Robust-by-Design Classification via Unitary-Gradient Neural Networks

Fabio Brau, Giulio Rossolini, Alessandro Biondi, and Giorgio Buttazzo

Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, Pisa,Italy
{fabio.brau, giulio.rossolini, alessandro.biondi, giorgio.buttazzo}@santannapisa.it

Abstract

The use of neural networks in safety-critical systems requires safe and robust models, due to the existence of adversarial attacks. Knowing the minimal adversarial perturbation of any input x, or, equivalently, knowing the distance of x from the classification boundary, allows evaluating the classification robustness, providing certifiable predictions. Unfortunately, state-of-the-art techniques for computing such a distance are computationally expensive and hence not suited for online applications. This work proposes a novel family of classifiers, namely *Signed Distance Classifiers* (SDCs), that, from a theoretical perspective, directly output the exact distance of x from the classification boundary, rather than a probability score (e.g., SoftMax). SDCs represent a family of robust-by-design classifiers. To practically address the theoretical requirements of a SDC, a novel network architecture named *Unitary-Gradient Neural Network* is presented. Experimental results show that the proposed architecture approximates a signed distance classifier, hence allowing an online certifiable classification of x at the cost of a single inference.

Introduction

Deep Neural Networks (DNNs) reached popularity due to the high capability of achieving super-human performance in various tasks, such as *Image Classification, Object Detection* and *Image Generation*. However, their usage in safety-critical systems, such as *autonomous cars*, is pushing the scientific community toward the definition and the achievement of certifiable guarantees.

In this regard, as independently shown by (Szegedy et al. 2013; Biggio et al. 2013), neural networks are highly sensitive to small perturbations of the input, also known as *adversarial examples*, which are not easy to detect (Biggio and Roli 2018; Carlini et al. 2018; Rossolini, Biondi, and Buttazzo 2022), and cause the model to produce a wrong classification. Informally speaking, a classifier is said to be ε-*robust* in a certain input x if the classification result does not change by perturbing x with all possible perturbations of a bounded magnitude ε.

In the last few years, a large number of methods for crafting adversarial examples have been presented (Goodfellow, Shlens, and Szegedy 2015; Moosavi-Dezfooli, Fawzi, and Frossard 2016; Rony et al. 2020; Madry et al. 2019). In particular, (Carlini and Wagner 2017; Rony et al. 2019) proposed methods to find the minimal adversarial perturbation (MAP) or, equivalently, the closest adversarial example for a given input x. Such a perturbation directly provides the distance of x from the classification boundary, which, given a maximum magnitude of perturbation, can be used to verify the trustworthiness of the prediction (Weng et al. 2018) and design robust classifiers (Wong et al. 2018; Cohen, Rosenfeld, and Kolter 2019). Note that, when the MAP is known, one can check on-line whether a certain input x can be perturbed with a bounded-magnitude perturbation to change the classification result. If this is the case, the network itself can signal the unsafeness of the result. Unfortunately, due the hard complexity of the algorithms for solving the MAP problem on classic models, the aforementioned strategies are not suited for efficiently certifying the robustness of classifiers (Brau et al. 2023).

To achieve provable guarantees, other works focused on designing network models with bounded Lipschitz constant that, by construction, offers a lower bound of the MAP as the

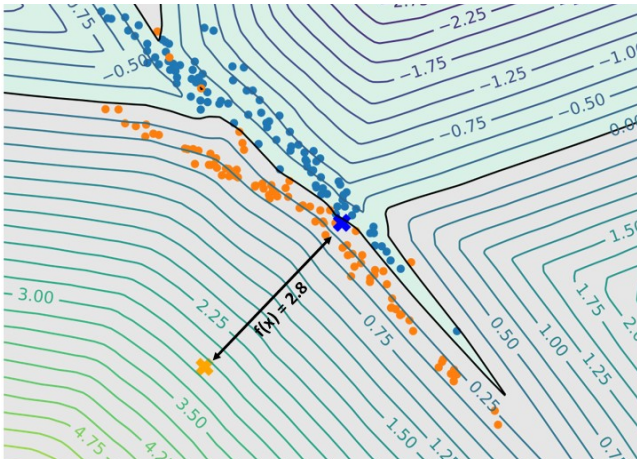

Figure 1: An example of a binary SDC. Observe that the countour lines are parallel-curves of the classification boundary (the black curve) and the output of the model in a x (the orange cross) directly provides the distance from the closest point in the classification boundary (the blue cross).

network output (Tsuzuku, Sato, and Sugiyama 2018). These particular models can be obtained by composing orthogonal layers (Cisse et al. 2017; Li et al. 2019; Trockman and Kolter 2021; Serrurier et al. 2021; Singla and Feizi 2021) and norm-preserving activation functions, such as those presented by (Anil, Lucas, and Grosse 2019; Chernodub and Nowicki 2017). However, despite the satisfaction of the Lipschitz inequality, these models do not provide the exact boundary distance but only a lower bound.

This work introduces a new family of classifiers, namely Signed Distance Classifiers (SDC), that *straighten the Lipschitz lower bound by outputting the exact distance of an input x from the classification boundary*. SDC can then solve the MAP problem as a result of the network inference (see Figure 1). From a theoretical point of view, we extend the characterization property of the signed distance functions to a multi-class classifier. From a practical perspective, we address such a theoretical model by presenting a new architecture, named Unitary-Gradient Neural Network (UGNN), *having unitary gradient (under the Euclidean norm) in each differentiable point*. In summary, this work provides the following contributions:

- It introduces a notable family of classifiers, named SDC, which provide as output the distance of an input x from the classification boundary.
- It provides a novel network architecture, named UGNN, which, to best of our knowledge, represents the first practical approximation of an SDC.
- It shows that the abs function can replace other more expensive norm-preserving activation functions without introducing a significant accuracy loss. Furthermore, it proposes a new layer named *Unitary Pair Difference*, which is a generalization of a fully-connected orthogonal layer.
- It assesses the performance, the advantages, and the limitations of the proposed architecture through a close comparison with the state-of-the-art models in the field of the Lipschitz-Bounded Neural Networks.

Related Work

The evaluation and the provable verification of the robustness of a classification model can be addressed by computing the MAP in a given point x (Carlini et al. 2018). Since that computation involves solving a minimum problem with non-linear constraints, the community focused on designing faster algorithms to provide an accurate estimation of the distance to the classification boundary (Rony et al. 2019, 2020; Pintor et al. 2021). However, all these algorithms require multiple forward and backward steps, hence are not suited for an online application (Brau et al. 2023).

On the other side, since the sensitiveness to input perturbations strictly depends on the Lipschitz constant of the model, knowing the local Lipschitz constant in a neighborhood of x provides a lower bound of the MAP in x (Hein and Andriushchenko 2017). In formulas, for a L-Lipschitz neural network f, a lower bound of MAP is deduced by considering $\frac{1}{L\sqrt{2}}(f_l(x) - f_s(x))$, where l, s are the first and the second top-2 components, respectively. However, for common DNNs, obtaining a precise estimation of L is still computationally expensive (Weng et al. 2018), thus also this strategy is not suited for an online application.

For these reasons, recently, other works focused on developing neural networks with a bounded Lipschitz constant by design (Huang et al. 2021). (Miyato et al. 2018) achieved 1-Lipschitz fully connected layers by bounding the spectral-norm of the weight matrices to be 1. Similarly, (Serrurier et al. 2021) considered neural networks f in which each component f_i is 1-Lipschitz, thus, differently from the 1-Lipschitz networks mentioned before, given a sample x, the lower bound of MAP is deduced by $\frac{1}{2}(f_l(x) - f_s(x))$.

Other authors leveraged *orthogonal* weight matrices to pursue the same objective. For instance, (Li et al. 2021) showed that a ReLU Multi-Layer Perceptron merely composed by orthogonal layers is 1-Lipschitz. Indeed, an orthogonal matrix W (i.e. such that $WW^T = I$ or $W^TW = I$) has a unitary spectral norm, $\|W\| = 1$. Roughly speaking, orthogonal fully connected and convolutional layers can be obtained by *Regularization* or *Parameterization*. The former methods include a regularization term in the training loss function to encourage the orthogonality of the layers, e.g (Cisse et al. 2017) use $\beta\|W^TW - Id\|^2$. The latter methods, instead, consider a parameterization of the weight $W(\theta)$ depending on a unconstrained parameter θ so that, for each θ, $W(\theta)$ is an orthogonal weight matrix (Anil, Lucas, and Grosse 2019; Trockman and Kolter 2021). For convolutional layers, a regularization strategy can be applied, since they can be written as matrix-vector product through a structured matrix (Wang et al. 2020). However, recent parameterized strategies as BCOP (Li et al. 2019), CayleyConv (Trockman and Kolter 2021), and Skew Convolution (Singla and Feizi 2021) come out as efficient and performant alternatives.

This work defines an SDC, as a function f that provides the MAP by computing $f_l(x) - f_s(x)$, thus tightening the lower bounds provided by L-Lipschitz classifiers. Furthermore, we present the UGNN, designed by properly leveraging the previous orthogonal parameterized strategies, as the first architecture that approximate a theoretical SDC.

Signed Distance Classifier

In this context, a classifier $\hat{k} : \mathcal{X} \to \mathcal{Y}$ maps the input domain into a finite set of labels $\mathcal{Y}$. The concept of *robustness* is formally stated in the following definition.

Definition 1 (robustness)**.** A classifier $\hat{k}$ is ε-robust in an input $x \in \mathbb{R}^n$ (or equivalently, a classification $\hat{k}(x)$ is ε-robust), if $\hat{k}(x + \delta) = \hat{k}(x)$ for any perturbation δ with $\|\delta\| < \varepsilon$, where $\|\cdot\|$ is the Euclidean norm.

Binary Classifiers

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a binary classifier that provides a classification of an input x based on its sign, i.e., $\hat{k}(x) = \mathrm{sgn}(f(x))$, and let $\mathcal{B}_f := \{x \in \mathbb{R}^n : f(x) = 0\}$ be the classification boundary of f. Given an input sample x, the *MAP* problem for a binary classifier is defined as follows:

$$\begin{aligned} d_f(x) := \inf_{p \in \mathbb{R}^n} \quad & \|p - x\| \\ \text{s.t.} \quad & f(p) = 0, \end{aligned} \tag{1}$$

where d_f represents the distance function from the boundary $\mathcal{B}_f$. The *closest adversarial example* to x is defined as the unique x^* (if any) such that $d_f(x) = \|x - x^*\|$ and $\text{sgn}(f(x)) \neq \text{sgn}(f(x^*))$. Observe that Problem (1) is equivalent to the definition of *Minimal Adversarial Perturbation* in (Moosavi-Dezfooli, Fawzi, and Frossard 2016).

Certifiable robustness. We refer to $\delta^* = x^* - x$ as the *perturbation* that realizes the MAP. By definition of $d_f(x)$, for each perturbation δ with $\|\delta\| < d_f(x)$ it holds $\hat{k}(x+\delta) = \hat{k}(x)$; hence, $\hat{k}$ is certifiable $d_f(x)$-robust in x.

A *Signed Distance Function* d_f^* is defined as follows:

$$d_f^*(x) = \begin{cases} d_f(x) & x \in R_+ \\ -d_f(x) & x \notin R_+. \end{cases} \tag{2}$$

where $R_+ = \{f > 0\}$. Following this definition, a signed distance function d_f^* satisfies intriguing properties that make it highly interesting for *robustness evaluation, verification, and certifiable prediction*. In particular, d_f^* provides the same classification of f, since $\text{sgn}(d_f^*(x)) = \text{sgn}(f(x))$ for each $x \in \mathbb{R}^n$. Furthermore, the gradient $\nabla d_f^*(x)$ coincides with the direction of the shortest path to reach the closest adversarial example to x (Federer 1959, Thm. 4.8).

Observation 1. Let $x \in \mathbb{R}^n$, if there exists a unique $x^* \in \mathcal{B}_f$ such that $d_f(x) = \|x - x^*\|$, then d_f^* is differentiable in x such that

$$\nabla d_f(x) = \frac{x - x^*}{\|x - x^*\|}, \tag{3}$$

and hence has a gradient with unitary Euclidean norm, i.e., $\|\nabla d_f^*(x)\| = 1$, referred to as *unitary gradient* (UG) for short in the following. Furthermore, d_f^* is such that:

1. It provides a trivial way to certify the robustness of $\hat{k}$ in x, since, by definition, $|d_f^*(x)|$ represents the MAP.
2. It explicitly provides the closest adversarial example to x, which can be computed $x^* = x - d_f^*(x)\nabla d_f^*(x)$.

Proof. Refer to (Federer 1959, Thm. 4.8) □

Inspired by these intriguing properties, this work aims at *investigating classifiers whose output provides the distance (with sign) from their own classification boundary.*

A Characterization Property

A trivial example of a binary classifier f that coincides with a signed distance function is given by any *affine function* with a unitary weight. Indeed, if $f(x) = w^T x + b$, where $\|w\| = 1$, then the MAP relative to f has the explicit unique solution of the form $x^* = x - \frac{f(x)}{\|w\|^2} w$, as already pointed out in (Moosavi-Dezfooli, Fawzi, and Frossard 2016), from which $d_f(x) = |f(x)|$.

As shown in Observation 1, a signed distance function has a unitary gradient. Under certain hypotheses, the opposite implication holds: a function f with a unitary gradient coincides with a signed distance function from $\mathcal{B}_f$. This result is formalized in the following theorem.

Theorem 1. *Let $\mathcal{U} \subseteq \mathbb{R}^n$ be an open set, and let $f : \mathbb{R}^n \to \mathbb{R}$ be a function, smooth in $\mathcal{U}$, such that $\mathcal{B}_f \subseteq \mathcal{U}$. If f has a unitary gradient in $\mathcal{U}$, then there exists an open set $\Omega_f \subseteq \mathcal{U}$ such that f coincides in Ω_f with the signed distance function from $\mathcal{B}_f$. Formally,*

$$\|\nabla f_{\restriction \mathcal{U}}\| \equiv 1 \quad \Rightarrow \quad \exists \Omega_f \subseteq \mathcal{U}, \quad f_{\restriction \Omega_f} \equiv d^*_{f \restriction \Omega_f}. \tag{4}$$

Proof. The proof is built upon (Sakai 1996, Prop.2.1). Any trajectory $\gamma : [0, 1] \to \mathcal{U}$ that solves the dynamical system $\dot{\gamma}(t) = \nabla f(\gamma(t)))$ coincides with the shortest path between the point $\gamma(0)$ and the hyper-surface $f^{-1}(\gamma(1))$. Details are reported in the Appendix. □

It is worth noting that, as pointed out in (Sakai 1996, Prop.2.1), this characterization holds for particular geometrical spaces, i.e., *Complete Riemannian Manifolds*. Unfortunately, as shown by the author, the only smooth functions with unitary gradient in a Complete Remannian Manifold with non-negative Ricci Curvature (e.g., $\mathbb{R}^n$) are the affine functions (Sakai 1996, Theorem A). However, an open set $\mathcal{U} \subset \mathbb{R}^n$ is a Remannian Manifold that does not satisfy the completeness property. Hence, the existence of a non-affine signed distance function is not in contradiction with (Sakai 1996, Theorem A). A trivial example is given by the binary classifier $f(x) = \|x\| - 1$ defined in $\mathcal{U} = \mathbb{R}^n \setminus \{0\}$. Further details are provided in the Appendix.

Extension to Multi-Class Classifiers

By following the *one-to-rest* strategy (Schölkopf et al. 2002), the results above can be extended to multi-class classifiers. Let $f : \mathbb{R}^n \to \mathbb{R}^C$ be a smooth function by which the predicted class of a sample $x \in \mathbb{R}^n$ is given by $\hat{k}(x) = \text{argmax}_i f_i(x)$, where $\hat{k}(x) = 0$ if there is no unique maximum component. Observe that, according to (Biggio et al. 2013; Szegedy et al. 2013; Moosavi-Dezfooli, Fawzi, and Frossard 2016), the MAP problem for a multi-class classifier can be stated as follows:

$$\begin{aligned} d_f(x) := \inf_{p \in \mathbb{R}^n} \quad & \|p - x\| \\ \text{s.t.} \quad & \hat{k}(p) \neq \hat{k}(x). \end{aligned} \tag{MAP}$$

Here, we extend the definition of signed distance function d_f^* to a multi-class *Signed Distance Classifier* f as follows.

Definition 2 (Signed Distance Classifier)**.** A function $f : \mathbb{R}^n \to \mathbb{R}^C$ is a *Signed Distance Classifier* if, for each pair i, j, with $i \neq j$, the difference $(f_i - f_j)$ corresponds to the signed distance from the one-to-one classification boundary $\mathcal{B}_{ij} := \{x \in \mathbb{R}^n : f_i - f_j = 0\}$.

The following observation shows that an SDC satisfies similar properties of Observation 1 for binary classifiers.

Observation 2. Let $f : \mathbb{R}^n \to \mathbb{R}^C$ be a signed distance function and let $x \in \mathbb{R}^n$ be a sample classified as $l = \hat{k}(x)$. Let $s := \text{argmax}_{j \neq l} f_j(x)$ be the second-highest component of $f(x)$. Hence, the classifier f:

1. Provides a fast way to certificate the robustness of $\hat{k}$ in x. In fact, $f_l(x) - f_s(x) = d_f(x)$, where $d_f(x)$ is the MAP.

2. Provides the closest adversarial example to x, i.e.,

$$x^* = x - (f_l(x) - f_s(x))\nabla(f_l - f_s)(x),$$

where x^* is the unique solution of Problem MAP in x.

Proof. The detailed steps are in the Appendix. □

Similarly to the binary case, an SDC is characterized by having a *unitary gradient* for each pair-wise difference of the output components. In details, by directly applying Theorem 1, a smooth classifier f is a signed distance classifier (in some open set) if and only if $\|\nabla(f_i - f_j)\| \equiv 1, \forall i \neq j$.

Unitary-Gradient Neural Networks

In the previous section, we showed that if a smooth classifier f satisfies the unitary gradient property in some open set $\mathcal{U} \supseteq \mathcal{B}_f$, then it admits an open set $\Omega_f \supseteq \mathcal{B}_f$ in which f coincides with the signed distance function with respect to the boundary $\mathcal{B}_f$. Furthermore, affine functions represents all and the only smooth SDCs in the whole $\mathbb{R}^n$.

Supported by these results, any DNN that globally satisfies the UG property would coincide with a trivial linear model, which hardly provides good classification performance for complex tasks. To approximate a non-trivial SDC with a well-performing DNN f_θ, we impose the UG property almost-everywhere.This section shows the proper requirements on f_θ to satisfy the hypothesis of Theorem 1, providing layer-wise sufficient conditions that ensure the UG property. To this end, we focus our analysis on the family $\mathcal{F}$ of feed-forward DNNs $f : \mathbb{R}^n \to \mathbb{R}^C$ of the form $f = g \circ h^{(L)} \circ \cdots \circ h^{(1)}$, where g is the output-layer and each $h^{(i)}$ is any canonical elementary layer (e.g., Fully Connected, Convolutional, etc.) or an activation function.

Observation 3 (Layer-wise sufficient conditions)**.** Let f be a DNN in $\mathcal{F}$. For each i, let $J^{(i)}(x)$ be the Jacobian of $h^{(i)}$ evaluated in $y = h^{(i-1)} \circ \cdots \circ h^{(1)}(x)$. For each $j = 1, \ldots, C$, let $W_j(x)$ be the Jacobian of g_j evaluated in $y = h^{(L)} \circ \cdots \circ h^{(1)}(x)$. Hence, if

$$J^{(i)}J^{(i)T} \equiv I, \quad \forall i = 1, \ldots, L \qquad \text{(GNP)}$$

$$(W_h - W_k)(W_h - W_k)^T \equiv 1, \quad \forall h \neq k, \qquad \text{(UPD)}$$

then, for all $h \neq k$, $f_h - f_k$ satisfies the UG property.

Proof. For a feed-forward neural network, the Jacobian matrix of each component f_j can be decomposed as

$$\mathrm{Jac}(f_j) = W_j \prod_{i=1}^{L} J^{(i)} = W_j J^{(L)} \cdots J^{(1)}. \qquad (5)$$

Hence, the thesis follows by the associative property and by observing that $(AB)^T = B^T A^T$ for any two matrices. □

Observe that Condition GNP, namely *Gradient Norm Preserving*, requires any layer to have an output dimension no higher than the input dimension. Indeed, a rectangular matrix $J \in \mathbb{R}^{M \times N}$ can be orthogonal by row, i.e, $JJ^T = I$, only if $M \leq N$. Condition GNP is also addressed in (Li et al. 2019; Trockman and Kolter 2021) to build Lipschitz-Bounded Neural Networks. However, for their purposes, the authors also consider DNNs that satisfy a weaker condition named *Contraction Property* (see (Trockman and Kolter 2021)), which includes the $M \geq N$ case.

Gradient Norm Preserving Layers

We now provide an overview of the most common layers that can satisfy the GNP property. For a shorter notation, let h be a generic internal layer.

Activation Function Activation functions $h : \mathbb{R}^n \to \mathbb{R}^n$ can be grouped in two main categories: *component-wise* and *tensor-wise* activation functions. Common component-wise activation functions as ReLU, tanh, and sigmoid do not satisfy the GNP property (Chernodub and Nowicki 2017). Moreover, since any component-wise function h that satisfies the GNP property is piece-wise linear with slopes ± 1 (see the appendix for further details), abs is GNP.

Tensor-wise activation functions have recently gained popularity thanks to (Chernodub and Nowicki 2017; Anil, Lucas, and Grosse 2019; Singla, Singla, and Feizi 2021), who introduced OPLU, GroupSort, HouseHolder activation functions, respectively, which are specifically designed to satisfy the GNP property. An overview of these activation functions is left in the appendix. In this work, we compare the abs with the OPLU and the GroupSort with a group size of 2, a.k.a MaxMin.

Fully Connected and Convolutional Layers A fully connected layer of the form $h(x) = Wx + b$ has a constant Jacobian matrix $\mathrm{Jac}(h)(x) = W$. This implies that h is GNP if and only if W is an orthogonal-by-row matrix. Similarly, for a convolutional layer with kernel $\mathcal{K}$ of shape $M \times C \times k \times k$, the GNP property can be satisfied only if $M \leq C$, i.e., the layer does not increase the number of channels of the input tensor (Li et al. 2021). As done in (Anil, Lucas, and Grosse 2019), in our model we consider the Björck parameterization strategy to guarantee the orthogonality of the fully connected layers and the CayleyConv strategy presented in (Trockman and Kolter 2021) for the convolutional layers.

Pooling, Normalization and Residual Layers Max-pooling two-dimensional layers with kernel $\boldsymbol{k} = (k_1, k_2) \in \mathbb{N}^2$, stride $\boldsymbol{s} = \boldsymbol{k}$, and without padding, satisfy the GNP property if applied to a tensor whose spatial dimensions H, W are multiples of k_1 and k_2, respectively. This can be proved by observing that the Jacobian matrix corresponds to an othogonal projection matrix (Li et al. 2021).

Batch-normalization layers with a non-unitary variance do not satisfies the GNP property (Li et al. 2021). For residual blocks, it is still not clear whether they can or cannot satisfy the GNP property. Indeed, a residual layer of the form $h(x) = x + \tilde{h}(x)$ is GNP if and only if $\mathrm{Jac}(\tilde{h})\mathrm{Jac}(\tilde{h})^T + \mathrm{Jac}(\tilde{h}) + \mathrm{Jac}(\tilde{h})^T \equiv 0$. For such reasons, the last two mentioned layers are not considered in our model.

Unitary Pair Difference Layers

This section focuses on the second condition stated in Observation 3: the *Unitary Pair Difference* (UPD).

Since most neural classifiers include a last fully-connected layer, we restrict our analysis to this case. Let

$g(x) = Wx+b$ be the last layer, since $\mathrm{Jac}(g) \equiv W$, then the UPD property requires that for each two rows W_h, W_k the difference $W_h - W_k$ has unitary norm. A matrix W satisfies the UPD property if the function $x \mapsto Wx$ is UPD.

Bounded UPD layer. An UPD matrix from any orthogonal-by-row matrix as stated by the next observation.

Observation 4. Let $Q \in \mathbb{R}^{m \times C}$ such that $QQ^T = I$. Then, $W = \frac{1}{\sqrt{2}}Q$ satisfies the UPD property, indeed

$$\|W_h - W_k\|^2 = \underbrace{\|W_h\|^2}_{1/2} + \underbrace{\|W_k\|^2}_{1/2} - 2\underbrace{W_h^T W_k}_{0} = 1. \quad (6)$$

An UPD layer with matrix W as above is said to be *bounded*, as each row of W is bounded to have norm $1/\sqrt{2}$.

As pointed out in (Singla, Singla, and Feizi 2021), this constraint makes it harder to train the model when the output dimension C is large (i.e., there are many classes).

Unbounded UPD layer. To avoid this issue, we considered an UPD layer with parametric weight matrix $W(U)$. Matrix $W(U)$ is obtained by iteratively applying the L-BFGS, an optimization algorithm for unconstrained minimum problems (Liu and Nocedal 1989), to the loss

$$\Psi(U) = \sum_{h<k} (\|U_h - U_k\|^2 - 1)^2. \quad (7)$$

More specifically, if `psi` is the routine that computes such a loss function and `L-BFGS` is the routine that performs one step of the L-BFGS optimization algorithm, then the weight matrix is obtained as $W = \texttt{UPD}(U)$, where `UPD` is the following procedure:

```
def UPD(U: Tensor):
    # Returns an UPD matrix
    W = U
    for in range(max_iter):
        W = L-BFGS(psi(U), W)
    return W
```

Listing 1: Psudo code that parameterizes an UPD matrix through a parameter U. The resulting W is obtained by performing few steps of the L-BFGS algorithm to find a minimum of Ψ with starting point U.

Note that the UPD layer $g(x) = W(U)\,x + b$ depends on the weights U, b and it is fully differentiable in U. This implies that such a procedure, like parameterization strategies for orthogonal layers, can be applied during training. Finally, note that the algorithm complexity strongly depends on the computational cost of the objective $\Psi(U)$. Our implementation exploits parallelism by implementing $\Psi(U)$ by means of a matrix product of the form $A^{(C)}U$, where $A^{(C)}$ is designed to compute all the pair differences between rows required by Eq. (7) (see Appendix).

Unitary-Gradient Neural Network Architecture

This section describes how to practically combine GNP and UPD layers to obtain a neural network f_θ such that all pairwise differences of its output vector have unitary gradient. The main difficulty in crafting such a network is due to the GNP property, which implies a decreasing dimension in both dense and convolutional layers. Indeed, most DNNs for image classification process a 3-channel image by gradually increasing the channel dimension of convolutional layers.

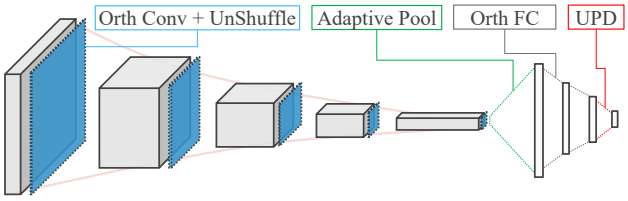

Figure 2: Tested UGNN architecture: 5 GNP conv-blocks, 2 FC GNP layers and 1 UPD layer.

To overcome this issue, we leverage a 2-dimensional *PixelUnshuffle* layer (Shi et al. 2016), which inputs a tensor of shape $C \times rH \times rW$ and outputs a tensor of shape $r^2C \times H \times W$. The output is obtained by only rearranging input components. As such, this layer satisfies the GNP property (proof available in Appendix). The PixelUnshuffle layer allows increasing the number of channels of hidden layers even in convolutional GNP networks.

It is worth pointing out that (Li et al. 2019; Trockman and Kolter 2021) also leveraged such a permutation layer, but only to emulate a convolution with stride 2. That said, the UGNN proposed in this work, shown in Fig. 2, consists of five GNP blocks, two fully connected GNP layers, a last UPD layer (bounded or unbounded), and GNP activation functions. Each GNP block consists of two GNP convolutional layers and one last PixelUnshuffle layer with scaling factor 2; a GNP activation function is applied after each convolution (see Tab. 1). Each convolutional layer has a circular padding to preserve the spatial dimension of the input. Furthermore, before the flattening stage, a max-pool layer with window size and stride $H/2^5$ is applied to process input of different spatial dimension $H = m \cdot 2^5$, for any $m \in \mathbb{N}$.

Note that, to the best of our records, this is the first instance of a convolutional DNN that aims at pratically implementing an SDC and that provably satisfies $\|\nabla(f_i - f_j)\| \equiv 1$ almost everywhere. (Béthune et al. 2021) only focused on fully-connected networks, while (Serrurier et al. 2021) only approximated an optimal f^* such that $\|\nabla f_i^*\| \equiv 1$. (Boczko and Young 2005) approximate a binary SDC via SVM. In conclusion, observe that, by design, each pair-difference $f_i - f_j$ of an UGNN satisfies the 1-Lipschitz prop-

Layers	**Output Shape**
`OrthConv`$(3 \cdot 4^i, 3 \cdot 4^i, 3)$	$3 \cdot 4^i \times \frac{H}{2^i} \times \frac{H}{2^i}$
`GNP Activation`	-
`OrthConv`$(3 \cdot 4^i, 3 \cdot 4^i, 3)$	-
`GNP Activation`	-
`PixelUnshuffle`(2)	$3 \cdot 4^{i+1} \times \frac{H}{2^{i+1}} \times \frac{H}{2^{i+1}}$

Table 1: An example of the (i+1)th internal GNP conv-block. Observe that the number of channels increases with a geometric progression of common ratio 4 and each spatial dimension decreases with a ratio of 2.

Models	Accuracy [%] Std.Norm	Raw
LargeConvNet	**79.0** $\pm$ 0.26	**72.2** $\pm$ 0.11
LargeConvNet+Abs	77.8 $\pm$ 0.33	71.8 $\pm$ 0.25
LipConvNet5	78.0 $\pm$ 0.26	68.8 $\pm$ 0.35
LipConvNet5+Abs	76.1 $\pm$ 0.31	65.5 $\pm$ 0.68
ResNet9	78.7 $\pm$ 0.22	66.4 $\pm$ 0.17
ResNet9+Abs	78.1 $\pm$ 0.34	65.6 $\pm$ 0.22
UGNN+Abs+updB	71.9 $\pm$ 0.29	69.2 $\pm$ 0.31
UGNN+Abs+updU	72.1 $\pm$ 0.54	68.9 $\pm$ 0.81
UGNN+MaxMin+updB	72.6 $\pm$ 0.79	70.4 $\pm$ 0.52
UGNN+MaxMin+updU	**72.7** $\pm$ 0.38	70.4 $\pm$ 0.86
UGNN+OPLU+updB	71.9 $\pm$ 0.09	70.5 $\pm$ 0.39
UGNN+OPLU+updU	72.0 $\pm$ 0.70	**70.6** $\pm$ 0.45

Table 2: Accuracy comparison between the 1-Lipschitz models and the UGNNs.

erty, hence the margin $\mathcal{M}_f(x) = f_l(x) - \max_{j \neq l} f_j(x)$ is a lower bound of the MAP in x.

Observation 5 (Certifiable Robustness)**.** If f is a UGNN, then $\hat{k}(x) = \text{argmax}_i\, f_i(x)$ is $\mathcal{M}_f(x)-$robust in x.

Proof. The proof is available in the Appendix. □

Experimental Results

Experiments were conducted to evaluate the classification accuracy of a UGNN and its capability of implementing an SDC. As done by related works, the experiments targeted the CIFAR10 datasets. We compared UGNN with the following 1-Lipschitz models: *LargeConvNet* (Li et al. 2019), *ResNet9* (Trockman and Kolter 2021), and *LipConvNet5* (Singla and Feizi 2021). For all the combinations of GNP activations, UPD layers, preprocessing, and input size, our model was trained for 300 epochs, using the Adam optimizer (Kingma and Ba 2017), with learning rate decreased by 0.5 after 100 and 200 epochs, and a batch of 1024 samples, containing randomly cropped and randomly horizontally flipped images. The other models were trained by following the original papers, leveraging a multi-margin loss function with a margin $m = \varepsilon\sqrt{2}$, with $\varepsilon = 0.5$. For a fair comparison, UGNN was trained with a margin $m = \varepsilon$, being the lower bound $\mathcal{M}_f$ of the MAP for UGNN different from the $\mathcal{M}_f/\sqrt{2}$ of the other DNNs, as discussed in Observation 5. For the experiments, we used 4 Nvidia Tesla-V100 with cuda 10.1 and PyTorch 1.8 (Paszke et al. 2019).

Accuracy Analysis

Table 2 summarizes the accuracy on the testset, where UGNN was tested with the (Abs, MaxMin, OPLU) activation, and the last UPD layers (bounded and unbounded). The other models were tested with the original configuration and with the abs activation. Experiments were performed with and without standard normalization (Std.Norm) of the input, and each configuration was trained four times with randomly initialized weights to obtain statically sound results. In summary, the take-away messages of the Tab. 2 are: (i) The unbounded UPD layer (named updU) increased the performance with respect to the bounded one (named updB) in almost all cases. (ii) Std.Norm pre-processing significantly increased the performance. We believe this is due to the GNP property of the layers, which cannot learn a channel re-scaling different from ± 1. (iii) The use of abs activations in the 1-Lipschitz models does not cause a significant performance loss with respect to the other GNP activations (that requires a more expensive sorting). (iv) Despite the strict constraints of the UGNN architecture, it achieves comparable performance in the raw case, while there is a clear gap of accuracy for the Std.Norm case.

Input Size	Last Layer	Accuracy [%] Std.Norm	Raw
64	updB	72.1 $\pm$ 0.27	72.4 $\pm$ 0.42
	updU	72.6 $\pm$ 0.69	72.8 $\pm$ 0.61
128	updB	74.5 $\pm$ 0.56	75.9 $\pm$ 0.07
	updU	74.9 $\pm$ 0.45	76.2 $\pm$ 0.30
256	updB	76.5 $\pm$ 0.35	78.4 $\pm$ 0.29
	updU	**76.8** $\pm$ 0.29	**78.5** $\pm$ 0.22

Table 3: Accuracy comparison of the UGNN models with different, pre-processed input sizes and output layers.

To improve the UGNN accuracy, we investigated for intrinsic learning characteristics of its architecture. In particular, we noted that a strong limitation of the model is in the last two GNP blocks (see Fig. 2), which process tensor with a high number of channels (thus higher learning capabilities) but with compressed spatial dimensions ($H/8$ and $H/16$). Hence, for small input images (e.g., 32×32), such layers cannot exploit the spatial capability of convolutions. Table 3 reports a performance evaluation of the UGNN (with MaxMin activation) for larger input sizes. Note that, differently from the UGNN, common DNNs do not benefit of an up-scaling image transformation, since it is possible to apply any number of channels on the first convolutions layers. Moreover, the compared models do not have adaptive layers, hence, do not handle different input sizes. This observation allows the UGNN to outperform the other models for the raw case and reach similar accuracy for the Std.Norm case.

MAP Estimation

This section evaluates the MAP estimation through the lower bound (LB) given by the UGNN discussed in Observation 2 and the other 1-Lipschitz models. Figure 3 compares the ratio of the LB and the MAP between the 1-Lipschitz DDNs and a UGNN with MaxMin and bounded upd, for the normalized inputs. The MAP is computed with the expensive Iterative Penalty procedure, as done in (Brau et al. 2023). Note that our analysis considers the worst-case MAP, i.e., without *box-constraints*, as also done by the compared 1-Lipschitz models. Indeed, since image pixels are bounded in $[0, 1]$, the MAP is itself a lower bound of the distance from the closest adversarial image. Table 4 reports

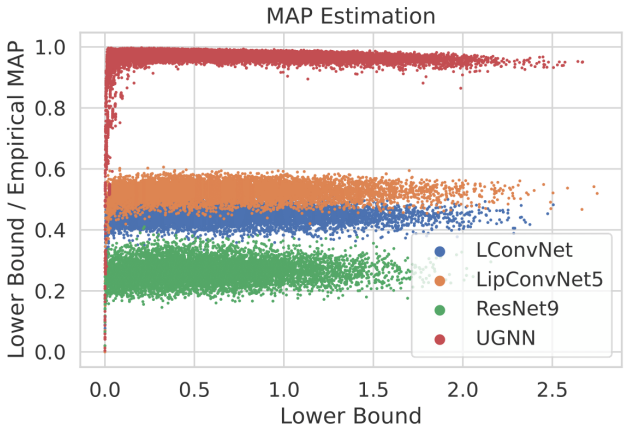

Figure 3: Evaluation of the lower-bound estimation of the MAP provided by the 1-Lipschitz DNNs and the UGNN. The y-axis reports the ratio of the given lower bound and the MAP computed through an iterative-penalty algorithm.

Model	LB/MAP	#N	BC
ResNet9 (raw)	.34±.063	6669	✓
LargeConvNet (raw)	.46±.057	7219	✓
LipConvNet5 (raw)	.58±.069	6911	✓
UGNN+OPLU+updU (raw)	**.70±.090**	7125	✓
UGNN+OPLU+updB (raw)	**.70±.093**	7098	✓
UGNN+MaxMin+updB (raw)	**.71±.087**	7114	✓
UGNN+MaxMin+updU (raw)	**.71±.088**	7118	✓
ResNet9 (norm)	.26±.036	7904	✗
LargeConvNet (norm)	.44±.027	7933	✗
LipConvNet5 (norm)	.52±.031	7840	✗
UGNN+OPLU+updB (norm)	**.96±.046**	7215	✗
UGNN+OPLU+updU (norm)	**.96±.047**	7282	✗
UGNN+MaxMin+updU (norm)	**.96±.051**	7316	✗
UGNN+MaxMin+updB (norm)	**.96±.044**	7327	✗

Table 4: Evaluation of the LB/MAP ratio deduced by the output of the models with/without Box Constraint.

statistics related to the LB/MAP ratio for different UGNNs, where the box-constrained (BC) MAPs were computed using the Decoupling Direction Norm strategy (Rony et al. 2019). The column #N contains the number of samples correctly classified by the model and for which the MAP algorithm reached convergence. Note that, in all the tested cases, the LB provided by the UGNN resulted to be tighter than the other 1-Lipschitz DNNs. Similar considerations hold for other model configurations (see Appendix).

Certifiable Robust Classification

Figure 4 shows a close comparison of the accuracy of the (certifiable) ε-robust classifications for different values of ε, i.e., the percentage of correctly classified samples with a LB lower than ε. We selected the UGNN with the highest accuracy (MaxMin-updB-256-raw). The tests for the 32x32 input size are provided in Appendix. The 1-Lipschitz models were trained on raw inputs, where the best run has been selected.

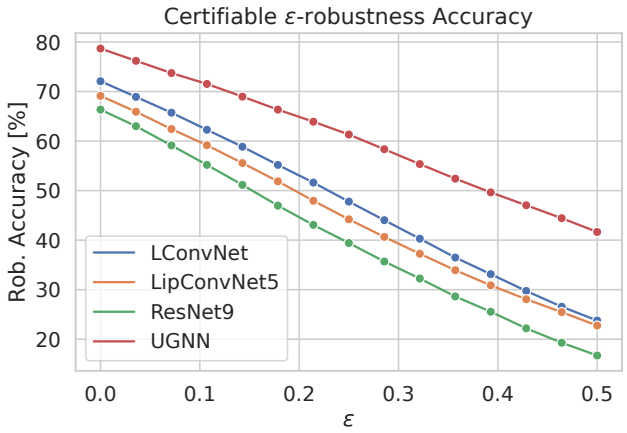

Figure 4: Accuracy of the certifiable ϵ-robust classifications.

For these models, to handle 256x256 images, an initial nearest interpolation from 256 to 32 is applied. This transformation is necessary since, differently from the UGNN, they are not adaptive to different input sizes. Note that the interpolation preserves both the accuracy and the 1-Lipschitz property. As shown in Fig. 4, the UGNN outperforms the other models for all the tested ε values.

Conclusion

This paper presented a novel family of classifiers, named *Signed-Distance Classifiers* (SDCs), which provides the *minimal adversarial perturbation* (MAP) by just computing the difference between the two highest output components, thus offering an online-certifiable prediction.

To practically implement an SDC, we developed a novel architecture, named *Unitary-Gradient Neural Network* (UGNN), which satisfies (almost-everywhere) the characterization property of an SDC. To design this model, we proposed a new fully-connected layer, named Unitary Pair Difference (UPD), which features unbounded weight matrix while preserving the unitary-gradient property.

Several experiments were conducted to compare the proposed architecture with the most related certifiable 1-Lipschitz models from previous work. The experiments highlighted the performance of the UGNN in terms of accuracy, certifiable robustness, and estimation of the MAP, showing promising results.

Future work will focus on improving the UGNN. Furthermore, as pointed out by other authors, additional investigations are needed to tackle practical open problems in this field, such as addressing dataset with many classes and improving learning strategies.

References

Anil, C.; Lucas, J.; and Grosse, R. 2019. Sorting Out Lipschitz Function Approximation. In *Proceedings of the 36th International Conference on Machine Learning*, 291–301. PMLR.

Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. *Evasion Attacks*

against Machine Learning at Test Time, volume 7908 of *Lecture Notes in Computer Science*, 387–402. Springer Berlin Heidelberg. ISBN 978-3-642-38708-1.

Biggio, B.; and Roli, F. 2018. Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, 2154–2156. New York, NY, USA: Association for Computing Machinery. ISBN 9781450356930.

Boczko, E. M.; and Young, T. R. 2005. The signed distance function: a new tool for binary classification. *arXiv preprint cs/0511105*.

Brau, F.; Rossolini, G.; Biondi, A.; and Buttazzo, G. 2023. On the Minimal Adversarial Perturbation for Deep Neural Networks With Provable Estimation Error. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4): 5038–5052.

Béthune, L.; González-Sanz, A.; Mamalet, F.; and Serrurier, M. 2021. The Many Faces of 1-Lipschitz Neural Networks. *arXiv:2104.05097 [cs, stat]*. ArXiv: 2104.05097.

Carlini, N.; Katz, G.; Barrett, C.; and Dill, D. L. 2018. Provably Minimally-Distorted Adversarial Examples. *arXiv:1709.10207 [cs]*. ArXiv: 1709.10207.

Carlini, N.; and Wagner, D. 2017. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, 39–57.

Chernodub, A.; and Nowicki, D. 2017. Norm-preserving Orthogonal Permutation Linear Unit Activation Functions (OPLU). arXiv:1604.02313.

Cisse, M.; Bojanowski, P.; Grave, E.; Dauphin, Y.; and Usunier, N. 2017. Parseval Networks: Improving Robustness to Adversarial Examples. *arXiv:1704.08847 [cs, stat]*. ArXiv: 1704.08847.

Cohen, J.; Rosenfeld, E.; and Kolter, Z. 2019. Certified Adversarial Robustness via Randomized Smoothing. In *Proceedings of the 36th International Conference on Machine Learning*, 1310–1320. PMLR.

Federer, H. 1959. Curvature measures. *Transactions of the American Mathematical Society*, 93(3): 418–491.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]*. ArXiv: 1412.6572.

Hein, M.; and Andriushchenko, M. 2017. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Huang, Y.; Zhang, H.; Shi, Y.; Kolter, J. Z.; and Anandkumar, A. 2021. Training certifiably robust neural networks with efficient local lipschitz bounds. *Advances in Neural Information Processing Systems*, 34: 22745–22757.

Kingma, D. P.; and Ba, J. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*.

Lang, S. 2012. *Fundamentals of differential geometry*, volume 191. Springer Science & Business Media.

Li, Q.; Haque, S.; Anil, C.; Lucas, J.; Grosse, R. B.; and Jacobsen, J.-H. 2019. Preventing Gradient Attenuation in Lipschitz Constrained Convolutional Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Li, S.; Jia, K.; Wen, Y.; Liu, T.; and Tao, D. 2021. Orthogonal Deep Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(4): 1352–1368.

Liu, D. C.; and Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1): 503–528.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2019. Towards Deep Learning Models Resistant to Adversarial Attacks. (arXiv:1706.06083). ArXiv:1706.06083 [cs, stat].

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral Normalization for Generative Adversarial Networks. *arXiv:1802.05957 [cs, stat]*. ArXiv: 1802.05957.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2574–2582.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Pintor, M.; Roli, F.; Brendel, W.; and Biggio, B. 2021. Fast minimum-norm adversarial attacks through adaptive norm constraints. *Advances in Neural Information Processing Systems*, 34: 20052–20062.

Rony, J.; Granger, E.; Pedersoli, M.; and Ayed, I. B. 2020. Augmented Lagrangian Adversarial Attacks. *arXiv:2011.11857 [cs]*. ArXiv: 2011.11857.

Rony, J.; Hafemann, L. G.; Oliveira, L. S.; Ayed, I. B.; Sabourin, R.; and Granger, E. 2019. Decoupling Direction and Norm for Efficient Gradient-Based L2 Adversarial Attacks and Defenses. *arXiv:1811.09600 [cs]*. ArXiv: 1811.09600.

Rossolini, G.; Biondi, A.; and Buttazzo, G. 2022. Increasing the Confidence of Deep Neural Networks by Coverage Analysis. *IEEE Transactions on Software Engineering*.

Sakai, T. 1996. On Riemannian manifolds admitting a function whose gradient is of constant norm. *Kodai Mathematical Journal*, 19(1).

Schölkopf, B.; Smola, A. J.; Bach, F.; et al. 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Serrurier, M.; Mamalet, F.; González-Sanz, A.; Boissin, T.; Loubes, J.-M.; and del Barrio, E. 2021. Achieving robustness in classification using optimal transport with hinge regularization. *arXiv:2006.06520 [cs, stat]*. ArXiv: 2006.06520.

Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A. P.; Bishop, R.; Rueckert, D.; and Wang, Z. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1874–1883.

Singla, S.; and Feizi, S. 2021. Skew Orthogonal Convolutions. In *Proceedings of the 38th International Conference on Machine Learning*, 9756–9766. PMLR.

Singla, S.; Singla, S.; and Feizi, S. 2021. Improved deterministic l2 robustness on CIFAR-10 and CIFAR-100. In *International Conference on Learning Representations*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Trockman, A.; and Kolter, J. Z. 2021. Orthogonalizing Convolutional Layers with the Cayley Transform. *arXiv:2104.07167 [cs, stat]*. ArXiv: 2104.07167.

Tsuzuku, Y.; Sato, I.; and Sugiyama, M. 2018. Lipschitz-Margin Training: Scalable Certification of Perturbation Invariance for Deep Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Wang, J.; Chen, Y.; Chakraborty, R.; and Yu, S. X. 2020. Orthogonal Convolutional Neural Networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11502–11512. IEEE. ISBN 978-1-72817-168-5.

Weng, T.-W.; Zhang, H.; Chen, P.-Y.; Yi, J.; Su, D.; Gao, Y.; Hsieh, C.-J.; and Daniel, L. 2018. Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach. *arXiv:1801.10578 [cs, stat]*. ArXiv: 1801.10578.

Wong, E.; Schmidt, F. R.; Metzen, J. H.; and Kolter, J. Z. 2018. Scaling provable adversarial defenses. *arXiv:1805.12514 [cs, math, stat]*. ArXiv: 1805.12514.