Are Safety Guarantees in Neural Networks Safe? How to Compute Trustworthy Robustness Certifications

Merkouris Papamichail[1,2], Konstantinos Varsos[1,2], Giorgos Flouris[1], and João Marques-Silva[3,4]

[1] Foundation for Research and Technology - Hellas, Heraklion, Greece
{mercoyris,varsosk,fgeo}@ics.forth.gr
[2] University of Crete, Heraklion, Greece
[3] Catalan Institution for Research and Advanced Studies, Barcelona, Spain
jpms@icrea.cat
[4] University of Lleida, Lleida, Spain

Abstract. A primary challenge in AI safety is the existence of adversarial examples —slightly distorted inputs that cause a neural network (NN) to misclassify. To mitigate this problem, recent research focuses on the computation of *robustness certifications*, which, for a given input, determine the largest distortion the input may receive without breaking the network's prediction. Robustness certifications can be interpreted as an axis-aligned hyper-rectangle (*multi-dimensional intervals*). Most existing approaches focus on maximizing the certification's *volume*, but recent intractability results prohibit the computation of volume-optimal certifications in reasonable time. We introduce the *apothem* measure — the minimum slack between the input and one of the interval's faces, and show how to compute *apothem-optimal* certifications in a *linear* number of calls to a NN verifier (*oracle*) w.r.t. the input domain's diameter. Moreover, we prove that we *cannot* have a volume-optimal, oracle-based algorithm, even if we *discard* the oracle costs. Also, we introduce *dual certifications* —an interval including all instances of a class— thus providing *apothem-minimum* upper bounds to a robustness certification. Further, we present the `ParallelepipedoNN`[1] system, which we evaluate on the standard MNIST and Fashion MNIST benchmarks. A preliminary comparison with existing work on the same datasets reveals at least two-fold improvement w.r.t. the minimum edge length.

Keywords: AI Safety · Adversarial Robustness · Interval Algebra.

1 Introduction

Neural Networks (NNs) have been successfully applied to a variety of machine learning tasks, achieving both high performance and empirical accuracy. However, it has been observed that slight distortions of the input can break the network's prediction. Such distorted inputs are called *adversarial examples* [30,7]

[1] https://github.com/merkouris148/parallelepipedonn

and motivated research on *robustness certification*. In robustness certification we are presented with an input and must compute the largest distortion the input may receive without causing a misclassification. By calculating tolerance intervals for each dimension and taking their Cartesian product, we are left with an *axis-aligned hyper-cube*, or *multi-dimensional interval*.

This geometrical interpretation allows us to express robustness certification as an *optimization problem* such that finding the best certification corresponds to optimizing w.r.t. a given measure. The most widely used approach in the area is to express robustness certification as a constrained convex optimization problem, and then solving it numerically using first order differential methods [32,20,18,12]. This approach also leverages powerful NN frameworks (e.g., PyTorch, TensorFlow) for efficiently computing the necessary gradients. Despite its efficiency, this approach comes at a cost: in order to be applied to real world NNs, the network needs to be *linearly relaxed*. Since the ReLU activation function is non-differentiable, traditional NNs can only be described using boolean variables. This leads to integer constraints. These constraints are relaxed, in order to obtain a convex program. In particular, linearly relaxing a NN results in false positives, flagging safe inputs as adversarial examples. This leads to artificially smaller robustness certifications, underestimating the NN's local stability.

This uncertainty hinders the applicability of robustness certifications in practice. For example, consider the common robustness certifications' quality measure, the *volume* [32,19,18]. Suppose that in order to consider the network robust, we need to achieve a volume threshold of v_1. However, we calculated a certification of volume v_2, with $v_2 < v_1$. *Does the network meet the desired safety guarantees? Should we retrain it, or is its robustness underestimated?* Since volume optimality cannot be guaranteed, these questions cannot be answered.

This work aims to mitigate this exact problem. Firstly, we introduce the *apothem* measure, which measures the minimum "slack" between the input and one of the faces of the hyper-rectangle. Next, we show how to compute apothem-optimal robustness certifications by iteratively querying a NN verifier (oracle). For arbitrary certifications, this can be done in a *linear* number of oracle calls w.r.t. the input domain's diameter. If we constrain ourselves to *uniform* certifications (allowing only uniform distortion), we achieve apothem optimality in *logarithmic* oracle calls w.r.t. the input domain's diameter. Further, we introduce the notion of *dual certifications*, which covers all the inputs of a specific class. Hence, a dual certification includes all possible robustness certifications of a class, constituting an upper bound to any increasing quality measure. Finally, we show that optimality cannot be achieved in polynomial time for the volume metric by an oracle-based algorithm, even if we discard the oracle costs.

Returning to our original example, our apothem-optimal algorithms can be used to compute a robustness certification. If the threshold is ϖ_1, but we computed ϖ_2, with $\varpi_2 < \varpi_1$, we *know* that the network is at fault and should be retrained. Moreover, if we need to use another quality measure, like volume, we can compute the dual certification. If the dual's volume is v_2 and we need to have v_1, with $v_1 > v_2$, we *know* that *no* such certification exists, for this NN.

We implemented our algorithms in the `ParallelepipedoNN` system, which we evaluate on the standard MNIST and Fashion MNIST datasets on multiple measures. Notably, a preliminary comparison with existing software reveals a two-fold improvement w.r.t. the certification's minimum edge length, and at least an order of magnitude improvement w.r.t. the certification's diameter.

2 Multi-Dimensional Intervals

We begin with some foundational results from Sunaga's *Interval Algebra* [29]. Firstly, we generalize the $\leq \subseteq \mathbb{R} \times \mathbb{R}$ relation to multi-dimensional spaces. For two vectors $\boldsymbol{\ell}, \boldsymbol{u} \in \mathbb{R}^d$ we write $\boldsymbol{\ell} \leq \boldsymbol{u}$ *iff* $\ell_i \leq u_i$ for all $i \in [d]$, and $\boldsymbol{\ell} < \boldsymbol{u}$ *iff* $\ell_i < u_i$ for all $i \in [d]$. Multi-dimensional intervals are defined as follows:

Definition 1 (Multi-Dimensional Intervals). *Let $\boldsymbol{\ell}, \boldsymbol{u} \in \mathbb{R}^d$, with $\boldsymbol{\ell} \leq \boldsymbol{u}$. An* interval $[\boldsymbol{\ell}, \boldsymbol{u}] \subset \mathbb{R}^d$ *is the set of points $\mathbf{x} \in \mathbb{R}^d$ such that $\boldsymbol{\ell} \leq \mathbf{x} \leq \boldsymbol{u}$. Finally, $\mathbb{I}(d)$ denotes the space of d–dimensional intervals, i.e., $\mathbb{I}(d) = \{S \subset \mathbb{R}^d \mid \exists\ \boldsymbol{\ell}, \boldsymbol{u} \in \mathbb{R}^d,\ \boldsymbol{\ell} \leq \boldsymbol{u},\ S = [\boldsymbol{\ell}, \boldsymbol{u}]\}$.*

The multi-dimensional intervals of Def. 1 are used to formalize the concept of robustness certification. Let $I_1, \ldots, I_d$ be the sequence of (one-dimensional) tolerance intervals of a robustness certification. Recall that we can perturb freely each feature i within I_i, without breaking the prediction. Concretely, this robustness certification can be represented by the multi-dimensional interval $I = I_1 \times \cdots \times I_d$.

Assume $\mathbb{F} \subset \mathbb{R}^d$ be the *input domain* of a particular neural network. For a given input $\mathbf{x} \in \mathbb{F}$, we can express the robustness certifications of $\mathbf{x}$ as intervals of the form $[\mathbf{x}-\mathbf{a}, \mathbf{x}+\mathbf{b}]$, where $\mathbf{a}, \mathbf{b} \geq \mathbf{0}$. Intuitively, the vectors $\mathbf{a}, \mathbf{b}$ correspond to the maximum subtractive and additive perturbations not breaking the network's prediction. Inversely, for any interval I containing $\mathbf{x}$, there are vectors $\mathbf{a}, \mathbf{b} \geq \mathbf{0}$, s.t. $I = [\mathbf{x}-\mathbf{a}, \mathbf{x}+\mathbf{b}]$. For the domain $\mathbb{F}$ and an input $\mathbf{x}$, $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ denotes the space of all the intervals of $\mathbb{F}$ containing $\mathbf{x}$.

For any $\mathbf{x} \in \mathbb{R}^d$, we call the interval $[\mathbf{x}, \mathbf{x}] = \{\mathbf{x}\}$ *trivial*. For a radius $\rho > 0$, we call intervals of the form $[\mathbf{x} - \rho\mathbf{1}, \mathbf{x} + \rho\mathbf{1}]$ *uniform*. Uniform intervals geometrically represent axis-aligned *hyper-cubes*. Moreover, for any vector $\mathbf{v} \geq \mathbf{0}$, we call intervals of the form $[\mathbf{x} - \mathbf{v}, \mathbf{x} + \mathbf{v}]$ *symmetric*. Finally, *arbitrary* intervals are characterized by two vectors $\mathbf{a}, \mathbf{b} \geq \mathbf{0}$, s.t. $[\mathbf{x} - \mathbf{a}, \mathbf{x} + \mathbf{b}]$. See Fig. 1 (left).

2.1 Operations on Intervals, and the Interval Lattice

Recall that we denote the space of intervals in $\mathbb{F}$ that contain an input $\mathbf{x}$ with $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$. The space $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ is highly structured by simple algebraic operations.

Definition 2. *Let $[\boldsymbol{\ell}, \boldsymbol{u}], [\boldsymbol{m}, \boldsymbol{n}] \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$. We define the operations:*

1. $[\boldsymbol{\ell}, \boldsymbol{u}] + [\boldsymbol{m}, \boldsymbol{n}] \stackrel{\Delta}{=} [\boldsymbol{\ell} + \boldsymbol{m}, \boldsymbol{u} + \boldsymbol{n}]$
2. $[\boldsymbol{\ell}, \boldsymbol{u}] \sqcup [\boldsymbol{m}, \boldsymbol{n}] \stackrel{\Delta}{=} [\min\{\boldsymbol{\ell}, \boldsymbol{m}\}, \max\{\boldsymbol{u}, \boldsymbol{n}\}]$
3. $[\boldsymbol{\ell}, \boldsymbol{u}] \sqcap [\boldsymbol{m}, \boldsymbol{n}] \stackrel{\Delta}{=} [\max\{\boldsymbol{\ell}, \boldsymbol{m}\}, \min\{\boldsymbol{u}, \boldsymbol{n}\}]$

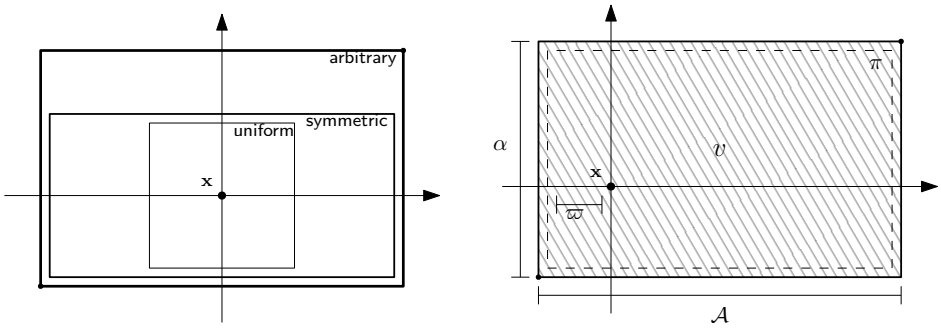


Fig. 1: Left: Interval Families: uniform, symmetric, and arbitrary. Right: Interval Measures: $\mathcal{A}$ diameter, π perimeter, v volume, α minimum edge, ϖ apothem.

It is easy to see that $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ is *closed* under the operations in Def. 2. The summation of two intervals, corresponds to the well-known *Minkowski sum*. We sometimes abuse notation, writing $\mathbf{x}+[\boldsymbol{\ell}, \boldsymbol{u}]$, instead of $[\mathbf{x}, \mathbf{x}]+[\boldsymbol{\ell}, \boldsymbol{u}]$ (analogously for the other operations). The $\sqcup, \sqcap$ operations were introduced by Sunaga [29], and reveal the underlying *lattice* structure. A partially ordered set is a lattice when for any two elements the *least upper bound (lub)* and *greatest lower bound (glb)* exist. In the $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ latttice structure, the lub is given by the *join* operation $\sqcup$, and the glb by the *meet* operation $\sqcap$.

Theorem 1 (Interval Lattice [29]). *The interval space $\mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, under $\subseteq$ constitutes a complete* lattice *with $\sqcup, \sqcap$ as the* meet *and* join *operations, respectively.*

2.2 Interval Objectives

We proceed by discussing interval measures. A *measure* $\mu\colon \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}} \to \mathbb{R}_{\geq 0}$ is a function assigning a positive real value to an interval. The trivial interval has to have zero measure, and a measure must be monotone w.r.t. set-inclusion. We also introduce the novel *apothem* measure.

Definition 3 (Interval Measures). *Consider an interval $I = \mathbf{x} + [\boldsymbol{\ell}, \boldsymbol{u}] \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, with $\boldsymbol{\ell} \leq \boldsymbol{u}$. Then we define the following measures:*

$$
\begin{aligned}
&\textit{Apothem:} && \varpi(I) = \min\Big\{ \min_{i\in[d]}(x_i - \ell_i), \min_{j\in[d]}(u_j - x_j) \Big\} \\
&\textit{Minimum Edge:} && \alpha(I) = \min_{i\in[d]} u_i - \ell_i \\
&\textit{Perimeter:} && \pi(I) = \textstyle\sum_{i\in[d]} u_i - \ell_i \\
&\textit{Volume:} && v(I) = \textstyle\prod_{i\in[d]} u_i - \ell_i. \\
&\textit{Diameter:} && \mathcal{A}(I) = \max_{i\in[d]} u_i - \ell_i = \|u_i - \ell_i\|_\infty
\end{aligned}
$$

All the measures of Def. 3 are depicted in Fig. 1 (right). In our experimental evaluation, we also use the *average edge length* $\overline{E}(I)$, defined as $\overline{E}(I) = \pi(I)/d$. Apothem[2] measures the minimum "slack" between the input $\mathbf{x}$ and one of the

[2] In Euclidean geometry *apothem* is a line between the polygon's center and a faces.

faces of the corresponding hyper-rectangle. In the sequel, we see that a robustness certification's faces will be bounded by adversarial examples. Hence, we can use the apothem to measure the minimum distance between the input $\mathbf{x}$ and the nearest adversarial example. Prop. 1 relates the above measures. For the omitted proofs we refer to our arxiv pre-print [25].

Proposition 1. *For an interval $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ and the measures of Def. 3 we have,*

$$\mathcal{A}(I) \geq \frac{1}{d} \cdot \pi(I) \geq \sqrt[d]{v(I)} \geq \alpha(I) \geq 2 \cdot \varpi(I) \tag{1}$$

3 Robustness Operators

Here we will work abstractly, in order to consider the combinatorial aspects of robustness certification and its relation to multi-dimensional intervals. In particular, consider the domain of a neural network $\mathbb{F}$ and a set of adversarial examples $\mathcal{V} \subset \mathbb{F}$. Moreover, let $\mathbf{x} \in \mathbb{F}$ be the given input. A robustness certification will be an interval $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in I$ *and* $I \cap \mathcal{V} = \varnothing$.

The technicalities on how we obtain the set of adversarial examples $\mathcal{V}$ will be discussed in Section 4. For now, let's assume that this set is complete, meaning that if we manage to exclude its members, there is no other input in the domain $\mathbb{F}$ breaking the prediction. This lets us focus on a subtle, but important problem. *From all the candidate intervals $J \subseteq \mathbb{F} \setminus \mathcal{V}$, which one should we choose?*

We proceed in two steps. Firstly, we consider the case of excluding a single adversarial example $\mathbf{v} \in \mathcal{V}$, defining the apothem-optimal, "small-step"[3], *constrain* operator. Then, we generalize for multiple adversarial examples, where we define the "big-step", *bottom* operator, also ensuring apothem-optimality. Finally, we discuss dual certifications, returned by the big-step, *top* operator.

3.1 Small-Step Robustness

Now we discuss our "small-step" robustness operator. For an input $\mathbf{x}$, consider an interval I s.t. $\mathbf{x} \in I$. Moreover, let $\mathbf{v} \in I$ be an adversarial example we want to exclude. Our small-step robustness operator will do just that, excluding the adversarial example $\mathbf{v}$ from the interval I, in an *apothem-optimal* manner. We call this operation *constrain*, and denote it by $I/\mathbf{v}$. Subsequently, let $I' = I/\mathbf{v}$.

Recall that the intervals are essentially sets of $2d$ inequalities, each corresponding to a lower or upper bound for one of the d coordinates of the input space, written compactly as $I = \mathbf{x} + [\boldsymbol{\ell}, \boldsymbol{u}]$, with $\boldsymbol{\ell} \leq \mathbf{0} \leq \boldsymbol{u}$. To ensure that $\mathbf{v} \notin I'$, it suffices to ensure that one of these inequalities fails. To achieve apothem-optimality, we choose the dimension $k \in [d]$ for which $|v_k - x_k|$ is maximum. If $v_k - x_k > 0$, then the upper bound must be reduced to exclude $\mathbf{v}$, by changing the k-th dimension of $\boldsymbol{u}$ to produce a $\boldsymbol{u}'$ that leaves $\mathbf{v}$ out of I'; otherwise the lower bound must be increased accordingly.

[3] "Small-step" operators deal with one example, while "big-step" with sets.

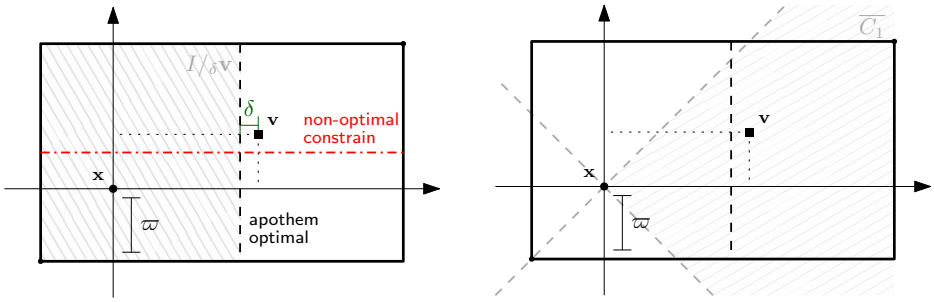


Fig. 2: Left: The constrain operator. Right: How the domain space can be partitioned in cones. The cone $\overline{C_i}$ contains all the adversarial examples that result in modifying the i-th coordinate of the upper endpoint $\boldsymbol{u}$.

An important technicality here is that we use a precision constant $\delta > 0$ to ensure a "meaningful" reduction of the interval. When modifying the upper bound, we set $u'_k = v_k - x_k - \delta$, and symmetrically for the lower bound. This serves two purposes. First, it excludes $\mathbf{v}$ in a "strict sense" (otherwise $\mathbf{v}$ would be in the border of I'). Second, it avoids infinitesimal modifications. As we will see later, this is necessary to guarantee the termination of our algorithms. This process is visualized in Fig. 2 (left) and defined formally below.

Definition 4 (Constrain Operator). *Consider an input $\mathbf{x} \in \mathbb{F}$, an interval $I = \mathbf{x} + [\boldsymbol{\ell}, \boldsymbol{u}]$, with $\boldsymbol{\ell} \leq \mathbf{0} \leq \boldsymbol{u}$, an adversarial example $\mathbf{v} \in I$, and a precision constant $\delta > 0$. Let $k = argmax_{i \in [d]}\{|v_i - x_i|\}$. We define the* constrain operation *$I/_\delta\ \mathbf{v} = \mathbf{x} + [\boldsymbol{\ell}', \boldsymbol{u}']$, where:*

- *If $v_k - x_k > 0$ then $\boldsymbol{\ell}' = \boldsymbol{\ell}$, $u'_i = u_i$ for all $i \in [d] \setminus \{k\}$ and $u'_k = \max\{0, v_k - x_k - \delta\}$.*
- *If $v_k - x_k < 0$ then $\ell'_i = \ell_i$ for all $i \in [d] \setminus \{k\}$, $\ell'_k = \min\{0, v_k - x_k + \delta\}$, and $\boldsymbol{u}' = \boldsymbol{u}$.*

Note that $v_k - x_k \neq 0$ because, by construction, $\mathbf{x} \neq \mathbf{v}$. The reason for max and min in the definition is to ensure that $\mathbf{x} \in I'$, i.e., $\boldsymbol{\ell}' \leq \mathbf{0} \leq \boldsymbol{u}'$. Subsequently, we omit δ, writing $I/\mathbf{v}$. The operator's apothem-optimality is established below.

Proposition 2. *Consider an interval $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ and a point $\mathbf{v} \in I$. For any interval $J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, with $\mathbf{v} \notin J$ and $J \subseteq I$, we have $\varpi(J) \leq \varpi(I/\mathbf{v})$.*

Proof. W.l.o.g. let $\mathbf{x} = \mathbf{0}$. For any $J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ and $\mathbf{v} \in \mathbb{F}$, it holds $\mathbf{v} \notin J$, *iff* $\max_{i \in [d]} |v_i| > \varpi(J)$. Also, $\varpi(I/\mathbf{v}) = \min\{\varpi(I), \max_{i \in [d]} |v_i|\}$. ■

3.2 Big-Step Robustness

Now we revert to the original problem of handling a set of adversarial examples. Observe that the small-step constrain operator implicitly defines a partition of $\mathbb{F}$ into $2d$ regions[4], denoted by $\overline{C_i}, \underline{C_i}$, for $i \in [d]$, see Fig. 2 (right).

[4] Corresponding to polyhedral cones, $C = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} \leq \mathbf{0}\}$, $A \in \mathbb{R}^{m \times n}$.

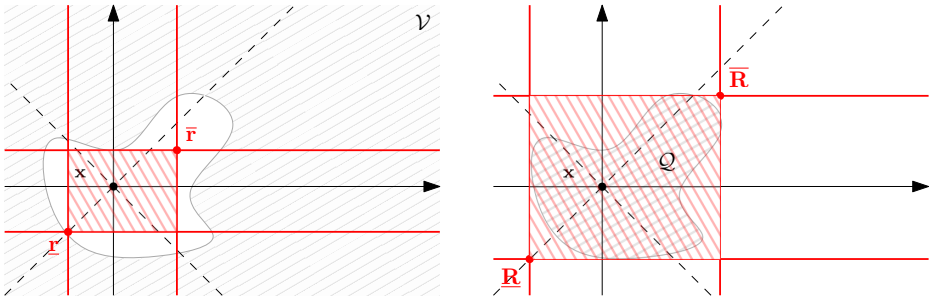


Fig. 3: Left: The bottom interval $\mathcal{V}\perp\mathbf{x}$. Right: The dual top interval $\mathcal{Q}\top\mathbf{x}$.

Definition 5 (Cone Partition). *Consider the domain $\mathbb{F} \subset \mathbb{R}^d$, and an input $\mathbf{x} \in \mathbb{F}$. The cone partition w.r.t. $\mathbf{x}$ is defined by the sequence $\{\underline{C_i}, \overline{C_i}\}_{i\in[d]}$, where*

$$\begin{aligned} \underline{C_i} &= \left\{\mathbf{v} \in \mathbb{F} \;\middle|\; v_i \leq x_i, \forall j \in [d]\ |v_i - x_i| \geq |v_j - x_j|\right\}, \\ \overline{C_i} &= \left\{\mathbf{v} \in \mathbb{F} \;\middle|\; v_i \geq x_i, \forall j \in [d]\ |v_i - x_i| \geq |v_j - x_j|\right\}. \end{aligned} \tag{2}$$

For a given set of adversarial examples $\mathcal{V}$, let $\mathcal{V}\perp\mathbf{x}$ denote the *bottom interval*, where $(\mathcal{V}\perp\mathbf{x})\cap\mathcal{V} = \varnothing$, see Figure 3 (Left). Since $\mathbf{x} \in (\mathcal{V}\perp\mathbf{x})$, we write the bottom interval as $\mathcal{V}\perp\mathbf{x} = \mathbf{x} + [-\underline{\mathbf{r}}, \overline{\mathbf{r}}]$, with $\underline{\mathbf{r}}, \overline{\mathbf{r}} \geq \mathbf{0}$. The k-th coordinate of the upper endpoint $\overline{\mathbf{r}}$ is set to the *minimum* coordinate-wise absolute distance $|x_k - v_k|$. We work similarly for the k-th coordinate of the lower endpoint $\underline{\mathbf{r}}$.

Definition 6 ($\perp$-Operator). *Consider a domain $\mathbb{F}$ and an input $\mathbf{x} \in \mathbb{F}$. Let $\mathcal{V} \subset \mathbb{F}$ be a set of adversarial examples. Moreover, assume the values $\underline{r_i}, \overline{r_i} > 0$, where $\underline{r_i} = \inf\{x_i - v_i \mid \mathbf{v} \in \underline{C_i} \cap \mathcal{V}\}$ and $\overline{r_i} = \inf\{v_i - x_i \mid \mathbf{v} \in \overline{C_i} \cap \mathcal{V}\}$. Then, $\mathcal{V}\perp\mathbf{x}$ denotes the bottom interval, where $\mathcal{V}\perp\mathbf{x} = \mathbf{x} + [-\underline{\mathbf{r}}, \overline{\mathbf{r}}]$.*

Observe that, by definition, the bottom interval can be *unbounded*. Indeed, if one of the cones is empty, e.g., $\overline{C_i} = \varnothing$, then the corresponding coordinate-wise absolute distance approaches infinity, i.e., $\overline{r_i} \to +\infty$. This happens trivially, when inputs lay at the border of the domain $\mathbb{F}$, thus $\overline{C_i} \cap \mathbb{F} = \varnothing$. Hence, no adversarial examples exist. Note that unbounded intervals can still have bounded apothem, unless equal to $\mathbb{R}^d$. We discuss this issue again in Sec. 5. Subsequently, we show that the bottom operator is apothem optimal.

Theorem 2 ($\perp$-Operator Optimality). *Consider a domain $\mathbb{F}$ and an input $\mathbf{x} \in \mathbb{F}$. Let $\mathcal{V} \subset \mathbb{F}$ be a set of adversarial examples. The following hold:*

1. $(\mathcal{V}\perp\mathbf{x}) \cap \mathcal{V} = \varnothing$.
2. *For every $J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in J$ and $J \cap \mathcal{V} = \varnothing$, we have $\varpi(J) \leq \varpi(\mathcal{V}\perp\mathbf{x})$.*

Proof. The first desideratum is trivial from Def. 6. We prove the second. W.l.o.g. let the apothem be achieved by the coordinate $\overline{r_k}$. From the Def. 6 there is a adversarial example $\mathbf{v} \in \mathcal{V}$, s.t. $\overline{r_k} = v_k$. Assume the interval $J = \mathbf{x} + [\boldsymbol{\ell}, \boldsymbol{u}]$,

$\boldsymbol{\ell} \leq \mathbf{0} \leq \boldsymbol{u}$, with $J \cap \mathcal{V} = \varnothing$. For sake of contradiction, $\varpi(J) > \varpi(\mathcal{V}\perp\mathbf{x})$. It holds, $u_k \geq \varpi(J) > r_k$. Thus, $u_k > v_k$. Since $J \cap \mathcal{V} = \varnothing$, there is a coordinate j, s.t. $u_j \leq v_j$. However, $\varpi(J) \leq u_j \leq v_j \leq v_k \leq r_k = \varpi(\mathcal{V}\perp\mathbf{x}) < \varpi(J)$, where we have $v_j \leq v_k$, since $\mathbf{v} \in \overline{C}_k$. A contradiction. ■

As we discussed before, we can *translate* the bottom operator to consecutive constrain operations. Below we establish that the resulting interval will always include the bottom interval, regardless of the order with which we perform the constrains. This enables us to apply the bottom operator in practice. Further, since in real-world cases adversarial examples will arrive in an on-line manner, we can exclude them upon arrival using the constrain operator.

Proposition 3. *Consider a domain $\mathbb{F}$, and an input $\mathbf{x} \in \mathbb{F}$. Let $\mathcal{V} \subset \mathbb{F}$ be a set of adversarial examples. We assume an ordering in $\mathcal{V}$, i.e., $\mathcal{V} = \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$. Let $I = \mathbb{F}\ /\mathbf{v}_1\ /\mathbf{v}_2\ \cdots\ /\mathbf{v}_n$[5]. Then, $\mathbf{x} \in I$, $I \cap \mathcal{V} = \varnothing$, and $I \supseteq \mathcal{V}\perp\mathbf{x}$.*

3.3 Dual Certifications

Here we discuss the dual problem, which enables us to efficiently compute upper bounds to the robustness certification. We consider the inverse situation: for the domain $\mathbb{F}$ and input $\mathbf{x}$, we take a set of positive inputs $\mathcal{Q} \subset \mathbb{F}$, and compute an interval $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in I$ and $I \cup \mathcal{Q} = I$.

The interval I is obtained by the top operator $\mathcal{Q}\top\mathbf{x}$, see Fig. 3 (Right). We work symmetrically to the bottom operator. In particular, the dual certification can be written in the form $\mathcal{Q}\top\mathbf{x} = \mathbf{x} + [-\underline{\mathbf{R}}, \overline{\mathbf{R}}]$, with $\underline{\mathbf{R}}, \overline{\mathbf{R}} \geq \mathbf{0}$. Now the i-th coordinate $\overline{R_i}$ is set to the *maximum* coordinate-wise absolute distance between $\mathbf{x}$ and the positive points of $\mathcal{Q}$. For $\underline{\mathbf{R}}$ we work similarly.

Definition 7 ($\top$-Operator). *Consider a domain $\mathbb{F}$ and an input $\mathbf{x} \in \mathbb{F}$. Let $\mathcal{Q} \subset \mathbb{F}$ be a set of positive points. Moreover, assume the values $\underline{R_i}, \overline{R_i} > 0$, where $\underline{R_i} = \sup\{x_i - q_i \mid \mathbf{q} \in \mathcal{Q}\}$ and $\overline{R_i} = \sup\{q_i - x_i \mid \mathbf{q} \in \cap\mathcal{Q}\}$. Then, $\mathcal{Q}\top\mathbf{x}$ denotes the top interval, where $\mathcal{Q}\top\mathbf{x} = \mathbf{x} + [-\underline{\mathbf{R}}, \overline{\mathbf{R}}]$.*

Note that since $\mathcal{Q} \subset \mathbb{F}$, the top interval is *always bounded*. We prove the optimality of the top operator in Thm. 3. However, we follow a different approach than the Thm. 2. First, we prove the correctness of the top operator. Then, we show that the bottom operator returns a *minimal* interval that includes all the points in $\mathcal{Q}$. Finally, we show that any minimal interval including $\mathcal{Q}$ is *unique*. This results in apothem optimality.

Theorem 3. ($\top$-Operator Optimality) *Consider a domain $\mathbb{F}$ and an input $\mathbf{x} \in \mathbb{F}$. Let $\mathcal{Q} \subset \mathbb{F}$ be a set of positive points. The following hold:*

1. $(\mathcal{Q}\top\mathbf{x}) \cup \mathcal{Q} = (\mathcal{Q}\top\mathbf{x})$.
2. *There is no interval $J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in J$, $J \cup \mathcal{Q} = J$, and $J \subset (\mathcal{Q}\top\mathbf{x})$.*

[5] We assume left associativity of the / operator.

3. *Let* $I, J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ *two intervals, s.t.* $(\mathcal{Q} \cup \{\mathbf{x}\}) \subseteq I, J$. *Moreover, let* I, J *be* minimal, *w.r.t. set inclusion, achieving this property. Then,* $I = J$.
4. *For every* $J \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, *s.t.* $\mathbf{x} \in J$, *and* $J \cup \mathcal{Q} = J$, *then* $\varpi(J) \geq \varpi(\mathcal{Q}\top\mathbf{x})$.

Proof (Sketch). *1.* Immediate consequence of Def. 7. *2.* We prove that if $J \subset (\mathcal{Q}\top\mathbf{x})$, then there is some $\mathbf{q} \in Q$, s.t. $\mathbf{q} \notin J$. *3.* Let I, J be minimal intervals. Then we prove that the intersection $I \cap J$ also includes $\mathcal{Q} \cup \{\mathbf{x}\}$. Thus, reaching a contradiction. *4.* Follows from the previous results. ■

Again, the top interval can be translated in a sequence of small-step interval operations. In particular, we use the join operator $\sqcup$ of Def. 2 to include the points in $\mathcal{Q}$. We abuse the notation writing $I \sqcup \mathbf{q}$, instead of $I \sqcup [\mathbf{q}, \mathbf{q}]$.

Proposition 4. *Consider a domain* $\mathbb{F}$ *and an input* $\mathbf{x} \in \mathbb{F}$. *Let* $\mathcal{Q} \subset \mathbb{F}$ *be a set of positive points. We assume an ordering in* $\mathcal{Q}$, *i.e.,* $\mathcal{Q} = \mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$. *Let* $I = [\mathbf{x}, \mathbf{x}] \sqcup \mathbf{q}_1 \sqcup \mathbf{q}_2 \cdots \sqcup \mathbf{q}_n$[6]. *Then,* $I = \mathcal{Q}\top\mathbf{x}$.

We close our discussion on the dual certification by establishing the (weak) duality of the bottom and top operators in the following theorem.

Theorem 4. (Weak $\bot, \top$-Duality) *Consider an input domain* $\mathbb{F}$, *and a partition into positive points* $\mathcal{Q}$ *and adversarial examples* $\mathcal{V}$, *i.e.,* $\mathcal{Q} \cap \mathcal{V} = \varnothing$ *and* $\mathcal{Q} \cup \mathcal{V} = \mathbb{F}$. *Then, for an input* $\mathbf{x} \in \mathbb{F}$, *we have* $(\mathcal{V}\bot\mathbf{x}) \cap \mathbb{F} \subseteq \mathcal{Q}\top\mathbf{x}$ *and for every increasing measure* $\mu \colon \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}} \to \mathbb{R}_{\geq 0}$ *of Def. 2, we have* $\mu[(\mathcal{V}\bot\mathbf{x}) \cap \mathbb{F}] \leq \mu[\mathcal{Q}\top\mathbf{x}]$.

Note that since the bottom interval may be unbounded, we take the intersection $(\mathcal{V}\bot\mathbf{x}) \cap \mathbb{F}$. In the sequel, we discuss how the above observations can be applied to adversarial robustness of NN classifiers. There, all the points of a particular class c are positive. The rest of the points are adversarial examples.

4 Algorithms

In this section we discuss how the notions of Sec. 3 can be applied to compute the adversarial robustness of neural network classifiers. This is achieved by encoding the bottom and top operators as small-step interval operations, applying Prop. 3 and 4. However, for NNs we cannot have explicit descriptions of the adversarial examples $\mathcal{V}$ or positive points $\mathcal{Q}$. Instead, we only have the description of the NN, its weights, biases and activation functions. Using this, we construct first order formulas $\mathscr{Q}, \mathscr{S}$ that are satisfied only for points in $\mathcal{V}, \mathcal{Q}$, respectively.

ReLU NNs can be expressed as a set of inequalities with real and boolean variables. This formalism is called in the literature *mixed integer linear programming (MILP)* [21,5]. Using existing MILP solvers (e.g. GLPK[7]), MILPs can be solved efficiently and accurately. Moreover, specific tools have been developed and optimized for NN MILPs. One such tool is the Marabou [15,33] software for NN verification, which we also use in our `ParallelepipedoNN` software.

[6] We assume left associativity of the $\sqcup$ operator.
[7] https://www.gnu.org/software/glpk/

Therefore, despite not having an explicit description of the $\mathcal{V}, \mathcal{Q}$ sets, we can construct *oracles* (first-order formulas, solvable with a MILP solver) $\mathscr{V}, \mathscr{Q}$, which either *assert* that an interval I is included in a set $\mathcal{P} \in \{\mathcal{V}, \mathcal{Q}\}$, or return a *counterexample* $\mathbf{p} \in I \setminus \mathcal{P}$. Then, we use the counterexample $\mathbf{p}$ to refine the current interval using an operator $\square \in \{/, \sqcup\}$.

4.1 From Abstract Sets to Neural Networks

Assume the input dimension d_{in} and the output dimension d_{out}. Let $\mathbb{F} \subset \mathbb{R}^{d_{\text{in}}}$ denote the input (feature) space and $\mathbb{S} \subset \mathbb{R}^{d_{\text{out}}}$ denote the output (score) space. A NN is a function of the form $\sigma\colon \mathbb{F} \to \mathbb{S}$. In this work we are interested in ReLU activated NNs. For the one-dimensional case, the ReLU function $r\colon \mathbb{R} \to \mathbb{R}_{\geq 0}$ is defined as $r(x) = \max(x, 0)$. For the d-dimensional case we have $\mathbf{r}\colon \mathbb{R}^d \to \mathbb{R}^d_{\geq \mathbf{0}}$, where $\mathbf{r}(\mathbf{x}) = (r(x_1), \ldots, r(x_d))$. We follow the definition in [4].

Definition 8 (Neural Network Classifiers). *A* neural network $\sigma\colon \mathbb{F} \to \mathbb{S}$, *with* $\mathbb{F} \subset \mathbb{R}^{d_{in}}, \mathbb{S} \subset \mathbb{R}^{d_{out}}$ *is described as the tuple* $\sigma = \langle L, D, W, B \rangle$, *where:*

- $L \in \mathbb{N}$ *denotes the number of* layers.
- D *denotes the* architecture, *i.e. a sequence* $d_{in} = d_0, d_1, \ldots, d_{L-1}, d_L = d_{out}$.
- W *denotes the sequence of* weight matrices, *s.t.* $W^{(i)} \in \mathbb{R}^{d_i \times d_{i-1}}$, *for* $i \in [L]$
- B *denotes a sequence of* biases *s.t.* $\mathbf{b}^{(i)} \in \mathbb{R}^{d_i}$ *for* $i \in [L]$.

The value of $\sigma(\mathbf{x})$ *is given as the value* $\sigma^{(L)}$ *in the system of equations below.*

$$\left.\begin{aligned} \sigma^{(0)} &= \mathbf{x} \\ \sigma^{(i)} &= \mathbf{r}[W^{(i)}\sigma^{(i-1)} + \mathbf{b}^{(i)}], \quad \forall i \in [L] \end{aligned}\right\} \tag{3}$$

For classification problems, we also assume a set of classes $\mathcal{C}$, *s.t.* $|\mathcal{C}| = d_{out}$. *A classifier is a function of the form* $\kappa\colon \mathbb{F} \to \mathcal{C}$, *where* $\kappa(\mathbf{x}) = \arg\max_{i \in d_{out}} \sigma(\mathbf{x})$.

As discussed we can formulate a NN as a MILP to formally describe the input/output relation of the neural network as a logic formula. Let ν be that formula. For every input $\mathbf{x} \in \mathbb{F}$ and every output $\mathbf{y} \in \mathbb{S}$ the pair *satisfies* ν if and only if the network $\sigma(\cdot)$ with input $\mathbf{x}$ outputs $\mathbf{y}$. Now assume a fixed input $\mathbf{x} \in \mathbb{F}$ and an interval $I = [\boldsymbol{\ell}, \boldsymbol{u}] \in \mathbb{I}(d_{\text{in}})|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in I$. We have the following.

$$\mathscr{V}_{I,\nu}(\mathbf{p}) \equiv \left(\bigwedge_{i \in [d_{\text{in}}]} \ell_i < p_i \wedge p_i < u_i\right) \wedge \nu(\mathbf{p}, \mathbf{y}) \wedge \left(\bigvee_{j \neq c} y_j - y_c > \epsilon\right) \tag{4}$$

$$\mathscr{Q}_{I,\nu}(\mathbf{p}) \equiv \left(\bigvee_{i \in [d_{\text{in}}]} p_i < \ell_i \vee u_i < p_i\right) \wedge \nu(\mathbf{p}, \mathbf{y}) \wedge \left(\bigwedge_{j \neq c} y_c - y_j > \epsilon\right) \tag{5}$$

Above, $\epsilon > 0$ is a precision constant. The predicate $\mathscr{V}_{I,\nu}$ is satisfied only by adversarial examples, i.e., for inputs that belong to the current interval I and maximize a different score than the target class c. Symmetrically, the predicate $\mathscr{Q}_{I,\nu}$ is satisfied by a c-instance not included in the current interval I. When there is no solution in I for $\mathscr{V}_{I,\nu}$, we reached a robust certification. Symmetrically, when there is no solution outside of I for $\mathscr{Q}_{I,\nu}$, we reached a dual certification.

Algorithm 1: RefineCheck$(I_o, \mathscr{P}_{I,\nu}, \square)$

Input: We assume the following arguments:
- $I_o \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, an initial interval.
- $\mathscr{P}_{I,\nu}$, a property to be falsified.
- $\square\colon \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}} \times \mathbb{F} \to \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, a refinement operator.

Output: I, an interval s.t. $\models \neg\mathscr{P}_{I,\nu}$

1 $I \leftarrow I_o$
2 **while** $\exists \mathbf{p}$, *s.t.* $\mathbf{p} \models \mathscr{P}_{I,\nu}$ **do**
3 $\quad I \leftarrow I \square \mathbf{p}$
4 **return** I

4.2 Refine & Check Algorithm

We now have all the components to develop our *refine & check* algorithm, whose pseudocode is given in Alg. 1. Starting from an initial interval I_o, the algorithm repeatedly checks whether the current interval I *falsifies* some given property $\mathscr{P}_{I,\nu}$. The check is delegated to a NN verifier (line 2), which receives as input the MILP encoding of the network, the interval I, and the property $\mathscr{P}_{I,\nu}$. If the property is not falsified, the verifier returns a *counterexample* $\mathbf{p}$ that satisfies the property. The interval is then *refined* using $\mathbf{p}$ via some given operator $\square$, producing the interval $I \square \mathbf{p}$ (line 3). The procedure continues until the verifier asserts that $\mathscr{P}_{I,\nu}$ is falsified, in which case the current interval is returned.

Different instantiations of Alg. 1 can be obtained by appropriately choosing its parameters. Eq. (6) shows how to compute the *apothem-optimal robustness certification*. In this setting, the initial interval is the whole domain $\mathbb{F}$, the property to be falsified is given by eq. (4), and the refinement operator is the constrain operator of Def. 4. The correctness of eq. (6) is given by Prop. 3, which shows that the bottom operator can be realized as a sequence of constrain operations. We call Alg. 1 instantiated in this manner as Top-Down Search (TDS). Finally, since we begin from the domain $\mathbb{F}$, TDS computes the intersection $(\mathscr{V} \bot\mathbf{x}) \cap \mathbb{F}$.

$$(\mathscr{V} \bot\mathbf{x}) \cap \mathbb{F} = \textsc{RefineCheck}(\mathbb{F}, \mathscr{V}_{I,\nu}, /) \quad \text{Top-Down Search (TDS)} \quad (6)$$

$$\mathscr{Q} \top\mathbf{x} = \textsc{RefineCheck}([\mathbf{x},\mathbf{x}], \mathscr{Q}_{I,\nu}, \sqcup) \quad \text{Bottom-Up Search (BUS)} \quad (7)$$

Eq. (7) presents the implementation for *dual certifications*. The construction proceeds symmetrically. The interval is initialized to the trivial $[\mathbf{x},\mathbf{x}]$, the property to be falsified is given by eq. (5), and the refinement operator is the join operator $\sqcup$ of Def. 2. The correctness of eq. (7) follows from Prop. 4, which shows that the top operator can be expressed as a sequence of join operations. We refer to the parametrization in eq. (7) as Bottom-Up Search (BUS).

Observe that the abstract sets $\mathcal{V}$ and $\mathcal{Q}$ introduced in Sec. 3 have now been translated into the verification properties $\mathscr{V}$ and $\mathscr{Q}$. This was possible only because we managed to translate the "global", big-step, bottom and top operations, to "local", small-step, operations. Based on the small-step operators, we are able

to construct the *on-line* refine & check method of Alg. 1. In this setting the set $\mathcal{P}$, that is $\mathcal{P} \in \{\mathcal{V}, \mathcal{Q}\}$, is constructed incrementally, and contains all the counterexamples $\mathbf{p}$ returned by the NN verifier during the execution of Alg. 1.

4.3 The Complexity of Apothem Optimality

Our *refine & check* algorithm allows us to calculate the computational complexity of both operators in a single proof. To that end, let $\mathcal{P}$ be the set of all counterexamples returned by the NN verifier, and $n = |\mathcal{P}|$. Moreover, let $t_{\mathscr{P}_{I,\nu}}$ be the time consumed by the NN verifier (in the worst case) to test the satisfiability of the property $\mathscr{P}_{I,\nu}$. Finally, let $t_{\square}$ be the execution time of the refinement operation $\square$. Then, in each iteration, Alg. 1 processes one counterexample, makes a query to the NN verifier, and applies a refinement operator.

Theorem 5. *Let n be the number of counterexamples returned by the NN verifier. Let $t_{\mathscr{P}_{I,\nu}}$ be the worst-case time to check whether property $\mathscr{P}_{I,\nu}$ holds. Finally, let $t_{\square}$ be the worst-case time required by the operator $\square$. Then, the refine & check method of Alg. 1 terminates after $O[n \cdot (t_{\square} + t_{\mathscr{P}_{I,\nu}})]$ steps. If $\square \in \{/, \sqcup\}$, then $t_{\square} = O(d)$, and Alg. 1 concludes in $O[nd + nt_{\mathscr{P}_{I,\nu}}]$ time.*

Although the number n of counterexamples is unknown, we can bound it from above using the precision constant δ. The constrain operator only modifies one inequality of the given interval at each refinement step, and each inequality can be modified *at most* $\mathcal{A}(\mathbb{F})/\delta$ times, where $\mathcal{A}(\mathbb{F})$ is the domain's diameter. Since each interval is defined by $2d$ inequalities, the total number of counterexamples is bounded by $2d \cdot \mathcal{A}(\mathbb{F})/\delta$. The join operator $\sqcup$ is more intervening, modifying all $2d$ inequalities at once, resulting in $\mathcal{A}(\mathbb{F})/\delta$ counterexamples.

Corollary 1. *Let $t_{\mathscr{P}_{I,\nu}}$ be the worst case time for verifying property $\mathscr{P}_{I,\nu}$. TDS terminates after $O[(\mathcal{A}(\mathbb{F})/\delta)(d^2 + d \cdot t_{\mathscr{P}_{I,\nu}})]$ time. BUS terminates after $O[(\mathcal{A}(\mathbb{F})/\delta)(d + t_{\mathscr{P}_{I,\nu}})]$ time.*

4.4 The Intractability of Volume Optimality

From Thm. 5 we can compute an *apothem optimal* robustness certification in $\mathsf{poly}(n, d, t_{\mathscr{P}_{I,\nu}})$ time. Namely, even if NN verification can be computed in constant time, i.e., $t_{\mathscr{P}_{I,\nu}} = O(1)$, our algorithm becomes polynomial. This does *not* hold for other measures, like the volume, where we *cannot* have polynomial time volume-optimality, even if NN verification costs are ignored.

To see that, recall the abstract robustness certification problem, introduced in Sec. 3. There, we had a set $\mathcal{V} \subset \mathbb{F}$ of adversarial examples and an input point $\mathbf{x} \in \mathbb{F}$. Our goal was to compute an interval $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, s.t. $\mathbf{x} \in I$ and $\mathcal{V} \cap I = \varnothing$. For a constant $\gamma > 0$, we want to decide if there is $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$ with the above properties and $v(I) \geq \gamma$. This problem is closely related to *Maximum Empty Rectangle (MER)* [24,3,1,2] and *query-Maximum Empty Square (q-MES)* [8] problems. Next, we establish an intractability result based on the proof of MER's intractability by Backer and Keil in [1].

Theorem 6. *Consider a set of adversarial examples $\mathcal{V} \subset \mathbb{F}$, an input $\mathbf{x} \in \mathbb{F}$, and a constant $\gamma > 0$. Moreover, let $n = |\mathcal{V}|$, let d be the dimension s.t. $\mathbb{F} \subset \mathbb{R}^d$, and let $t_{\mathscr{P}_{I,\nu}}$ be the worst-case time check whether property $\mathscr{P}_{I,\nu}$ holds. Then, the existence of a robustness certification $I \in \mathbb{I}(d)|^{\mathbb{F}}_{\mathbf{x}}$, with $\mathbf{x} \in I$, $I \cap \mathcal{V} = \varnothing$ and $v(I) \geq \gamma$* cannot *be decided in $\mathsf{poly}(n, d, t_{\mathscr{P}_{I,\nu}})$ time, unless $\boldsymbol{P} = \boldsymbol{NP}$.*

Proof (Sketch). We use the argument of [1] reducing from the independent set problem to maximum volume adversarial robustness. In [1] each vertex of a given graph maps to a dimension, each edge maps to a point. The graph has an independent set of size k iff there is a sufficiently large empty interval.

Thus, given a graph G, we use the methodology in [1] to construct a set of adversarial examples $\mathcal{V}$. However, we also need to define an input $\mathbf{x}$ that must be included to the interval. We set $\mathbf{x} = \mathbf{0}$. From Lemma 2 in [1] the origin will always be included in a maximum volume interval. Thus, our choice of $\mathbf{x}$ does not effect the validity of the argument. This concludes the reduction. ■

4.5 Notes on Uniform Certifications

We close this section with some notes on uniform robustness certifications (originally used in [32]), i.e., certifications of the form $I = [\mathbf{x} - \rho\mathbf{1}, \mathbf{x} + \rho\mathbf{1}]$ for $\rho > 0$ and some input $\mathbf{x} \in \mathbb{F}$. In our setting, we can use binary search to determine the maximum radius excluding the adversarial examples in $\mathcal{V}$, or the minimum radius including the positive examples $\mathcal{Q}$. The top and bottom operators of Sec. 3, *when restricted to uniform certifications*, are denoted by $\top_{\mathbb{B}}$ and $\bot_{\mathbb{B}}$ respectively, while the corresponding algorithms $\mathbb{B}$-BUS and $\mathbb{B}$-TDS. These operators can be computed in logarithmic $\log(\mathcal{A}(\mathbb{F})/\delta)$ oracle calls. We use uniform certifications as a *baseline* in our following experimentation.

5 Implementation

In this section, we review the `ParallelepipedoNN` system, where we implement the algorithms and operations discussed earlier. In particular, we present two case studies based on the MNIST [17] and the Fashion MNIST [35] datasets. Specifically, we train two ReLU NNs and compute robustness and dual certifications for each dataset-network pair. For robustness certifications, we use the TDS algorithm of eq. (6) and its *uniform variant* $\mathbb{B}$-TDS (see Subsec. 4.5). For dual certifications, we use the BUS algorithm of eq. (7) and its *uniform variant* $\mathbb{B}$-BUS. We comment on the CPU time, oracle queries and edge lengths.

Experimental Setup. We ran our experiments in parallel on an Ubuntu 18.04 machine, with Intel Xeon E5-2640 v4 CPU at 2.394GHz with 38 cores, with 128GB RAM. We utilized 35 cores. `ParallelepipedoNN` is written in Python v3.8.16. We used the Marabou v2.0 [15,33] NN verifier. Our implementation takes as input a NN in open neural network exchange (ONNX) v1.16.0[8] format. For linear

[8] See https://github.com/onnx/onnx.

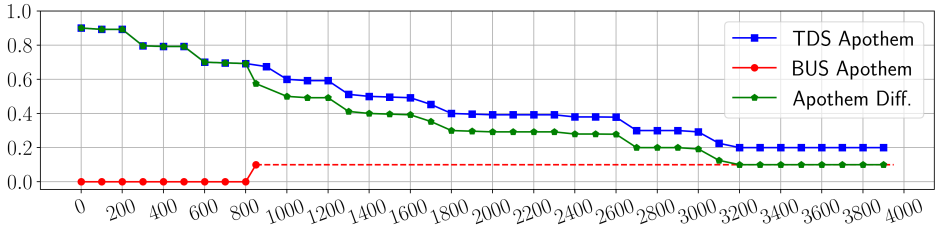


Fig. 4: Apothems in TDS (blue), BUS (red), and their difference (green), w.r.t. iterations, on a "7" MNIST image. BUS stops after 860 iterations.

algebra computations, we used the NumPy v1.23.5 library. For visualization, we used the Matplotlib v3.7.2 library. The NNs used were trained from scratch, using TensorFlow v2.12.0. We evaluated all the algorithms using the same parameters. We set the precision constant δ to $\delta = 0.1$, and a *timeout* to 1 hour. We use 2 datasets, namely MNIST [17] and Fashion MNIST [35]. Both datasets consist of 28×28 grayscale images, belonging to 10 classes. The NN architecture is $D = \langle 784, 32, 10, 10 \rangle$, w.r.t. Def. 8. The last layer is fully connected, without activation function. This corresponds to 25,450 trainable parameters. For training, we used the Adam [16] algorithm, Glorot [6] weight initialization, and the Categorical Crossentropy loss. The achieved test-set accuracy is 94% and 82% for the MNIST and Fashion MNIST, respectively. For each NN, we randomly chose 5 images of the 10 classes of the test set (a total of 50 images per NN).

Measures in Practice. Note that one totally black or totally white pixel suffices to place a grayscale image to the border of the input space, making $\mathscr{V} \perp \mathbf{x}$, and most measures, trivially unbounded. Therefore, for all measures, *except the apothem*, we use the intersection $(\mathscr{V} \perp \mathbf{x}) \cap \mathbb{F}$. The apothem is computed in the (possibly unbounded) bottom interval $\mathscr{V} \perp \mathbf{x}$. This is achieved by excluding the interval's faces that stayed still during TDS' execution. Hence, *only adversarially bounded faces are considered*, and the apothem returns the distance between the input and the *nearest* adversarial example.

Running One Instance. In Fig. 4 we plot the *apothem* of the TDS and BUS algorithms as a function of the *iterations*, on a "7" MNIST image. TDS computes an apothem-maximum robustness certification, beginning from the whole domain, and gradually shrinking. In plateaus, the non-minimum edges are shrinking, thus the apothem stays constant. BUS computes an apothem-minimum dual certification. Note that BUS terminates in much fewer iterations than TDS, because it updates all edges in each iteration. Finally, we depict the difference in apothems between the i-th robustness and dual certifications. In this case, the two apothems happen to converge at 0.1 distance.

Experimental Results. Our experimental results are presented in Tbl 1. There, we observe a tradeoff between quality and performance. The uniform algorithms

Data	Alg.	Time (min.)	Verif. Perc.	# Verif. Calls	Apoth. ϖ	Min. Edge α	Avg. Edge $\overline{E}$	Diam. $\mathcal{A}$	Perim. π
MNIST	TDS	43.08	95%	3602.73	**.014**	**.14**	**.47**	**1.00**	**368.16**
	$\mathbb{B}$–TDS	**.34**	99%	**4.00**	.000	.07	.07	.13	57.87
	BUS	16.15	96%	895.73	**.022**	.87	1.00	1.00	781.40
	$\mathbb{B}$–BUS	**.05**	97%	**4.00**	.023	**.81**	**.83**	1.00	**650.92**
Fashion MNIST	TDS	31.17	94%	2860.42	**.005**	**.20**	**.59**	**1.00**	**460.86**
	$\mathbb{B}$–TDS	**.08**	98%	**4.00**	.000	.12	.14	.24	127.71
	BUS	9.55	96%	511.94	**.004**	**.10**	**.79**	1.00	**618.66**
	$\mathbb{B}$–BUS	**.06**	97%	**4.00**	.005	.81	.89	1.00	697.82

Table 1: Experiments on MNIST and Fashion MNIST datasets. Averages over 50 samples from each datasets. Values are rounded up to two decimal places.

perform much faster than their arbitrary counterpart. Converging only after 4 iterations. This is due to their sub-linear (logarithmic) dependency from the feature space's diameter. Nevertheless, in both datasets we observe that the fail to approximate the optimal apothem, for robustness certifications. On the other hand, the quality of uniform algorithms' results is improved in dual certification calculation, offering similar results to BUS. Thus, offering the best quality vs performance trade-off, in dual certifications. For robustness certifications arbitrary intervals remain the best option.

The above asymmetry should be expected. Recall that both BUS and $\mathbb{B}$-BUS perform drastic changes in each iteration. BUS updates d coordinates in each iteration, by updating all the coordinates of one endpoint. However, $\mathbb{B}$-BUS updates $2d$ coordinates, by refining both endpoints at every step. Thus, BUS and $\mathbb{B}$-BUS behave much more similar than their top-down counterparts.

In Tbl. 1 we also observe the duality gap, between robustness and dual certifications. For MNIST the duality gap is $\simeq .006$, while in Fashion MNIST is essentially zero. This provides strong empirical evidence on our algorithms' apothem optimality. Moreover, in Fashion MNIST the duality gap for the other qualitative metrics is small. However, this is not the case for MNIST. How and if the apothem duality gap affects the other measures is left as future work.

TDS ablation study. Here we present an ablation study on δ for TDS. We use the MNIST dataset and NN described previously. The precision constant δ ranges from 0.1 to 0.9, with step 0.1. Our results are presented in Fig. 5. As expected, the number of oracle calls decreases as the δ increases. However, the time per oracle call increases. Hence, the total time remains about the same.

Nevertheless, we observe a non-monotonic corelation between the apothem ϖ and δ. This is the result of approximating an unknown real value by a rational number, known in the literature as *Diophantine approximation* [27]. Indeed, let ϖ^* be the *true* apothem. Then, TDS returns an approximation of the form $\widehat{\varpi} = 1 - N \cdot \delta$, where $N = \lceil (1 - \varpi^*)/\delta \rceil$, the number of oracle calls (steps).

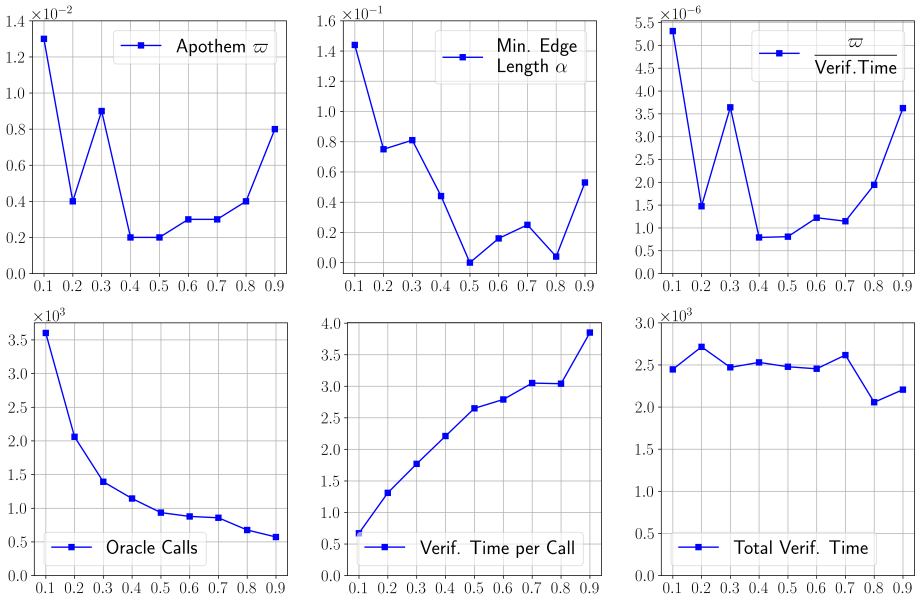


Fig. 5: TDS ablation study for $\delta = 0.1, \ldots, 0.9$. Averages over 50 MNIST images.

The quality of the approximation depends on the divisibility of $1 - \varpi^*$ by δ. If δ divides well $1 - \varpi^*$ (leaves a small reminder), then all its *divisors* will divide $1 - \varpi^*$ well. For instance, assume that $\varpi^* = 0.134$. Then, for $\delta = 0.1$, we have $N = 9$ and $\widehat{\varpi} = 1 - 9 \cdot 0.1 = 0.1$. The same $\widehat{\varpi} = 0.1$ is obtained for $\delta = 0.9$ or $\delta = 0.3$. This *periodicity* is exhibited in Fig. 5, with the *sawtooth-shaped* graph.

In any case, choosing $\delta = 0.1$ is the most preferable choice as depicted in the apothem to total verification time ratio in Fig. 5. Nevertheless, choosing $\delta = 0.9$ indeed seems to offer a viable alternative with better performance, should someone be able to afford a slight decrease in results quality. The ablation study for Fashion MNIST offers similar results, reported on our arxiv pre-print [25].

Comparison with Existing Software. A direct comparison with existing software is currently impossible. Software incompatibilities (e.g., between PyTorch, TensorFlow, and ONNX, or Marabou and ERAN, etc.) raise implementation obstacles. Moreover, most existing algorithms do not output their certifications, but only a final average. Thus, a direct experimental comparison is impossible. Here we make a preliminary comparison based on the reported results in related works, on the same datasets. We observe a *two-fold* improvement w.r.t. α, and *an order of magnitude* improvement w.r.t. $\overline{E}$.

Liu et al. [20] train 3 networks for MNIST, with 3 hidden layers and architectures $\langle 3 \times 100 \rangle = \langle 100, 100, 100 \rangle, \langle 3 \times 300 \rangle$ and $\langle 3 \times 500 \rangle$. In each case, they report minimum edge length of $\alpha \simeq 0.06$, for uniform certifications, and $\alpha \simeq 0.07$, for symmetric certifications. For Fashion MNIST, they use a network with architec-

ture $\langle 3 \times 1024 \rangle$. They report $\alpha \simeq 0.026$, for uniform and $\alpha \simeq 0.028$, for symmetric certifications. The implementation of Li et al. [18] is not publicly available. They train 3 NNs for MNIST (MNIST100, MNIST300, MNIST500), and one NN for Fashion MNIST (FMNIST100). They report α of 0.058, 0.041, 0.045, and 0.072, for MNIST100, MNIST300, MNIST500, and FMNIST100, respectively. Finally, Kabahala and Drachsler-Cohen [12] report average edge length $\overline{E}$ of 0.196 and 0.1 for two MNIST NNs ($\langle 3 \times 50 \rangle$, $\langle 3 \times 100 \rangle$), whereas for two Fashion MNIST NNs ($\langle 3 \times 50 \rangle$, $\langle 3 \times 250 \rangle$) they report $\overline{E}$ of 0.191 and 0.037.

6 Related Work

Interval arithmetic [29,23] has been used in NN adversarial analysis (e.g. [32,14,36]). However, they do not make explicit use of the underlying lattice structure [29].

Adversarial robustness is the problem of *deciding* whether an area is free of adversarial examples. Several approaches exist [22], such as abstraction (e.g. [28]), satisfiability modulo theories (SMT) (e.g. [13,14]), and MILP (e.g. [15,33]).

In this work we focus on *robustness certification* which is the optimization variant of the above. Most approaches rely on differentiable optimization, where the NN is linearly relaxed (by taking the convex approximation of the ReLU activation) and then solved either by computing a feasible solution to the dual convex program [32], or by the augmented Langrage method [20,18]. These methods are scalable and can be applied to large NNs, but suffer from approximation errors [26] and lack optimality guarantees. More recent work [12] follows a hybrid approach, achieving *maximality* by querying a NN verifier to check for adversarial examples before deciding on the edge to refine using differentiable methods. Optimization happens w.r.t. $\overline{E}$, but optimality is not proved. Similar oracle-based iterative schemes have been applied in formal AI-explainability [9,34,10,11].

Existing work on robustness certification mostly focuses on optimizing the volume [32,20,18], through optimizing minimum edge length. Deciding on the adversarial robustness of a *uniform* area is **NP**-hard [13], and even approximating the optimal α is intractable for uniform certifications [31]. However, volume optimality stays intractable even if the oracle costs are discarded.

7 Conclusions

We proposed a method to compute trustworthy robustness certifications by ensuring optimality w.r.t. the apothem. Moreover, we introduced the notion of dual certifications, which gives an upper bound to the certification's quality. We also developed the `ParallelepipedoNN` system, which was evaluated over the MNIST and Fashion MNIST datasets. Our evaluation shows promising results w.r.t. existing implementations. Nevertheless, a direct experimental comparison is needed. We leave the latter as future work.

Acknowledgements. This work was partially funded by COFLED project.

References

1. Backer, J., Keil, J.M.: The Mono- and Bichromatic Empty Rectangle and Square Problems in All Dimensions. In: Latin American Theoretical Informatics Symposium (2010)
2. Chan, T.M.: Faster Algorithms for Largest Empty Rectangles and Boxes. Discrete & Computational Geometry (2023)
3. Chazelle, B., III, R.L.S.D., Lee, D.T.: Computing the Largest Empty Rectangle. SIAM Journal on Computing (1986)
4. Fefferman, C.: Reconstructing a neural net from its output. Revista Matemática Iberoamericana (1994)
5. Fischetti, M., Jo, J.: Deep neural networks and mixed integer linear optimization. Constraints - An International Journal (2018)
6. Glorot, X., Bengio, Y.: Understanding the Difficulty of Training Deep Feedforward Neural Networks. In: Conference on Artificial Intelligence and Statistics (2010)
7. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and Harnessing Adversarial Examples. In: International Conference on Learning Representations (2015)
8. Gutiérrez, G., Paramá, J.R.: Finding the Largest Empty Rectangle Containing Only a Query Point in Large Multidimensional Databases. In: Scalable Scientific Data Management (2012)
9. Ignatiev, A., Narodytska, N., Marques-Silva, J.: Abduction-Based Explanations for Machine Learning Models. In: Association for the Advancement of Artificial Intelligence (2019)
10. Izza, Y., Huang, X., Morgado, A., Planes, J., Ignatiev, A., Marques-Silva, J.: Distance-Restricted Explanations: Theoretical Underpinnings & Efficient Implementation. Principles of Knowledge Representation and Reasoning (2024)
11. Izza, Y., Ignatiev, A., Stuckey, P.J., Marques-Silva, J.: Delivering Inflated Explanations. In: Association for the Advancement of Artificial Intelligence (2024)
12. Kabaha, A., Drachsler-Cohen, D.: Maximal Robust Neural Network Specifications via Oracle-Guided Numerical Optimization. In: International Conference on Verification, Model Checking, and Abstract Interpretation (2023)
13. Katz, G., Barrett, C.W., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In: Computer-Aided Verification (2017)
14. Katz, G., Barrett, C.W., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: a Calculus for Reasoning about Deep Neural Networks. Formal Methods System Design (2022)
15. 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: Computer-Aided Verification (2019)
16. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations (2015)
17. LeCun, Y., Cortes, C., Burges, C.J.: MNIST handwritten digit database. ATT Labs (2010)
18. Li, C., Ji, S., Weng, H., Li, B., Shi, J., Beyah, R., Guo, S., Wang, Z., Wang, T.: Towards Certifying the Asymmetric Robustness for Neural Networks: Quantification and Applications. IEEE Transactions on Dependable and Secure Computing (2022)

19. Liu, C., Arnon, T., Lazarus, C., Barrett, C.W., Kochenderfer, M.J.: Algorithms for Verifying Deep Neural Networks. Foundations and Trends in Optimization (2019)
20. Liu, C., Tomioka, R., Cevher, V.: On Certifying Non-Uniform Bounds against Adversarial Attacks. In: International Conference on Machine Learning (2019)
21. Lomuscio, A., Maganti, L.: An approach to reachability analysis for feed-forward ReLU neural networks. Computer Research Repository (2017)
22. Meng, M.H., Bai, G., Teo, S.G., Hou, Z., Xiao, Y., Lin, Y., Dong, J.S.: Adversarial Robustness of Deep Neural Networks: A Survey from a Formal Verification Perspective. Computing Research Repository (2022)
23. Moore, R.E., Kearfott, R.B., Cloud, M.J.: Introduction to Interval Analysis. Society for Industrial and Applied Mathematics (2009)
24. Naamad, A., Lee, D.T., Hsu, W.L.: On the maximum empty rectangle problem. Discrete Applied Mathematics (1984)
25. Papamichail, M., Varsos, K., Flouris, G., Marques-Silva, J.: Are safety guarantees in neural networks safe? How to compute trustworthy robustness certifications. Computer Research Repository (2026)
26. Salman, H., Yang, G., Zhang, H., Hsieh, C.J., Zhang, P.: A Convex Relaxation Barrier to Tight Robustness Verification of Neural Networks. In: Advances in Neural Information Processing Systems (2019)
27. Schmidt, W.M.: Diophantine approximation, pp. 34–72. Springer Berlin Heidelberg (1991)
28. Singh, G., Gehr, T., Püschel, M., Vechev, M.T.: An Abstract Domain for Certifying Neural Networks. Proceeding of the ACM on Programming Languages (2019)
29. Sunaga, T.: Theory of an Interval Algebra and its Application to Numerical Analysis. Research Association of Applied Geometry (1958)
30. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. In: International Conference on Learning Representations (2014)
31. Weng, T.W., Zhang, H., Chen, H., Song, Z., Hsieh, C.J., Daniel, L., Boning, D.S., Dhillon, I.S.: Towards Fast Computation of Certified Robustness for ReLU Networks. In: International Conference on Machine Learning (2018)
32. Wong, E., Kolter, J.Z.: Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In: International Conference on Machine Learning (2018)
33. 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: Computer-Aided Verification (2024)
34. Wu, M., Wu, H., Barrett, C.W.: VeriX: Towards Verified Explainability of Deep Neural Networks. In: Advances in Neural Information Processing Systems (2023)
35. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. Computer Research Repository (2017)
36. Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., Hsieh, C.J.: Fast and Complete: Enabling Complete Neural Network Verification with Rapid and Massively Parallel Incomplete Verifiers. In: International Conference on Learning Representations (2021)