathcal{A}_{R,\mathbb{R}}(\hat{\mathrm{x}}_j - \alpha) + \mathcal{A}_{L,\mathbb{R}}(\hat{\mathrm{x}}_j + \alpha)) \\ &\geq \hat{1}_j (f(\hat{\mathrm{x}}_j + \alpha) + f(\hat{\mathrm{x}}_j - \alpha)) \end{aligned}$$

which is a contradiction. □

B EXTENDED PROOFS FOR GENERAL IMPOSSIBILITY RESULTS

Here we restate the theorems and show the full proofs for the results in Section 5.

B.1 Proofs for Relative Interior Lemmas

In the following lemmas, let A, A', B, B', C be bounded and non-empty subsets of $\mathbb{R}^d$:

Lemma 5.3 *(Respects Projection).* $A \sqsubset B$ implies $A|_i \sqsubset B|_i$.

Proof of 5.3. Let $y \in A|_i$. Then there is some $x \in A$ such that $x_i = y$. Then $x \in \text{relint}(B)$ by $A \sqsubset B$. Then there is some $\epsilon > 0$ such that $N_\epsilon(x) \cap \text{aff}(B) \subseteq B$. Then $N_\epsilon(x)|_i \cap \text{aff}(B)|_i \subseteq B|_i$. We know $\text{aff}(B|_i) \subseteq \text{aff}(B)|_i$: given $z \in \text{aff}(B|_i)$, it must be an affine combination of the i'th dimension of elements of B. Letting z' be the same affine combination of those elements, $z'_i = z$, so $z \in \text{aff}(B)|_i$. Then $N_\epsilon(x|_i) \cap \text{aff}(B|_i) \subseteq B|_i$ and thus $y \in \text{relint}(B|_i)$. □

Lemma 5.4 *(Respects Cartesian Product).* $A \sqsubset B$ and $A' \sqsubset B'$ implies $A \times A' \sqsubset B \times B'$.

Proof of 5.4. Let $(x, x') \in A \times A'$. Then because $x \in A$ we know $x \in \text{relint}(B)$ and respectively $x' \in \text{relint}(B')$. Then there is some $\epsilon > 0$ such that $N_\epsilon(x) \cap \text{aff}(B) \subseteq B$, and $\epsilon' > 0$ such that $N_{\epsilon'}(x') \cap \text{aff}(B') \subseteq B'$. We know $(A \cap A') \times (B \cap B') \subseteq (A \times B) \cap (A' \times B')$: $(a, b) \in (A \cap A') \times (B \cap B')$ implies $a \in A \cap A'$ and $b \in B \cap B'$, so $(a, b) \in A \times B$ and $(a, b) \in A' \times B'$ so $(a, b) \in (A \times B) \cap (A' \times B')$.
Then, we know $\text{aff}(B \times B') \subseteq \text{aff}(B) \times \text{aff}(B')$: $(b, b') \in \text{aff}(B \times B')$ implies (b, b') is an affine combination of elements of $B \times B'$ which implies b is an affine combination of elements from B and b' is an affine combination of elements from B', so $(b, b') \in \text{aff}(B) \times \text{aff}(B')$.
Thus for $\lambda = \min\{\epsilon, \epsilon'\}$ we know $N_\lambda(x, x') \cap \text{aff}(B \times B') \subseteq B \times B'$. Thus $(x, x') \in \text{relint}(B \times B')$. □

Lemma 5.5 *(Downward Union).* $A \sqsubset C$ and $B \sqsubset C$ implies $A \cup B \sqsubset C$.

Proof of 5.5. Suppose $x \in A \cup B$. Then $x \in A$ or $x \in B$. Either way, we know $x \in \text{relint}(C)$. □

Lemma 5.6 *(Downward Hull).* $C \in \mathcal{B}^d$ and $A \sqsubset C$ implies $\mathcal{H}_\infty(A) \sqsubset C$.

Proof of 5.6. $C \in \mathcal{B}^d$ implies $C|_i \in \mathcal{B}$, and thus $C|_i$ is convex so $\text{relint}(C|_i)$ is convex. We know $A|_i \sqsubset C|_i$ by Lemma 5.3, so $\mathcal{H}_\infty(A|_i) \sqsubset C|_i$ by convexity of $C|_i$ and that $\mathcal{H}_\infty$ is the convex hull in one dimension. Thus, by Lemma 5.4, we know $\mathcal{H}_\infty(A|_1) \times \cdots \times \mathcal{H}_\infty(A|_d) \sqsubset C|_1 \times \cdots \times C|_d$. Because $C \in \mathcal{B}^d$ we know $C = C|_1 \times \cdots \times C|_d$ and similarly that $\mathcal{H}_\infty(A) = \mathcal{H}_\infty(A|_1) \times \cdots \times \mathcal{H}_\infty(A|_d)$. Thus, $\mathcal{H}_\infty(A) \sqsubset C$. □

B.2 Proofs for Inversion and Impossibility Theorems

Lemma 5.9 *(Concrete Relative Inversion).* Suppose f is a feed-forward network with ReLU-activations and $Y, X' \in \mathcal{B}^d$ and X is compact and non-empty. Then

$$Y \sqsubset \mathcal{H}_\infty(f(X)) \implies \exists X' \sqsubset \mathcal{H}_\infty(X).\, Y \subseteq f^\#(X').$$

Proof of 5.9. The proof is by structural induction on the construction of the network f, assuming the lemma itself as the induction hypothesis for any network with fewer operations than f. First, assume $Y \sqsubset \mathcal{H}_\infty(f(X))$.

Case: $f = g \circ h$. *(Sequential Computation)*

Subproof. By definition, $Y \sqsubset \mathcal{H}_\infty(g \circ h(X))$. Thus, there exists some $H \sqsubset \mathcal{H}_\infty(h(X))$ such that $Y \subseteq g^\#(H)$ by the induction hypothesis on g. Applying the induction hypothesis again with the network h, we get a set $X' \sqsubset \mathcal{H}_\infty(X)$ such that $H \subseteq h^\#(X')$. Thus, $Y \subseteq g(H) \subseteq g^\# \circ h^\#(X') = f^\#(X')$. ◃

Case: $f(x) = (x, x)$. *(Relational Duplication)*

Subproof. Then $Y|_1 \sqsubset \mathcal{H}_\infty(X)$ and $Y|_2 \sqsubset \mathcal{H}_\infty(X)$ by Lemma 5.3, We choose $X' = \mathcal{H}_\infty(Y|_1 \cup Y|_2)$ which we know by Lemma 5.5, is such that $X' \sqsubset \mathcal{H}_\infty(X)$. Thus, $Y|_1 \subseteq X'$ and $Y|_2 \subseteq X'$, so $Y \subseteq X' \times X' = f^\#(X')$. ◃

Case: $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$. *(Non-Relational Parallel Computation)*

Subproof. Then $Y|_1 \sqsubset \mathcal{H}_\infty(g_1(X|_1))$ and $Y|_2 \sqsubset \mathcal{H}_\infty(g_1(X|_1))$ by definition. Then applying the induction hypothesis twice produces $L \sqsubset \mathcal{H}_\infty(X|_1)$ and $R \sqsubset \mathcal{H}_\infty(X|_2)$ such that $Y|_1 \subseteq g_1^\#(L)$ and $Y|_2 \subseteq g_1^\#(R)$. Then we choose $X' = L \times R$ which we know by Lemma 5.4, is such that $X' \sqsubset \mathcal{H}_\infty(X)$. Then $Y \subseteq g_1^\#(X'|_1) \times g_2^\#(X'|_2) = f^\#(X')$. ◃

Case: $f(x) = c$. *(Constant)*

Subproof. Here, any subset $X' \sqsubset X$ will suffice. Then we can let $X' = \{C(X)\}$. ◃

Case: $f(x) = c \cdot x$ for $c \neq 0$. *(Multiplication by a Constant)*

Subproof. Let $X' = \mathbb{B}_{|c^{-1}|\mathcal{R}(Y)}(c^{-1}C(Y))$. Then clearly, $Y \subseteq f(X')$. It remains to show that $X' \sqsubseteq \mathcal{H}_\infty(X)$. For the remainder of this subproof, because we know that $d = 1$ we will write $x_l = \inf X$, $x_u = \sup X$, $y_l = \inf Y$ and $y_u = \sup Y$. We note that $x_l = C(\mathcal{H}_\infty(X)) - \mathcal{R}(\mathcal{H}_\infty(X))$ and so on. Supposing $c > 0$ (the other case is analogous) and $x_l < x_u$ (the proof is similar when they are equal), we have $cx_l < y_l \leq y_u < cx_u$ by $Y \sqsubset \mathcal{H}_\infty(f(X))$.

Then we know $x_l < |c^{-1}|y_l \leq |c^{-1}|y_u < x_u$, and thus $X' \sqsubset \mathcal{H}_\infty(X)$. ◃

Case: $f(x) = \text{ReLU}(x)$. *(Activation)*

Subproof. Again, because we know that $d = 1$ we will write $x_l := \inf X$, $x_u := \sup X$, and $y_l := \inf Y$ and $y_u := \sup Y$. We know that $\inf \mathcal{H}_\infty(f(X)) = \mathrm{ReLU}(x_l)$ and $\sup \mathcal{H}_\infty(f(X)) = \mathrm{ReLU}(x_u)$. Thus by $Y \sqsubset \mathcal{H}_\infty(f(X))$ we know $\mathrm{ReLU}(x_l) \le y_l \le y_u \le \mathrm{ReLU}(x_u)$. We then have two cases we need to address:

Suppose: $x_u > 0$.

Subproof. Here we define $X' = [y_l, y_u]$. We thus have $x_l \le \mathrm{ReLU}(x_l) < y_l \le y_u < \mathrm{ReLU}(x_u) = x_u$ provided $x_l < x_u$. Otherwise we know $x_l = \mathrm{ReLU}(x_l) = y_l = y_u = \mathrm{ReLU}(x_u) = x_u$ so we have $X' \sqsubset \mathcal{H}_\infty(X)$. ◁

Suppose: $x_u \le 0$.

Subproof. Define $X' = \{\frac{x_u + x_l}{2}\}$. We know $\mathrm{ReLU}(m) = 0 = y_l = y_u$ and thus $X' \sqsubset \mathcal{H}_\infty(X)$. ◁

Thus, in both cases we can find $X' \sqsubset \mathcal{H}_\infty(X)$ such that $Y \subseteq f(X') \subseteq f^\#(X')$ ◁

Case: $f(x_1, x_2) = x_1 + x_2$. *(Addition)*

Subproof. Conveniently again, Y is one-dimensional. Either $\mathcal{H}_\infty(f(X))$ is a single point or it is not:

Assume: $\mathcal{H}_\infty(f(X))$ is a single point.

Subproof. Then $\inf Y = \sup Y = \inf \mathcal{H}_\infty(f(X)) = \sup \mathcal{H}_\infty(f(X))$. Let $y = \inf Y$. In this case, we know there is some compact and non-empty set $Z \subseteq \mathbb{R}$ such that $X = \{(x', y - x') : x' \in Z\}$. Then we can pick $X' = \{(C(\mathcal{H}_\infty(Z)), y - C(\mathcal{H}_\infty(Z)))\}$ which is the singleton-set containing the center of the l_∞-hull of X and thus $X' \sqsubset \mathcal{H}_\infty(X)$ by Lemma 5.8. ◁

Otherwise: $\mathcal{H}_\infty(f(X))$ is not a single point.

Subproof. We know $\inf \mathcal{H}_\infty(f(X)) < \inf Y \le \sup Y < \sup \mathcal{H}_\infty(f(X))$. Because Y is one-dimensional and a relative subset of the non-singular $\mathcal{H}_\infty(f(X))$ we know $Y \sqsubset \mathcal{H}_\infty(f(\mathcal{H}_\infty(X)))$.

Let a, b, r_a, r_b, y, r_y be as follows:

$$\begin{aligned} a &= C(\mathcal{H}_\infty(X|_1)), & r_a &= \mathcal{R}(\mathcal{H}_\infty(X|_1)), \\ b &= C(\mathcal{H}_\infty(X|_2)), & r_b &= \mathcal{R}(\mathcal{H}_\infty(X|_2)), \\ y &= C(Y), \quad \text{and} & r_y &= \mathcal{R}(Y). \end{aligned}$$

Then $\mathcal{H}_\infty(X) = \mathbb{B}_{r_a}(a) \times \mathbb{B}_{r_b}(b)$ and thus $Y \sqsubset f(\mathbb{B}_{r_a}(a) \times \mathbb{B}_{r_b}(b)) = \mathbb{B}_{r_a + r_b}(a + b)$.

Choose $X' = \mathbb{B}_r(x)$ for x and r defined as:

$$\begin{aligned} x_1 &= a + r_a \frac{y - a - b}{r_a + r_b}, & r_1 &= \frac{r_y r_a}{r_a + r_b}, \\ x_2 &= b + r_b \frac{y - a - b}{r_a + r_b}, \quad \text{and} & r_2 &= \frac{r_y r_b}{r_a + r_b}. \end{aligned}$$

Then clearly, $x_1 + x_2 = y$, and $r_1 + r_2 = r_y$ so $Y \subseteq f(X')$.

Thus $r_y < r_a + r_b$ by $Y \sqsubset B_{r_a + r_b}(a + b)$.

This also tells us that $y + r_y < a + b + r_a + r_b$ and $y - r_y > a + b - r_a - r_b$. If $r_a \ne 0$ we can derive $x_1 + r_1 < a + r_a$ and $x_1 - r_1 > a - r_a$. Similarly, if $r_b \ne 0$ we can derive $x_2 + r_2 < b + r_b$

and $x_2 - r_2 > b - r_b$. Thus, $X_1 \sqsubset \mathbb{B}_{r_a}(a)$ and $X_2 \sqsubset \mathbb{B}_{r_b}(b)$, and thus by Lemma 5.4 we have $X' \sqsubset \mathbb{B}_{r_a}(a) \times \mathbb{B}_{r_b}(b) = \mathcal{H}_\infty(X)$. ◃

Thus, in both cases, there exists an $X' \sqsubset \mathcal{H}_\infty(X)$ such that $Y \subseteq f(X') \subseteq f^\#(X')$. ◃

As any feed forward neural network (without input-value dependent loops) can be expressed using these operations without modifying the result under interval analysis, by induction $\exists X' \sqsubset \mathcal{H}_\infty(X).\, Y \subseteq f^\#(X')$. □

Theorem 5.10 *(Non-Invertibility Induces Interval Imprecision).* Suppose $f : \mathbb{R}^n \to \mathbb{R}^m$ is a feed-forward network with ReLU-activations and $x \in \mathbb{R}^m$ and $N \subseteq f^{-1}(x)$ is compact and non-empty. Then (assuming M is a box):

$$\exists M \sqsubset \mathcal{H}_\infty(N).\, x \in f^\#(M).$$

Proof of 5.10. The proof is by structural induction on the construction of the network f, assuming the theorem itself as the induction hypothesis for any network with fewer operations than f.

Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a feed forward network with ReLU activations, and let $x \in \mathbb{R}^m$ and let $N \subseteq f^{-1}(x)$ be compact and non-empty. Then f is one of the following cases:

Case: $f = g \circ h$ *(Sequential Computation)*

Subproof. We first know that $N \subseteq h^{-1} \circ g^{-1}(x)$ by the definition of f. We then infer that $h(N) \subseteq \mathcal{H}_\infty(N)$ is compact and non-empty by application of the continuous function h. Thus, by induction on $h(N)$ and g there is some $M' \sqsubset \mathcal{H}_\infty(h(N))$ such that $x \in g^\#(M')$. Thus, by Lemma 5.9, we know that there is some $M \sqsubset \mathcal{H}_\infty(N)$ such that $M' \subseteq h^\#(M)$. Thus, $x \in g^\# \circ h^\#(M) = f^\#(M)$. ◁

Case: $f(y) = (y, y)$. *(Relational Duplication)*

Subproof. We have $N \subseteq \{x_1\} \cap \{x_2\}$ so $N = \{x_1\}$. By singleton reflexivity, $N \sqsubset \mathcal{H}_\infty(N)$. Thus, $x \in f^\#(N)$. ◁

Case: $f(x_1, x_2) = (g_1(x_1), g_2(x_2))$. *(Non-Relational Parallel Computation)*

Subproof. First we know $N|_1 \subseteq g_1^{-1}(x_1)$ and $N|_2 \subseteq g_2^{-1}(x_2)$ by projection and that $N|_1$ and $N|_2$ are still compact and non-empty. Thus, by the induction hypothesis twice we see that there are boxes $M_1 \sqsubset \mathcal{H}_\infty(N|_1)$ and $M_2 \sqsubset \mathcal{H}_\infty(N|_2)$ such that $x_1 \in g_1^\#(M_1)$ and $x_2 \in g_2^\#(M_2)$. Then $M_1 \times M_2 \sqsubset \mathcal{H}_\infty(N)$ by Lemma 5.4. Then $x_1 \in g_1^\#(M_1 \times M_2)$ and $x_2 \in g_2^\#(M_1 \times M_2)$ by soundness. Thus, there is some box $M \sqsubset \mathcal{H}_\infty(N)$ such that $x \in f^\#(M)$. ◁

Case: $f(y) = c$. *(Constant)*

Subproof. We know $x = c$ and thus $f^{-1} = \mathbb{R}$. If we let $M = \{C(\mathcal{H}_\infty(N))\} \sqsubset \mathcal{H}_\infty(N)$, then $x \in f^\#(M)$. ◁

Case: $f(y) = c \cdot y$ for $c \neq 0$. *(Multiplication by a Constant)*

Subproof. We know $f^{-1}(x) = \{c^{-1} \cdot x\} = N \sqsubset \mathcal{H}_\infty(N)$ by N being non-empty and thus $x \in f^\#(N)$. ◁

Case: $f(y) = \text{ReLU}(y)$. *(Activation)*

Subproof. Then $x = \text{ReLU}(y)$ can either be zero or greater than zero.

Case: $x > 0$

Subproof. $N = \{x\}$ by def. of ReLU, and we know $\{x\} \sqsubset \mathcal{H}_\infty(N)$ and $x \in f^\#(\{x\})$. ◁

Case: $x = 0$

Subproof. $N = (-\infty, 0]$ by def. of ReLU. Thus $\{C(\mathcal{H}_\infty(N))\} \sqsubset \mathcal{H}_\infty(N)$ and $x \in f^{\#}(\{C(\mathcal{H}_\infty(N))\})$. ◁

Because x is the result of a ReLU, it must have been one of these two possibilities, and in both cases we could find some $M \sqsubset \mathcal{H}_\infty(N)$ such that $x \in f^{\#}(M)$. ◁

Case: $f(y_1, y_2) = y_1 + y_2$. *(Addition)*

Subproof. In this case, we know $N \subseteq f^{-1}(x) = \{(a, x - a) : a \in \mathbb{R}\}$. We pick $M = \{C(\mathcal{H}_\infty(N))\} \sqsubset \mathcal{H}_\infty(N)$. Given N is bounded, we know:

$$\begin{aligned} C(\mathcal{H}_\infty(N)) &= \left(\frac{\inf N|_1 + \sup N|_1}{2}, \frac{\inf N|_2 + \sup N|_2}{2} \right) \\ &= \left(\frac{\inf N|_1 + \sup N|_1}{2}, \frac{2x - \inf N|_1 - \sup N|_1}{2} \right). \end{aligned}$$

We can rewrite $f(C(\mathcal{H}_\infty(N)))$ as

$$f(C(\mathcal{H}_\infty(N))) = \frac{\inf N|_1 + \sup N|_1}{2} + \frac{2x - \inf N|_1 - \sup N|_1}{2} = x.$$

Thus, $x = f(C(\mathcal{H}_\infty(N))) \in f^{\#}(M)$ ◁

Thus, $\exists M \sqsubset \mathcal{H}_\infty(N).\, x \in f^{\#}(M)$. □