Robustness Certificates Against Adversarial Examples for ReLU Networks

Preprint · February 2019

CITATIONS
0

READS
29

2 authors:

Sahil Singla
University of Maryland, College Park
13 PUBLICATIONS **35** CITATIONS
SEE PROFILE

Soheil Feizi
Massachusetts Institute of Technology
106 PUBLICATIONS **5,103** CITATIONS
SEE PROFILE

Robustness Certificates Against Adversarial Examples for ReLU Networks

Sahil Singla [1] **Soheil Feizi** [1]

Abstract

While neural networks have achieved high performance in different learning tasks, their accuracy drops significantly in the presence of small adversarial perturbations to inputs. Defenses based on regularization and adversarial training are often followed by new attacks to defeat them. In this paper, we propose attack-agnostic robustness certificates for a multi-label classification problem using a deep ReLU network. Although computing the exact distance of a given input sample to the classification decision boundary requires solving a non-convex optimization, we characterize two lower bounds for such distances, namely the simplex certificate and the decision boundary certificate. These robustness certificates leverage the piece-wise linear structure of ReLU networks and use the fact that in a polyhedron around a given sample, the prediction function is linear. In particular, the proposed simplex certificate has a closed-form, is differentiable and is an order of magnitude faster to compute than the existing methods even for deep networks. In addition to theoretical bounds, we provide numerical results for our certificates over MNIST and compare them with some existing upper bounds.

1. Introduction

Although neural network models have achieved state-of-the-art results on several learning tasks, in the last couple of years, researchers have demonstrated their lack of robustness with respect to adversarial perturbations. For example, in image classification, adversarial examples have been crafted to mislead the classifier while being visually indistinguishable from *normal* examples (Goodfellow et al., 2014; Szegedy et al., 2013; Shafahi et al., 2018).

In the last couple of years, a pattern has been emerged that defense mechanisms against existing attacks are often followed by stronger attacks to break them. Even detecting the presence of adversarial examples in a dataset seems to be difficult (Hendrik Metzen et al., 2017; Carlini & Wagner, 2017). Moreover, different references have shown that adversarial examples can exist in the physical world as well (Sharif et al., 2016; Kurakin et al., 2016a; Eykholt et al., 2017). This can be a significant issue in deploying neural networks in applications such as self-driving cars, authentication systems, malware detection etc.

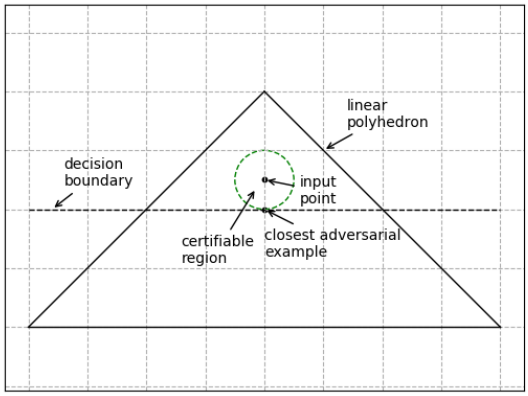
Studying adversarial examples for neural networks has twofold purposes: (i) devising stronger attack algorithms for crafting adversarial examples that can break the existing defense mechanisms, and (ii) developing defenses and evaluating their robustness to adversarial perturbations. In theory, the evaluation of a neural networks robustness should be *agnostic* to the attack methods. However, existing methods use the distortions obtained by different attacks as an empirical robustness measure of a target neural network. As highlighted by (Goodfellow, 2018), attack based methodology provides merely an *upper bound* on the size of perturbation needed to fool the prediction model while security guarantees require a *lower bound* on the size of the adversarial perturbation.

The robustness evaluation based on such attack approaches can cause biases in the analysis. For example, adversarial training retrains the network by adding crafted adversarial examples using some attack methods to the training set. Although a network trained using adversarial training can be robust to the attack used to craft the adversarial examples, it can be susceptible to other types of attacks (Athalye et al., 2018; Athalye & Sutskever, 2017; Carlini & Wagner, 2016).

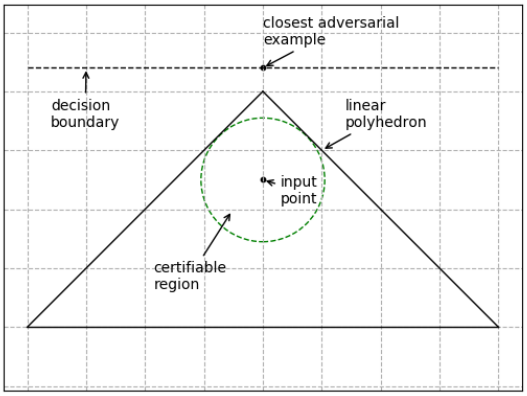
In this work, we propose attack-agnostic certificates of robustness for a multi-label classification problem using a deep ReLU network. Our certificates leverage the piece-wise linear structure of deep ReLU networks and use the fact that the prediction function is linear in a polyhedron around a given sample. The key advantage of our certificate compared to other existing lower bounds (e.g. (Zhang et al., 2018; Weng et al., 2018a)) is its extremely efficient computation even for very deep networks. Below we explain the key ideas of our proposed certificates.

For a ReLU network, we define an *activation pattern* θ that represents whether or not each neuron is active (on or off) in

[1]University of Maryland, College Park. Correspondence to: Sahil Singla <ssingla@cs.umd.edu>, Soheil Feizi <sfeizi@cs.umd.edu>.



(a) Closest adversarial example lies inside the linear region



(b) Closest adversarial example lies outside the linear region

Figure 1. In our robustness certificates, we exploit the piecewise linear structure of ReLU networks, i.e. the prediction function is linear in some convex region (polyhedron) around an input point. The closest adversarial example, then, either lies inside or outside of this region. In case (a), we use the linear function to get the certificate while in case (b), we use the boundaries of the linear region to obtain a robustness certificate. In both cases, the radius of the green dotted circle gives the proposed *Simplex* certificate.

the network. We show that for a given activation pattern θ, the region in which all inputs induce that pattern on network ReLUs forms a polyhedron. We refer to this convex region by $S(\theta)$ and show that for $\mathbf{x} \in S(\theta)$, the neural network is a linear function $d_\theta(.)$ where $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$. In Section 3, we explain how to efficiently compute $S(\theta)$ and $d_\theta(.)$ for a deep ReLU network. For a class k, we say that $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ defines $K-1$ decision boundaries where K is the total number of classes. That is,

$$(\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{x} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)} = 0$$

is a decision boundary for all $j \neq k$. $\mathbf{W}_j^{(\theta)}$ is the j^{th} row of $\mathbf{W}^{(\theta)}$ and $\mathbf{b}_j^{(\theta)}$ is the scalar in the j^{th} position of $\mathbf{b}^{(\theta)}$.

For example, consider a two-layer neural network for the binary classification task where the input dimension is $D = 2$ and the number of neurons in the hidden layer is $N_1 = 3$. In this case, the activation pattern θ is a binary vector of length three ($\theta \in \{0,1\}^3$.) Figure 1 represents an example polyhedron defined by θ (in this case, the polyhedron is a triangle since we have three affine constraints).

Exploiting this geometric structure of the ReLU network, we characterize two lower bounds for the distance of a given input to the closest adversarial example:

- **The Simplex Certificate:** This certificate applies to a ReLU network with any arbitrary depth and for a multi-label classification problem. For a given input $\mathbf{u}$ (and therefore an activation pattern θ and a predicted class k), let $S(\theta) = \{\mathbf{x} : \mathbf{P}^{(\theta)}\mathbf{x} + \mathbf{q}^{(\theta)} \geq 0\}$ and $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$.

  We consider two cases: The first case is when the closest adversarial example lies inside $S(\theta)$ (e.g. see Figure 1a). In this case, the lower bound to the closest adversarial example is given by the minimum distance of $\mathbf{u}$ to all $K-1$ decision boundaries defined by $d_\theta(.)$:

$$\min_{j \neq k} \frac{|\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + (\mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)})\ |}{\|(\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\|_2}$$

  In this case, the exact distance can be obtained by solving a linear program for every $j \neq k$. However, the above closed-form formula provides a lower bound which is very efficient to compute even for deep networks.

  The second case is where the closest adversarial example does not lie inside $S(\theta)$ (e.g. see Figure 1b). In this case, a lower bound to the closest adversarial example can be characterized as the distance of the point to the closest face of the polyhedron $S(\theta)$ as follows:

$$\min_i \frac{|\ \mathbf{P}_i^{(\theta)}\mathbf{u} + \mathbf{q}_i^{(\theta)}\ |}{\|\mathbf{P}_i^{(\theta)}\|_2}.$$

  The minimum of these two quantities act as our *Simplex certificate*. We present details of this certificate in Section 4.1.

- **The Decision Boundary Certificate:** Unlike the simplex certificate which can be used for a ReLU network for an arbitrary depth, this certificate applies to a two-layer ReLU network.

For a given input $\mathbf{u}$ with a predicted class k, we can write the certificate in terms of the minimum distance to all possible decision boundaries as follows:

$$\min_{j \neq k} \min_{\theta^{(1)} \in \{0,1\}^{N_1}} \frac{| (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)} |}{\|\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)}\|_2} \tag{1}$$

where $\theta^{(1)}$ denotes the indicator vector for the activations of the first hidden layer and N_1 is the number of hidden units in the first hidden layer.
We show that $\|\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)}\|_2 \leq \|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2$ and relax $\theta^{(1)} \in \{0,1\}^{N_1}$ to $\theta^{(1)} \in [0,1]^{N_1}$. This leads to the following lower bound:

$$\min_{j \neq k} \frac{\min_{\theta^{(1)} \in [0,1]^{N_1}} | (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)} |}{\|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2}.$$

For a two-layer network, we show that the numerator is linear in $\theta^{(1)}$. Thus, the above optimization can be solved using convex optimization. For deeper networks, the numerator term can be a higher-order polynomial in θ making the optimization difficult to solve. Thus, for the decision boundary certificate, we only focus on two-layer neural networks. We discuss details of this certificate in Section 4.2.

In what follows, we explain details of these results. All proofs have been presented in Appendix.

2. Background and related work

The adversarial attacks and defenses have received significant attention from the machine learning community in the last couple of years (Szegedy et al., 2013; Uesato et al., 2018; Goodfellow et al., 2014; Athalye et al., 2018; Athalye & Sutskever, 2017; Buckman et al., 2018; Kurakin et al., 2016b; Papernot et al., 2016; Zantedeschi et al., 2017; Papernot et al., 2016; Tramèr et al., 2017; Carlini & Wagner, 2016a; 2017; Kannan et al., 2018; Madry et al., 2017; Eykholt et al., 2017). A wide range of defenses have been proposed to harden neural networks against adversarial attacks. However, a pattern has emerged in which the majority of adversarial defenses are broken by new attacks. For example,(Carlini & Wagner, 2016), (Athalye et al., 2018), (Athalye & Sutskever, 2017), (Uesato et al., 2018) broke several of these proposed defenses.

2.1. Creating Adversarial Examples

The current state-of-the-art white-box attack methods are the iterative fast gradient sign method (I-FGSM) (Goodfellow et al., 2014; Kurakin et al., 2016b), DeepFool (Moosavi-Dezfooli et al., 2016), Carlini and Wagners attack (CW attack) (Carlini & Wagner, 2016), elastic-net attacks to deep neural networks (Chen et al., 2017a), robust physical perturbations attack (Eykholt et al., 2017), EOT attack (Athalye et al., 2017). In the white-box attacks, the network parameters are assumed to be visible to the attacker. Black-box adversarial attacks are also possible by computing universal adversarial perturbations (Moosavi-Dezfooli et al., 2016), using ensemble approaches (Liu et al., 2017), using substitute models (Papernot et al., 2016; Ilyas et al., 2018a), employing zero-order optimization-based attacks (Chen et al., 2017b; Ilyas et al., 2018b).

2.2. Certifiable Defenses

A number of "certifiable" defense mechanisms have been developed for certain classifiers. (Raghunathan et al., 2018) harden a two-layer classifier using semidefinite programming, and (Sinha et al., 2018) proposes a convex duality-based approach to adversarial training that works on sufficiently small adversarial perturbations with a quadratic adversarial loss, while (Kolter & Wong, 2017) considers training a robust classifier using the convex outer adversarial polytope. (Gowal et al., 2018) shows how applying interval bound propagation during training, combined with MILP-based exact verification, can lead to provably robust networks. These provable defenses, although very insightful, are either restrictive or are computationally expensive.

2.3. Theoretical Robustness Guarantees against Adversarial Examples

In recent years, formal verification methods were developed to verify robustness of neural networks. Most of these methods use satisfiability modulo theory (SMT) solvers (Ehlers, 2017; Katz et al., 2017; Carlini & Wagner, 2016b) or Large scale Mixed integer Linear Programming (MILP) solvers (Cheng et al., 2017; Lomuscio & Maganti, 2017; Tjeng et al., 2017). However, these methods scale poorly with the number of ReLUs in a network, making them prohibitively slow in practice even for medium-sized models. (Katz et al., 2017) has illustrated the difficulty of exact verification by proving that it is NP-complete.

Some methods computes robustness certificates by computing the Lipschitz constants. For example, (Szegedy et al., 2013) evaluates the Lipschitz constant for each layer of the network and uses the product of these Lipschitz constants to demonstrate the robustness issue in neural networks. Reference (Hein & Andriushchenko, 2017) derives a closed-form robustness certificate using a local Lipschitz continuous condition for a single-hidden layer feed-forward network. However, a closed-form lower bound for an arbitrary depth multi-layer-perceptron (similar to our simplex certificate) seems to be difficult to compute. (Weng et al., 2018b) computes a characterization of the lower bound for distances to

the closest adversarial examples. However, as highlighted by (Goodfellow, 2018), their proposed method computes an empirical estimate of this theoretical lower bound and hence is not a robustness certificate in general. (Zhang et al., 2018; Weng et al., 2018a) compute a theoretical lower bound but it can be expensive to compute for large networks.

An *upper bound* to the distance of a point to its closest adversarial example can be computed using a certain attack (Bastani et al., 2016). This differs from our proposed robustness certificates because our certificate is a lower bound on the minimum distortion and is attack agnostic. Additionally, our simplex certificate is differentiable and can be computed in closed form for a given input for any deep multi-layer perceptron with ReLU activations. We show that simplex certificate is an order of magnitude faster to compute than the state of the art methods (Zhang et al., 2018; Weng et al., 2018a).

3. Piecewise Linear Structures of ReLU Networks

In this section, we provide notation and definitions that will be used in characterizing our robustness certificates. To simplify the exposition, the approaches are developed under the notation of fully connected networks with ReLU activations for a multi-label classification problem.

3.1. Notation

We consider a neural network with M layers and N_i neurons in the i^{th} layer ($M \geq 2$ and $i \in [M]$) for a multi-class classification problem. Let N be the total number of hidden neurons in the network. The number of classes is equal to K (or N_M). The corresponding function of neural network is $f : \mathbf{R}^D \to \mathbf{R}^K$ where D is the dimension of the input.

We use $\mathbf{x}$ to represent an input instance in $\mathbf{R}^D$. We use $[L]$ to denote the set $\{1, \ldots, L\}$. For an input $\mathbf{x}$, we use $\mathbf{z}^{(i)}(\mathbf{x}) \in \mathbf{R}^{N_i}$ and $\mathbf{a}^{(i)}(\mathbf{x}) \in \mathbf{R}^{N_i}$ to denote the input (*before* applying ReLU activations) and output (*after* applying ReLU activations) of neurons in the i-th hidden layer of the network, respectively. For simplicity, we refer to $\mathbf{z}^{(i)}(\mathbf{x})$ and $\mathbf{a}^{(i)}(\mathbf{x})$ as raw and activated neurons in the i-th layer, respectively. The raw and activated neurons in the j-th position of the i-th hidden layer are given by $\mathbf{z}_j^{(i)}(\mathbf{x})$ and $\mathbf{a}_j^{(i)}(\mathbf{x})$ respectively. To simplify notation and when no confusion arises, we make the dependency of $\mathbf{z}^{(i)}$ and $\mathbf{a}^{(i)}$ to $\mathbf{x}$ implicit. We define $\mathbf{a}^{(0)}(\mathbf{x}) = \mathbf{x}$ and $N_0 = D$.

With a fully connected architecture and ReLU activations, each $\mathbf{z}^{(i)}$ and $\mathbf{a}^{(i)}$ (for $i \in [M]$) is computed using a transformation matrix $\mathbf{W}^{(i)} \in R^{N_i \times N_{i-1}}$ and the bias vector $\mathbf{b}^{(i)} \in R^{N_i}$ as follows:

$$\mathbf{z}^{(i)}(\mathbf{x}) = \mathbf{W}^{(i)}\mathbf{a}^{(i-1)}(\mathbf{x}) + \mathbf{b}^{(i)}$$
$$\mathbf{a}^{(i)}(\mathbf{x}) = \text{ReLU}\left(\mathbf{z}^{(i)}(\mathbf{x})\right) = \max\left(0, \mathbf{z}^{(i)}(\mathbf{x})\right)$$
$$\mathbf{a}^{(M)}(\mathbf{x}) = \mathbf{z}^{(M)}(\mathbf{x}).$$

The weight and bias vectors for the j-th row of $\mathbf{W}^{(i)}$ and $\mathbf{b}^{(i)}$ are given by $\mathbf{W}_j^{(i)}$ and $\mathbf{b}_j^{(i)}$, respectively. For a given input $\mathbf{x}$, the vector of logits is given by,

$$f(\mathbf{x}) = \mathbf{z}^{(M)}(\mathbf{x}).$$

We use $f_i(\mathbf{x})$ to denote the logit for the class i where $i \in [K]$. The predicted class is given by

$$\hat{\mathbf{y}}(\mathbf{x}) = \underset{i \in [K]}{\text{argmax}}\ f_i(\mathbf{x}).$$

We use $\odot$ to denote the Hadamard Product. We use $\mathbf{A}_i$ to denote the i^{th} row of the matrix $\mathbf{A}$. For a vector $\mathbf{v}$, we use $\mathbf{v}_i$ to denote the element in the i^{th} position of the vector. We use $\text{diag}(\mathbf{v})$ to denote the diagonal matrix formed by placing each element of $\mathbf{v}$ along the diagonal.

Definition 1 (Activation pattern)**.** *An activation pattern θ is a set of indicator vectors for each hidden layer of the network. The indicator vector for i^{th} hidden layer ($i \in [M-1]$) is denoted by $\theta^{(i)}$ and specifies the following functional constraints:*

$$\theta_j^{(i)} = \begin{cases} 1, & \mathbf{z}_j^{(i)}(\mathbf{x}) \geq 0 \\ 0, & \mathbf{z}_j^{(i)}(\mathbf{x}) < 0 \end{cases}$$

We denote θ as $\theta = \{\theta^{(1)}, \ldots, \theta^{(M-1)}\}$.

We say that an input $\mathbf{x}$ induces an activation pattern θ in f if the activation pattern defined by the neurons $\mathbf{z}_j^i(\mathbf{x})$ is equal to θ. Informally, an activation pattern θ represents a configuration of all ReLUs in the network as either being "on" or "off".

Definition 2 (Activation Region)**.** *For a given activation pattern θ, we define the activation region $S(\theta)$ such that:*

$$\mathbf{x} \textit{ induces } \theta \textit{ in } f \iff \mathbf{x} \in S(\theta)$$

Thus the activation region for an activation pattern θ, is the largest region such that all inputs in the region induce the activation pattern θ.

Definition 3 (Decision function)**.** *For a given activation pattern θ, we define the decision function $d_\theta : \mathbf{R}^d \to \mathbf{R}^K$ such that for every $\mathbf{x}$ that induces the activation pattern θ:*

$$d_\theta(\mathbf{x}) = f(\mathbf{x}), \quad \forall\, \mathbf{x} \in S(\theta).$$

The decision function is the function such that for all inputs that induce a certain activation pattern, the neural network and the decision function are the same.

3.2. Polyhedral Structures of ReLU Networks

Because ReLU networks are piece-wise linear functions, they are linear in some region around a given input. In this section, we prove that for an arbitrarily deep ReLU network, the activation region for an activation pattern is a convex polyhedron and in this region, the neural network is linear. Furthermore, we derive the exact polyhedron in which the activation pattern is constant.

We first explain our results for a two-layer neural network and then present our them for a neural network with an arbitrary depth. For a two-layer ReLU networks, the activation pattern θ is merely composed of one vector $\theta^{(1)}$ (since $M = 2$ in this case). In this case, we have:

Theorem 1. *Given an activation pattern $\theta = \{\theta^{(1)}\}$ for a two-layer network, we have:*

(a) The activation region is $S(\theta)$ where

$$S(\theta) = \cap_{i=1}^{i=N_1} S_i^{(1)}(\theta)\,,$$
$$S_i^1(\theta) = \begin{cases} \mathbf{x} : \mathbf{W}_i^{(1)}\mathbf{x} + \mathbf{b}_i^{(1)} \geq 0 & \text{if } \theta_i^{(1)} = 1 \\ \mathbf{x} : \mathbf{W}_i^{(1)}\mathbf{x} + \mathbf{b}_i^{(1)} < 0 & \text{if } \theta_i^{(1)} = 0 \end{cases}$$

(b) The decision function is $d_\theta(.)$ where

$$d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)},$$
$$\mathbf{W}^{(\theta)} = \mathbf{W}^{(2)}\text{diag}(\theta^{(1)})\mathbf{W}^{(1)}\,,$$
$$\mathbf{b}^{(\theta)} = \mathbf{W}^{(2)}\text{diag}(\theta^{(1)})\mathbf{b}^{(1)} + \mathbf{b}^{(2)}$$

Note that in this case $S(\theta)$ is a polyhedron characterized using N_1 linear constraints (i.e. N_1 faces). At a vertex, D linearly independent constraints are tight (i.e. equal to zero). In total, $\binom{N1}{D}$ combinations are possible. But all the $\binom{N1}{D}$ combinations may not have D linearly independent equations and some of them may not have satisfy the other inequality constraints of the polyhedron.

A similar result can be stated for a ReLU network with an arbitrary depth (although the notation is a bit more complex than that of the two-layer case).

We present this result in the following theorem:

Theorem 2. *Consider an M layer ($M \geq 3$) neural network (denoted by $f(.)$) and an activation pattern $\theta = \{\theta^{(1)}, \ldots, \theta^{(M-1)}\}$. Consider the $M-1$ layer neural network (denoted by $g(.)$) constructed by removing the last weight layer and the last activation layer such that*

$$g(\mathbf{x}) = \mathbf{z}^{(M-1)}(\mathbf{x}) \quad \forall\, \mathbf{x} \in \mathbf{R}^D$$

and the activation pattern $\phi = \{\theta^{(1)}, \ldots, \theta^{(M-2)}\}$. Given the decision function $d_\phi(\mathbf{x}) = \mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)}$ and the activation region $T(\phi)$ for g(.), (a) the activation region for $f(.)$ is $S(\theta)$ where:

$$S(\theta) = T(\phi) \cap S^{(M-1)}(\theta) \text{ where,}$$
$$S^{(M-1)}(\theta) = \cap_{i=1}^{i=N_{M-1}} S_i^{(M-1)}(\theta) \text{ and}$$
$$S_i^{(M-1)}(\theta) = \begin{cases} \mathbf{x} : \mathbf{W}_i^{(\phi)}\mathbf{x} + \mathbf{b}_i^{(\phi)} \geq 0 & \text{if } \theta_i^{(M-1)} = 1 \\ \mathbf{x} : \mathbf{W}_i^{(\phi)}\mathbf{x} + \mathbf{b}_i^{(\phi)} < 0 & \text{if } \theta_i^{(M-1)} = 0 \end{cases}$$

and (b) the decision function for $f(.)$ is $d_\theta(.)$ where:

$$d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)} \text{ where,}$$
$$\mathbf{W}^{(\theta)} = \mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{W}^{(\phi)} \text{ and}$$
$$\mathbf{b}^{(\theta)} = \mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{b}^{(\phi)} + \mathbf{b}^{(M)}$$

As a consequence of Theorem 2, the decision function at a layer is a function of the weights and biases of all layers upto (and including) the current layer, the activation region can be constructed using the decision function at the previous layer and the indicator vector of the current layer. Moreover, the number of linear constraints needed to define the activation region $S(\theta)$ for a deep ReLU network grows linearly with respect to the depth of the network:

Corollary 1. *For an M layer network and activation pattern θ, the activation region is a polyhedron with $\sum_{i=1}^{i=M-1} N_i$ inequalities and the decision function is linear in the input.*

Thus, the activation region $S(\theta)$ is a polyhedron and can be represented in terms of linear inequalities.

$$S(\theta) = \{\mathbf{x} : \mathbf{P}^{(\theta)}\mathbf{x} + \mathbf{q}^{(\theta)} \geq 0\}$$

where $\mathbf{P}^{(\theta)}$ and $\mathbf{q}^{(\theta)}$ are constant given θ, $\mathbf{P}^{(\theta)}$ is a matrix of dimensions $(\sum_{i=1}^{M-1} N_i) \times D$ and $\mathbf{q}^{(\theta)}$ is a vector of dimension $(\sum_{i=1}^{M-1} N_i)$.

We emphasize that the activation region may not be the largest linear region around the input point. Indeed the linear region could be larger than the activation region if the linear function remains the same in some adjoining region with a different activation pattern. However, the activation region is provably the largest region in which the activation pattern is the same.

Since the decision function is locally linear in $\mathbf{x}$, at the i^{th} layer (with the decision function $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)} = \mathbf{z}^{(i)}(\mathbf{x})$), we can compute the j^{th} row $\mathbf{W}_j^{(\theta)}$ using the gradient of $\mathbf{z}_j^{(i)}$ with respect to the input, i.e., $\nabla_{\mathbf{x}}\mathbf{z}_j^{(i)}$, in any autograd software. However, current autograd implementations do not support the gradient of a vector with respect to the neuron's input, i.e. $\nabla_{\mathbf{x}}\mathbf{z}^{(i)}$. Thus, to compute $\mathbf{W}^{(\theta)}$, we need to call $\nabla_{\mathbf{x}}\mathbf{z}_j^{(i)}$ once for each row of $\mathbf{W}^{(\theta)}$ (N_i times total) making the gradient based implementation expensive. To circumvent these issues, we propose an efficient iterative method to compute the activation region and decision function for a multi-layer-perceptron in Algorithm 1.

Algorithm 1
Computing Activation Region and Decision Function

Input:
Input point: $\mathbf{x}$
Network weights: $\{\mathbf{W}^{(i)}, \mathbf{b}^{(i)} \quad \forall\, i \in \{M\}\}$
Initialize:
$\mathbf{a}^{(0)} \leftarrow \mathbf{x}$, $\mathbf{C} \leftarrow \mathbf{W}^{(1)}$, $\mathbf{d} \leftarrow \mathbf{b}^{(1)}$, $\mathbf{P} \leftarrow \emptyset$, $\mathbf{q} \leftarrow \emptyset$
for $i = 1$ **to** $M - 1$ **do**
  $\mathbf{z}^{(i)} \leftarrow \mathbf{W}^{(i)}\mathbf{a}^{(i-1)} + \mathbf{b}^{(i)}$
  $\theta^{(i)} \leftarrow \mathbf{z}^{(i)} \geq 0$
  $\mathbf{a}^{(i)} \leftarrow \mathbf{z}^{(i)} \odot \theta^{(i)}$
  $\mathbf{P}^{(i)} \leftarrow \text{diag}(2\theta^{(i)} - 1)\mathbf{C}$
  $\mathbf{q}^{(i)} \leftarrow \text{diag}(2\theta^{(i)} - 1)\mathbf{d}$
  $\mathbf{P} \leftarrow$ concat($\mathbf{P}$, $\mathbf{P}^{(i)}$, axis=0)
  $\mathbf{q} \leftarrow$ concat($\mathbf{q}$, $\mathbf{q}^{(i)}$, axis=0)
  $\mathbf{C} \leftarrow \mathbf{W}^{(i+1)}\text{diag}(\theta^{(i)})\mathbf{C}$
  $\mathbf{d} \leftarrow \mathbf{W}^{(i+1)}\text{diag}(\theta^{(i)})\mathbf{d} + \mathbf{b}^{(i+1)}$
end for
return $(\mathbf{P}, \mathbf{q}), (\mathbf{C}, \mathbf{d})$

Algorithm 1 computes the activation region $S(\theta)$ and the decision function $d_\theta(.)$ efficiently for the activation pattern θ induced by an input $\mathbf{x}$ so that we can write them in terms of the returned matrices $(\mathbf{P}, \mathbf{q})$, $(\mathbf{C}, \mathbf{d})$:

$$d_\theta(\mathbf{x}) = \mathbf{C}\mathbf{x} + \mathbf{d}$$
$$S(\theta) = \{\mathbf{x} : \mathbf{P}\mathbf{x} + \mathbf{q} \geq 0\}$$

Multiplication by $\text{diag}(2\theta^{(i)} - 1)$ flips the sign of the row $\mathbf{C}_j$ if $\theta_j^{(i)}$ is zero and keeps the same sign otherwise. Similarly, multiplication by $\text{diag}(\theta^{(i)})$ zeros out the row $\mathbf{C}_j$ if $\theta_j^{(i)}$ is zero and keeps the same row otherwise. Thus Algorithm 1 can compute $S(\theta)$ and $d_\theta(.)$ efficiently in one forward pass.

4. Robustness Certificates for ReLU Networks

In this section, we present our robustness certificates (lower bounds of the distance between a point to the closest adversarial examples) for ReLU networks. The closest adversarial example to a given point $\mathbf{x}$ is defined as follows:

Definition 4 (Closest adversarial example)**.** *For a given input* $\mathbf{u}$ *with predicted class* k*, we define the closest adversarial example* $\mathbf{u}_0$ *such that: (a) its assigned label is different than* k*, i.e.*

$$f_k(\mathbf{u}_0) = f_j(\mathbf{u}_0) \geq f_i(\mathbf{u}_0) \quad j \neq k, \forall\, i \in [K] \qquad (2)$$

(b) its distance is minimum to $\mathbf{x}$ *compared to all vectors* $\mathbf{v}$ *satisfying 2, i.e.*

$$\|\mathbf{u} - \mathbf{u}_0\|_2 \leq \|\mathbf{u} - \mathbf{v}\|_2$$

4.1. The Simplex Certificate

In this section we derive a differentiable certificate against adversarial examples that can be computed for any arbitrarily deep multi-layer-perceptron. We have illustrated the key intuition behind this certificate which we refer to it as the *simplex certificate* in Figure 1. If the nearest adversarial example lies inside the polyhedron $S(\theta)$, we know the linear function characterizing the decision boundary and we can compute a lower bound on the distance to the closest adversarial example (Panel (a) in Figure 1). On the other hand, if the closest adversarial example lies outside the polyhedron $S(\theta)$, we can compute a lower bound using the minimum distance of the point $\mathbf{x}$ to the boundaries of the Polyhedron $S(\theta)$ (Panel (b) in Figure 1). This procedure leads to the following theorem:

Theorem 3. *Given a test input* $\mathbf{u}$ *with predicted class* k *and the activation pattern* θ *it induces, let* $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ *be the decision function and let* $S(\theta) = \{\mathbf{x} : \mathbf{P}^{(\theta)}\mathbf{x} + \mathbf{q}^{(\theta)} \geq 0\}$ *be the activation region. We define:*

$$P_{min} = \min_i \frac{|\ \mathbf{P}_i^{(\theta)}\mathbf{u} + \mathbf{q}_i^{(\theta)}\ |}{\|\mathbf{P}_i^{(\theta)}\|_2}$$
$$d_{min} = \min_{j \neq k} \frac{|\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + (\mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)})\ |}{\|(\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\|_2}$$

Let $\mathbf{u}_0$ *be the closest adversarial example. We have that:*

$$\|\mathbf{u} - \mathbf{u}_0\|_2 \geq \min(d_{min}, P_{min})$$

Hence, $\min(d_{min}, P_{min})$ *defines a provable lower bound to the closest adversarial example.*

Note that N_1 rows of $\mathbf{P}^{(\theta)}$ and $\mathbf{q}^{(\theta)}$ are a function of the weights and biases of the first layer, N_2 rows of $\mathbf{P}^{(\theta)}$ and $\mathbf{q}^{(\theta)}$ are a function of the weights and biases of *both* the first and the second layers and so on. Since both d_{min} and P_{min} are differentiable and can be computed in a single forward pass using Algorithm 1, we can use this certificate for training robust classifiers as well. In this work, however, we focus on just characterizing this quantity as a certificate for robustness for an arbitrarily deep pre-trained ReLU network. Training robust classifiers using a regularization based on the proposed simplex certificate can be an interesting direction for the future work.

4.2. The Decision Boundary Certificate

In this section, we derive another lower bound to the closest adversarial example. Unlike the simplex lower bound which can be used for a ReLU network for an arbitrary depth, this lower bound is characterized for a two-layer network.
Let $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ be the decision function for the activation pattern θ. Then for a given input $\mathbf{u}$ with a

predicted class k, we can write the certificate in terms of the minimum distance to all possible decision boundaries as in (1). Due to the dependence on θ in denominator and the combinatorial constraint $\theta^{(1)} \in \{0,1\}^{N_1}$, optimization (1) is difficult to solve in general. To further simplify this optimization, we first show that:

$$\begin{aligned}\|\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)}\|_2 &= \|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\text{diag}(\theta^{(1)})\mathbf{W}^{(1)}\|_2 \\ &\leq \|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2\end{aligned}$$

W relax $\theta^{(1)} \in \{0,1\}^{N_1}$ to $\theta^{(1)} \in [0,1]^{N_1}$ and solve:

$$\min_{j \neq k} \frac{\min\limits_{\theta^{(1)} \in [0,1]^{N_1}} |\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)}\ |}{\|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2}$$

Finally, we prove that for a two-layer network $\mathbf{W}_j^{(\theta)}$ and $\mathbf{b}_j^{(\theta)}$ are linear in θ and hence the above optimization can be solved efficiently. We summarize our result in the following theorem:

Theorem 4. *Given a test input* $\mathbf{u}$ *for a 2 layer neural network with predicted class k, let* $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ *be the decision function for the activation pattern* θ *where:*

$$\begin{aligned}\mathbf{W}^{(\theta)} &= \mathbf{W}^{(2)}\text{diag}(\theta^{(1)})\mathbf{W}^{(1)} \\ \mathbf{b}^{(\theta)} &= \mathbf{W}^{(2)}\text{diag}(\theta^{(1)})\mathbf{b}^{1} + \mathbf{b}^{(2)}\end{aligned}$$

Let $\mathbf{u}_0$ *be the closest adversarial example. We have that:*

$$\begin{aligned}&\|\mathbf{u} - \mathbf{u}_0\|_2 \\ &\geq \min_{j \neq k} \frac{\min\limits_{\theta^{(1)} \in [0,1]^{N_1}} |\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + (\mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)})\ |}{\|\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)}\|_2 \|\mathbf{W}^{(1)}\|_2}\end{aligned}$$

(a) RHS defines a provable lower bound to the closest adversarial example (b) $\forall\ i \in \{1,..,K\}$, $\mathbf{W}_i^{(\theta)}$, $\mathbf{b}_i^{(\theta)}$ *are linear in* $\theta^{(1)}$ *and RHS can be solved using convex optimization.*

For deeper networks, the numerator of the decision boundary certificate will be polynomial in θ making the optimization difficult to solve. Developing efficient computational approaches for this bound for deeper networks are among interesting directions for the future work. Nevertheless, in the next section and for a two-layer network, we include a comparison of this bound against the Simplex certificate.

5. Experiments

In this section, we numerically assess the performance of our proposed robustness certificates. We also compare our proposed lower bounds with some existing *upper bounds* including the Iterative FGSM (Kurakin et al., 2016a) and DeepFool (Moosavi-Dezfooli et al., 2016).

Note that DeepFool and Iterative FGSM provide upper bounds on the distance of a point to its closest adversarial example. Nevertheless, we compare our proposed lower bounds with these upper bounds (1) to provide a partial empirical validation of the correctness of our lower bounds, and (2) to assess the gap between the proposed lower bounds and existing upper bounds.

First, we compare the two proposed certificates, namely the simplex certificate (Section 4.1) and the decision boundary certificate (Section 4.2) and DeepFool and Iterative FGSM upper bounds over a two-layer ReLU network with 1024 hidden units for an MNIST-binary classification task. The network was to classify a digit as either being ≥ 5 or < 5.

Figure 2a demonstrates our numerical results in this case. We observe that in all examples, the simplex certificate significantly outperforms the decision boundary one (note that a good certificate obtains large values of the lower bound). However, it may be possible that an extension of the decision boundary certificate to deeper networks performs better than the simplex certificate in some examples. We leave exploring this direction for the future work.

Moreover, from Figure 2a, as expected, we observe that the values of simplex and decision boundary certificates are smaller than the values computed by Iterative FGSM and DeepFool upper bounds. We also observe that DeepFool provides a better upper bound compared to Iterative FGSM.

In our next experiment, we consider a deeper network than the previous case. In this case we only evaluate the performance of the simplex certificate since the decision boundary certificate is valid for two-layer networks. More specifically, we consider a three-layer MLP with ReLU activations on the MNIST dataset for digit classification. The number of hidden units were 1024 and 512 at the first and second hidden layers, respectively.

Figure 2b shows comparison between our simplex lower bound and DeepFool and Iterative FGSM upper bounds. We validate that our simplex certificate values are smaller than both upper bound values. Similar to the two-layer case, we observe that DeepFool provides a better upper bound than that of Iterative FGSM. In some examples, the differences between the DeepFool upper bound values and our simplex lower bound values are small, indicating the tightness of our simplex bounds in those cases.

One key advantage of the proposed simplex certificate is its efficient computation even for deep networks. This can allow using a simplex regularization in training robust classifiers. Table 1 shows a comparison of running times of our proposed simplex certificate and some other existing lower bounds including the Fast Lin, Fast-Lip proposed in (Weng et al., 2018a) and CROWN-Ada proposed in (Zhang et al., 2018). As shown in this table, our simplex certificate

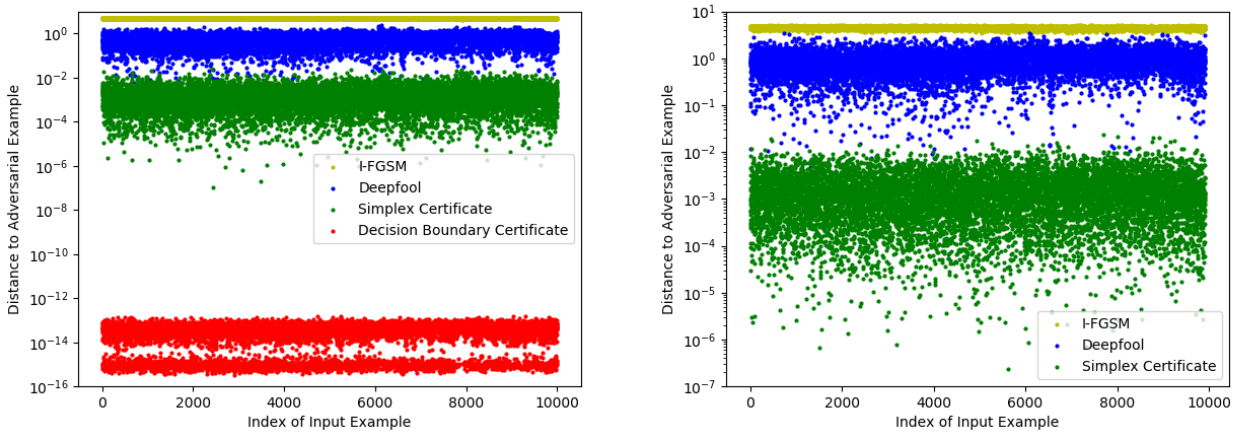


Figure 2. (a) Comparison between simplex and decision boundary lower bounds and Iterative FGSM and DeepFool upper bounds for two-layer ReLU networks. (b) Comparison between the Simplex lower bound, and Iterative FGSM and DeepFool upper bounds for three-layer ReLU networks. Note that Decision Boundary certificate cannot be used in this case.

Table 1. Running time comparison between our proposed simplex certificates and other methods. $m \times [n]$ denotes a neural network with m hidden layers each with n units. Running times are in seconds.

Config	Fast-Lin	Fast-Lip	CROWN-Ada	**Simplex**
MNIST 3×[1024]	0.732	1.060	1.260	**0.0037**
MNIST 4×[1024]	1.790	2.580	3.520	**0.0063**
CIFAR 7×[1024]	12.70	20.90	20.70	**0.0387**

is an order of magnitude more efficient to compute compared to these approaches. We note that running times for other approaches reported in Table 1 are taken directly from respective references and thus can be subject to variations due to implementation differences. A comprehensive study of different aspects of these lower bounds is an interesting direction for future work.

6. Conclusion and Future Work

In this paper, we characterized two robustness certificates for ReLU networks, namely the simplex certificate and the decision boundary certificate. Both of these certificates exploit the piecewise linear structure of ReLU network, i.e. for a given input, the decision boundary is linear in a polyhedron around that point. In particular, our simplex certificate is differentiable and is very efficient to compute even for deep networks.

A regularization based on the simplex certificate can be used for training neural networks to make them robust against adversarial examples. This can be an interesting direction for the future work. Another future work direction is to make the robustness lower bounds closer to the true values by exploiting piecewise linear structures of ReLU networks in neighboring polyhedra for a given point. Finally, extensions of our results to convolutional neural networks (CNNs) with ReLU activations can be another promising direction for the future work.

References

Athalye, A. and Sutskever, I. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing Robust Adversarial Examples. *arXiv e-prints*, July 2017.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Bastani, O., Ioannou, Y., Lampropoulos, L., Vytiniotis, D., Nori, A., and Criminisi, A. Measuring Neural Net Robustness with Constraints. *arXiv e-prints*, May 2016.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. Thermometer encoding: One hot way to resist adversarial examples. *OpenReview*, 2018.

Carlini, N. and Wagner, D. Towards Evaluating the Robustness of Neural Networks. *arXiv e-prints*, August 2016.

Carlini, N. and Wagner, D. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016a.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. *arXiv preprint arXiv:1608.04644*, 2016b.

Carlini, N. and Wagner, D. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. *arXiv e-prints*, May 2017.

Carlini, N. and Wagner, D. Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017.

Chen, P.-Y., Sharma, Y., Zhang, H., Yi, J., and Hsieh, C.-J. EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. *arXiv e-prints*, September 2017a.

Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J. ZOO: Zeroth Order Optimization based Black-box Attacks to Deep Neural Networks without Training Substitute Models. *arXiv e-prints*, August 2017b.

Cheng, C.-H., Nührenberg, G., and Ruess, H. Maximum Resilience of Artificial Neural Networks. *arXiv e-prints*, April 2017.

Ehlers, R. Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks. *arXiv e-prints*, May 2017.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. Robust Physical-World Attacks on Deep Learning Models. *arXiv e-prints*, July 2017.

Goodfellow, I. Gradient Masking Causes CLEVER to Overestimate Adversarial Perturbation Size. *arXiv e-prints*, April 2018.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. *arXiv e-prints*, December 2014.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models. *arXiv e-prints*, October 2018.

Hein, M. and Andriushchenko, M. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. *arXiv e-prints*, May 2017.

Hendrik Metzen, J., Genewein, T., Fischer, V., and Bischoff, B. On Detecting Adversarial Perturbations. *arXiv e-prints*, February 2017.

Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box Adversarial Attacks with Limited Queries and Information. *arXiv e-prints*, April 2018a.

Ilyas, A., Engstrom, L., and Madry, A. Prior Convictions: Black-Box Adversarial Attacks with Bandits and Priors. *arXiv e-prints*, July 2018b.

Kannan, H., Kurakin, A., and Goodfellow, I. Adversarial Logit Pairing. *arXiv e-prints*, March 2018.

Katz, G., Barrett, C., Dill, D., Julian, K., and Kochenderfer, M. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. *arXiv e-prints*, February 2017.

Kolter, J. Z. and Wong, E. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv e-prints*, July 2016a.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial Machine Learning at Scale. *arXiv e-prints*, November 2016b.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016a.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016b.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. Towards Robust Neural Networks via Random Self-ensemble. *arXiv e-prints*, December 2017.

Lomuscio, A. and Maganti, L. An approach to reachability analysis for feed-forward ReLU neural networks. *arXiv e-prints*, June 2017.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv e-prints*, June 2017.

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. Universal adversarial perturbations. *arXiv e-prints*, October 2016.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582, 2016.

Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Berkay Celik, Z., and Swami, A. Practical Black-Box Attacks against Machine Learning. *arXiv e-prints*, February 2016.

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597. IEEE, 2016.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Shafahi, A., Huang, W. R., Studer, C., Feizi, S., and Goldstein, T. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018.

Sharif, M., Bhagavatula, S., Bauer, L., and Reiter, M. K. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1528–1540. ACM, 2016.

Sinha, A., Namkoong, H., and Duchi, J. Certifying some distributional robustness with principled adversarial training. *OpenReview*, 2018.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv e-prints*, December 2013.

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

Tjeng, V., Xiao, K., and Tedrake, R. Evaluating Robustness of Neural Networks with Mixed Integer Programming. *arXiv e-prints*, November 2017.

Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

Uesato, J., O'Donoghue, B., van den Oord, A., and Kohli, P. Adversarial Risk and the Dangers of Evaluating Against Weak Attacks. *arXiv e-prints*, February 2018.

Weng, T.-W., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Boning, D., Dhillon, I. S., and Daniel, L. Towards Fast Computation of Certified Robustness for ReLU Networks. *arXiv e-prints*, April 2018a.

Weng, T.-W., Zhang, H., Chen, P.-Y., Yi, J., Su, D., Gao, Y., Hsieh, C.-J., and Daniel, L. Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach. *arXiv e-prints*, January 2018b.

Zantedeschi, V., Nicolae, M.-I., and Rawat, A. Efficient defenses against adversarial attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 39–49. ACM, 2017.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient Neural Network Robustness Certification with General Activation Functions. *arXiv e-prints*, November 2018.

A. Proofs

A.1. Proof of Theorem 1

(a) The activation pattern for $\mathbf{x}$ is θ if and only if:

$$\forall\, i \in [N_1],\ \begin{cases} \mathbf{W}_i^{(1)}\mathbf{x} + \mathbf{b}_i^{(1)} \geq 0 \text{ if } \theta_i^{(1)} = 1 \\ \mathbf{W}_i^{(1)}\mathbf{x} + \mathbf{b}_i^{(1)} < 0 \text{ if } \theta_i^{(1)} = 0 \end{cases}$$

Thus, $S(\theta)$ gives the activation region for θ.

$(b)\ \forall\, \mathbf{x} \in S(\theta)$,

$\mathbf{a}^{(1)}(\mathbf{x}) = \max(\mathbf{z}^{(1)}(\mathbf{x}), 0) \ = \ \theta^{(1)} \odot \mathbf{z}^{(1)}(\mathbf{x})$

Since $\mathbf{z}^{(1)}(\mathbf{x}) = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}$,

$\mathbf{a}^{(1)}(\mathbf{x}) = \theta^{(1)} \odot (\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$

$\mathbf{a}^{(1)}(\mathbf{x}) = \mathrm{diag}(\theta^{(1)})(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$

$\mathbf{a}^{(2)}(\mathbf{x}) = \mathbf{z}^{(2)}(\mathbf{x}) = \mathbf{W}^{(2)}\mathbf{a}^{(1)}(\mathbf{x}) + \mathbf{b}^{(2)}$

$\mathbf{a}^{(2)}(\mathbf{x}) = \mathbf{W}^{(2)}(\mathrm{diag}(\theta^{(1)})(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})) + \mathbf{b}^{(2)}$

$$\mathbf{a}^{(2)}(\mathbf{x}) = \underbrace{\mathbf{W}^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{W}^{(1)}}_{\text{Weight term}}\mathbf{x} + \underbrace{\mathbf{W}^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{b}^{(1)} + \mathbf{b}^{(2)}}_{\text{Bias term}}$$

$\mathbf{a}^{(2)}(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ where ,

$\mathbf{W}^{(\theta)} = \mathbf{W}^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{W}^{(1)}$ and ,

$\mathbf{b}^{(\theta)} = \mathbf{W}^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{b}^{(1)} + \mathbf{b}^{(2)}$

Thus, $d_\theta(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)}$ is the decision function for θ. Since $\mathbf{W}^{(\theta)}$ and $\mathbf{b}^{(\theta)}$ are constant for a given θ, $\mathbf{a}^{(2)}$ is linear in $\mathbf{x}$ with weights $\mathbf{W}^{(\theta)}$ and bias $\mathbf{b}^{(\theta)}$.

A.2. Proof of Theorem 2

(a) We first prove $S^{(M-1)}(\theta) \cap T(\phi) \subseteq S(\theta)$. Consider $\mathbf{x} \in S^{(M-1)}(\theta) \cap T(\phi)$. Thus, $\mathbf{x} \in T(\phi)$,

$$\implies \mathbf{z}^{(M-1)}(\mathbf{x}) = g(\mathbf{x}) = \mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)}$$
$$\mathbf{a}^{M-1}(\mathbf{x}) = \max(\mathbf{z}^{(M-1)}(\mathbf{x}), 0)$$
$$\mathbf{a}^{M-1}(\mathbf{x}) = \max(\mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)}, 0)$$

Since $\mathbf{x} \in S^{(M-1)}(\theta)$, the indicator vector of $(M-1)^{th}$ layer is $\theta^{(M-1)}$. Thus,

$\implies \mathbf{x} \in S(\theta)$

$\implies S^{(M-1)}(\theta) \cap T(\phi) \subseteq S(\theta)$

Now we prove $S(\theta) \subseteq S^{(M-1)}(\theta) \cap T(\phi)$,

Consider $\mathbf{x} \in S(\theta)$,

Since $\mathbf{x}$ induces θ in $f(.)$, it must induce ϕ in $g(.)$

Hence $\mathbf{x} \in T(\phi)$,

$\implies \mathbf{z}^{(M-1)}(\mathbf{x}) = g(\mathbf{x}) = \mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)}$

But since $\mathbf{x} \in S(\theta)$,

The indicator vector of $(M-1)^{th}$ layer must be $\theta^{(M-1)}$.

$(M-1)^{th}$ indicator can be $\theta^{(M-1)}$ **iff** $\mathbf{x} \in S^{(M-1)}(\theta)$

$\implies \mathbf{x} \in S^{(M-1)}(\theta)$ and $\mathbf{x} \in T(\phi)$

Thus, $S(\theta) \subseteq S^{(M-1)}(\theta) \cap T(\phi)$

$\implies S(\theta) = S^{(M-1)}(\theta) \cap T(\phi)$

$(b) \forall\ \mathbf{x} \in S(\theta)$,

$$\mathbf{a}^{(M-1)}(\mathbf{x}) = \max(\mathbf{z}^{(M-1)}(\mathbf{x}), 0)$$
$$\mathbf{a}^{(M-1)}(\mathbf{x}) = \theta^{(M-1)} \odot \mathbf{z}^{(M-1)}(\mathbf{x})$$

Since $\mathbf{z}^{(M-1)}(\mathbf{x}) = \mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)}$,

$$\mathbf{a}^{(M-1)}(\mathbf{x}) = \theta^{(M-1)} \odot (\mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)})$$
$$\mathbf{a}^{(M-1)}(\mathbf{x}) = \text{diag}(\theta^{(M-1)})(\mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)})$$
$$\mathbf{a}^{(M)}(\mathbf{x}) = \mathbf{z}^{(M)}(\mathbf{x}) = \mathbf{W}^{(M)}\mathbf{a}^{(M-1)}(\mathbf{x}) + \mathbf{b}^{(M)}$$
$$\mathbf{a}^{(M)}(\mathbf{x}) = \mathbf{W}^{(M)}(\text{diag}(\theta^{(M-1)})(\mathbf{W}^{(\phi)}\mathbf{x} + \mathbf{b}^{(\phi)})) + \mathbf{b}^{(M)}$$
$$\mathbf{a}^{(M)}(\mathbf{x}) = \underbrace{\mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{W}^{(\phi)}}_{\text{Weight term}}\mathbf{x} + \underbrace{\mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{b}^{(\phi)} + \mathbf{b}^{(M)}}_{\text{Bias term}}$$
$$\mathbf{a}^{(M)}(\mathbf{x}) = \mathbf{W}^{(\theta)}\mathbf{x} + \mathbf{b}^{(\theta)} \quad \text{where,}$$
$$\mathbf{W}^{(\theta)} = \mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{W}^{(\phi)},$$
$$\mathbf{b}^{(\theta)} = \mathbf{W}^{(M)}\text{diag}(\theta^{(M-1)})\mathbf{b}^{(\phi)} + \mathbf{b}^{(M)}$$

Hence, $\mathbf{W}^{(\theta)}$ and $\mathbf{b}^{(\theta)}$ are constant for a given θ and $\mathbf{a}^{(M)}$ is linear in $\mathbf{x}$ with weights $\mathbf{W}^{(\theta)}$ and bias $\mathbf{b}^{(\theta)}$.

A.3. Proof of Corollary 1

Using Theorem 1 and Theorem 2, the i^{th} hidden layer adds N_i inequalities to the activation region and the decision function is constant given θ. The proof follows using induction.

A.4. Proof of Theorem 3

P_{min} is the minimum of the distances of the input from all the faces of the Polyhedron $S(\theta)$. d_{min} denotes the minimum of the distances of the input from all the decision boundaries defined by the decision function d_θ.
Consider two cases:
Case 1: $\mathbf{u}_0$ lies inside the polyhedron $S(\theta)$
Since $\mathbf{u}_0$ lies inside the polyhedron,

$$f(\mathbf{u}_0) = \mathbf{W}^{(\theta)}\mathbf{u}_0 + \mathbf{b}^{(\theta)}$$

Since $\mathbf{u}_0$ is the closest adversarial example,

$$f_j(\mathbf{u}_0) = f_k(\mathbf{u}_0) \text{ (for some } j \neq k)$$

and it must lie on the decision boundary:

$$\mathbf{W}_j^{(\theta)}\mathbf{u}_0 + \mathbf{b}_j^{(\theta)} = \mathbf{W}_k^{(\theta)}\mathbf{u}_0 + \mathbf{b}_k^{(\theta)} \text{ (where } j \neq k)$$

Since d_{min} is the minimum distance of $\mathbf{u}$ from all such decision boundaries.

$$\implies \|\mathbf{u} - \mathbf{u}_0\|_2 \geq d_{min} \tag{3}$$

Case 2: $\mathbf{u}_0$ lies outside the polyhedron $S(\theta)$
Since $\mathbf{u}_0$ lies outside the polyhedron, minimum distance of $\mathbf{u}$ from all the faces of the polyhedron $S(\theta)$ gives a lower bound to the distance from $\mathbf{u}_0$. Since each $\mathbf{P}_i^{(\theta)}\mathbf{x} + \mathbf{q}_i^{(\theta)} = 0$, defines a face of the polyhedra,

$$\implies \|\mathbf{u} - \mathbf{u}_0\|_2 \geq P_{min} \tag{4}$$

Using 3 and 4,

$$\|\mathbf{u} - \mathbf{u}_0\|_2 \geq \min(d_{min}, P_{min})$$

A.5. Proof of Theorem 4

Let $\mathbf{u}_0$ is the closest adversarial example, and ϕ be the activation pattern for $\mathbf{u}_0$. Since $\mathbf{u}_0$ must lie on a decision boundary, we assume for some l:

$$\mathbf{c}^{(\phi)}\mathbf{u}_0 + \mathbf{d}^{(\phi)} = 0 \text{ where,}$$
$$\mathbf{c}^{(\phi)} = \mathbf{W}_l^{(\phi)} - \mathbf{W}_k^{(\phi)}$$
$$\mathbf{d}^{(\phi)} = \mathbf{b}_l^{(\phi)} - \mathbf{b}_k^{(\phi)}$$
$$\|\mathbf{u} - \mathbf{u}_0\|_2 \geq \frac{|\ \mathbf{c}^{(\phi)}\mathbf{u} + \mathbf{d}^{(\phi)}\ |}{\|\mathbf{c}^{\phi}\|_2}$$

Simplifying the denominator term $\|\mathbf{c}^{(\phi)}\|_2$,

$$\begin{aligned}
&\|\mathbf{c}^{(\phi)}\|_2 \\
&= \|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\mathrm{diag}(\phi^{(1)})\mathbf{W}^{(1)}\|_2 \\
&\leq \|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathrm{diag}(\phi^{(1)})\mathbf{W}^{(1)}\|_2 \\
&\leq \|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathrm{diag}(\phi^{(1)})\|_2 \|\mathbf{W}^{(1)}\|_2 \\
&\text{Since } \|\mathrm{diag}(\phi^{(1)})\|_2 = \max(\phi^{(1)}) \leq 1 \\
&\implies \|\mathbf{c}^{(\phi)}\|_2 \leq \|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2 \\
&\frac{1}{\|\mathbf{c}^{(\phi)}\|_2} \geq \frac{1}{\|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2} \\
&\text{Since } \|\mathbf{u} - \mathbf{u}_0\|_2 \geq \frac{|\ \mathbf{c}^{(\phi)}\mathbf{u} + \mathbf{d}^{(\phi)}\ |}{\|\mathbf{c}^{\phi}\|_2} \\
&\text{Substituting } \frac{1}{\|\mathbf{c}^{(\phi)}\|_2},
\end{aligned}$$

$$\begin{aligned}
&\|\mathbf{u} - \mathbf{u}_0\|_2 \\
&\geq \frac{|\ \mathbf{c}^{(\phi)}\mathbf{u} + \mathbf{d}^{(\phi)}\ |}{\|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2} \\
&= \frac{|\ (\mathbf{W}_l^{(\phi)} - \mathbf{W}_k^{(\phi)})\mathbf{u} + \mathbf{b}_l^{(\phi)} - \mathbf{b}_k^{(\phi)}\ |}{\|(\mathbf{W}_l^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2} \\
&\geq \min_{j \neq k} \frac{|\ (\mathbf{W}_j^{(\phi)} - \mathbf{W}_k^{(\phi)})\mathbf{u} + \mathbf{b}_j^{(\phi)} - \mathbf{b}_k^{(\phi)}\ |}{\|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2} \\
&\geq \min_{j \neq k} \frac{\min\limits_{\theta_i^{(1)} \in \{0,1\}} |\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)}\ |}{\|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2} \\
&\geq \min_{j \neq k} \frac{\min\limits_{0 \leq \theta_i^{(1)} \leq 1} |\ (\mathbf{W}_j^{(\theta)} - \mathbf{W}_k^{(\theta)})\mathbf{u} + \mathbf{b}_j^{(\theta)} - \mathbf{b}_k^{(\theta)}\ |}{\|(\mathbf{W}_j^{(2)} - \mathbf{W}_k^{(2)})\|_2 \|\mathbf{W}^{(1)}\|_2}
\end{aligned}$$

Since $\mathbf{W}_j^{(\theta)}$ and $\mathbf{W}_k^{(\theta)}$ are row vectors, we can simplify,

$$\begin{aligned}
\mathbf{W}_j^{(\theta)} &= \mathbf{W}_j^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{W}^{(1)} \\
&= (\theta^1)^T \mathrm{diag}(\mathbf{W}_j^{(2)})\mathbf{W}^{(1)}, \\
\mathbf{b}_j^{(\theta)} &= \mathbf{W}_j^{(2)}\mathrm{diag}(\theta^{(1)})\mathbf{b}^{(1)} + \mathbf{b}_j^{(2)} \\
&= (\theta^{(1)})^T \mathrm{diag}(\mathbf{W}_j^{(2)})\mathbf{b}^{(1)} + \mathbf{b}_j^{(2)}
\end{aligned}$$

Similarly for $\mathbf{W}_k^{(\theta)}$ and $\mathbf{b}_k^{(\theta)}$.
Since $\mathbf{W}_j^{(\theta)}$, $\mathbf{b}_j^{(\theta)}$ and $\mathbf{W}_k^{(\theta)}$, $\mathbf{b}_k^{(\theta)}$ are linear in θ, RHS can be solved using convex optimization.

B. Details of Experiments

B.1. Details of Experiments reported in Figure 2a

Hyper-parameters used in this experiment are reported in Table 2.

Table 2. Hyper-parameters used in experiments of Figure 2a

Parameter	Config
Optimizer	Adam
Network architecture	[784, 1024, 2]
Batch size	64
Number of epochs	20
Learning rate	0.001
Initialization	Glorot

B.2. Details of Experiments reported in Figure 2b

Hyper-parameters used in this experiment are reported in Table 3.

Table 3. Hyper-parameter used in experiments of Figure 2b

Parameter	Config
Optimizer	Adam
Network architecture	[784, 1024, 512, 10]
Batch size	64
Number of epochs	20
Learning rate	0.001
Initialization	Glorot

B.3. Details of Experiments reported in Table 1 in the Main text

Hyper-parameters used in this experiment are reported in Table 4.

Table 4. Hyper-parameter used in experiments of Table 1 in the main text

Parameter	Config
Optimizer	Adam
Batch size	64
Number of epochs	20
Learning rate	0.001
Initialization	Glorot