Post-processing Counterexample-guided Fairness Guarantees in Neural Networks

Kiarash Mohammadi,[1,2] **Aishwarya Sivaraman,** [2] **Golnoosh Farnadi** [1,2,4]

[1] Mila, [2] University of Montreal, [3] UCLA, [4] HEC Montreal

Abstract

There is an increasing interest in adopting high-capacity machine learning models such as deep neural networks to semi-automate human decisions. Hence, it is crucial that these models guarantee similar decisions for similar individuals. To ensure such a fair decision, it is necessary to construct tools capable of verifying and enforcing fairness constraints. In this work, we propose methods to guarantee fairness of a neural network via verification using mixed-integer programming. We show, that it is possible to guarantee individual fair prediction without intervening in the model, efficiently and with little to no loss in accuracy.

Introduction

Deep neural networks are increasingly used to make sensitive decisions, including financial decisions such as whether to give a loan to an applicant (Hardt, Price, and Srebro 2016), recidivism risk assessments (Julia Angwin and Kirchner 2016), salary prediction (BBC 2018), etc. In these settings, for ethical, and legal reasons, it is of utmost importance that decisions made are fair. For example, all else being equal, one would expect two individuals with different gender receive the same hiring decision. However, prior studies have shown that models trained on data are prone to bias on the basis of sensitive attributes such as race, gender, age, etc. (Larson et al. 2016; Buolamwini and Gebru 2018)

It has been shown that even if sensitive features such as race and gender are withheld from the model, the model can still be unfair as it is often possible to internally reconstruct sensitive features that are encoded in data. Guaranteeing fairness not only helps organizations to address laws against discrimination, but also helps users to better trust and understand the learned model (Bastani, Zhang, and Solar-Lezama 2019).

Most prominent definitions of fairness in machine learning can be largely categorized into *individual fairness* and *group fairness*. While group fairness minimizes the impact that discrimination has on the groups of individuals on average, individual fairness is based on the intuition that similar individuals should be treated similarly. Recently, various definitions of group fairness and individual fairness have been introduced. Group fairness measures define specific groups in the population and require that particular statistics, computed based on model decisions, should be equal for all groups (Hardt, Price, and Srebro 2016; Dwork et al. 2012). The group fairness notions are generally hard to formally guarantee fairness for all input points (Kearns et al. 2018; Ruoss et al. 2020). In contrast, individual notions of fairness (Galhotra, Brun, and Meliou 2017; Dwork et al. 2012) are easier to specify as logical constraints to guarantee fairness, as they explicitly require that similar individuals in the population are treated similarly. Specifically, in this paper, we focus on an individual fairness notion introduced by (Galhotra, Brun, and Meliou 2017). This notion says that a model is fair if, the decision of the model is the same for any two individuals with various combination of sensitive attributes, when nonsensitive attributes are fixed.

Our proposed approach to guarantee individual fairness is through formal verification methods. We focus on guaranteeing fairness at prediction time via verification of a trained model. By verification, we mean evaluating whether a classifier satisfies a specified notion of fairness. Recently a few works started to research on formal verification of neural network models (Liu et al. 2019). Often the verification problem is formulated as a satisfiability problem in which given a property to prove, they attempt to discover a counterexample that would make the property false. Prior work on individual fairness verification have focused on identifying the absence of unfair predictions using verification (John, Vijaykeerthy, and Saha 2020), or they only guarantee fairness for training data points (Ruoss et al. 2020). If a trained model is not fair, these methods fail to guarantee fair predictions for all points in the input domain. In this work, we propose the first method for addressing this challenge. At a high level, our approach is based on the observation that we can guarantee fairness at prediction time - as a post-processing step, without changing the model- by counting the number of fairness counterexamples for a given test point.

Our experimental evaluation on three datasets, i.e., COMPAS, German, and Adult, shows that we can guarantee fairness at prediction time with little to no loss in accuracy. Further, we show that our approach to guarantee fairness as a post-processing step, do not affect the prediction time.

Methodology

In this section, we formalize individual fairness verification and construct definitions for guaranteed fair predictions. We start with preliminaries on Mixed-Integer Program (MIP) encoding of neural networks for fairness verification. Then, we explain how we use verification as a tool to guarantee fair predictions.

ReLU Neural Networks (NN) generalize well and are widely used (Glorot, Bordes, and Bengio 2011; Xu, Choy, and Li 2016; dos Santos et al. 2019), particularly in the context of verification (Katz et al. 2017; Huang et al. 2017) and robustness. Hence, in this paper we guarantee fair predictions for functions produced by ReLU networks. While the methods discussed here can be extended to any NN architecture with piece-wise linear activation functions, we focus on fully connected ReLU networks for simplicity.

Let $\mathcal{X}$ as the input space consisting of d features, and suppose that it is a compact finite subset $\mathcal{X} = [L', U']^d$ of $\mathbb{R}^d$ where L, U define the domain of features. Let $\mathcal{Y}$ be the output space. We consider binary classification tasks where $\mathcal{Y} \in \{0, 1\}$. Our goal will be to guarantee *individually fair* predictions from f in some sensitive input features. We refer to the specific notion of individual fairness called *Causal Discrimination* that is proposed by Galhotra, Brun, and Meliou (2017). Causal Discrimination is defined as:

Definition 1. (Causal Discrimination) Assume a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X} \equiv N \times S$ and S denotes protected or sensitive attributes, and N denotes all additional attributes describing the individual, such that $\mathcal{X}[1 \dots k] \in N$ and $\mathcal{X}[k+1 \dots d] \in S$. We define f to be *individually fair* in sensitive features S iff for any two points $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ where $\boldsymbol{x}[i] = \boldsymbol{x}'[i],\ \forall i \in \{1 \dots k\}$, we have that $f(\boldsymbol{x}) = f(\boldsymbol{x}')$.

For example, when two individuals apply for a loan, a model is fair according to the causal discrimination, if the decision to give loans to identical individuals of one race and another race is the same.

Neural Networks Encoding using MIP

The field of neural network verification (Liu et al. 2019), motivated by *adversarial examples*, aims at formally verifying properties of trained models. While several approaches to encode neural networks for verification within problem-solving frameworks have been studied (Liu et al. 2019; Bunel et al. 2018); we use the encoding and optimization approach presented in (Mohammadi et al. 2021) for the context of decision-making scenarios. Their techniques are significantly faster than other verification approaches, which is crucial to our goal of guaranteeing fairness at prediction time.

Given an n-layer fully-connected ReLU neural network with a single output where the width of each layer is represented by t_i, the values of neurons before applying ReLU is represented by vector $\boldsymbol{z}_i,\ \forall i \in \{0 \dots n\}$ ($\boldsymbol{z}_0$ being the input), and their values after ReLU by $\hat{\boldsymbol{z}}_i,\ \forall i \in \{1 \dots n\}$, Tjeng and Tedrake (2017) propose the following MIP encoding, $\forall i \in \{1 \dots n\}$:

$$\boldsymbol{z}_i = \boldsymbol{W}_i \hat{\boldsymbol{z}}_{i-1} + \boldsymbol{b}_i \tag{1a}$$

$$\begin{aligned} \boldsymbol{\delta}_i \in \{0,1\}^{t_i}, \quad \hat{\boldsymbol{z}}_i \geqslant 0, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{u}_i \cdot \boldsymbol{\delta}_i, \\ \hat{\boldsymbol{z}}_i \geqslant \boldsymbol{z}_i, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{z}_i - \boldsymbol{l}_i \cdot (1 - \boldsymbol{\delta}_i) \end{aligned} \tag{1b}$$

The first part (1a) encodes the linear relationship before ReLU. The second part (1b) encodes the ReLU activation function, for $\hat{\boldsymbol{z}} = ReLU(z) = \max(0, z)$. $\boldsymbol{\delta}_i$ is a vector of binary variables representing the state of each ReLU as *non-active* or *active*. This encoding relies on bounds on the values of neurons, $\boldsymbol{l}_i, \boldsymbol{u}_i$. These bounds are computed using a linear approximation of the network proposed by Ehlers (2017), given the bounds on input $\boldsymbol{l}_0 = L', \boldsymbol{u}_0 = U'$. Moreover, in this work, we assume discrete domains over the input variables, hence the *mixed-integer* program. This encoding of neural networks allow us to add additional constraints, and we can verify different properties of a model by adding property-based constraints to (1) to achieve verification through optimization, which we will discuss in the next section.

Individual Fairness Verification

Formal properties of functions are often characterized in terms of their counterexamples. Counterexample-guided algorithms are prevalent in the field of formal methods (Clarke et al. 2000; Solar-Lezama et al. 2006) and neural network verification (Sivaraman et al. 2020). The techniques proposed in this paper will be centered around using counterexamples to the individual fairness specification defined in Definition 1.

Definition 2. A pair of inputs $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ is an *individual fairness counterexample (CE) pair* in sensitive features S of function $f : \mathcal{X} \rightarrow \mathcal{Y}$ iff (i) $\boldsymbol{x}[i] = \boldsymbol{x}'[i],\ \forall i \in \{1 \dots k\}$, and (ii) $f(\boldsymbol{x}) \neq f(\boldsymbol{x}')$.

Given a binary classifier f implemented via a ReLU NN and an individual $\boldsymbol{x}$, we identify fairness counterexamples to this individual using Definition 2. This process involves adding the two kinds of constraints defined in Definition 2 to the MIP encoding from Equation 1. The feasible set of this optimization problem is explored using an optimizer backend (Gurobi Optimization 2020) and if there exists a solution satisfying these constraints, we will have a fairness counterexample. Formally, the following constraints will be added:

$$\boldsymbol{z}_{0,i} = \boldsymbol{x}[i],\ \forall i \in \{1 \dots k\} \tag{2a}$$

$$\hat{z}_n = 1 - f(\boldsymbol{x}) \tag{2b}$$

where $\boldsymbol{z}_{0,i}$ is the variable associated with the i-th neuron in layer 0 (input layer). Concretely, this fairness verification approach searches for a counterexample with the same non-sensitive features as $\boldsymbol{x}$ and any assignments to sensitive features ($\boldsymbol{z}_{0,i}$ where $i \in \{k+1 \dots d\}$), constraining the output of the NN, $\hat{z}_n$, to have the flipped label compared to $f(\boldsymbol{x})$.

Our approach to fairness verification looks for a counterexample of individual fairness for a given point $\boldsymbol{x}$, i.e., if a counterexample is found, individual fairness is not satisfied for this individual w.r.t. Definition 1 and the absence of such counterexamples verify the fairness only for this particular individual. To extend verification to individual fairness in the general case, it is enough to define two MIP encodings given in Equation (1) for the same neural network where (i) a set of constraints on their input layer guarantee the variables corresponding to nonsensitive features to be equal, and (ii) a constraint on the output of the two NNs guarantees the prediction to be different. If a solution $(\boldsymbol{x}, \boldsymbol{x}')$ is found for this aggregate MIP, then individual fairness is not met, otherwise it holds in the general case. Formally, the following constraints are added to the aggregate MIP encoding:

$$\boldsymbol{z}_{0,i} = \boldsymbol{z}'_{0,i},\ \forall i \in \{1 \dots k\} \tag{3a}$$

$$\hat{z}_n = 1 - \hat{z}'_n \tag{3b}$$

where $\boldsymbol{z}$ denotes pre-activation variables associated with the first MIP encoding of the network, and $\boldsymbol{z}'$ for the second MIP encoding of the same network.

While this setup allows us to verify fairness of a learned function, it is not clear how to *guarantee* fairness. In the next section we show how to use individual fairness verification to guarantee fair predictions for all individuals.

Guaranteed Fair Predictions

A naive approach to guarantee fair predictions w.r.t. Definition 1, would be to return the same output for all individuals, e.g., the most frequent label in the training set. While this satisfies individual fairness, it leads to poor model performance (see Table 2 in Section Evaluation). However, this gives us intuition to a better approach, instead of returning majority decision for all individuals in the domain, we could return the majority decision for a group of individuals who share nonsensitive attributes.

More precisely, we apply function h to the output of f to obtain individually fair output for a given input x:

$$h\left(f(\boldsymbol{x})\right) = \mathbb{1}\left(\left(\sum_{\boldsymbol{x}' \in A(\boldsymbol{x})} f(\boldsymbol{x}') - \mathbb{1}(f(\boldsymbol{x}') = 0)\right) \geqslant 0\right) \tag{4}$$

where:

$$\begin{aligned} A(\boldsymbol{x}) := \{X \mid\ & X[1] = \boldsymbol{x}[1], \dots, X[k] = \boldsymbol{x}[k], \\ & X[k+1] = a_{k+1}, \dots, X[d] = a_d; \\ & \forall a_{k+1}, \dots, a_d \in [L'_{k+1,\dots,d}, U'_{k+1,\dots,d}]^{d-k}\} \end{aligned} \tag{5}$$

Theorem 1. For any function f and for any input $\boldsymbol{x} \in \mathcal{X}$ with S as sensitive features, $h(f(\boldsymbol{x}))$ is individually fair in S.

Proof. The proof is trivial: h outputs the same decision for all points within the group of all assignments to the sensitive attributes given fixed nonsensitive attributes of $\boldsymbol{x}$, thus, no fairness CE pair exists. □

So far we have established a way to guarantee fair predictions for all input points based on majority decision captured in function h. To identify the majority decision, we need a way of counting the frequency of each label within the given group of assignments specified by fixing the nonsensitive features in $\boldsymbol{x}$. We propose two ways to do so.

Counting by Enumeration The first approach to computing majority decision would be to enumerate all possible assignments of sensitive attributes for a fixed set of nonsensitive attributes. Concretely, given a test point $\boldsymbol{x}$, we use backtracking and traverse all possible assignments to the sensitive features, counting the frequency of each label. We stop when we have found $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ assignments that result in a specific label, assuring that this is the majority decision.

Computational complexity of this approach grows with the size of sensitive attributes and the domain size of each sensitive attribute. Therefore, exhaustively enumerating even $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ will increase prediction time significantly. This motivates the next approach.

Counterexample-guided Counting To compute h and identify majority decision, we leverage our efficient MIP framework to find counterexamples. The key idea is to count if there are $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ number of counterexamples for the MIP encoding of f constrained to $\boldsymbol{x}$. Concretely, given a test sample $\boldsymbol{x}$, we use the pool search mode of Gurobi Optimization (2020) to explore the MIP search tree in pursuit of $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ counterexamples where their labels is opposite to that of $f(\boldsymbol{x})$. If that many solutions are found, then the majority decision for the group of assignments specified by $\boldsymbol{x}$ is opposite to that of $f(\boldsymbol{x})$, otherwise the prediction remains unchanged. Note that this choice of the label is empirical, i.e., we could as well always look for solutions having constant label 1 to figure out the majority label. The general scheme of this approach is shown in Algorithm 1. The algorithm takes as input the learned weights and biases $\boldsymbol{W}, \boldsymbol{B}$ (containing $\boldsymbol{W_i}, \boldsymbol{b}_i$ in Equation (1)) of a ReLU NN, as well as a sample $\boldsymbol{x}$. The initial label of $\boldsymbol{x}$ is specified by y in line 3. In line 4, the MIP encoding of the NN is obtained as per Equation 1 and constraints from Equation 2 specifying a counterexample for $\boldsymbol{x}$ are obtained in the following line. In line 6, the MIP search tree is explored to find the specified number of such counterexamples; if it finds less than that, the final prediction does not change, otherwise it flips.

Algorithm 1: Counterexample-guided Counting to Guarantee Fair Predictions

```
1: Input: $\boldsymbol{W}$, $\boldsymbol{B}$, $\boldsymbol{x}$
2: Output: $l \in \{0, 1\}$
3: $y = f_{\boldsymbol{W},\boldsymbol{B}}(\boldsymbol{x})$
4: $\phi_N \leftarrow$ MIPEncoding($\boldsymbol{W}, \boldsymbol{b}$)    ▷ MIP Encoding in Equation 1
5: $\phi_{CE} \leftarrow$ CounterExampleEncoding($\phi_N, \boldsymbol{x}$)    ▷ MIP Constraints in Equation 2
6: $S \leftarrow$ findSolutions($\phi_N, \phi_{CE}, \left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$)
7: if $|S| < \left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ then
8:     return $y$
9: else
10:    return $1 - y$
```

Evaluation

In this section, we evaluate the effectiveness of our methods on three widely known fairness datasets.

- COMPAS (Larson et al. 2016): This contain 12k samples with binary labels ”Low“ and ”High“ for the predicted recidivism risk, as well as 8 features where 3 of them are sensitive : sex (binary), ethnicity ($\in [0, 8]$), and marital status ($\in [0, 6]$).
- German (Bache and Lichman 2013): This consists of 1000 samples with dimensionality 20 and 3 sensitive attributes: sex /marital status ($\in [0, 3]$), age ($\in [19, 75]$), and foreign worker (binary). It is used for binary classification of good or bad credit risks.
- Adult (Adult data 1996): This consists of 30k samples and 14 features with four sensitive features: marital status ($\in [0, 6]$), race ($\in [0, 4]$), sex (binary), and native country ($\in [0, 40]$). The main task is to classify whether an individual's income exceeds $50K per year.

All experiments were run on a machine with 8 GiB RAM and 2.5GHz processor. The implementations are all single-threaded. We use Gurobi-9.1.2[1] as our solver. The baseline model is a constant predictor that returns the most frequent label in the training set. The architecture used for the ReLU Neural Network is 2 hidden layers of size 16. It is trained via a 5-fold cross-validation with grid search over learning rate, number of epochs, and batch size to find the best model.

We investigate the following research questions:

Q1: To what extent does our initial learned model violate individual fairness at prediction?

This can be interpreted with two metrics. i) CE rate: for how many test samples there exists an individual fairness counterexample, i.e., it is possible to change the outcome of the model by only changing the sensitive attributes, and ii) Flip rate: for how many test samples the prediction of the model should be flipped in order to guarantee individual fairness, i.e., not only a counterexample exists for them, but also the label of the counterexample is the majority vote.

Note that to compute CE rate we only need the counterexample finding procedure, while for flip rate we need the CE-guided counting procedure to find the majority vote. Table 1 shows CE rate and flip rate across different datasets. We can observe that there exists fairness counterexamples in the test set of all datasets, so fairness as in Definition 1 does not hold, thus, there is no need to perform the general fairness verification procedure (Equation 3). Also, to guarantee fairness, the final prediction flips for a small subset of the samples for which a counterexample is found.

Data	COMPAS	German	Adult
CE rate	0.25 ± 0.35	30.15 ± 7.89	0.43 ± 0.29
Flip rate	0.03 ± 0.01	7.83 ± 2.95	0.09 ± 0.06

Table 1: CE rate on and Flip rate (%) on the unseen test set

Q2:What is the effect of verification-guided fair prediction on performance?

Our approach only aims to enforce individual fairness as a post-processing step, and does not take the model accuracy into account, thus, we could expect a decay in performance. In Table 2, we report performance of the baseline model (i.e., the constant predictor returning the most frequent label), the best model from grid search (with no fairness criteria), and fairness-guaranteed prediction model. We observe that individually fair models of all three datasets perform better than the baseline models. While there is a small decay in performance from the best model to the fair model on German dataset, no decrease is observed in COMPAS and Adult. This is expected given the low flip rate as shown in Table 1.

	Accuracy (Before)		Accuracy (After)
Data	Baseline	ReLU NN	ReLU NN
COMPAS	83.99 ± 0.42	91.96 ± 0.94	91.96 ± 0.92
German	69.94 ± 1.99	74.87 ± 2.86	74.67 ± 2.83
Adult	75.10 ± 0.49	78.60 ± 1.75	78.61 ± 1.75

Table 2: Performance of the models (%)

Q3: How scalable is our verification-guided fair prediction approach?

We compare scalability of the proposed approaches to guaranteeing fairness for a given input. In table 3 we compare them on a single fold and observe that the CE-guided counting approach performs two orders of magnitude faster than enumeration on German dataset. On Adult, where there are many more test samples, the enumeration approach times out (> 3 hours), while CE-guided counting takes 10 minutes.

Data	# instance	Enumeration (s)	CE-guided Counting (s)
COMPAS	2500	6780.6	158.8
German	200	9374.9	21.9
Adult	6000	timed-out	621.3

Table 3: Runtime in seconds

Conclusion

We propose a novel approach to guarantee individual fairness in neural networks at prediction time using verification. While we show that our approach is capable of efficiently enforcing fairness, an open path to explore is to study the scalability of our method for various NN architectures.

The individual fairness definition is limited, for example, it does not capture the relations among features. An interesting future work would be to extend this with causal fairness measures. Another direction, could be to try binding the counterexamples to follow the distribution of the data. Currently, there are no distribution constraints in our MIP formulation which might produce Out-of-Distribution counterexamples. Finally, our approach to use fairness counterexamples could be used to fine-tune an unfair model towards being fair, which is a promising future work.

[1] https://www.gurobi.com

Acknowledgements

Funding support for project activities has been partially provided by Canada CIFAR AI Chair, Google scholar award, and NSERC Discovery Grants program.

References

Adult data. 1996. https://archive.ics.uci.edu/ml/datasets/adult.

Bache, K.; and Lichman, M. 2013. UCI machine learning repository.

Bastani, O.; Zhang, X.; and Solar-Lezama, A. 2019. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA): 1–27.

BBC. 2018. Gender pay gap: Men still earn more than women at most firms.

Bunel, R.; Turkaslan, I.; Torr, P. H.; Kohli, P.; and Kumar, M. P. 2018. A Unified View of Piecewise Linear Neural Network Verification. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, 4795–4804. Red Hook, NY, USA: Curran Associates Inc.

Buolamwini, J.; and Gebru, T. 2018. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In Friedler, S. A.; and Wilson, C., eds., *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, 77–91. PMLR.

Clarke, E.; Grumberg, O.; Jha, S.; Lu, Y.; and Veith, H. 2000. Counterexample-guided abstraction refinement. In *International Conference on Computer Aided Verification*, 154–169. Springer.

dos Santos, M. C.; Pinheiro, V. H. C.; do Desterro, F. S. M.; de Avellar, R. K.; Schirru, R.; dos Santos Nicolau, A.; and de Lima, A. M. M. 2019. Deep rectifier neural network applied to the accident identification problem in a PWR nuclear power plant. *Annals of Nuclear Energy*, 133: 400–408.

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*, 214–226.

Ehlers, R. 2017. Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks. *CoRR*, abs/1705.01320.

Galhotra, S.; Brun, Y.; and Meliou, A. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, 498–510.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 315–323.

Gurobi Optimization, L. 2020. Gurobi Optimizer Reference Manual.

Hardt, M.; Price, E.; and Srebro, N. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29: 3315–3323.

Huang, X.; Kwiatkowska, M.; Wang, S.; and Wu, M. 2017. Safety verification of deep neural networks. In *International Conference on Computer Aided Verification*, 3–29. Springer.

John, P. G.; Vijaykeerthy, D.; and Saha, D. 2020. Verifying individual fairness in machine learning models. In *Conference on Uncertainty in Artificial Intelligence*, 749–758. PMLR.

Julia Angwin, S. M., Jeff Larson; and Kirchner, L. 2016. Machine Bias.

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 *International Conference on Computer Aided Verification*, 97–117. Springer.

Kearns, M.; Neel, S.; Roth, A.; and Wu, Z. S. 2018. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, 2564–2572. PMLR.

Larson, J.; Mattu, S.; Kirchner, L.; and Angwin, J. 2016. https://github.com/propublica/compas-analysis.

Liu, C.; Arnon, T.; Lazarus, C.; Barrett, C. W.; and Kochenderfer, M. J. 2019. Algorithms for Verifying Deep Neural Networks. *CoRR*, abs/1903.06758.

Mohammadi, K.; Karimi, A.-H.; Barthe, G.; and Valera, I. 2021. Scaling Guarantees for Nearest Counterfactual Explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '21, 177–187. New York, NY, USA: Association for Computing Machinery. ISBN 9781450384735.

Ruoss, A.; Balunović, M.; Fischer, M.; and Vechev, M. 2020. Learning certified individually fair representations. *arXiv preprint arXiv:2002.10312*.

Sivaraman, A.; Farnadi, G.; Millstein, T.; and Van den Broeck, G. 2020. Counterexample-Guided Learning of Monotonic Neural Networks. *Advances in Neural Information Processing Systems*, 33: 11936–11948.

Solar-Lezama, A.; Tancau, L.; Bodik, R.; Seshia, S.; and Saraswat, V. 2006. Combinatorial sketching for finite programs. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, 404–415.

Tjeng, V.; and Tedrake, R. 2017. Verifying Neural Networks with Mixed Integer Programming. *CoRR*, abs/1711.07356.

Xu, L.; Choy, C.-s.; and Li, Y.-W. 2016. Deep sparse rectifier neural networks for speech denoising. In *2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 1–5. IEEE.