REGLO: Provable Neural Network Repair for Global Robustness Properties

Feisi Fu[1]*, Zhilu Wang[2]*, Weichao Zhou[1], Yixuan Wang[2], Jiameng Fan[1], Chao Huang[3], Qi Zhu[2], Xin Chen[4], Wenchao Li[1]

[1]Department of Electrical and Computer Engineering, Boston University, Boston, MA, USA
[2]Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL, USA
[3]Department of Computer Science, University of Liverpool, Liverpool, UK
[4] University of New Mexico, Albuquerque, NM, USA
fufeisi@bu.edu, zhilu.wang@u.northwestern.edu, zwc662@bu.edu, yixuanwang2024@u.northwestern.edu, jmfan@bu.edu, chao.huang2@liverpool.ac.uk, qzhu@northwestern.edu, chenxin@unm.edu, wenchao@bu.edu

Abstract

We present REGLO, a novel methodology for repairing pre-trained neural networks to satisfy global robustness and individual fairness properties. A neural network is said to be globally robust with respect to a given input region if and only if all the input points in the region are locally robust. This notion of global robustness also captures the notion of individual fairness as a special case. We prove that any counterexample to a global robustness property must exhibit a corresponding large gradient. For ReLU networks, this result allows us to efficiently identify the linear regions that violate a given global robustness property. By formulating and solving a suitable robust convex optimization problem, REGLO then computes a minimal weight change that will provably repair these violating linear regions.

Introduction

Motivated by the fragility of deep neural networks (DNNs) to small input perturbations known as *adversarial examples* (Goodfellow, Shlens, and Szegedy 2014), there is a large, growing body of research on measuring, verifying, and improving the robustness of DNNs against those perturbations (Tramèr et al. 2017; Madry et al. 2018; Shafahi et al. 2019; Wang et al. 2019; Wong, Rice, and Kolter 2020; Mirman, Gehr, and Vechev 2018; Zhang et al. 2020; Fan and Li 2021; Huang et al. 2020). Various notions of robustness have been considered (Casadio et al. 2022; Seshia et al. 2018; Chen et al. 2021; Katz et al. 2017; Carlini et al. 2019), and can be largely categorized into two groups: *local robustness* and *global robustness*. Local robustness is about the robustness of individual input points. Intuitively, it means that for an input x, a small change of the input (e.g., any p-norm-bounded Δx) would not result in a significant change in the output (e.g., change of a classification result). On the other hand, global robustness requires local robustness on *all* (infinitely many) points within some given input region X. Global robustness is thus strictly stronger than local robustness, and has the advantage of also enforcing robustness on any unseen input within the given input region.

The existing approaches for improving the global robustness of neural networks focus on altering their training process (Leino, Wang, and Fredrikson 2021; Chen et al. 2021; Wadsworth, Vera, and Piech 2018; Celis and Keswani 2019; Adel et al. 2019; Tao et al. 2022; Khedr and Shoukry 2022). These approaches mainly involve introducing a loss function related to global robustness such that global robustness is taken into account during gradient descent. However, training-based methods that use such a loss term *cannot* guarantee the satisfaction of global robustness properties as there is no way to guarantee that the loss term will be minimized to 0 via training. Moreover, applying training-based methods to fine-tune pre-trained networks can be prohibitively expensive. The application of these methods to pre-trained networks can also be hampered by the unavailability of training data, e.g., when the neural network is obtained from a third party or the training data is private.

In this paper, we consider the problem of *repairing* a pre-trained DNN to satisfy a given global robustness property. At a high level, neural network repair is the class of techniques that involves directly modifying a pre-trained neural network so that the resulting network satisfies some given property. A repair method is considered *sound* if it can guarantee the removal of the discovered violations or the satisfaction of the given property. In contrast to training-based methods that requires many iterations, repair can be applied once as a *post-hoc* modification. Existing DNN repair methods mainly consist of weight modification (Dong et al. 2020; Goldberger et al. 2020; Chen, Li, and Zhang 2022; Refaeli and Katz 2022), either via constraint solving or fine-tuning, and DNN architecture extension (Sotoudeh and Thakur 2021; Fu and Li 2021; Mitchell et al. 2021; Leino et al. 2022). However, these repair methods cannot handle global robustness or individual fairness properties. The technique proposed in this paper fills precisely this gap. We summarize our contributions below.

1. We propose `REGLO`, the *first DNN repair technique with provable guarantees on satisfying global robustness properties*.

2. We show that any input that is a counterexample to a global robustness property must have a corresponding large gradient that indicates the violation.

3. For piecewise linear DNNs, our approach is both sound and complete – the resulting network is guaranteed to satisfy the given global robustness property, and a repair is guaranteed to be found if one exists.

* The first two authors contributed equally to this paper.

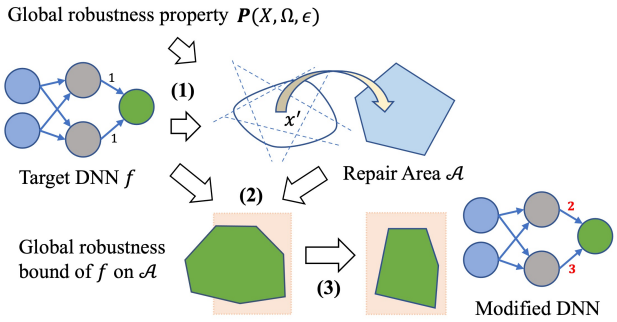

Figure 1: An illustration of REGLO's verification-guided algorithm: (1) identify repair areas that violate the global robustness property, (2) compute the global robustness bound for each repair area (in green with the light coral area showing the desired bound according to the global robustness property), and (3) solve a robust convex optimization problem to modify certain weights of the target DNN so that the modified DNN is guaranteed to satisfy the global robustness property on those repair areas.

4. Across a variety of benchmarks, we show that REGLO can significantly enhance global robustness while maintaining performance.

Related Works

Global Robustness Training

Current methods aimed at improving the overall resilience of neural networks primarily concentrate on modifying their training procedures. Leino, Wang, and Fredrikson (2021) present a method for training globally-robust classification networks by using an additional output class that labels inputs as "non-locally-robust". Chen et al. (2021) employ a counterexample-guided framework, referred to as the booster-fixer training framework, to iteratively train and fix a decision tree-like classifier until it satisfies the specified global robustness properties.

Existing works also leverage adversarial-training schemes by using a discriminator to force the classifier to be unbiased towards the sensitive features (Wadsworth, Vera, and Piech 2018; Celis and Keswani 2019; Adel et al. 2019; Tao et al. 2022). Khedr and Shoukry (2022) propose to use output bounds obtained from global robustness verification as a regularizer during training. Compared with REGLO, training-based methods with robustness-related loss terms cannot ensure global robustness, as there is no guarantee that the loss will reach 0 during training. Additionally, applying training-based methods to fine-tune pre-trained networks can be costly and hindered by unavailable training data, such as when the network is from a third party or private sources.

Individual Fairness

The notion of *individual fairness* (IF), which requires two inputs that differ only on some sensitive features to have similar outputs, can be viewed as a global robustness property (John, Vijaykeerthy, and Saha 2020). Benussi et al. (2022) present an MILP formulation whose solution can be used to verify IF properties and guide the training process by modifying the training loss. Yeom and Fredrikson (2020) employs randomized smoothing to modify the inference process of neural networks and achieve provably individual fairness.

Similar to most papers on global robustness, training-based methods (Benussi et al. 2022) for individual fairness with related loss terms cannot guarantee satisfaction. This often leads to safety issues when deploying such trained neural networks. The major issue with the randomized smoothing method (Yeom and Fredrikson 2020) is that it can lead to increased computational costs during the inference process of a neural network, which can be unacceptable in many applications. Our proposed REGLO method, however, completely avoids these issues.

Neural Network Repair

Neural Network Repair refers to the process of modifying a trained neural network so that the resulting network satisfies some given input-output property. Existing techniques include weight modification (Dong et al. 2020; Goldberger et al. 2020; Chen, Li, and Zhang 2022; Refaeli and Katz 2022), either via constraint solving or fine-tuning, and architecture extension (Sotoudeh and Thakur 2021; Fu and Li 2021; Mitchell et al. 2021; Leino et al. 2022). For instance, in (Sotoudeh and Thakur 2021), a decoupling technique is employed to separate the activation functions of a neural network, thereby achieving provable modifications for specific inputs. In (Fu and Li 2021), a sub-network is constructed by targeting specific linear regions of a ReLU neural network. The output of this sub-network is used as a modification to the original neural network, ultimately enabling it to satisfy the desired input-output constraints.

To the best of our knowledge, existing repair techniques are currently designed to fix errors in neural network outputs. In comparison to these methods, REGLO is the first DNN repair technique that can handle global robustness properties. It is important to note that in this context, "global robustness properties" pertain to the consideration of two related inputs and outputs, rather than individual ones.

Verification for Global Robustness Property

A standard way to verify global robustness is to reduce it to verifying local robustness by constructing a twin-network such that the input to one copy of the DNN represents the original input x and the input to the other copy represents an adversarial perturbation of x (Singh et al. 2019; Katz et al. 2017). *ITNE* (Wang, Huang, and Zhu 2022; Wang et al. 2022) is the state-of-the-art global robustness verification technique. It adds interleaving dependencies in the twin-network encoding and leverages bound propagation techniques (Zhang et al. 2018; Wang et al. 2018; Huang et al. 2020; Wang et al. 2021) to compute the output bound for a given input area on the twin-network. In REGLO, we use global robustness verification to guide the repair process, which we will describe in detail in the next section.

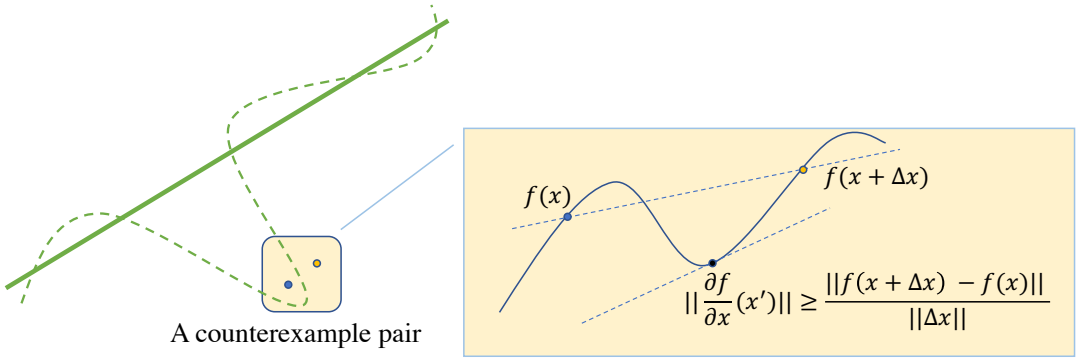

Figure 2: The solid green line and the dashed green line are the unknown, ground-truth decision boundary for the input data distribution and the decision boundary of the trained DNN respectively. The deviation of DNN's decision boundary leads to a violation of the global robustness property. Our key observation is that if $(x, x + \Delta x)$ is a counterexample pair of a global robustness property for DNN f, then there exists x' with a large gradient.

Background

Deep Neural Networks (DNNs). An R-layer feed-forward DNN $f : \mathcal{X} \to \mathcal{Y}$ is a composition of linear functions and activation function σ, where $\mathcal{X} \subseteq \mathbb{R}^m$ is a bounded input domain and $\mathcal{Y} \subseteq \mathbb{R}^n$ is the output domain. The weights and biases of the linear function are parameters of the DNN. We call the first $R-1$ layers hidden layers and the R-th layer the output layer. We use z_j^i to denote the i-th neuron (before activation) in the j-th hidden layer.

For DNNs that use only the ReLU activation function $\sigma(x) = \max(x, 0)$, we call them ReLU DNNs. For any neuron z_j^i, we say the neuron is activated for an input if and only if the neuron's value $\sigma(z_j^i) = z_j^i$. We use a binary variable α_j^i to represent the activation status of z_j^i (where $\alpha_j^i = 1$ means the neuron is activated). The set of activation statuses $\{\alpha_j^i\}$ of all the neurons is called an activation pattern. It is known that an $\mathbb{R}^m \to \mathbb{R}$ function is representable by a ReLU DNN *if and only if* it is a continuous piecewise linear (CPWL) function (Arora et al. 2016).

Linear Regions. A *linear region* is the set of inputs that are subject to the same activation pattern in a ReLU DNN (Serra, Tjandraatmadja, and Ramalingam 2017).

Lemma 1 *(Lee, Alvarez-Melis, and Jaakkola 2019) Consider a ReLU DNN f and an input $x \in \mathbb{R}^m$. For every neuron z_j^i, it induces a feasible set*

$$\mathcal{A}_j^i(x) = \begin{cases} \{\bar{x} \in \mathbb{R}^m | (\nabla_x z_j^i)^T \bar{x} + z_j^i - (\nabla_x z_j^i)^T x \geq 0\} \\ \quad \textit{if } z_j^i \geq 0 \textit{ or } \alpha_j^i = 1 \\ \{\bar{x} \in \mathbb{R}^m | (\nabla_x z_j^i)^T \bar{x} + z_j^i - (\nabla_x z_j^i)^T x \leq 0\} \\ \quad \textit{if } z_j^i < 0 \textit{ or } \alpha_j^i = 0 \end{cases}$$

where the symbol $\nabla_x z$ represents the gradient vector of z with respect to x and the intersection $\mathcal{A}(x) = \bigcap_{i,j} \mathcal{A}_j^i(x)$ is the linear region that includes x.

Global Robustness and Individual Fairness. We consider a global robustness property $\mathbf{P}$ on $X \subseteq \mathcal{X}$ that is defined as follows.

Definition 1 (Global Robustness) *A DNN f satisfies a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$ on an input set $X \subseteq \mathcal{X}$ along with a perturbation set $\Omega \subseteq \mathbb{R}^m$ if and only if for any $x \in X$ and $\Delta x \in \Omega$, $\|f(x) - f(x + \Delta x)\| \leq \epsilon$ holds.*

In essence, a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$ requires every point in X to be locally robust with respect to Ω and ϵ. We can also obtain the common norm-bounded notion of robustness by instantiating the defintion above with $\Omega = \{\Delta x \mid \|\Delta x\| \leq \delta\}$.

Definition 2 (Norm-Bounded Global Robustness) *A DNN f is (δ, ϵ)-globally robust on an input set $X \subseteq \mathcal{X}$ if and only if for any $x \in X$, we have that $\|\Delta x\| \leq \delta \Rightarrow \|f(x) - f(x + \Delta x)\| \leq \epsilon$.* [1]

It is worth noting that this definition of global robustness captures the notion of *individual fairness* (Dwork et al. 2012; John, Vijaykeerthy, and Saha 2020) as a special case, by taking $\Omega = \{\Delta x \mid \Delta x_{\text{SF}} = 0\}$ where SF indicates sensitive features and NF indicates the remaining, non-sensitive features, as follows.

Definition 3 (Individual Fairness) *A DNN f is ϵ-fair with respect to some sensitive input features SF if and only if for any $x \in \mathcal{X}$, if $x_{SF} = (x+\Delta x)_{SF}$, then $\|f(x)-f(x+\Delta x)\| \leq \epsilon$.*

Based on the definitions above, we define the *repair* problem for global robustness as follows.

Definition 4 (Repair for Global Robustness Property) *Given a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$ and a target DNN $f \not\models \mathbf{P}(X, \Omega, \epsilon)$, the repair problem is to find a modified DNN $\widehat{f}$ such that $\widehat{f} \models \mathbf{P}(X, \Omega, \epsilon)$.* [2]

[1] The norm in this definition can be any p norm.

[2] We consider the case where f and $\widehat{f}$ share the same structure,

The REGLO Approach

We present REGLO's approach below, starting with an observation that allows us to identify repair regions that violate a given global robustness property. A preview of the high-level approach in REGLO is also given in Figure 1.

Key Observation. A counterexample to a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$ is a pair of inputs $(x, \Delta x)$ that violates the property $\mathbf{P}$, i.e., $x \in X$ and $\Delta x \in \Omega$ but $\|f(x) - f(x + \Delta x)\| \geq \epsilon$.

Lemma 2 (Mean Value Inequality (Hörmander 2015)) *For a continuous function $f : [a, b](\subseteq \mathbb{R}) \to \mathbb{R}^n$, if f is differentiable on (a, b), then*

$$\|f(b) - f(a)\| \leq (b - a) \sup_{x \in (a,b)} \|f'(x)\| \tag{1}$$

Inspired by Lemma 2, we derive the following theorem[3], which enables us to convert the search of a potential *violation region* (a region that contains a counterexample pair) for a global robustness property to the search of a single point with a large gradient, as illustrated in Figure 2.

Theorem 1 (Gradients of Any Counterexamples) *For a DNN f and a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$, if there is a counterexample $(x, \Delta x)$ such that $\Delta x \in \Omega$ and $\|f(x + \Delta x) - f(x)\| \geq \epsilon$, then there exists a differentiable point x' between x and $x + \Delta x$, such that $\|x - x'\| \leq \frac{dia(\Omega)}{2}$ and $\|\frac{\partial f}{\partial x}(x')\| > \frac{\epsilon}{dia(\Omega)}$, where $dia(\Omega)$ is the diameter of Ω*[4].

Proof: Consider $g : [0, 1] \to \mathbb{R}^n$ defined as $g(t) = f(x + t\Delta x)$ and g is piecewise differentiable on $[0, 1]$. Since $\epsilon < \|g(1) - g(0)\|$, there must exist a differentiable interval (t_i, t_{i+1}), such that

$$(t_{i+1} - t_i)\epsilon < \|g(t_{i+1}) - g(t_i)\|$$

Then by Inequality 2, we have,

$$\begin{aligned}(t_{i+1} - t_i)\epsilon &< \|g(t_{i+1}) - g(t_i)\| \\ &\leq (t_{i+1} - t_i) \sup_{t \in (t_i, t_{i+1})} \|g'(t)\| \\ = (t_{i+1} - t_i) \sup_{t \in (0,1)} &\|\frac{\partial f}{\partial x}(x + t\Delta x)\Delta x\|\end{aligned}$$

Thus, there exists a $x' = x + t'\Delta x$, $\|x - x'\| \leq \frac{dia(\Omega)}{2}$ (we can simply switch x and $x + \Delta x$ if $\|x - x'\| > \frac{dia(\Omega)}{2}$), and

$$\|\frac{\partial f}{\partial x}(x')\Delta x\| > \epsilon \Rightarrow \|\frac{\partial f}{\partial x}(x')\| > \frac{\epsilon}{dia(\Omega)}$$

□

i.e. the same number of layers and the same number of neurons.

[3]The *Mean Value Inequality* requires differentiability of the function f on (a, b). For non-differentiable DNNs, the non-differentiability comes from its activation functions, e.g., ReLU, LeakReLU, Heaviside, or Maxpooling. Since the measure of the non-differentiable area is zero for those operations, we can still apply the *Mean Value Inequality* in a piecewise manner to obtain the same result.

[4]Here we take the operator norm of a matrix.

Instead of sampling a pair of points, Theorem 1 states that sampling a single point can be used to check for counterexamples to a given global robustness property. For ReLU DNNs, as all the points within the same linear region have the same gradient, we can use this result to efficiently identify violating linear regions.

Remark: 1. Note that this theorem specifies a necessary but not sufficient condition. In other words, the presence of a counterexample must exhibit a large corresponding gradient but the reverse is not necessarily true.

2. For individual fairness, we can define the norm $\|x\| = \|x_{SF}\|$ if $x_{NF} = 0$ and else $\|x\| = +\infty$, and only consider the gradient with respect to the sensitive features $\|\frac{\partial f}{\partial x}(x')\| = \|\frac{\partial f}{\partial x_{SF}}(x')\|$.

Repair Areas. By leveraging Theorem 1, the search for counterexamples is transformed into searching for linear regions with gradients larger than $\frac{\epsilon}{dia(\Omega)}$. A mixed-integer programming (MILP) problem can be used to encode the search for the maximum gradient in a ReLU DNN (Fischetti and Jo 2017; Runje and Shankaranarayana 2022). Since the number of linear regions on X is finite, we can find all the linear regions with gradient greater than $\frac{\epsilon}{dia(\Omega)}$ by iteratively solving MILPs and accumulating exclusion constraints (to exclude the same activation pattern $\hat{\alpha}$ that we have already found) until the optimal solution c^* is smaller than $\frac{\epsilon}{dia(\Omega)}$. Note that while the total number of linear regions can be very large, the number of violating regions is typically much smaller.

For non-CPWL DNNs, we can use random sampling to search for input x' that satisfies the violation constraint $\|\frac{\partial f}{\partial x}(x')\| > \frac{\epsilon}{dia(\Omega)}$. If the target DNN f is twice differentiable, we can apply a projected gradient descent method to improve the sample efficiency. For CPWL DNNs, we can also start with random sampling and then pivot to the more expensive MILP-based method for better search efficiency.

We now define the repair area $\mathcal{A}$ as follows. $\mathcal{A} = \{x | Ax \leq b\}$ by solving MILP or $\mathcal{A} = \{x | \|x - x'\| \leq \frac{\delta}{2}\}$ via random sampling. We use $\{\mathcal{A}_i\}_{i \in I}$ to denote all the repair areas found via the aforementioned procedure.

Verification-Guided Constraints. For each repair area $\mathcal{A}_i$, *ITNE* (Wang et al. 2022) uses the following optimization problem to estimate the global robustness bound on $\mathcal{A}_i$,

$$\begin{cases} \max_{[\Delta x, \Delta z_i] \in \Omega \times Z_i} \|\theta \cdot \Delta z_i\| \\ Z_i = \{\Delta z \mid D_i^l \Delta x + e_i^l \leq \Delta z \leq D_i^u \Delta x + e_i^u\} \end{cases} \tag{2}$$

where θ is the weight in the last hidden layer of the DNN, D_i^l, e_i^l, D_i^u and e_i^u are parameters for the linear bounds of Δz, and Δz is the neurons value difference of the last hidden layer between x and $x + \Delta x$.

Remark: We consider weight modification in the last hidden layer because it does not change the activation pattern of any input and in turn preserves the boundaries of the linear regions. Thus, by repairing the violating linear regions iteratively, we can guarantee satisfaction of the given global robustness property. In theory, we can also consider modifying

Algorithm 1: Iterative Repair for ReLU DNN

Input: The target ReLU DNN f and the global robustness property $\mathbf{P}(X, \Omega, \epsilon)$.
Parameter: Maximum iteration T and maximum number of repair areas M.
Output: The repaired DNN.

Let $t = 0$.
Let $Areas = \emptyset$
while $t < T$ **do**
  while $|Areas| < M$ **do**
    Let c^* and x^* be the optimal value and solution of MILP, respectively.
    if $c^* \leq \frac{\epsilon}{dia(\Omega)}$ **then**
      Break
    end if
    Let $\mathcal{A}_i$ be the linear region containing x^* computed using Lemma 1.
    $Areas$.add($\mathcal{A}_i$).
    Add a new exclusion constraint to MILP.
  end while
  Formulate the constraints on $Areas$ as a robust optimization problem (3), solve it via Algorithm 2, and obtain an optimal solution $\Delta\theta$.
  Update f's last-layer weight $\theta = \theta + \Delta\theta$.
end while
return f with updated last-layer weight.

the weights of an intermediate layer. However, the objective function of optimization problem (2) will include the subsequent layers and the optimization problem will no longer be convex.

Repair as Robust Optimization. We use $\Delta\theta$ to denote the weight change to the DNN's last hidden layer. In order to preserve the functionality (e.g., accuracy) of the network, we aim to find a *minimal* weight change $\Delta\theta$ that can guarantee the satisfaction of global robustness property on all the repair areas $\{\mathcal{A}_i\}_{i\in I}$. Formally, $\Delta\theta$ is the solution to the following optimization problem.

$$\begin{cases} \min \|\Delta\theta\| \\ \max_{[\Delta x, \Delta z_i]\in\Omega\times Z_i} \|(\theta+\Delta\theta)\Delta z_i\| \leq \epsilon \\ Z_i = \{\Delta z \mid D_i^l \Delta x + e_i^l \leq \Delta z \leq D_i^u \Delta x + e_i^u\} \end{cases} \tag{3}$$

Property 1 *Optimization problem (3) is convex and thus any local minimum also achieves the global minimum.*

Proof: By the definition of a norm, we have any norm $\|.\|$ is a convex function.

For any $\Delta\theta_1$, $\Delta\theta_2$, $\lambda \in [0, 1]$ and any Δz_i, we have

$$\begin{aligned} &\max_{[\Delta x, \Delta z_i]} \|(\theta + \lambda\Delta\theta_1 + (1-\lambda)\Delta\theta_2)\Delta z_i\| \\ =&\max_{[\Delta x, \Delta z_i]} \|\lambda(\theta+\Delta\theta_1)\Delta z_i + (1-\lambda)(\theta+\Delta\theta_2)\Delta z_i\| \\ \leq&\max_{[\Delta x, \Delta z_i]} [\|\lambda(\theta+\Delta\theta_1)\Delta z_i\| + \|(1-\lambda)(\theta+\Delta\theta_2)\Delta z_i\|] \\ \leq&\lambda\epsilon + (1-\lambda)\epsilon = \epsilon \end{aligned}$$

Both the objective and the constraints are convex. Therefore, we have that optimization programming (3) is convex. Thus, any local minimum achieves the global minimum. □

This minimization problem with inner maximal constraints is a form of *robust optimization* (Ben-Tal, El Ghaoui, and Nemirovski 2009). For minimizing the L_1 or L_∞ norm of $\Delta\theta$, such *robust optimization* problems can be solved by taking the *robust counterpart* of the inner constraints and converting it to a linear programming (LP) problem.

Repair via Barrier Method. For general L_p norm of $\Delta\theta$, we apply the *barrier method* from (Boyd and Vandenberghe 2004) and formulate it as an unconstrained convex optimization problem. We use $[\Delta x_i^*(\Delta\theta), \Delta z_i^*(\Delta\theta)]$ and $\epsilon_i^*(\theta+\Delta\theta)$ to denote the optimal solution and the optimal value respectively for optimization problem (2). We have, for a sufficiently large t, the solution of the following barrier problem converges to the solution of the optimization problem (3).

$$\min_{\Delta\theta} \|\Delta\theta\| - \frac{1}{t}\sum_{i\in I} \log(\epsilon - \epsilon_i^*(\theta+\Delta\theta)) \tag{4}$$

To solve the optimization problem (4), we compute the gradient of $\epsilon_i^*(\theta+\Delta\theta)$ by

$$\frac{\partial\epsilon_i^*(\theta+\Delta\theta)}{\partial\Delta\theta} = \frac{\partial\|(\theta+\Delta\theta)^T\Delta z_i\|}{\partial\Delta\theta}|_{\Delta z_i = \Delta z_i^*(\Delta\theta)} \tag{5}$$

Iterative Repair. In the previous sections, we enumerate all the repair areas $\{\mathcal{A}_i\}_{i\in I}$ that violate $\mathbf{P}(X, \Omega, \epsilon)$ and repair these areas via weight modification. However, it is possible that the repair increases the gradient $\|\frac{\partial f}{\partial x}(x')\|$ for $x' \in X \setminus \cup_{i\in I}\mathcal{A}_i$ and results in a new violation to $\mathbf{P}(X, \Omega, \epsilon)$. To ensure a sound repair, we iteratively search for repair areas to repair. When solving the optimization problem, we consider both the previously repaired areas and the new repair areas.

We call one iteration, including the search of repair areas and the repair itself, a *Single Iteration Repair*. Algorithm 1 presents the *Iterative Repair* procedure.

Theoretical Guarantees

In this section, we present the theoretical guarantees that REGLO provides.

Theorem 2 (Completeness Guarantees) *We have the following completeness guarantees for REGLO:*

1. For a Single Iteration Repair, REGLO can always find a solution to optimization problem (3).

2. For an Iterative Repair on a piecewise linear DNN, REGLO always terminates with no more repair areas to be found by solving MILP.

Proof: For a *Single Iteration Repair*, the optimization problem (3) is convex over a close domain. Since $\Delta\theta = -\theta$ is a feasible solution, the domain where $\Delta\theta$ is in is always feasible. Therefore, optimization problem (3) has an optimal solution. Since the optimization problem is convex, the optimal solution is unique and REGLO can always find the optimal solution to the optimization problem (3).

For an *Iterative Repair* on a CPWL DNN, REGLO can always find an optimal solution for every *Single Iteration*

		ST	AT	ST+Finetune	AT+Finetune	ST+REGLO	AT+REGLO
All age	VB	12.5	4.69	9.0	4.52	0.29	**0.26**
	PGD-B	1.31	0.135	0.7	0.11	0.028	**0.008**
	PGD-R	75.5%	26.9%	57.4%	1.5%	**0.0%**	**0.0%**
	ACC	**76.6%**	69%	68.3%	69%	75.6%	69%
	T(s)	25.5	38.3	25.5+18.4	38.3+17.9	25.5+51.8	38.3+50.3
Age below 24	VB	1.11	0.42	1.04	0.4	**0.08**	0.12
	PGD-B	0.15	0.015	0.058	0.008	0.009	**0.006**
	PGD-R	7.8%	0.5%	0.3%	**0.0%**	**0.0%**	**0.0%**
	ACC	**76.6%**	69%	68.3%	69%	76.3%	69%
	T(s)	25.5	38.3	25.5+18.1	38.3+17.6	25.5+71.3	38.3+80.1
Age from 25 to 54	VB	6.46	2.43	6.04	2.34	**0.039**	0.045
	PGD-B	0.68	0.084	0.28	0.044	0.0053	**0.0015**
	PGD-R	58%	3.3%	39.1%	**0.0%**	**0.0%**	**0.0%**
	ACC	**76.6%**	69%	69.7%	69%	69.3%	69%
	T(s)	25.5	38.3	25.5+17.3	38.3+18.2	25.5+55.5	38.3+47.9
Age from 55 to 64	VB	2	0.75	1.88	0.73	**0.057**	0.11
	PGD-B	0.24	0.0265	0.11	0.016	0.0065	**0.004**
	PGD-R	23.4%	**0.0%**	5.2%	**0.0%**	**0.0%**	**0.0%**
	ACC	**76.6%**	69%	71.7%	69%	75.7%	69%
	T(s)	25.5	38.3	25.5+18.5	38.3+17.6	25.5+47.9	38.3+39.4
Age above 65	VB	2.23	0.84	2.1	0.81	**0.065**	0.094
	PGD-B	0.33	0.029	0.144	0.0192	0.0101	**0.0033**
	PGD-R	33.6%	**0.0%**	7.4%	**0.0%**	**0.0%**	**0.0%**
	ACC	**76.6%**	69%	72.7%	69%	75.7%	69%
	T(s)	25.5	38.3	25.5+18.6	38.3+18.3	25.5+55.4	38.3+45.2

Table 1: Individual Fairness repair on German Credit for different age groups. Observe that after using REGLO, global robustness-related metrics such as VB, PGD-B, and PGD-R were significantly reduced by up to 99.3% with less than 0.9% accuracy drop.

Repair. Given that the number of linear regions for a CPWL DNN is finite, REGLO always terminates with no more repair areas to be found by solving the MILP. □

Ideally, if the target DNN is a ReLU DNN, we can enumerate all the linear regions that violate the global robustness property $\mathbf{P}(X, \Omega, \epsilon)$ by solving multiple MILPs and applying REGLO. By *Iterative Repair*, REGLO will terminate when no more repair area can be found. Thus, we have the soundness guarantee for the resulting DNN.

Specifically, for a *Single Iteration Repair*, the weight change $\Delta\theta$ ensures the satisfaction of $\mathbf{P}(X, \Omega, \epsilon)$ on all the repair areas $\{\mathcal{A}_i\}_{i\in I}$. As we discussed in the *Iterative Repair Section*, the repair may increase the gradient $\|\frac{\partial f}{\partial x}(x')\|$ for $x' \in X \setminus \cup_{i\in I}\mathcal{A}_i$ and causes it to violate $\mathbf{P}(X, \Omega, \epsilon)$. We call any such violation in $X \setminus \cup_{i\in I}\mathcal{A}_i$ a *side effect* of our repair. The following theorem shows that we have guarantees on limiting the side effects of a *Single Iteration Repair*.

Theorem 3 (Limited Side Effect) *Given a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$, a target DNN f, and weight change $\Delta\theta$ from a Single Iteration Repair, we have*

1. For any area $\mathcal{B} \subset \cup_{i\in I}\mathcal{A}_i \subset X$, $\widehat{f} \models \mathbf{P}(\mathcal{B}, \Omega, \epsilon)$;

2. For any area $\mathcal{C} \subset X$ which is not a subset of $\cup_{i\in I}\mathcal{A}_i$, $\widehat{f} \models \mathbf{P}(\mathcal{C}, \Omega, \epsilon + 2L\|\Delta\theta\|\|X\|)$, where L is the Lipschitz constant of f from the input layer to the last hidden layer.

Proof: The constraints of optimization problem (3) ask the resulting DNN must satisfy the global robustness property on $\mathcal{B} \subset \cup_{i\in I}\mathcal{A}_i \subset X$, a subset of the repair areas.

The difference between the resulting DNN $\widehat{f}$ and the target DNN f on any input x can be bounded by the norm of $\Delta\theta$: $\|\widehat{f}(x) - f(x)\| = \|(\theta + \Delta\theta)f_{n-1}(x) - \theta f_{n-1}(x)\| = \|\Delta\theta f_{n-1}(x)\| \leq \|\Delta\theta\| \cdot \|f_{n-1}\| \cdot \|x\| = L\|\Delta\theta\| \cdot \|x\|$, where f_{n-1} is the DNN function from the input layer to the last hidden layer.

For any $x \in \mathcal{C} \subset X$, which is not in a subset of $\cup_{i\in I}\mathcal{A}_i$, and any $\Delta x \in \Omega$:

$$\|\widehat{f}(x + \Delta x) - \widehat{f}(x)\| = \|\widehat{f}(x + \Delta x) - f(x + \Delta x) +$$
$$f(x + \Delta x) - f(x) + f(x) - \widehat{f}(x)\| \leq \epsilon + 2L\|\Delta\theta\|\|X\|$$

□

Corollary 1 (Soundness Guarantees for CPWL DNNs) *Given a global robustness property $\mathbf{P}(X, \Omega, \epsilon)$, a piecewise linear DNN f, and weight change from Iterative Repair, REGLO will terminate with no more repair areas to be found and the resulting DNN $\widehat{f} \models \mathbf{P}(X, \Omega, \epsilon)$.*

Proof: For *Iterative Repair* on a CPWL DNN f, by Theorem 2, REGLO will terminate with no more repair areas to be found. Therefore, the resulting DNN satisfies $\mathbf{P}(X, \Omega, \epsilon)$ outside the repair areas. In addition, by Theorem 3, we have that the resulting DNN satisfies $\mathbf{P}(\mathcal{C}, \Omega, \epsilon)$ for any $\mathcal{C} \subset X$. Combining these results, we have that REGLO will terminate with no more repair areas to be found and the resulting DNN $\widehat{f} \models \mathbf{P}(X, \Omega, \epsilon)$. □

For DNNs that are not piecewise linear, we have the following weaker soundness guarantee.

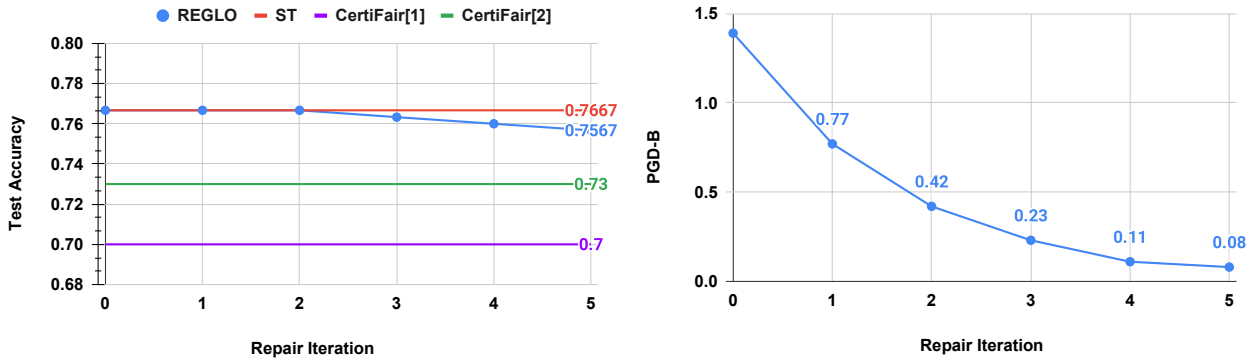

Figure 3: Comparing REGLO with CertiFair on the German Credit dataset. The left figure tracks the changes in accuracy of the REGLO repaired DNN over repair iterations. The three horizontal lines represent the accuracies of ST, CertiFair[1], and CertiFair[2] respectively. The results clearly demonstrate that REGLO outperforms both CertiFair[1] and CertiFair[2] in terms of test accuracy. The figure on the right shows the reduction of PGD-B for the REGLO-repaired DNN over repair iterations.

Algorithm 2: Repair via Barrier Method

Input: Current last-layer weight θ, repair areas $\{\mathcal{A}_i\}_{i\in I}$, and Ω.
Parameter: Initial step size α, initial weight of barrier function t, an early stop threshold δ, and maximal steps K.
Output: $\Delta\theta$

Let $k = 0$.
Let $\Delta\theta = -\theta$ (start from a feasible solution).
while $k < K$ **do**
  Let ϵ_i^* and Δz_i^* be the optimal value and optimal solution of optimization problem (2), respectively.
  Compute the gradient $g = \frac{\partial \epsilon_i^*(\theta+\Delta\theta)}{\partial \Delta\theta}$ according to Equation (5).
  if $\|g\| < \delta$ **then**
    Break
  end if
  Update $\Delta\theta = \Delta\theta - \alpha \cdot g$
  Update α and t.
end while
return $\Delta\theta$.

Corollary 2 (Soundness Guarantee for General DNNs) *Given a global robustness property* $\mathbf{P}(X, \Omega, \epsilon)$*, a DNN* f*, and weight change* $\Delta\theta$ *from Iterative Repair,* REGLO *returns a DNN* $\widehat{f} \models \mathbf{P}(\mathcal{C}, \Omega, \epsilon + 2L\|\Delta\theta\|\|X\|)$ *for any repair area* $\mathcal{C}$*.*

Proof: Since *Iterative Repair* collects both the previously repaired areas and the new repair areas found at one iteration, the last *Single Iteration Repair* will repair all those collected repair areas. Therefore, by Theorem 3, the last *Single Iteration Repair* will return a DNN $\widehat{f}$ that satisfies $\mathbf{P}(\mathcal{C}, \Omega, \epsilon + 2L\|\Delta\theta\|\|X\|)$. □

Experiments

The goal of the experiments is to validate the effectiveness of REGLO in enhancing the global robustness of trained neural networks while preserving their performance. We compare REGLO against six other methods, including four baseline methods, CertiFair (Khedr and Shoukry 2022), a well-known method for training provably fair DNN, and VGRP (Chen et al. 2021), a training procedure to train classifiers with verified global robustness properties.

Experimental Setup

Our prototype tool is implemented in Python. We use Gurobi (Gurobi Optimization, LLC 2021) to solve MILP and use CVXPY (Diamond and Boyd 2016) to solve optimization problem (2). The global robustness bounds used in the optimization problem (2) are derived by ITNE (Wang et al. 2022) with a bound propagation technique similar to β-CROWN (Wang et al. 2021) (without bound refinements for efficiency). The verification bounds (VBs) in the experiments are evaluated using ITNE with bound refinements for tighter estimations. All the experiments were run on machines with CPUs similar to ten-core Intel Xeon E5-2660v3 @ 2.6 GHz with a single GeForce GTX 1080 Graphics Card. Our code is available on GitHub: https://github.com/BU-DEPEND-Lab/REGLO/tree/main/net_repair.

Baseline Methods. To the best of our knowledge, REGLO is the *first* study on DNN repair to satisfy a global robustness property. We consider the following four baseline methods for comparison with REGLO: **ST** is standard training. **AT** is adversarial training using PGD (Madry et al. 2018) on the training data. **ST+Finetune** is standard training followed by fine-tuning for the global robustness property. Specifically, Finetune is to fine-tune (train with a smaller learning rate) a pre-trained DNN with additional counterexamples generated by applying PGD on randomly sampled points in X. **AT+Finetune** is adversarial training followed by fine-tuning for the global robustness property.

For REGLO, we consider two settings where the repair is applied after standard training and after adversarial training, respectively: **ST+REGLO** is standard training followed by REGLO. **AT+REGLO** is adversarial training followed by

Method	TPR(%)	FPR(%)	Acc(%)	F1	Global Robustness Property	Running Time
Cryptojacking						
REGLO	**98.51**	**2.19**	**98.19**	**0.9816**	**Satisfy**	**3.11 seconds**
VGRP	98.40	2.20	98.10	0.9810	**Satisfy**	138.9 mins
Twitter Spam Accounts						
REGLO	**94.64**	9.19	**93.08**	**0.9277**	**Satisfy**	**3.5 mins**
VGRP	87.1	**2.3**	92.20	0.9200	Not Satisfy	Timed Out
Twitter Spam URLs						
REGLO	98.95	**1.4**	**98.60**	**0.9875**	**Satisfy**	**8.7 mins**
VGRP	**99.40**	5.80	96.50	0.9620	**Satisfy**	129.5 mins

Table 2: Comparing REGLO (ST+REGLO) with VGRP. We used the code (https://github.com/surrealyz/verified-global-properties) from the VGRP paper and ran it on our machine. The VGRP results for this benchmark are from the last epoch before the tool timed out after 24 hours. On average, REGLO surpasses VGRP in terms of accuracy, F1, and running time across all three datasets. Specifically, REGLO demonstrates up to 0.88% higher accuracy than VGRP. Furthermore, REGLO is much more efficient than VGRP. The running time for REGLO is up to 99.96% less compared to that of VGRP.

REGLO.

Remark: It has been shown that AT can result in a drop in test accuracy (Tsipras et al. 2018). We refer the readers to Zhang et al. (2019) for a detailed theoretical analysis of the trade-off between robustness and accuracy in neural network training. As our experiments will show, similar to the findings in these studies, AT+REGLO can produce higher improvements in global robustness, but at the cost of lower performance compared with ST+REGLO.

Evaluation Metrics. Given that there is no efficient method for *exact* verification of a global robustness property, we use ITNE (Wang, Huang, and Zhu 2022) to compute an upper-bound of the true global robustness ϵ^*. In addition, we use PGD to evaluate the empirical robustness of randomly sampled inputs in X, as a lower-bound of ϵ^*. We also report accuracy on testing data and runtime. Specifically, we consider the following metrics: **VB** is the verification bound given by ITNE for the global robustness property. **PGD-B** is the maximum norm difference on outputs between sampled input pairs $(x, x+\Delta x)$ as computed by PGD. **PGD-R** is the ratio of sampled input pairs that violate the global robustness property as computed by PGD. **ACC** is the accuracy on testing data (for classification problems). **T(s)** is the runtime in seconds.

Remark: Note that PGD-B $\leq \epsilon^* \leq$ VB, where ϵ^* is the unknown, true global robustness of the target DNN for X and Ω. Note that Finetune essentially aims at reducing PGD-B as PGD-B is used as a regularizer in its loss function during fine-tuning.

Repair for Individual Fairness. We train a ReLU DNN on the German Credit dataset (Dua and Graff 2017) to predict the credit risks (good or bad) for a person based on certain input features. An ideal DNN predictor should be fair with respect to the sensitive input feature 'age', that is the resulting DNN should satisfy a global robustness property [5] $\mathbf{P}(X, \Omega, \epsilon)$ for $\Omega = \{\Delta x \mid \Delta x_{SF} = 0\}$ with $\epsilon = 0.01$, where SF are all input features other than 'age'. We consider the global robustness properties on input domain $\mathcal{X}$ as well as regions based on age groups: $X_0 = \{x \,|\, x_{age} \leq 24\}$, $X_1 = \{x \mid 25 \leq x_{age} \leq 54\}$, $X_2 = \{x \mid 55 \leq x_{age} \leq 64\}$, or $X_3 = \{x \mid 65 \leq x_{age}\}$ according to (Wikipedia contributors 2022). For REGLO, we search for repair areas by random sampling on X and randomly choose 30 of them to repair. The results are shown in Table 1. REGLO enhances the global robustness-related metrics (VB, PGD-B and PGD-R) by up to 99.3% with a minor drop of 0.9% in accuracy.

Comparing REGLO with CertiFair. In this study, we compare REGLO with CertiFair (Khedr and Shoukry 2022) on the German Credit dataset (for the sensitive input feature 'age'). CertiFair employs two different fairness regularizers: CertiFair[1] is based on the standard binary cross-entropy loss, while CertiFair[2] is based on the output difference of the twin-network. The results, as shown in Figure 3, show that the reduction in accuracy with REGLO is significantly smaller than those with CertiFair. Moreover, the CertiFair-trained DNN still contains a substantial number of counterexamples (found through random sampling of pairs of input points): 4.94% for CertiFair[1] and 13.59% for CertiFair[2] (Khedr and Shoukry 2022). In contrast, even random sampling with PGD attacks is unable to find any counterexample for the REGLO-repaired network (PGD-R of ST+REGLO is 0.0%). Furthermore, REGLO does not require access to the original training data which CertiFair does. This makes REGLO a more practical and desirable option for applications where data access may be restricted or limited.

Comparing REGLO with VGRP. In this experiment, we compare REGLO with VGRP (Chen et al. 2021) on three datasets, Cryptojacking (Kharraz et al. 2019), Twitter Spam Accounts (Lee, Eoff, and Caverlee 2011), and Twitter Spam URLs (Kwon, Baig, and Akoglu 2017) (the same dataset used in VGRP paper). VGRP uses a so-called booster-fixer scheme to train classifiers that can guarantee the satisfaction of global robustness properties. Table 4 gives statistics about these datasets in terms of the sizes of training and test

[5] We define any individual fairness property as a global robustness property in Definition

		ST	AT	ST+Finetune	AT+Finetune	ST+REGLO	AT+REGLO
X=X_0	VB	232.3	59.4	229.5	58.7	**7.6**	9.2
	PGD-B	5.6	0.7	3.8	0.4	0.2	**0.08**
	PGD-R	100.0%	94.4%	100.0%	18.7%	**0.0%**	**0.0%**
	ACC	**98.1%**	91.5%	96.5%	85.9%	97.4%	86.5%
	T(s)	163.8	2787.2	163.8+238.3	2787.2+229.7	163.8+916.3	2787.2+612.8
X=X_{20}	VB	240.2	54.3	236.2	49.8	**2.3**	4.5
	PGD-B	8.3	0.4	5.2	0.3	0.08	**0.06**
	PGD-R	100.0%	37.4%	100.0%	0.3%	**0.0%**	**0.0%**
	ACC	**98.1%**	91.5%	95.8%	86.7%	90.9%	79.9%
	T(s)	163.8	2787.2	163.8+235.3	2787.2+222.7	163.8+941.6	2787.2+607.2
X=X_{40}	VB	246.4	63.8	240.8	61.1	**4.1**	5.8
	PGD-B	6.4	0.7	4.5	0.5	0.1	**0.09**
	PGD-R	100.0%	99.9%	100%	36.8%	**0.0%**	**0.0%**
	ACC	**98.1%**	91.5%	97.0%	86.3%	96.0%	79.5%
	T(s)	163.8	2787.2	163.8+221.3	2787.2+222.9	163.8+956.7	2787.2+865.3

Table 3: It can be observed that after using REGLO, global robustness-related metrics such as VB, PGD-B, and PGD-R were significantly reduced by up to $99\%, 99\%$ and 100% while keeping the accuracy above 90%. Furthermore, in most cases, after using REGLO, VB of the trained networks becomes less than 5 which is lower than PGD-B of ST (note that PGD-B is a lower bound of the unknown, true global robustness whereas VB is an upper bound). Although the DNNs trained via AT has already achieved a PGD-B of less than 1.0 as well as PGD-R lower than that of ST, AT+REGLO could further reduce PGD-B to less than 0.1 and PGD-R to 0%. In contrast, Finetune only produced in minor improvements on these metrics.

Dataset	# Training	# Test	# Features
Cryptojacking	2,800	1,200	7
Twitter Spam Accounts	36,000	4,000	15
Twitter Spam URLs	295,870	63,401	25

Table 4: The three datasets that we use to evaluate REGLO against VGRP (Chen et al. 2021).

sets and the number of input features. The global robustness property is defined as $\mathbf{P}(X, \{\Delta x \mid \|\Delta x\| \leq \delta\}, \epsilon)$, where $\delta = 0.2, \epsilon = 0.1$ for Cryptojacking, $\delta = 0.1, \epsilon = 5$ for Twitter Spam Accounts, and $\delta = 1.5, \epsilon = 10$ for Twitter Spam URLs (the same setting as those in (Chen et al. 2021)). The VGRP's booster-fixer training procedure requires a costly verification in each epoch, and the results in Table 2 indicate that while VGRP can train classifiers that satisfy the specified global robustness properties (when it finishes before timing out after 24 hours), it results in a sizeable drop in accuracy (up to 7.54%). In comparison, REGLO is much more efficient (reduce the running time by up to 99.86%) and does not compromise accuracy.

Repair for Norm-bounded Global Robustness. We train a convolutional DNN with ReLU units on the GTSRB dataset (Stallkamp et al. 2011). We consider a norm-bounded global robustness property $\mathbf{P}(X, \Omega, \epsilon)$, where $\Omega = \{\Delta x \mid \|\Delta x\| \leq \delta\}$, $\delta = 1/255$, $\epsilon = 0.5$, X is a hyper-rectangle obtained from the training inputs with the same class label, i.e., $X_k = \{x \mid x^{l_k} \leq x \leq x^{u_k}\}$, where $x_i^{l_k}$ and $x_i^{u_k}$ are the 30th and 70th percentile of the data in the input space for every dimension i and a fixed class k. For REGLO, we randomly choose 30 areas to repair. We train the models with the data of all 42 classes in GTSRB and present results on 3 of those classes (due to limited space) in Table 3. REGLO reduces VB and PGD-B of the classifiers by up to 99% while Finetune reduces those by no more than 10%. Furthermore, REGLO is able to reduce PGD-R of ST from 100% to 0% while Finetune only reduces PGD-R slightly and also causes a considerable drop in accuracy.

Conclusion and Future Work

REGLO is the *first* work that enables provable repair of neural networks for global robustness properties. Experimental results demonstrate the effectiveness of the approach against multiple baselines. Achieving such deterministic guarantees, however, can be challenging when we apply the technique to larger networks. Current state-of-the-art global robustness verification tools such as ITNE (which we use as a subroutine in REGLO) only scale to moderate-size networks. Another challenge to scalability lies in finding the violating linear regions. As the size of the network increases, solving the MILP problem can become very expensive. In addition, the number of linear regions of a ReLU network grows exponentially in the number of layers and polynomially in the number of neurons (or layer width) (Montúfar et al. 2014). Thus, for large DNNs we have to resort to random sampling for finding the violating regions as described in Repair Areas Section. One future direction is to consider statistical techniques to sidestep the inherent complexity of aiming for such deterministic guarantees. Another direction is to investigate alternative repair strategies other than weight modifications as weight changes can result in compromise in performance.

Acknowledgments

We gratefully acknowledge the support from the National Science Foundation awards CCF-1646497, CNS-1834701,

IIS-1724341, CNS-2038853, ONR grant N00014-19-1-2496, the US Air Force Research Laboratory (AFRL) under contract number FA8650-16-C-2642. This work is also supported by the grant EP/Y002644/1 under the EPSRC ECR International Collaboration Grants program, funded by the International Science Partnerships Fund (ISPF) and the UK Research and Innovation.

References

Adel, T.; Valera, I.; Ghahramani, Z.; and Weller, A. 2019. One-network adversarial fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2412–2420.

Arora, R.; Basu, A.; Mianjy, P.; and Mukherjee, A. 2016. Understanding Deep Neural Networks with Rectified Linear Units. *CoRR*, abs/1611.01491.

Ben-Tal, A.; El Ghaoui, L.; and Nemirovski, A. 2009. *Robust optimization*, volume 28. Princeton university press.

Benussi, E.; Patane, A.; Wicker, M.; Laurenti, L.; and Kwiatkowska, M. 2022. Individual Fairness Guarantees for Neural Networks. *arXiv preprint arXiv:2205.05763*.

Boyd, S.; and Vandenberghe, L. 2004. *Convex optimization*. Cambridge university press.

Carlini, N.; Athalye, A.; Papernot, N.; Brendel, W.; Rauber, J.; Tsipras, D.; Goodfellow, I.; Madry, A.; and Kurakin, A. 2019. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*.

Casadio, M.; Komendantskaya, E.; Daggitt, M. L.; Kokke, W.; Katz, G.; Amir, G.; and Refaeli, I. 2022. Neural network robustness as a verification property: A principled case study. In *International Conference on Computer Aided Verification*, 219–231. Springer.

Celis, L. E.; and Keswani, V. 2019. Improved adversarial learning for fair classification. *arXiv preprint arXiv:1901.10443*.

Chen, Y.; Wang, S.; Qin, Y.; Liao, X.; Jana, S.; and Wagner, D. 2021. Learning security classifiers with verified global robustness properties. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 477–494.

Chen, Z.; Li, Q.; and Zhang, Z. 2022. Self-Healing Robust Neural Networks via Closed-Loop Control. *arXiv preprint arXiv:2206.12963*.

Diamond, S.; and Boyd, S. 2016. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83): 1–5.

Dong, G.; Sun, J.; Wang, J.; Wang, X.; and Dai, T. 2020. Towards Repairing Neural Networks Correctly. *arXiv preprint arXiv:2012.01872*.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Dwork, C.; Hardt, M.; Pitassi, T.; Reingold, O.; and Zemel, R. 2012. Fairness through Awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS'12, 214—-226. New York, NY, USA: Association for Computing Machinery. ISBN 9781450311151.

Fan, J.; and Li, W. 2021. Adversarial Training and Provable Robustness: A Tale of Two Objectives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35-8, 7367–7376.

Fischetti, M.; and Jo, J. 2017. Deep neural networks as 0-1 mixed integer linear programs: A feasibility study. *arXiv preprint arXiv:1712.06174*.

Fu, F.; and Li, W. 2021. Sound and Complete Neural Network Repair with Minimality and Locality Guarantees. In *International Conference on Learning Representations*.

Goldberger, B.; Katz, G.; Adi, Y.; and Keshet, J. 2020. Minimal Modifications of Deep Neural Networks using Verification. In *LPAR*, volume 2020, 23rd.

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

Gurobi Optimization, LLC. 2021. Gurobi Optimizer Reference Manual.

Hörmander, L. 2015. *The analysis of linear partial differential operators I: Distribution theory and Fourier analysis*. Springer.

Huang, C.; Fan, J.; Chen, X.; Li, W.; and Zhu, Q. 2020. Divide and Slide: Layer-Wise Refinement for Output Range Analysis of Deep Neural Networks. In *International Conference on Embedded Software (EMSOFT)*.

John, P. G.; Vijaykeerthy, D.; and Saha, D. 2020. Verifying Individual Fairness in Machine Learning Models. *CoRR*, abs/2006.11737.

Katz, G.; Barrett, C.; Dill, D. L.; Julian, K.; and Kochenderfer, M. J. 2017. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In Majumdar, R.; and Kunčak, V., eds., *Computer Aided Verification*, 97–117. Cham: Springer International Publishing. ISBN 978-3-319-63387-9.

Kharraz, A.; Ma, Z.; Murley, P.; Lever, C.; Mason, J.; Miller, A.; Borisov, N.; Antonakakis, M.; and Bailey, M. 2019. Outguard: Detecting in-browser covert cryptocurrency mining in the wild. In *The World Wide Web Conference*, 840–852.

Khedr, H.; and Shoukry, Y. 2022. Certifair: A framework for certified global fairness of neural networks. *arXiv preprint arXiv:2205.09927*.

Kwon, H.; Baig, M. B.; and Akoglu, L. 2017. A domain-agnostic approach to spam-url detection via redirects. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 220–232. Springer.

Lee, G.-H.; Alvarez-Melis, D.; and Jaakkola, T. S. 2019. Towards robust, locally linear deep networks. *arXiv preprint arXiv:1907.03207*.

Lee, K.; Eoff, B.; and Caverlee, J. 2011. Seven months with the devils: A long-term study of content polluters on twitter. In *Proceedings of the international AAAI conference on web and social media*, volume 5, 185–192.

Leino, K.; Fromherz, A.; Mangal, R.; Fredrikson, M.; Parno, B.; and Păsăreanu, C. 2022. Self-correcting Neural Networks for Safe Classification. In *Software Verification and Formal Methods for ML-Enabled Autonomous Systems: 5th*

International Workshop, FoMLAS 2022, and 15th International Workshop, NSV 2022, Haifa, Israel, July 31-August 1, and August 11, 2022, Proceedings, 96–130. Springer.

Leino, K.; Wang, Z.; and Fredrikson, M. 2021. Globally-robust neural networks. In *International Conference on Machine Learning*, 6212–6222. PMLR.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.

Mirman, M.; Gehr, T.; and Vechev, M. 2018. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, 3578–3586.

Mitchell, E.; Lin, C.; Bosselut, A.; Finn, C.; and Manning, C. D. 2021. Fast Model Editing at Scale. *CoRR*, abs/2110.11309.

Montúfar, G.; Pascanu, R.; Cho, K.; and Bengio, Y. 2014. On the Number of Linear Regions of Deep Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2924—-2932. Cambridge, MA, USA: MIT Press.

Refaeli, I.; and Katz, G. 2022. Minimal multi-layer modifications of deep neural networks. In *Software Verification and Formal Methods for ML-Enabled Autonomous Systems: 5th International Workshop, FoMLAS 2022, and 15th International Workshop, NSV 2022, Haifa, Israel, July 31-August 1, and August 11, 2022, Proceedings*, 46–66. Springer.

Runje, D.; and Shankaranarayana, S. M. 2022. Constrained Monotonic Neural Networks. *arXiv preprint arXiv:2205.11775*.

Serra, T.; Tjandraatmadja, C.; and Ramalingam, S. 2017. Bounding and Counting Linear Regions of Deep Neural Networks. *CoRR*, abs/1711.02114.

Seshia, S. A.; Desai, A.; Dreossi, T.; Fremont, D. J.; Ghosh, S.; Kim, E.; Shivakumar, S.; Vazquez-Chanlatte, M.; and Yue, X. 2018. Formal specification for deep neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, 20–34. Springer.

Shafahi, A.; Najibi, M.; Ghiasi, M. A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

Singh, G.; Gehr, T.; Püschel, M.; and Vechev, M. 2019. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL): 1–30.

Sotoudeh, M.; and Thakur, A. V. 2021. Provable repair of deep neural networks. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, 588–603.

Stallkamp, J.; Schlipsing, M.; Salmen, J.; and Igel, C. 2011. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *The 2011 International Joint Conference on Neural Networks*, 1453–1460.

Tao, G.; Sun, W.; Han, T.; Fang, C.; and Zhang, X. 2022. RULER: discriminative and iterative adversarial training for deep neural network fairness. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 1173–1184.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2018. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*.

Wadsworth, C.; Vera, F.; and Piech, C. 2018. Achieving fairness through adversarial learning: an application to recidivism prediction. *arXiv preprint arXiv:1807.00199*.

Wang, S.; Pei, K.; Whitehouse, J.; Yang, J.; and Jana, S. 2018. Formal security analysis of neural networks using symbolic intervals. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 1599–1614.

Wang, S.; Zhang, H.; Xu, K.; Lin, X.; Jana, S.; Hsieh, C.-J.; and Kolter, J. Z. 2021. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. In *Advances in Neural Information Processing Systems*, volume 34.

Wang, Y.; Ma, X.; Bailey, J.; Yi, J.; Zhou, B.; and Gu, Q. 2019. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning*, 6586–6595.

Wang, Z.; Huang, C.; and Zhu, Q. 2022. Efficient global robustness certification of neural networks via interleaving twin-network encoding. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 1087–1092. IEEE.

Wang, Z.; Wang, Y.; Fu, F.; Jiao, R.; Huang, C.; Li, W.; and Zhu, Q. 2022. A Tool for Neural Network Global Robustness Certification and Training. *arXiv preprint arXiv:2208.07289*.

Wikipedia contributors. 2022. Demographic profile — Wikipedia, The Free Encyclopedia. [Online; accessed 10-August-2022].

Wong, E.; Rice, L.; and Kolter, J. Z. 2020. Fast is better than free: Revisiting adversarial training. *International Conferences on Learning Representations*.

Yeom, S.; and Fredrikson, M. 2020. Individual fairness revisited: Transferring techniques from adversarial robustness. *arXiv preprint arXiv:2002.07738*.

Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Boning, D.; and Hsieh, C.-J. 2020. Towards Stable and Efficient Training of Verifiably Robust Neural Networks. *International Conference on Learning Representations*.

Zhang, H.; Weng, T.-W.; Chen, P.-Y.; Hsieh, C.-J.; and Daniel, L. 2018. Efficient neural network robustness certification with general activation functions. In *Advances in neural information processing systems*, 4939–4948.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; El Ghaoui, L.; and Jordan, M. 2019. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, 7472–7482. PMLR.