Université de Montréal

FETA: Fairness Enforced Verifying, Training, and Predicting Algorithms for Neural Networks

par

Kiarash Mohammadi

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

June 15, 2023

Résumé

L'automatisation de la prise de décision dans des applications qui affectent directement la qualité de vie des individus grâce aux algorithmes de réseaux de neurones est devenue monnaie courante. Ce mémoire porte sur les enjeux d'équité individuelle qui surviennent lors de la vérification, de l'entraînement et de la prédiction des réseaux de neurones. Une approche populaire pour garantir l'équité consiste à traduire une notion d'équité en contraintes sur les paramètres du modèle. Néanmoins, cette approche ne garantit pas toujours des prédictions équitables des modèles de réseaux de neurones entraînés. Pour relever ce défi, nous avons développé une technique de post-traitement guidée par les contre-exemples afin de faire respecter des contraintes d'équité lors de la prédiction. Contrairement aux travaux antérieurs qui ne garantissent l'équité qu'aux points entourant les données de test ou d'entraînement, nous sommes en mesure de garantir l'équité sur tous les points du domaine. En outre, nous proposons une technique de prétraitement qui repose sur l'utilisation de l'équité comme biais inductif. Cette technique consiste à incorporer itérativement des contre-exemples plus équitables dans le processus d'apprentissage à travers la fonction de perte. Les techniques que nous avons développé ont été implémentées dans un outil appelé FETA. Une évaluation empirique sur des données réelles indique que FETA est non seulement capable de garantir l'équité au moment de la prédiction, mais aussi d'entraîner des modèles précis plus équitables.

Mots clés: Équité, Réseaux de Neurones, Vérification

Abstract

Algorithmic decision-making driven by neural networks has become very prominent in applications that directly affect people's quality of life. This paper focuses on the problem of ensuring individual fairness in neural network models during verification, training, and prediction. A popular approach for enforcing fairness is to translate a fairness notion into constraints over the parameters of the model. However, such a translation does not always guarantee fair predictions of the trained neural network model. To address this challenge, we develop a counterexample-guided post-processing technique to provably enforce fairness constraints at prediction time. Contrary to prior work that enforces fairness only on points around test or train data, we are able to enforce and guarantee fairness on all points in the domain. Additionally, we propose a counterexample-guided loss as an in-processing technique to use fairness as an inductive bias by iteratively incorporating fairness counterexamples in the learning process. We have implemented these techniques in a tool called FETA. Empirical evaluation on real-world datasets indicates that FETA is not only able to guarantee fairness on-the-fly at prediction time but also is able to train accurate models exhibiting a much higher degree of individual fairness.

Keywords: Fairness, Bias Mitigation, Neural Networks, Verification

Contents

List of tables

List of figures

List of acronyms and abbreviations

DL	Deep Learning
NN	Neural Network
ReLU	Rectified Linear Unit
SMT	Satisfiability Modulo Theories
MILP	Mixed-Integer Linear Programs
BCE	Binary Cross Entropy
SCM	Structural Causal Model

Acknowledgements

I would like to express my genuine gratitude to the following individuals and organizations who have contributed to the completion of this Master's thesis:

First and foremost, I would like to thank my supervisor, Golnoosh Farnadi, for their invaluable guidance, expertise, and support throughout the entire research process. Their mentorship and insightful feedback have been instrumental in shaping the direction and quality of this work.

I would like to express my appreciation to the thesis committee Dhanya Sridhar, Laurent Charlin, and Golnoosh Farnadi for dedicating time and effort to this thesis.

Moreover, I am deeply grateful to the research community at Mila and Université de Montréal for their assistance, collaboration, and meaningful discussions during the course of this research. Especially, my dear friends and colleagues, Rebecca Salganik, Nicola Neophytou, and Rohan Sukumaran. Their feedback and support have been invaluable in addressing various challenges.

I am grateful to the academic institutions and organizations that provided resources and scholarships that supported my research. Their support has been crucial in facilitating the research process.

I would like to express my deep appreciation to my family and friends for their immense encouragement, understanding, and love throughout this academic journey. Their continuous support has been a constant source of motivation and inspiration.

Finally, I would like to acknowledge the broader research community on trustworthy AI and fairness in machine learning and algorithmic decision-making, whose collective efforts have led in the field moving forward. I am grateful to be part of this community and I wish it foster even more; we need it now more than ever.

Chapter 1

Introduction

Machine learning models, especially end-to-end trained models such as Deep Learning (DL) models [**28**], are being widely used to assist high-risk decision-making scenarios, ranging from university admissions to recidivism risk assessments [**39**]. These models have demonstrated significant performance in capturing complex relationships between features. Among those, Neural Networks (NNs) have become prevalent in many domains, especially with simple non-linearities such as the Rectified Linear Unit (ReLU) [**1**]. While these models have shown superior performance, unanticipated issues have arisen in their deployment. There are numerous examples in consequential scenarios in which machine learning models have demonstrated discriminatory behavior, i.e., unjustifiably favoring a majority over a demographic minority in the most general case. This can range from loan approval [**32**], recidivism risk assessments [**39**], salary prediction [**5**], etc., and has motivated a line of research on *fairness* in machine learning. In these scenarios, we normally face a two-class classification task where a binary decision (also referred to as *intervention*) is made for an individual. This decision or intervention could either be assistive (when the model is classifying for a positive outcome, e.g., job interview) or punitive (when the model is screening for a negative outcome, e.g., high or low risk for loan approval) [**24**]. As motivating examples, we revisit cases in which deployed machine learning models have been the cause of discrimination toward demographic groups.

Punitive intervention case (ProPublica COMPAS Study)[65]. One of the classical examples is the case of the COMPAS algorithm. This is an algorithm used to predict recidivism among defendants in the criminal justice system and it was later discovered that it exhibits discriminatory behavior. The task of the algorithm is to classify individuals as high-risk or low-risk to assist in the decision of granting bail. It has been found that African-Americans were more likely to be classified as high-risk compared to white defendants, even in the case that other features of the individuals are controlled, implying that the decision of the algorithm is biased by skin color.

Assistive intervention case (Amazon's Hiring Tool) [67]. In 2018, it was found that the tool Amazon uses for filtering the resumes of applicants is indeed biased. The algorithm was giving less chance to women, amplifying and perpetuating the bias in its training data, i.e., the historical gender imbalance in the tech industry.

Apple Credit Card Application [6]. There are many more examples for either case and examples that are not necessarily assistive or punitive. For instance, in 2019, Apple launched a credit card application that was accused of gender bias [6]. The scandal went viral when a couple who shared all of their bank accounts, assets, and credit cards, received different credit limits while only their gender was different in their applications. This *difference* in outputs among individuals that are *similar* in inputs can be considered *unfair*.

These examples are only a few of many, that motivate the problem of studying *fairness* in machine learning from an ethical point of view. That already being enough motivation, there are indeed legal grounds for which the problem of fairness in machine learning becomes important. Many countries have anti-discrimination laws in place that prohibit the use of protected attributes such as race, gender, or age as the basis of decision-making in different domains. For example, in the United States, the disparate impact regulations ensure "that public funds, to which all taxpayers of all races contribute, not be spent in any fashion which encourages, entrenches, subsidizes, or results in racial discrimination." [59].

In the context of training machine learning models, 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. [51, 12] 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. It is often the case that either there are still proxies of sensitive features present in the data, or it is possible to reconstruct sensitive features that are encoded in data internally. Thus, there is no straightforward solution.

Prior work on solving this problem can be categorized into addressing two broad fairness criteria: i) *group*-based notions of fairness, and ii) *individual*-based notions of fairness. The former tries to equalize a statistical measure among different demographic groups on average, e.g., demographic parity [20] or equalized odds [32]. On the other hand, individual fairness requires that similar individuals should be treated similarly [20]; it is less straightforward to define since the similarity measure on both the input and output spaces needs to be set. Group-based notions, despite their prevalence, are generally difficult to be formally guaranteed for all the input space [44, 69]. Furthermore, an algorithm that satisfies group fairness could be blatantly unfair from the point of view of individual users [20] as group fairness only considers average measures among demographic groups. The choice of a fairness notion in general is very context-dependent and requires domain knowledge of the complete sociotechnical system. For instance, following the examples mentioned above, for the COMPAS [65] and Amazon [67] examples it seems more appropriate to choose a group fairness notion, and for the Apple

[**6**] example we could leverage an individual fairness notion; however, choosing a specific notion within group/individual fairness notions requires more context. While no fairness notion is superior to another, in this work we focus on satisfying individual fairness [**20**] which states that the distance between the outcome for two individuals should be bounded according to the degree of their similarity.

Working with a fairness notion, prior work either aims at auditing (aka, verifying) fairness or mitigating bias. Auditing group fairness can mostly be done by empirical computation of the fairness metric such as demographic parity [**20**] or equalized odds [**32**]. It is, however, more difficult to audit individual fairness. Prior work either constructs verifiers using Satisfiability Modulo Theories (SMT) [**10**] or Mixed-Integer Linear Programs (MILP) [**8, 45**], or aims at performing adversarial attacks to find violations of individual fairness [**55**]. Bias mitigation techniques normally fall into three categories: i) pre-processing, ii) in-processing, and iii) post-processing. This thesis targets both verification and mitigation problems w.r.t. individual fairness. The closest work to ours in this regard is LCIFR [**69**]. It combines pre-processing and in-processing techniques to propose an approach to learning certified fair representation that maps similar individuals close to each other in the latent space. Moreover, it proposes a technique to certify fairness w.r.t. an individual fairness notion that is similar to ours.

Prior work on providing certified individual fairness, mainly aims at *local* guarantees, however, in this work, we propose algorithms to guarantee individual fairness *globally* and on the whole input space. To the best of our knowledge, this is the first work that provides global guarantees for the individual fairness notion of *causal discrimination* [**25**]. Moreover, our approach has the flexibility to either only improve fairness by regularizing the model for individual fairness, or guarantee fairness at the cost of a slight decrease in accuracy.

Our approach. In this work, we propose algorithms to verify, incorporate, and guarantee individual fairness globally for ReLU Neural Networks. We target the individual fairness notion proposed by [**25**] which states that individuals who differ in sensitive attributes but share non-sensitive attributes must receive the same outcome. This work has three main components:

- CE-Fair Verification: We leverage the rich body of research on automated theorem provers for NNs [**52**] to build a verifier that looks for individual fairness counterexamples, i.e., pair of similar individuals in the whole input domain that do not receive the same outcome.
- CE-Fair Prediction: We propose a pure post-processing approach that tweaks the outputs of the model to satisfy the fairness criteria. Our counterexample-guided approach performs a majority vote over individuals sharing non-sensitive attributes to detect the *fair* output at prediction time. This can be applied to any arbitrary ReLU NN.

- CE-Fair Training: This novel in-processing algorithm acts as a counterexample-guided regularizer that incorporates explicit counterexamples to fine-tune an already trained model toward being fair. We identify individual fairness counterexamples on the training data, inducing additional supervision for training the network, and perform this process iteratively. CE-Fair Training can be complemented by CE-Fair Prediction to guarantee individual fairness with much less damage to accuracy.

We have implemented our tool in an open-source package called "**F**airness **E**nforced Verifying, **T**raining, and Predicting **A**lgorithm" (FETA). We perform extensive experiments on three real-world datasets to showcase the effectiveness of our approach. Figure 1.1 demonstrates a high-level overview of the components of the FETA pipeline. Given a trained ReLU neural network, CE-Fair Verification is performed to examine whether any counterexamples exist and find the counterexample pairs. At this point, CE-Fair Prediction can already be performed to guarantee fairness as post-processing; however, the cost (i.e., accuracy drop) might be high. Instead, one can first fine-tune the model to be fairer by applying the in-processing CE-Fair Training and then apply CE-Fair Prediction for the ultimate model that both guarantees fairness and is accurate.

Main contributions. Our main contributions are as follows.

(1) A practical audit tool to verify individual fairness of arbitrary ReLU NNs that searches for counterexamples violating the fairness criteria

(2) A counterexample-guided prediction-time algorithm that guarantees individual fairness as a post-processing approach

(3) A counterexample-guided re-training algorithm that imposes fairness supervision as an in-processing approach

(4) An open-source package containing the implementation of all the algorithms, along with extensive experiments on real-world datasets

This thesis is the result of two papers: i) a paper titled "Post-processing Counterexample-guided Fairness Guarantees in Neural Networks", accepted in the AAAI 2022 Workshop on Combining Learning and Reasoning, ii) a paper titled "FETA: Fairness Enforced Verifying, Training, and Predicting Algorithms for Neural Networks", submitted as a conference paper and under review at the time of writing this thesis. The former paper has become a part of the latter. The writer of this thesis has: i) developed the idea and theoretical foundation under the guidance of the supervisor, ii) developed the open-source tool (FETA) and conducted extensive experiments on real-world datasets, and iii) has written the presented papers as the first author.

Thesis layout. Chapter 2 studies the related work on fairness mitigation and verification, as well as related work on neural network verification and adversarial learning. It is continued by providing some necessary background on basic machine learning and deep learning concepts, mixed-integer linear programs, and the background on encoding neural networks as MILPs.

Chapter 3 presents the paper titled "FETA: Fairness Enforced Verifying, Training, and Predicting Algorithms for Neural Networks".

Chapter 4 concludes by summarizing the work, mentioning its limitations and societal impact, and discussing the ongoing future efforts on extending the work to causal notions of fairness to address the limitations.

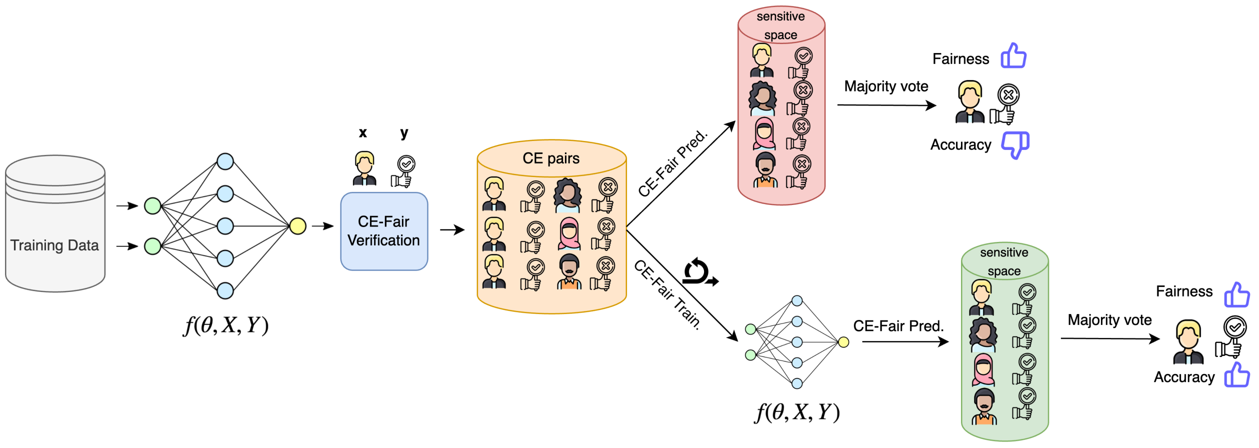

Fig. 1.1. Overview of the FETA pipeline. CE-Fair Verification is performed on an already-trained ReLU neural network and counterexample pairs are found. CE-Fair Prediction can already be applied at this stage to guarantee fairness but the accuracy cost might be high. For that reason, one can perform a few epochs of CE-Fair Training to achieve a more desirable model in terms of fairness and then apply CE-Fair Prediction for the final model.

Chapter 2

Related Work and Background

This chapter studies the closest and most recent work related to different elements of our work. The goal is to situate our work in the literature, motivate the differences, and showcase the novelty and contribution to the literature. Section 2.1 studies our work in the context of fairness in machine learning literature while section 2.2 situates our work in the neural network verification and adversarial example literature. It is followed by sections revisiting some fundamental backgrounds: some basic machine learning concepts in Section 2.3, and mixed-integer linear programs and how neural networks are encoded within this framework in Section 2.4.

2.1. Bias Mitigation Algorithms

Our work contributes to the research on both mitigating and verifying fairness in machine learning models. As discussed earlier, methods that aim to mitigate bias, normally fall into three categories: i) *pre-processing*, ii) *in-processing*, and iii) *post-processing*.

2.1.1. Pre-processing methods

Pre-processing approaches aim to mitigate bias at a data level. They are normally model-agnostic approaches trying to transform the data such that the encoded bias is removed [**13, 41**]. This could be done through data reweighing that aims to associate weights to different group-label pairs such that bias is mitigated [**41**], fair representation learning that tries to map the data into a latent space where bias is mitigated [**85**], or by balancing the representation of different demographic groups to remove disparate impact [**23**]. The closest work to ours in this context is [**69**] which learns representations in a way that similar individuals in the input space, are mapped close to each other in the latent space. It uses the same fairness definition as we do. We compare our approach against [**69**] and report the results in Section 3.5.1.

2.1.2. In-processing methods

These methods normally define extra soft or hard constraints on the optimization problem and aim at regularizing a model through training in order to obey a fairness criterion. In this case, the learning algorithm itself is implicitly or explicitly regularized to prefer a fair model w.r.t. a fairness definition. This is one of the more popular approaches and several works have been proposed to enforce fairness as an in-processing method, mostly for group fairness metrics [**33, 44, 56**]. As for in-processing techniques for individual fairness, prior work falls into two categories based on assuming an individual fairness metric is at hand or not. When it is not possible to compute the metric, prior work relies on having access to an oracle that evaluates the individual fairness of the learning algorithm [**26, 40**]. Otherwise, when the metric is amenable to compute, prior work relies on adversarial learning techniques to enforce fairness [**82, 83**]. Another line of work assumes a Structural Causal Model (SCM) [**61**] underlying the data and aims at constraining the solution space w.r.t. the causal model [**48, 49**]; this is discussed in more detail in Chapter 4.

2.1.3. Post-processing methods

This group of approaches works by tweaking model predictions at inference time in a way that the predictions obey certain fairness criteria [**32, 64, 53, 63, 2**]. They are flexible in the sense that they do not require to re-train the model and are normally model-agnostic. They use approaches such as threshold adjustments for different demographic groups to compensate for the model bias.

In this work, we propose both in-processing (CE-Fair Training) and post-processing (CE-Fair Prediction) bias mitigation algorithms. Our in-processing approach, unlike prior work on in-processing for individual fairness [**26, 40**], does not rely on an oracle for feedback, rather it relies on our CE-Fair Verification algorithm to search for counterexamples and evaluate individual fairness. Moreover, our in-processing CE-Fair Training approach uses exact counterexamples that are guaranteed to violate fairness maximally, as opposed to approximate examples in adversarial learning [**82, 83**]. While we are similar to [**45, 8**] in terms of in-processing, we propose a post-processing approach that, unlike others, guarantees individual fairness. Note that, our in-processing technique is only a means to help improve the quality of the ultimate model when post-processing is applied, i.e., it can be interpreted as a step of a more general algorithm.

Existing post-processing algorithms either focus on group fairness or on individual fairness notions that are different from the one we adopt [**32, 64, 53, 63, 2**]. Our adopted notion is the causal discrimination definition proposed by [**25**]. The post-processing bias mitigation approach proposed in [**53**] relies on the same fairness definition, however, it only supports a single binary sensitive attribute while we support multiple sensitive attributes of different types. Their approach works by flipping the prediction for samples that are likely to have individual fairness bias. To do so, they train a bias detector model on an auxiliary dataset that labels the unprivileged group as either prone to high bias or not. The bias detector is then used at inference time to judge whether an incoming sample is likely to have been discriminated against and flip the prediction accordingly. In contrast, our approach does not rely on a proxy model (that itself could be prone to bias) to determine whether discrimination has happened or not; CE-Fair Verification algorithm can provably determine if a fairness counterexample exists. Moreover, we guarantee individual fairness at prediction time for all the input domain using our post-precessing CE-Fair Prediction algorithm. This makes our approach more versatile and robust, with the ability to accommodate a wider range of real-world scenarios while ensuring fairness.

2.2. Verifying Machine Learning Systems and Adversarial Examples

Our work is related to the literature on neural network verification and adversarial examples [**52**] in the sense that we leverage formal verification tools and mixed-integer programs to encode a ReLU neural network, as well as the fairness notion and different data-type constraints. In this domain, several works have been proposed to formally verify different properties of ReLU-activated neural networks such as robustness to adversarial attacks [**52, 43, 21**]. Prior work on machine learning verification can be classified into (i) verification using satisfiability modulo theory (SMT) or mixed-integer linear programming (MILP) [**43, 76**], and (ii) verification using convex relaxations [**71, 19**]. The first group of works aims at proposing efficient ways to encode neural networks and the desired properties using first-order predicate logic or mixed-integer linear programs. Later, they leverage a backend SMT solver or a MILP optimizer to solve the verification problem. The second group proposes relaxations to the problem such that the optimization is more amenable to solving. In this work, we rely on prior work [**75, 11, 57**] from the first group to encode neural networks and other constraints as MILPs and build a practical and flexible individual fairness verification framework (our CE-Fair Verification and CE-Fair Prediction algorithms).

Prior work on fairness verification has been proposed in the context of probabilistic programs [**3, 4**] or linear kernels [**38**]. In the context of adversarially robust algorithms (which can be divided into empirical [**47, 54**] and certified defenses [**79, 72, 66, 34**]), specifically, [**80, 58, 83**] propose adversarial training-based algorithms for fairness. In our CE-Fair Training we use individual fairness counterexamples and that makes us closely related to these works. In general, we differ in two main ways. First, we are using the fairness definition from [**25**] and, to the best of our knowledge, there is no related work in adversarial training literature for enforcing this notion of individual fairness. Second, the literature in adversarial training mostly focuses on *local* robustness, i.e., they try to enforce robustness criteria in the neighborhood of a training sample rather than globally; however, we search for counterexamples in the whole input domain and ensure fairness on a *global* scale. Our algorithm is able to detect unfairness and guarantee fairness not only in the neighborhood of training samples but for the whole input space.

2.3. Background on Machine Learning

For the sake of completion and definition, we revisit a few fundamental concepts from machine learning and deep learning.

Definition 1. (**Single-output Feed-forward Neural Network**) Simply referred to as a neural network in the thesis, it is a function $F : \mathbb{R}^d \rightarrow \mathbb{R}$ implementing a series of matrix multiplications, each happening in a *layer*. Let $\boldsymbol{x} \in \mathbb{R}^d$ be an input, we define $\boldsymbol{z}_i$ and $\hat{\boldsymbol{z}}_i$ as the input and output of layer i respectively, where $\boldsymbol{z}_0 = \boldsymbol{x}$ and $\boldsymbol{z}_n = F(\boldsymbol{x})$ is the final output of the neural network at layer n. Each $\boldsymbol{z}_i$ is then recursively defined as:

$$\boldsymbol{z}_i = \boldsymbol{W}_i \hat{\boldsymbol{z}}_{i-1} + \boldsymbol{b}_i \tag{2.3.1}$$

where in this case, $\boldsymbol{z}_i = \hat{\boldsymbol{z}}_i$, and $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weights and biases.

Definition 2. (**Rectified Linear Unit**) Simply referred to as ReLU, it is a popular *activation* function in deep learning defined as:

$$ReLU(x) = max(0, x) \tag{2.3.2}$$

It is used to inject non-linearities within a neural network, as defined next.

Definition 3. (**Sigmoid Activation Function**) Sigmoid is a curve-shaped function $\sigma : \mathbb{R} \rightarrow [0, 1]$ defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.3.3}$$

Definition 4. (**Single-output ReLU-activated Feed-forward Neural Network**) Simply referred to as a ReLU neural network in this thesis, it is a neural network (Definition 1) in which the output of each layer is passed through a ReLU function (Definition 2). More formally, following Definition 1, we re-define $\hat{\boldsymbol{z}}_i$ as:

$$\begin{aligned} \boldsymbol{z}_i &= \boldsymbol{W}_i \hat{\boldsymbol{z}}_{i-1} + \boldsymbol{b}_i \\ \hat{\boldsymbol{z}}_i &= ReLU(\boldsymbol{z}_i) \end{aligned} \tag{2.3.4}$$

Figure 2.1 shows a simple ReLU neural network along with its weights. The activation in the output layer is sometimes replaced by a Sigmoid function (Definition 3) that maps the logits to probabilities (e.g., in the case of using neural nets for binary classification), i.e., $F(\boldsymbol{x}) = \sigma(\boldsymbol{z}_n)$.

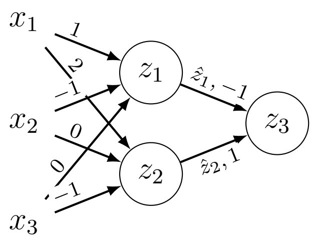

Fig. 2.1. A single-output ReLU-activated feed-forward neural network

The weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ are called the parameters of the neural network, often written as F_θ to show the dependence of the function on these parameters. To estimate these parameters, an optimization scheme and algorithm, such as Gradient Decent [**68**] and the Backpropagation algorithm [**30**] are required. Moreover, one would require a *loss* function that is an empirical metric evaluating the estimation quality of the parameters and guiding the optimization procedure. In this work, we use ReLU neural networks for the task of binary classification and the loss function for that is defined as follows.

Definition 5. (**Binary Cross Entropy**) Let $f_\theta : \mathbb{R} \to [0, 1]$ be a parametric function and $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_M, y_M)\}$ be a set of pairs of input-output samples where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$. The binary cross entropy loss is then defined as:

$$L_{BCE}(f_\theta) = -\frac{1}{M}\sum_{i=1}^{M} y_i \log(f_\theta(\boldsymbol{x}_i)) + (1 - y_i)\log(1 - f_\theta(\boldsymbol{x}_i)) \tag{2.3.5}$$

Intuitively, it is an empirical metric that measures how close the parametric distribution f_θ is to the true distribution of labels, i.e., how well f_θ is doing in estimating the probability of a sample $\boldsymbol{x}$ belonging to class 1.

2.4. Background on Encoding NNs as Mixed-Integer Linear Programs

We start with a quick introduction to Mixed-Integer Linear Programs (MILP). We will then explain the MILP encoding of ReLU neural networks [**11, 75**] adopted in this work to construct the CE-Fair Verification algorithm.

Definition 6. (**Linear Program (LP)**) Constrained linear optimization problems that can be expressed in the following form are linear programs.

$$\begin{aligned} &\textit{maximize} && \boldsymbol{c}^T\boldsymbol{x} \\ &\textit{s.t.} && \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b} \\ &\textit{and} && \boldsymbol{x} \ge \boldsymbol{0} \end{aligned} \tag{2.4.1}$$

where vectors $\boldsymbol{c}$ and $\boldsymbol{b}$ and matrix $\boldsymbol{A}$ are given and variable $\boldsymbol{x}$ is to be derived. $\boldsymbol{c}^T\boldsymbol{x}$ is called the *objective* function and $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$ and $\boldsymbol{x} \geq \boldsymbol{0}$ specify the constraints that form a convex polytope over which the objective function is to be optimized.

Definition 7. (**Mixed-Integer Linear Program (MILP)**) A linear program (Definition 6) is called mixed-integer when a subset of the variables to be derived are constrained to be integers. More specifically, an LP of the general following form:

$$\begin{aligned} \textit{maximize} \qquad & \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{c}'^T\boldsymbol{x}' \\ \textit{s.t.} \qquad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{A}'\boldsymbol{x}' \leq \boldsymbol{b}' \\ \textit{and} \qquad & \boldsymbol{x}' \in \mathbb{Z}^n \end{aligned} \tag{2.4.2}$$

where n is the dimension of vector $\boldsymbol{x}'$.

2.4.1. Unbounded MILP Encoding of ReLU Neural Networks

We would like to encode the defined ReLU NN as a mixed-integer linear program; by that, we mean constraints that would later be part of a larger MILP with an objective, but for now, we do not discuss the objective. Recall the details of Definition 4, the multiplication of weights is a linear operator and is straightforward to encode; however, it is more tricky to encode the ReLU activations.

Definition 8. (**Unbounded MILP Encoding of NNs**) Similar to Definition 2, let $\boldsymbol{W}$, $\boldsymbol{b}$ be the weights and biases of an n-layered ReLU NN. Moreover, let $\boldsymbol{z}_i$ and $\hat{\boldsymbol{z}}_i$ be the pre-ReLU and post-ReLU values of hidden layer $i \in \{1, \ldots, n\}$. Finally, let k_i be the width of layer i and $\boldsymbol{\delta}_i$ be binary vectors of dimension k_i. Then for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, k_i\}$ we would have the following MILP:

$$\mathbf{z}_i = \mathbf{W}_i\hat{\mathbf{z}}_{i-1} + \mathbf{b}_i \tag{2.4.3a}$$

$$\begin{aligned} \boldsymbol{\delta}_i \in \{0,1\}^{k_i},\ \hat{\mathbf{z}}_i = \mathbf{z}_i \cdot \boldsymbol{\delta}_i, \\ \delta_{i,j} = 1 \Rightarrow z_{i,j} \geq 0, \\ \delta_{i,j} = 0 \Rightarrow z_{i,j} < 0 \end{aligned} \tag{2.4.3b}$$

Intuitively, the first part (2.4.3a) defines the constraints for the linear operator of weights multiplication and the second part (2.4.3b) uses the auxiliary binary $\boldsymbol{\delta}$ variables to encode the state of each ReLU.

2.4.2. Bounded MILP Encoding of ReLU Neural Networks

The optimization backend adopted in this work [**31**] uses Branch-and-Bound (B&B) optimization internally. Thus, it is very useful to limit the bounds of the variables of the MILP for more runtime-efficient optimization. For that purpose, we make the justified assumption that there are some lower/upper bounds on the input domain. We adopt the MILP proposed by [**75**].

Definition 9. (**Bounded MILP Encoding of NNs**) Following Definition 2.4.3, let $(\boldsymbol{l}_i, \boldsymbol{u}_i)$ be lower bounds and upper bounds on the values of the hidden units in layer i. We then define the following bounded MILP encoding:

$$\boldsymbol{z}_i = \boldsymbol{W}_i \hat{\boldsymbol{z}}_{i-1} + \boldsymbol{b}_i \tag{2.4.4a}$$

$$\begin{gathered} \boldsymbol{\delta}_i \in \{0,1\}^{t_i}, \quad \hat{\boldsymbol{z}}_i \geqslant 0, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{u}_i \cdot \boldsymbol{\delta}_i, \\ \hat{\boldsymbol{z}}_i \geqslant \boldsymbol{z}_i, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{z}_i - \boldsymbol{l}_i \cdot (1 - \boldsymbol{\delta}_i) \end{gathered} \tag{2.4.4b}$$

Intuitively, the first part (2.4.4a) is again only the linear multiplication of weights; the second part (2.4.4b) ensures that $\delta_{i,j} = 0 \Leftrightarrow \hat{z}_{i,j} = 0$ and $\delta_{i,j} = 1 \Leftrightarrow \hat{z}_{i,j} = z_{i,j}$, this time while having bounds on the variables which would significantly boost optimization efficiency.

While it is beneficial to have bounds on the variables of the MILP, it is important how the bounds are computed; the tighter the bounds are, the more efficient the optimization will be. There are numerous approaches presented in the literature [**52**] for computing these bounds. Next, we explain the bounds computation approach adopted here.

2.4.2.1. Linear over-approximation of ReLUs for Bounds Computation. We leverage an intermediate relaxed MILP encoding proposed by [**21**] that over-approximates the ReLUs only for the purpose of computing the lower/upper bounds to be plugged into the exact encoding 2.4.4.

Definition 10. (**Over-approximation of ReLUs MILP**) Similar to the notation of Definition 2.4.4, for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, k_i\}$, we have the following relaxed encoding:

$$\mathbf{z}_i = \mathbf{W}_i \hat{\mathbf{z}}_{i-1} + \mathbf{b}_i \tag{2.4.5a}$$

$$\hat{\mathbf{z}}_i \geqslant \mathbf{z}_i, \quad \hat{\mathbf{z}}_i \geqslant 0, \quad \hat{z}_{i,j} \leqslant u_{i,j} \frac{z_{i,j} - l_{i,j}}{u_{i,j} - l_{i,j}} \tag{2.4.5b}$$

This is a fully linear MILP that no longer encodes ReLUs as binary variables and thus is much more efficient in terms of optimization runtime. Starting from the bounds on the input layer (which we assume is derived from, e.g., the dataset), it progressively computes tight bounds by solving two MILPs at each layer (with only the constraints up until the current layer): one minimization to find the lower bound and one maximization to find the upper bound. The bounds are then propagated to the next layer using interval arithmetic [**35**]. Finally, all the computed tight bounds are plugged into encoding 2.4.4 to have an *exact* and efficient MILP encoding of ReLU neural networks.

Chapter 3

FETA: Fairness Enforced Verifying, Training, and Predicting Algorithms for Neural Networks

by

Kiarash Mohammadi, Aishwarya Sivaraman, Golnoosh Farnadi[1]

([1]) Mila, Université de Montréal

This article was submitted in a conference and is under review at the time of writing this thesis; part of it was accepted at AAAI 2022 Workshop on Combining Learning and Reasoning.

The main contributions of Kiarash Mohammadi for this articles are presented.

- Development of the idea and theoretical foundation under the guidance of the supervisor;
- Development of the open-source tool (FETA) and conducting extensive experiments on real-world datasets;
- Writing the presented paper as the first author;

Co-author Aishwarya Sivaraman has helped with feedback through the project and writing revisions. Co-author Golnoosh Farnadi has guided the project as the principal investigator (PI).

Résumé. L'automatisation de la prise de décision dans des applications qui affectent directement la qualité de vie des individus grâce aux algorithmes de réseaux de neurones est devenue monnaie courante. Ce mémoire porte sur les enjeux d'équité individuelle qui surviennent lors de la vérification, de l'entraînement et de la prédiction des réseaux de neurones. Une approche populaire pour garantir l'équité consiste à traduire une notion d'équité en contraintes sur les paramètres du modèle. Néanmoins, cette approche ne garantit pas toujours des prédictions équitables des modèles de réseaux de neurones entraînés. Pour relever ce défi, nous avons développé une technique de post-traitement guidée par les contre-exemples afin de faire respecter des contraintes d'équité lors de la prédiction. Contrairement aux travaux antérieurs qui ne garantissent l'équité qu'aux points entourant les données de test ou d'entraînement, nous sommes en mesure de garantir l'équité sur tous les points du domaine. En outre, nous proposons une technique de prétraitement qui repose sur l'utilisation de l'équité comme biais inductif. Cette technique consiste à incorporer itérativement des contre-exemples plus équitables dans le processus d'apprentissage à travers la fonction de perte. Les techniques que nous avons développé ont été implémentées dans un outil appelé FETA. Une évaluation empirique sur des données réelles indique que FETA est non seulement capable de garantir l'équité au moment de la prédiction, mais aussi d'entraîner des modèles précis plus équitables.
Mots clés : Équité, Réseaux de Neurones, Vérification

Abstract. Algorithmic decision-making driven by neural networks has become very prominent in applications that directly affect people's quality of life. This paper focuses on the problem of ensuring individual fairness in neural network models during verification, training, and prediction. A popular approach for enforcing fairness is to translate a fairness notion into constraints over the parameters of the model. However, such a translation does not always guarantee fair predictions of the trained neural network model. To address this challenge, we develop a counterexample-guided post-processing technique to provably enforce fairness constraints at prediction time. Contrary to prior work that enforces fairness only on points around test or train data, we are able to enforce and guarantee fairness on all points in the domain. Additionally, we propose a counterexample-guided loss as an in-processing technique to use fairness as an inductive bias by iteratively incorporating fairness counterexamples in the learning process. We have implemented these techniques in a tool called FETA. Empirical evaluation on real-world datasets indicates that FETA is not only able to guarantee fairness on-the-fly at prediction time but also is able to train accurate models exhibiting a much higher degree of individual fairness.
Keywords: Fairness, Neural Networks, Verification

3.1. Introduction

Deep neural networks are increasingly used to make sensitive decisions, including financial decisions such as loan approval [**32**], recidivism risk assessments [**39**], salary prediction [**5**], etc. In these settings, for ethical, and legal reasons, it is of utmost importance that decisions are fair. For example, all else being equal, one would expect two individuals of a different gender to 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. [**51, 12**] 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 reconstruct sensitive features that are encoded in data internally. Guaranteeing fairness not only helps organizations to address laws against discrimination but also helps users to better trust and understand the learned model [**4**].

Training neural network models such that fairness properties hold in their prediction is not always straightforward or possible. Existing approaches to the problem, either identify the absence of unfair predictions using verification [**38, 77**] or guarantee fairness only for points during the training phase by constrained optimization techniques [**14**] or add fairness regularizers to the loss function [**42**]. Other works which focus on test data points, provide fair models by using robustness techniques [**69, 82**]. While these techniques are successful in mitigating discrimination, they fail to provide global fairness guarantees for all points in the input domain at the prediction time. Over the past years, multiple definitions of fairness have been introduced. It is believed that none of these definitions dominates the others, and each of them is suitable for different settings. Recent works on fairness consider group-based notions of fairness [**29, 84**], e.g., demographic parity [**20**] or equalized odds [**32**], that indicate that two populations of individuals should be treated equally on average. Despite their prevalence, group fairness notions are generally hard to formally guarantee fairness for all input points [**44, 69**]. Further, an algorithm that satisfies group fairness could be blatantly unfair from the point of view of individual users [**20**]. In this paper, we focus on individual fairness [**20**] which states that the distance between the outcome for two individuals should be bounded according to the degree of their similarity.

Our approach. This paper develops techniques to detect, incorporate and guarantee *individual* fairness constraints for all points in the input space to a standard ReLU neural network without imposing further restrictions on the hypothesis space. These techniques leverage recent work that employs automated theorem provers to formally verify the properties of neural networks. We focus on the individual fairness notion introduced by [**25**]. This notion says that a model is fair if, the decision of the model is the same for any two individuals with various combinations of sensitive attributes when nonsensitive attributes are fixed. To guarantee fairness, we present a counterexample-guided algorithm that detects and provably guarantees fairness at prediction time, given an arbitrary ReLU neural network. For any given model, our post-processing approach works by computing a majority decision for a group of individuals who share nonsensitive attributes on-the-fly via verification counterexamples. Furthermore, we propose a novel counterexample-guided algorithm to incorporate fairness during training. We identify individual fairness counterexamples on the training data, inducing additional supervision for training the network, and perform this process iteratively. We have implemented our algorithms in a tool called "**F**airness **E**nforced Verifying, **T**raining, and Predicting **A**lgorithm" (FETA). Empirical evaluations on real-world benchmark datasets demonstrate the effectiveness of our solutions to train fair and accurate models, while provably guaranteeing fairness at the prediction time. Empirically, the two algorithms, when used in conjunction, enable better generalization while guaranteeing fairness.

Main contributions. Our key contributions are: 1) A practical individual fairness verification approach that detects discrimination through counterexamples given an arbitrary ReLU neural network (see Section 3.3: CE-Fair Verification). 2) A counterexample-guided online algorithm that provably guarantees individual fairness at prediction time (see Section 3.4: CE-Fair Prediction). 3) A counterexample-guided re-training algorithm that incorporates individual fairness during training (see Section 3.5: CE-Fair Training). 4) An end-to-end available implementation of our methods in an open-source tool called FETA, together with an extensive evaluation of real-world datasets (see Section 3.4.1 and Section 3.5.1).

3.2. Preliminaries

We begin by introducing some common notations. Let $\mathcal{X}$ be the input space consisting of d features where $\mathcal{X} \equiv N \times S$ and S denotes protected or sensitive features, and N denotes remaining input features, and suppose that it is a compact finite subset $\mathcal{X} = [L, U]^d$ of $N \in \mathbb{R}^k$ and $S \in \mathbb{N}^{d-k}$. Let $\mathcal{Y} \in \{0, 1\}$ be the output space. We consider a supervised binary classification task, where $f_\theta : \mathcal{X} \to [0, 1]$ outputs a probability distribution over classes, θ denotes the parameters and the classifier $g_\theta : f_\theta(\mathcal{X}) \to \mathcal{Y}$ is defined as $g_\theta := \mathbb{1}(f_\theta(\boldsymbol{x}) \geq \Delta)$ where Δ is some classification threshold and it assigns an input to a category identified by numeric code $\mathcal{Y}$. Let $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_M, y_M)\}$ be the training dataset containing M samples with $\boldsymbol{x}_m$ and y_m respectively denoting the mth individual and the corresponding output. The most commonly used *Binary Cross Entropy (BCE)* loss for this task is:

$$L_{BCE}(f_\theta) = -\frac{1}{M} \sum_{i=1}^{M} y_i \log(f_\theta(\boldsymbol{x}_i)) + (1 - y_i) \log(1 - f_\theta(\boldsymbol{x}_i)) \tag{3.2.1}$$

where the goal is to find the best θ that minimizes L_{BCE} across the data-generation distribution rather than just over the finite D.

Our goal will be to verify, guarantee and train an *individually fair* model in some sensitive input features. We refer to the *Causal Discrimination* definition, a notion of individual fairness proposed by [**25**].

Definition 11. (Causal Discrimination) Assume a function $g_\theta : f_\theta(\mathcal{X}) \to \mathcal{Y}$, such that $\mathcal{X}[1 \ldots k] \in N$ and $\mathcal{X}[k+1 \ldots d] \in S$. We define g_θ to be *individually fair* in sensitive features S iff for any two points $\boldsymbol{x}$, $\boldsymbol{x}' \in \mathcal{X}$ where $\boldsymbol{x}[i] = \boldsymbol{x}'[i]$, $\forall i \in \{1 \ldots k\}$, we have that $g_\theta(\boldsymbol{x}) = g_\theta(\boldsymbol{x}')$.

In Neural Networks (NN), various nonlinear activation functions have been introduced. Among those, ReLU has been used widely and generalized well [**27, 81, 18**], particularly in the context of verification [**43, 37**] and robustness. Hence, we will assume f_θ is a ReLU neural network. Formal properties of neural networks are often verified by encoding the semantics of neural networks (f_θ) as logical constraints. While several approaches to encoding neural networks for verification have been studied [**52, 11**]; we use the encoding and optimization approach similar to [**57**] that propose significantly faster techniques for our use-case than the ones relying on Satisfiability Modulo Theories. This is crucial to our goal of guaranteeing fairness at prediction time.

Definition 12. (MILP Encoding of Neural Networks) Let f_θ be an n-layer fully-connected ReLU neural network with a single output where $\theta = (\boldsymbol{W}, \boldsymbol{B})$ and $\boldsymbol{W}_i$ and $\boldsymbol{b}_i$ respectively denoting weights and bias of ith layer. The width of each layer is represented by t_i, the values of neurons before applying ReLU are represented by vector $\boldsymbol{z}_i$, $\forall i \in \{0 \dots n\}$ ($\boldsymbol{z}_0$ being the input), and their values after ReLU by $\hat{\boldsymbol{z}}_i$, $\forall i \in \{1 \dots n\}$, [**75**] proposes the following MILP encoding, $\forall i \in \{1 \dots n\}$:

$$\boldsymbol{z}_i = \boldsymbol{W}_i \hat{\boldsymbol{z}}_{i-1} + \boldsymbol{b}_i \tag{3.2.2a}$$

$$\begin{gathered} \boldsymbol{\delta}_i \in \{0,1\}^{t_i}, \quad \hat{\boldsymbol{z}}_i \geqslant 0, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{u}_i \cdot \boldsymbol{\delta}_i, \\ \hat{\boldsymbol{z}}_i \geqslant \boldsymbol{z}_i, \quad \hat{\boldsymbol{z}}_i \leqslant \boldsymbol{z}_i - \boldsymbol{l}_i \cdot (1 - \boldsymbol{\delta}_i) \end{gathered} \tag{3.2.2b}$$

Equation (3.2.2a) encodes the linear relationship, while Equation (3.2.2b) encodes the ReLU activation function, i.e., $\hat{\boldsymbol{z}} = ReLU(z) = \max(0, z)$. $\boldsymbol{\delta}_i$ 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 [**21**], given the bounds on input $\boldsymbol{l}_0 = L, \boldsymbol{u}_0 = U$. Moreover, in this work, we assume both continuous and discrete domains over the input variables, hence the *mixed-integer linear program* (MILP).

Formal properties of functions are often characterized in terms of their counterexamples, e.g., [**74, 15, 73**]. The techniques proposed in this paper will be centred around using counterexamples to the fairness specification. Counterexample-guided algorithms rely on the ability to *find* counterexamples, which require that both the counterexample specification and the object of interest (f_θ) to be encoded in a language amenable to automated reasoning. Definition 12 provides such an encoding for f_θ. In the next section, we show how to add fairness constraints per Definition 11, to identify counterexamples.

3.3. CE-Fair Verification

In 2019, Apple launched a credit card application that was accused of gender bias [**6**]. The scandal went viral when a couple who shared all of their bank accounts, assets, and credit cards, received different credit limits while only their gender was different in their applications. Inspired by this real-world example, in this section, we propose an approach that focuses on auditing and verifying a trained neural network model to detect such discriminatory outcomes. We envision scenarios where the classifier is a proprietary model, belonging e.g. to a company or a bank, and an external party wants to inspect the model to ensure that it is operating fairly. We introduce *Counterexample-guided fair (CE-Fair) verification* that can identify these fairness violations.

Definition 13. (CE-Fair Verification) Consider example $x \in \mathcal{X}$, function $f_\theta : \mathcal{X} \rightarrow [0,1]$ and sensitive features S and non-sensitive features N. Then a *fairness counterexample* for example x, function f_θ, and S is x' such that (i) $\boldsymbol{x}[i] = \boldsymbol{x}'[i],\ \forall i \in \{1 \dots k\} \in N$, and (ii) $g_\theta(\boldsymbol{x}) \neq g_\theta(\boldsymbol{x}')$. We then define the CE-Fair verification function v for sensitive feature set S that takes as input a function $f_\theta : \mathcal{X} \rightarrow [0,1]$ and $x \in \mathcal{X}$ as follows:

$$v(f_\theta, x) = \begin{cases} x' & \text{where } x' \text{ is a fairness counterexample} \\ \emptyset & \text{if no fairness counterexample exists} \end{cases}$$

CE-Fair verification function $v(f_\theta, x)$ can find counterexample x' by solving the optimization problem in which the two kinds of constraints defined in Definition 13 are added to the MILP encoding from Equation 3.2.2. The feasible set of this optimization problem is explored using an optimizer backend and if there exists a solution satisfying these constraints, we will have a fairness counterexample. Formally, the following constraints will be added to the MILP formulated in Equations 3.2.2a, 3.2.2b to encode fairness counterexamples:

$$\boldsymbol{z}_0[i] = \boldsymbol{x}[i],\ \forall i \in \{1 \dots k\} \tag{3.3.1a}$$

$$\mathbb{1}(z_n \geq \Delta) = 1 - g_\theta(\boldsymbol{x}) \tag{3.3.1b}$$

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 nonsensitive 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 model to be opposite to $g_\theta(\boldsymbol{x})$. Here we only search for a counterexample violating fairness as a constraint by adding the following as an objective to the MILP:

$$|z_n - f_\theta(\boldsymbol{x})| \tag{3.3.2}$$

However, we can also search for the counterexample with maximum violation (see Definition 16). While CE-Fair verification is sufficient to audit and verify a trained model to identify counterexamples as per Definition 11, in the case where there are counterexamples –which is often the case– it is not clear how to *guarantee* fairness, or how to enforce it during training. The next two sections present the counterexample-guided algorithms that address these challenges.

3.4. CE-Fair Prediction

For ethical or legal reasons, a company may want to ensure that their model outcome is the same for all individuals irrespective of their sensitive attributes. Further, they may want to ensure fair predictions using techniques that do not require modifying and re-training the model under study. Such requirements can be due to the utilization of outsourced models without having any access to the original training data or having limited resources to re-train the model. In this section, we leverage our counterexample-guided verification approach to provably guarantee individual fairness at prediction time without any requirements to re-train the model. We propose an *online* technique that leverages counterexamples to Definition 11 to construct fair predictions on-the-fly at prediction time.

A naive approach to guarantee fair predictions 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 3.1 in Section 3.4.1). However, this gives us intuition to return the majority decision for a group of individuals who share nonsensitive attributes. We define CE-Fair prediction that produces individually fair output for a given input x and f_θ as:

Definition 14. (CE-Fair Prediction) For an example $x \in \mathcal{X}$, function $f_\theta : \mathcal{X} \to [0,1]$, we define a post-processing CE-Fair Prediction function h such that:

$$h\left(f_\theta(\boldsymbol{x})\right) = \mathbb{1}\left(\left(\sum_{\boldsymbol{x}' \in A(\boldsymbol{x})} g_\theta(\boldsymbol{x}') - \mathbb{1}(g_\theta(\boldsymbol{x}') = 0)\right) \geqslant 0\right) \tag{3.4.1}$$

where:

$$\begin{aligned} A(\boldsymbol{x}) := \{X \mid X[1] = \boldsymbol{x}[1], \ldots, X[k] = \boldsymbol{x}[k], \\ X[k+1] = a_{k+1}, \ldots, X[d] = a_d; \\ \forall a_{k+1}, \ldots, a_d \in [L'_{k+1,\ldots,d}, U'_{k+1,\ldots,d}]^{d-k}\} \end{aligned} \tag{3.4.2}$$

Basically, this is contrasting the 0 and 1 outputs of g_θ in the space of sensitive features; the counter increments when $g_\theta(\boldsymbol{x}')$ is 1 and decrements otherwise.

Theorem 1. For any function f_θ and for any input $\boldsymbol{x} \in \mathcal{X}$ with S as sensitive features, $h(f_\theta(\boldsymbol{x}))$ 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 counterexample exists. □

So far we have established a way to guarantee fair predictions for all input points based on the majority decision captured in function h. To identify the majority decision, the simple approach is to enumerate all possible assignments of sensitive attributes. Concretely, given a test point $\boldsymbol{x}$, we could traverse all possible assignments to the sensitive features, counting the frequency of each label. The computational complexity of this approach grows with the size of sensitive attributes and the domain size of each sensitive attribute. This approach, however, is not practical and increases prediction time significantly (See Appendix). This motivates the next approach in which we compute h and identify the majority decision by leveraging our MILP framework to find counterexamples.

Definition 15. (CE-Fair Counting) All counterexamples of a test sample $\boldsymbol{x}$ (i.e., $\mathcal{S}$) can be determined in an iterative way by adding the following constraints to the verification problem in Definition 13 and solve in iteration $K+1$ as:

$$x' = v(f_\theta, x) \tag{3.4.3a}$$

$$\sum_{i=1}^{d}\Big(\sum_{s\in\mathcal{S}:x'^{k}_{s}[i]=0} x'[i] + \sum_{s\in\mathcal{S}:x'^{k}_{s}[i]=1} (1 - x'[i])\Big) \geq 1 \tag{3.4.3b}$$

$$k = 1, \ldots, K \tag{3.4.3c}$$

where S is a set of all counterexamples of x and x'^{k}_{s} is a counterexample of x which is included in the set S from the previous iteration k. To satisfy Constraints 3.4.3c, the solution must differ in at least one entry for each x'^{k}. Once Problem 3.4.3 becomes infeasible, then all counterexamples have been determined. Since there is a finite number of feasible assignments (e.g., Equation 3.4.2), this iterative method will stop in a finite time.

The lazy constraints generation approach defined in Definition 15 can be implemented more efficiently by using the optimization backend [**31**]. Hence, instead of iteratively finding counterexamples, we explore the MILP search tree in pursuit of $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ counterexamples (rather than only one) where their labels are opposite to $g_\theta(\boldsymbol{x})$. If that many solutions are found, then the majority decision for the group of assignments specified by $\boldsymbol{x}$ is opposite to $g_\theta(\boldsymbol{x})$, otherwise, the prediction remains unchanged. The general scheme of *CE-Fair Counting* approach is shown in Algorithm 1. The algorithm takes as an input f_θ, as well as a sample $\boldsymbol{x}$. In line 3, the MILP encoding of f_θ is obtained as per Equation 3.2.2 and constraints from Equation 3.3.1 specifying a counterexample for $\boldsymbol{x}$ are obtained in the following line. In line 5, the MILP search tree is explored to find counterexamples; if it finds less than $\left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$, the final prediction does not change, otherwise, it flips (lines 6-9).

3.4.1. Empirical Evaluation of CE-Fair Prediction

This section shows the effectiveness of CE-Fair prediction approach through empirical evaluations on three widely known real-world benchmark datasets: German [**36**], IPUMS Adult, and Law School [**78**].

3.4.1.1. Dataset details. Here we overview the datasets we leverage:

- **German credit dataset** [**36**]

  This dataset consists of 1k samples with dimensionality 61 and sensitive features: age

Algorithm 1 Counterexample-guided Counting to Guarantee Fair Predictions

```
Input: $f_\theta$, $\boldsymbol{x}$
Output: CE-Fair Prediction: $h(f_\theta(\boldsymbol{x})) \in \{0, 1\}$
$\phi_N \leftarrow$ ModelMIPEncoding($f_\theta$) {Constraints in Equation 3.2.2}
$\phi_{CE} \leftarrow$ FairCEEncoding($\phi_N, \boldsymbol{x}, f_\theta(\boldsymbol{x})$) {Constraints in Equation 3.3.1}
$\mathcal{S} \leftarrow$ FairCECounting($\phi_N, \phi_{CE}$) {Constraints in Equation 3.4.3}
if $|\mathcal{S}| < \left\lceil \frac{|A(\boldsymbol{x})|}{2} \right\rceil$ then
    return $g_\theta(\boldsymbol{x})$
else
    return $1 - g_\theta(\boldsymbol{x})$
end if
```

$\in [19, 75]$, sex/marital status with 4 categories, and foreign worker with 2 categories. The main task is binary classification of good or bad credit risks.

- **IPUMS Adult dataset (aka, *the new Adult*)** [**16**]

  The initial Adult dataset [**46**] is used for binary classification of whether an individual's salary is above or below \$50k. [**16**] discuss some limitations of this dataset and propose a reconstruction of the Adult [**46**] dataset in which the actual *income* of the individuals are available. Thus, one can re-define the binary classification task with some threshold other than \$50k. In our experiments, this threshold is set to \$30k as the experiments by [**16**] indicate the most severe *unfairness* to occur around the 30k threshold.

  The dataset consists of 49k samples with dimensionality 103 and sensitive features: age $\in [17, 90]$, marital status with 7 categories, race with 5 categories, native country with 41 categories, and sex with 2 categories. This is the largest sensitive feature space among datasets used for our experiments.

- **Law School dataset** [**78**]

  This dataset, consisting of 86k samples, gathers law school admission records and is used for predicting if an individual would pass the bar exam. The input dimension is 37 and the sensitive features are: race with 3 categories and gender with 2 categories

3.4.1.2. Experimental Setup. Experiments are implemented in Python using Pytorch [**60**]. All experiments were run on a machine with 10 GiB RAM and a 2.1GHz Intel Xeon processor. We use Gurobi-9.5.1[1] as our backend solver to generate counterexamples. We make all code, datasets, and preprocessing pipelines publicly available. Below, we overview the experimental setup.

- **Data**

  The data is divided into 5 folds of 80/20 train/test sets. Moreover, 10% of the train set is sliced for validation. Experiment results are gathered within 5-fold cross-validation (CV). As for data types, we support categorical and numerical features.

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

For the CE-Fair Prediction part, where we have counting over individuals sharing sensitive attributes, numerical features are considered discrete. Numerical features of nonsensitive attributes are considered real-valued.

- **Model Architecture**
  The ReLU neural network model used across all experiments is a fixed architecture of 3 hidden layers of width 16. This is a reasonably complex model for the tabular datasets used in such scenarios.
- **Pre-training**
  To pre-train the initial model ($\mathbf{NN_b}$), we run a grid search over learning rate (10^{-2}, 10^{-3}, 10^{-4}) and batch size (64, 128). We train each configuration for 500 epochs and select the model with the best loss on the validation set. This is the case with all datasets except for Law School which is more tricky to train on; for that, we use a learning rate of 0.01, a batch size of 256, train for 100 epochs, and take the last epoch model to get the best initial pre-trained model for a fair comparison.

Q1: Does a deep neural network trained on data obey individual fairness? To quantify the degree of *unfairness* in the initial model trained on data, we introduce, Counterexample Rate (CE RATE), which computes how many test samples have counterexamples as per Definition 13. As shown in Figure 3.1, the degree of fairness violation based on these metrics is high for all our datasets, motivating the need for *guaranteed* fair predictions. The percentage of data points that have counterexample can be as high as 89% for IPUMS ADULT dataset.

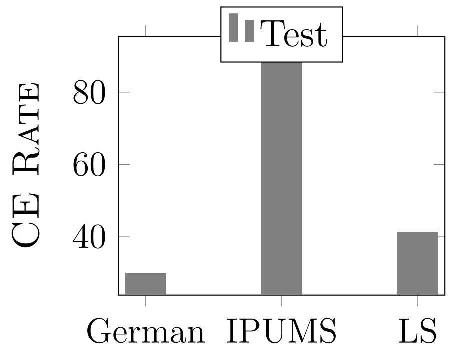

Fig. 3.1. Empirically, the best learned NN model is not fair. The figure presents the CE Rate for German, IPUMS Adult (IPUMS), and Law School (LS).

Q2: What is the effect of guaranteed fair predictions on performance and overall model fairness? In this experiment, we compare the accuracy of the best baseline model ($\mathrm{NN_b}$) with CE-FAIR predictions on test data. Further, to quantify the effect of fair predictions on model fairness, we introduce *flip rate* which computes the number of samples in test data where the prediction of the model was modified to satisfy individual fairness. Table 3.1 demonstrates that you can use CE-FAIR predictions to guarantee fairness with accuracy loss up to 8%. This can be explained as follows: since the flip rate of the best-trained model (NN_b) is high, it is expected that the drop in accuracy has a relation with the model flip rate, as seen with GERMAN with the lowest flip rate and smallest decrease in accuracy. Further, we compare our approach against a naive majority-based baseline which is a constant predictor that returns the most frequent label. We observe that, on average, CE-FAIR predictions perform 8% better than the majority baseline. In Section 3.5, we propose a counterexample-guided re-training approach to reduce the performance drop while improving fairness metrics.

Table 3.1. The effect of guaranteed fair predictions on performance and model fairness – 5-fold CV results

Dataset	**NN_b**		**Majority Baseline**		**CE-Fair Prediction**	
	Accuracy	**Flip Rate**	**Accuracy**	**Flip Rate**	**Accuracy**	**Flip Rate**
German	76.70 ± 2.78	8.90 ± 0.73	70.00 ± 1.78	0.0 ± 0.0	74.20 ± 3.50	0.0 ± 0.0
IPUMS Adult	81.57 ± 0.47	24.39 ± 1.52	54.61 ± 0.55	0.0 ± 0.0	73.23 ± 0.93	0.0 ± 0.0
Law School	82.72 ± 0.19	17.53 ± 0.59	72.98 ± 0.50	0.0 ± 0.0	74.36 ± 1.95	0.0 ± 0.0

Q3: How does CE-Fair prediction affect inference time? Figure 3.2 plots the ratio of inference time of NN_b and CE-FAIR predictions for test data on all datasets. We observe that the increase in inference time is proportional to the model flip rate (see Table 3.1). It also highly depends on the dimension of the sensitive features as seen with IPUMS ADULT. This is expected since a larger flip rate means more samples are unfair and therefore more calls to the verification engine. Of course, when violating fairness leads to ethical or legal problems, the question is not whether we can afford fairness enforcement, but whether it is correct to use machine learning at all. In this context, the computational price of enforcing fairness, even if it ends up being significant, is entirely warranted.

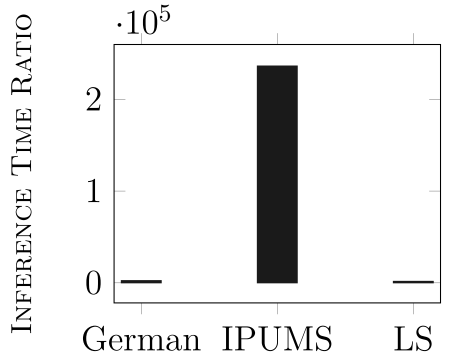

Fig. 3.2. Ratio of Inference time increases when using CE-Fair predictions to guarantee fairness.

3.5. CE-Fair Training

In this section, we propose an algorithm for learning fair neural networks with counterexamples. While in the previous section, we guaranteed fair predictions as a *post-processing* approach, in this section, we propose an *in-processing* approach that drops the guaranteed fairness requirement in exchange for a relatively fair model with efficient inference. Our learning algorithm is orthogonal to the prediction technique of the previous section and both approaches can be combined to acquire guaranteed fairness with boosted performance and more efficient inference time (see evaluation results in Section 3.5.1).

To define our learning paradigm, we first define a specific variation of the verification function defined in Definition 13 called $v_{max}(f_\theta, x)$. The CE-Fair verification function can produce a counterexample given an arbitrary sample. To make sure that the counterexamples are representative and not out-of-distribution examples for training, we generate counterexamples relative to each training point. Moreover, to make sure that the counterexamples are effective in reducing the degree of fairness violation, we instead appeal to Definition 16 to generate counterexamples with maximal violation relative to each training point.

Definition 16. (Maximum Violation CE-Fair Verification) Given a data point $\boldsymbol{x}$ and $f_\theta : \mathcal{X} \to [0,1]$, we find the maximum violation counterexample $\boldsymbol{x}'_{max}$ using $v_{max}(f_\theta, \boldsymbol{x})$ defined as:

$$\arg\max_{\boldsymbol{x}'} |f_\theta(\boldsymbol{x}') - f_\theta(\boldsymbol{x})| \quad s.t. \quad x' = v(f_\theta, \boldsymbol{x}) \tag{3.5.1}$$

Suppose a sample data point $\boldsymbol{x}$, is a young single Asian female applicant with a negative decision on her loan application in the training set. Using $v_{max}(f_\theta, \boldsymbol{x})$, we are able to find a counterfactual applicant $\boldsymbol{x}'$ with a different combination of sensitive attributes, e.g., an old married American man with the same credit history who can potentially receive a positive decision from the model with the highest level of violation compared to the sample data $\boldsymbol{x}$. Next, we show how we use such counterfactual examples, i.e., counterexamples, to train a fair model.

Definition 17. (CE-Fair Training) The loss function for counterexample-guided training can be written as:

$$\begin{aligned} L_{CE-Fair}(f_\theta) = \quad & \underbrace{L_{BCE}(f_\theta)}_{\text{Classification loss}} \quad + \\ & \underbrace{\sum_{i=1}^{M} y_i \log(f_\theta(v_{max}(f_\theta, \boldsymbol{x}_i)))) + (1 - y_i)\log(1 - f_\theta(v_{max}(f_\theta, \boldsymbol{x}_i))))}_{\text{Fairness Counterexample-guided loss}} \end{aligned} \tag{3.5.2}$$

where L_{BCE} is the classification loss defined in Equation 3.2.1 and M denotes samples in the training dataset. Note that $v_{max}(f_\theta, x_i)$ is obtained for each datapoint x_i by solving a MILP program with an objective that is defined in Equation 3.3.2 and all the constraints in Equations 3.2.2 and 3.3.1.

Algorithm 2 summarizes the counterexample-guided training. Note that our CE-Fair training incorporates data augmentation through counterexamples which can cause drift in the model quality. Our approach guards against this by recomputing counterexamples for each batch at every epoch (i.e., lines 7-9). This ensures that: i) an incorrect old counterexample does not burden the learning, and ii) learning incorporates multiple counterexamples from the training set at a time and so is less sensitive to any particular one. There are different heuristics that one could adopt to use counterexamples and encourage the learned function to become fairer. In Line 11, we allow the user to keep all the original samples in the batch (aka, *full batch*) to preserve accuracy. Contrary to this, we can only keep the original samples for which we have found a counterexample (aka, *CE batch*). The latter is effective in reducing discrimination, specifically when the counterexample-guided training is initiated with an optimal f_θ. This introduces a tradeoff as to what portion of the original samples to keep. In our empirical evaluation, we only evaluate based on *full batch* and *CE batch* for simplification.

Algorithm 2 Counterexample-guided Fairness Enforced Training

```
Input: $f_\theta$, $\boldsymbol{D}$, $\rho$
Output: CE-Fair Fine-tuned model: $f_\theta$
for $epoch \in \{1 \dots e\}$ do
    for all $batch \in$ shuffled($\boldsymbol{D}$) do
        $sampled\_batch \leftarrow$ RandomSample($batch, \rho$)
        for all $(\boldsymbol{x}, y) \in sampled\_batch$ do
            $\phi_N \leftarrow$ ModelMIPEncoding($f_\theta$) {Constraints in Equation 3.2.2}
            $\phi_{CE} \leftarrow$ FairCEEncoding($\phi_N, \boldsymbol{x}, f_\theta(\boldsymbol{x})$) {Constraints in Equation 3.3.1}
            $x'_{max} \leftarrow$ FindMaxViolationCE($\phi_N, \phi_{CE}$) {Objective as in Equation 3.5.1}
            if $x'_{max}$ exists then
                Append($sampled\_batch, (x'_{max}, y)$)
            end if
        end for
        $\theta \leftarrow$ OptimizationStep($f_\theta, sampled\_batch$)
    end for
end for
```

Practical Considerations. We highlight the fact that the search for counterexamples is an expensive process and the run time grows with the size of the D, the number of epochs e, and the dimensionality of sensitive features. To make our approach scalable, we introduce a hyperparameter ρ that indicates what portion of the dataset we are taking. In Section 3.5.1, we show that even a small value of ρ, as small as 1%, is effective to fine-tune the model to become fairer in only a few epochs.

Multi-objective Model Selection. In fairness-enforced re-training, we are concerned with accuracy and unfairness at the same time. We thus opt for choosing a *Pareto frontier* by selecting the epoch whose accuracy and unfairness, when seen as a point in the $2D$ space, have the minimum ℓ_2 distance to the point corresponding to maximum accuracy and minimum unfairness.

FETA Extension. Extending CE-Fair prediction and training to multi-class classification (one-vs-rest approach) is straightforward. To extend these approaches to regression, e.g., to predict credit limit, we need to modify Equation 3.3.1b to encode fairness counterexamples as $|f_\theta(\boldsymbol{x}) - f_\theta(\boldsymbol{x}')| > \epsilon$ where ϵ is a hyperparameter that needs to be defined based on the context. Also, we need to consider an appropriate loss function, e.g., ℓ_2 loss for regression, to include counterexamples in the training. Note that our neural network MILP encoding is not limited to ReLU and can encode any piece-wise linear activation function.

3.5.1. Empirical Evaluation of CE-Fair Training

In this section, we evaluate the learning algorithm both on its own and in conjunction with the prediction technique. We use the same datasets and hardware as in Section 3.4.1.

3.5.1.1. Experimental Setup. . Most of the experimental details are the same as the ones in Section 3.4.1. Here, we only outline the details specific to CE-Fair Training. As discussed earlier, for CE-Fair training, we have $\rho = 100\%, 1\%, 2\%$ for German, IPUMS Adult, and Law School, respectively. However, for the final evaluation on the test set (i.e., all the results in the paper), models have been evaluated on the full test set for all datasets.

We fine-tune the pre-trained $\mathbf{NN_b}$ model through CE-Fair Training with the same learning rate and batch size used for $\mathbf{NN_b}$. Each model is CE-Fair trained for 50 epochs with mentioned ρ. Finally, we take the Nadir point of perfect *accuracy* and perfect *CE rate* and take the model from the epoch with minimum ℓ_2 distance to this Nadir point w.r.t. its train metrics.

It is worth mentioning that for CE-Fair Training, the numerical sensitive attributes (like "age") are considered continuous to support a larger space of counterexamples for better regularization.

Q4: Does our CE-Fair training algorithm make the original unfair model fairer? In this experiment, we re-train $\mathrm{NN_b}$ model with $\rho = 100\%, 1\%, 2\%$ for German, IPUMS Adult, and Law School, respectively. We train for 50 epochs and select the best model based on the *Pareto frontier* discussed in Section 3.5. Figure 3.6 summarizes the learning curves. We observe that while loss oscillates due to the fairness-performance tradeoff, the average of maximum violation substantially decreases. The results presented here are chosen among the full batch and CE batch options. CE batch decreases average violation almost to its minimum only in the first few epochs. This is because it is focusing only on the counterexamples while FULL BATCH experiences a more smooth curve.

Table 3.2. The effect of CE-Fair re-training on fairness metrics – 5-fold CV results

Dataset	**Approach**	**Accuracy**	**Flip Rate**	**CE Rate**
German	NN_b	76.70 ± 2.78	8.90 ± 0.73	29.80 ± 2.15
	CE-Fair Training	75.70 ± 3.77	3.40 ± 0.73	17.70 ± 4.67
IPUMS Adult	NN_b	81.57 ± 0.47	24.39 ± 1.52	89.40 ± 1.80
	CE-Fair Training	80.03 ± 0.57	2.80 ± 0.67	15.34 ± 2.53
Law School	NN_b	82.72 ± 0.19	17.53 ± 0.59	41.18 ± 2.05
	CE-Fair Training	84.99 ± 0.28	5.06 ± 0.97	7.64 ± 0.89

To quantify if the function is fairer, we compare two fairness metrics FLIP RATE and CE RATE defined in Section 3.4.1. As shown in Table 3.2, counterexample-guided retraining leads to better fairness metrics on all datasets by reducing the number of fairness violations. In fact, in the IPUMS Adult dataset, we see the highest decrease of 74% w.r.t. CE RATE. These results indicate the usefulness of CE-FAIR learning to make the original unfair model fairer. Further, the drop in accuracy when enforcing fairness is negligible when compared to the original model; In fact, we observe an increase in accuracy for Law School. While CE-FAIR training significantly reduces fairness violations, it does not guarantee fair predictions for all points in the input domain. This motivates the need for using CE-FAIR predictions in conjunction with the counterexample-guided learning algorithm, to guarantee fair predictions.

Table 3.3. Comparison of applying CE-FAIR Prediction on $\mathbf{NN_b}$ vs. on the CE-FAIR re-trained model – 5-fold CV results

Dataset	**Approach**	**Accuracy**	**Inference Time (s)**
German	CE-Fair Prediction	74.20 ± 3.50	0.45 ± 0.06
	CE-Fair Training+ CE-Fair Prediction	75.30 ± 3.65	0.39 ± 0.03
IPUMS Adult	CE-Fair Prediction	73.23 ± 0.93	119.64 ± 33.45
	CE-Fair Training+ CE-Fair Prediction	79.46 ± 0.71	10.20 ± 2.90
Law School	CE-Fair Prediction	74.36 ± 1.95	0.39 ± 0.14
	CE-Fair Training+ CE-Fair Prediction	84.14 ± 0.97	0.30 ± 0.13

Q5: Does counterexample-guided learning improve the quality of the guaranteed prediction model? As shown in Table 3.3, by additionally enforcing fairness constraints through counterexample-guided re-training, we improve both accuracy and inference time of CE-FAIR predictions. Running CE-FAIR predictions on the re-trained model improves inference runtime significantly on IPUMS Adult which has the largest sensitive feature space. The maximum drop in accuracy compared to NN_b is only 1.5%. Whereas, running CE-FAIR predictions directly on NN_b leads to a maximum accuracy loss of 8.3%.

Thus, with CE-Fair training, we get both a fairness guarantee and better runtime and model performance.

Q6: How does FETA perform compared to fairness under unaware model?

We train a model that is *unaware* of the sensitive features. Such a model would satisfy fairness definition 11 as its decision is not prone to changes in the sensitive attributes. We call this model the *blind* model.

Table 3.4. Accuracy of different fair models compared – 5-fold CV

Dataset	**Blind Model**	**CE-Fair Training + Prediction**
German	70.30 ± 2.20	75.30 ± 3.65
IPUMS Adult	78.90 ± 0.15	79.46 ± 0.71
Law School	74.19 ± 1.20	84.14 ± 0.97

In Table 3.4, we compare the blind model with the CE-Fair Training + Prediction models. We train the blind model for the same amount of epochs and the same ρ (ratio) as CE-Fair Training for a fair comparison. Note that the blind model is fair by design but similar to CE-Fair Training, it requires re-training the model. We observe that CE-Fair Training + Prediction, aka, the FETA pipeline, gives consistently better accuracy, up to 10% better compared to the blind model, while providing similar fairness guarantees.

Q7: How does FETA compare against existing work?

The literature on fairness in machine learning contains several well-established notions, particularly group fairness notions like demographic parity, equalized odds, and equal opportunity [**32**]. However, as previously demonstrated [**9**], it is not straightforward to achieve both group fairness and individual fairness in a single model. To compare our framework with group fairness mitigation techniques, an extension to group fairness notions would be required, but that falls outside the scope of our current work. Table 3.5 reports the accuracy and CE Rate of FETA compared to a recent method called *LCIFR* [**69**] that mitigates individual fairness using the same fairness definition as ours (i.e., Definition 11). To the best of our knowledge, *LCIFR* is the closest related work to ours that includes the same fairness notion. In *LCIFR* [**69**], the authors propose a fair representation learning approach and adversarial classification to address individual fairness. We fine-tune both methods on all datasets and extend *LCIFR* to accept multiple sensitive attributes similar to our setting. To gather the results of this section, we extend the sensitive features of *LCIFR* and use our own train/test split, we also report the empirical results using the original implementation in the Appendix.We set $\gamma = 1.0$ (their loss balancing factor) for the sake of fair comparison. The results in Table 3.5 indicate that the accuracy of FETA outperforms *LCIFR* on German and Law School, and only for IPUMS Adult, *LCIFR* trains a more accurate model. This finding is noteworthy because *LCIFR* combines both pre-processing and in-processing techniques to enhance performance while our approach focuses on in-processing and post-processing techniques. It is worth saying that our approach guarantees to have no CE rate for all three datasets while *LCIFR* cannot provide any guarantees, i.e., *LCIFR* results are empirical and even if they exhibit close to zero unfairness on the test set, it does not imply guaranteed fairness over all the data points in the input space. Also, we use a combination of categorical and continious sensitive features while for *LCIFR* we only use the categorical ones due to their design. Note that using continuous sensitive features results in a larger counterexample space.

Table 3.5. Comparison of FETA and LCIFR

Dataset	**Approach**	**Accuracy (%)**	**CE Rate (%)**
German	NN_b	76.70 ± 2.78	29.80 ± 2.15
	LCIFR [**69**]	72.30 ± 1.43	0.20 ± 0.24
	FETA(CE-Fair Train. + Pred.)	75.30 ± 3.65	0.0 ± 0.0
IPUMS Adult	NN_b	81.57 ± 0.47	89.40 ± 1.80
	LCIFR [**69**]	81.52 ± 0.34	0.05 ± 0.04
	FETA(CE-Fair Train. + Pred.)	79.46 ± 0.71	0.0 ± 0.0
Law School	NN_b	82.72 ± 0.19	41.18 ± 2.05
	LCIFR [**69**]	74.13 ± 0.76	0.008 ± 0.007
	FETA(CE-Fair Train. + Pred.)	84.14 ± 0.97	0.0 ± 0.0

3.6. Extensions to FETA

We would like to highlight the fact that any extensions to FETA that are expressible as MILP constraints are straightforward. The goal of this work is to present a working pipeline of FETA that results in a provable post-processed model with high utility and guaranteed fairness. Within this framework, one can take the liberty to leverage the flexibility of MILP for straightforward extensions to FETA in different directions, as highlighted by the examples below.

- **Model architecture**. While our study focuses on ReLU NNs, it is feasible to extend our approach to other piece-wise linear activation functions such as Leaky ReLU and the PReLU (Parametric ReLU). Moreover, there are several non-linear activation functions that are not piece-wise linear, but they can be approximated using piece-wise linear functions. To give an example, the sigmoid function is a smooth, S-shaped curve that can be approximated using piece-wise linear functions [**7**]. Similarly, the hyperbolic tangent (tanh) function, which is another commonly used activation function in neural networks, can be approximated using piece-wise linear functions [**70**]. Finally, the softmax function which is often used as the activation function in the output layer of neural networks for multi-class classification tasks, can be approximated using quantiles. In this method, the output range of the Softmax function is divided into several intervals or quantiles. Each interval is then approximated by a linear function that passes through two points: the lower and upper bound of the interval. Although approximating non-linear activation functions with piece-wise linear functions can simplify the computations involved in neural network training and inference, such approximations would result in losing the post-processing guarantees presented in this work. Finally, extension to more diverse architectures that are suitable for non-tabular data, including max pooling or batch normalization layers, is also theoretically straightforward, however, it would incur extra computational costs.

- **Fairness notion and similarity measure**. One will face many options to define similarity for individual fairness. Examples include an ℓ_1 distance with a threshold ϵ or even a separate NN trained to output a similarity; both may be encoded as MILP constraints.
  In this paper, we evaluate FETA using the casual discrimination notion of individual fairness as defined in [**25**] (see Definition 11). However, extending FETA to leverage other notions of individual fairness is feasible. A simple extension to our fairness notion is ϵ-individual fairness which indicates that for any two individuals x and x' whose feature vectors differ by at most ϵ, their model outputs should be the same. This can be encoded as a piece-wise linear constraint using the ℓ_1 distance with a threshold ϵ, which can be encoded easily as MILP constraints.
  Another plausible extension is to consider fairness notions that are capable of capturing relations among features. A notable example is counterfactual fairness [**48**] which is a causal fairness measure. This notion requires that the model's predictions should not depend on an individual's protected attributes (e.g., race or gender) except through the attributes that are causally related to the outcome. Counterfactuals are often defined within Pearl's Structural Causal Model (SCM) framework [**62**] to capture relations among features. This framework defines a causal model by a set of so called *structural equations*. If one defines the *structural equations* in a piece-wise linear format (see [**49**]), counterfactual fairness metric can be easily encoded in FETA. However, it is important to acknowledge that constructing an SCM for counterfactual fairness requires carefully identification of relevant variables and their causal relationships, precise specification of functional relationships, and simulation of various scenarios to evaluate fairness. This is a complex undertaking that demands specialized expertise.
  Finally, another approach is to train a separate neural network on a population data to output a similarity score between two individuals given the features in the input space of the model. The output of the neural network can then be discretized into several

intervals, and a separate linear function can be used to approximate the similarity within each interval, which can then encoded as MILP constraints.

- **Expert constraints**. Another way to expand FETA is by incorporating constraints that capture domain knowledge. For example, constraints can be defined on how features are permitted to change when searching for a counterexample. As an example, in FETA we make sure that a categorical feature H with k values remains as 1-hot encoding by imposing constraints such as $\sum_{i=1}^{k} h_i = 1$. Or in the credit card scenario, a constraint can be added to ensure that *credit history* is always less than *age*.

3.7. Related Work

Our paper is a contribution to the extensive literature on fairness in machine learning. In this section, we will contextualize our work and compare it to existing techniques for mitigating and verifying fairness in machine learning.

Bias Mitigation Algorithms. Methods that seek to introduce fairness into machine learning systems broadly fall into one of three categories: *pre-processing*, *in-processing*, and *post-processing.* The *pre-processing* approaches try to transform training data [**13, 41**] to mitigate bias. E.g., the most related work to ours is a fair representation learning technique by [**69, 50**] that uses the same fairness notion as we used in our paper (see Definition 11) to learn a fair representation of the individuals and train a certified machine learning model that accepts the fair representation as input. *In-processing* techniques mitigate discrimination via model regularization by directly modifying the learning algorithm to meet the fairness criteria. The regularization implicitly or explicitly optimizes a fairness metric. Several in-processing techniques have been proposed for group-fairness metrics [**33, 44, 56**]. Further, in-processing techniques for individual fairness fall into two categories based on whether they enforce fairness with or without access to individual fairness metrics. The first category of work circumvents the need for a fair metric by assuming the learner has access to an oracle that provides feedback on violations of individual fairness [**26, 40**]. The second category of work enforces fairness by assuming access to a fairness metric and using adversarial learning techniques to enforce fairness [**82, 83**]. *Post-processing* techniques modify the model's prediction during the inference time to make sure that the prediction distribution approaches a specific fairness metric [**32, 64, 53, 63, 2**].

In this paper, we propose both in-processing and post-processing techniques for individual fairness. Unlike prior work, our proposed in-processing technique does not depend on oracles for feedback on violations [**26, 40**] and the fair training algorithm uses counterexamples rather than adversarial training [**82, 83**] to enforce fairness. Further, adversarial training is approximate adversary examples, whereas we find exact counterexamples. While we are similar to [**45, 8**] in terms of in-processing, we propose a post-processing approach that, unlike others, guarantees individual fairness. Moreover, our in-processing technique is only a help to improve the quality of the ultimate post-processed model with guarantees. While most of the existing work on post-processing focus on group fairness or other definitions of individual fairness [**32, 64, 53, 63, 2**], we propose guaranteed predictions via post-processing using the individual fairness definition from [**25**]. Although Post-processing by [**53**] employs the same definition of individual fairness as our method, it only supports a single binary sensitive attribute. Moreover, their method works by training a new model on the predictions of the original model, with the goal of transforming the predictions to be more fair. The new model is trained to optimize a fairness objective, such as minimizing the distance between the predicted outcomes of similar individuals with different sensitive attributes. In contrast, our approach can handle multiple binary/continuous sensitive attributes and do not rely on a trained model to generalize at the prediction time and provide formal provable guarantees of fair predictions for all points in the domain. This makes our approach more versatile and robust, with the ability to accommodate a wider range of real-world scenarios while ensuring fairness.

Verifying machine learning systems and Adversarial Learning. Prior work on machine learning verification can be classified into (i) verification using satisfiability modulo theory (SMT) or mixed-integer linear programming (MILP) [**43, 76**], and (ii) verification using convex relaxations [**71, 19**]. Our proposed approach uses the MILP encoding from prior work [**57**] to build a practical individual fairness verification approach. Further, prior works on fairness verification have been proposed in the context of probabilistic programs [**3, 4**] or linear kernels [**38**]. Recent works propose adversarially robust algorithms which can be divided into empirical [**47, 54**] and certified defenses [**79, 72, 66, 34**]. Specifically [**80, 58, 83**] propose adversarial training-based algorithms for fairness. We are closely related to these works, in that we carry out adversarial training using counterexamples. However, we differ in two ways. First, to the best of our knowledge, there is no related work in the adversarial robustness literature for ensuring individual fairness using the definition from [**25**]. Second, related work in adversarial training only ensures correctness in the neighborhood of a training point, while we globally search for a counterexample and are able to discover long-range fairness violations.

3.8. Conclusion & Future Directions

In this work, we propose 1) a counterexample-guided fairness verification framework, 2) a counterexample-guided approach to guarantee fairness as a post-processing approach without intervening in the model, 3) a counterexample-guided approach to adapt an already trained model toward being fair, which cannot guarantee fairness on its own but can be combined with the former approach to provide better accuracy and faster inference, 4) An open-source tool called FETA that facilitates the integration of multiple techniques for optimal results. We showed using real-world datasets that in practice we can have efficient and fair models with little damage to accuracy. While the results of our approaches are promising, we note that the causal discrimination fairness notion adopted in this work is limited; it is important to recognize that features may be interrelated in certain domains and that our concept of fairness has limitations. However, we must also stress the complexity of achieving individual fairness in both the training and post-processing stages. It should be noted that ensuring individual fairness means ensuring that the model does not discriminate between any two similar pairs of individuals in the outcome space. This is distinct from placing constraints on the parameters of the trained model during the training process. An interesting future work would be to extend FETA with other fairness notions that are capable of capturing the relations among features. A notable example is counterfactual fairness [**48**] which is a causal fairness measure based on SCM (structural causal model). Another future avenue to explore is to bind the counterexamples to follow the distribution of the data. One can extend our framework by adding distribution constraints to our MILP formulation to restrict counterexamples to be Out-of-Distribution. Another limitation of FETAleft for future work which is typical of approaches where such guarantees are provided, is scalability. Every year, state-of-the-art neural networks grow in size with a large number of parameters which poses incredible challenges for constraint-based verification approaches. Although the neural network model used in our experiments is fairly complex for tabular data, this approach might not scale to very deep networks. Indeed, it would be an interesting direction to address and explore the limits of scaling the model architecture. For example, future work could study how to modify the neural network learning algorithms to enable scalable constraint-based analysis. Finally, solvers tend to use floating-point approximations leading to numeric instabilities. Problem-specific solutions to make the approach more numerically stable could be a potential future work as well.

3.9. Societal Impact

In this paper, we propose three approaches to enhance the fairness of neural network models by fairness verification, fair training, and fair prediction. The impact of using these approaches to have fair neural network models on society is considerable as they prevent these models from incurring unfair biases or discrimination against individuals. Having access to fair models can enhance the accessibility of resources, opportunities, and services for historically marginalized people, such as people of color, women, people with disabilities, and low-income individuals.

In addition, individually fair neural network models can enhance trust and transparency in decision-making systems, particularly in critical areas such as criminal justice, hiring, and lending, where biased or unfair decisions can severely impact individuals.

Moreover, fair neural network models can help promote diversity and inclusion. By recognizing the importance of fairness and diversity in their models, companies can attract a more diverse group of users and employees, which can lead to better products and services for everyone.

However, while fair neural network models have the potential to promote fairness and reduce discrimination, it is important to carefully consider their potential drawbacks and limitations. In this paper, we focus on a specific fairness notion to verify, train and guarantee fair prediction of neural network models. However, we acknowledge that there is a huge literature on various notions of fairness, and individual fairness is context-dependent and should be defined relative to a task. Hence, our framework cannot and should not be used in every application domain.

Furthermore, there is a risk that fair neural network models could be misused or misinterpreted. One potential disadvantage of fair neural network models is that they may not always be able to achieve perfect fairness. It is worth mentioning that many aspects of fairness are not captured by mathematical measures. Our framework is highly dependent on a fairness notion, and the result change by changing the notion of fairness. Although individual fairness can be defined in different ways in FETA as explained in Section 3.6, it is necessary to choose only one definition for deployment; various individual fairness metrics may conflict with each other, making it difficult to optimize for all of them simultaneously.

Finally, although our approach can produce fair predictions, it is still based on a model produced by a machine learning algorithm. And it is important to note that FETA could suffer from the same disadvantages as the original model in aspects that we did not consider in this work, such as privacy, explanation, safety, security, and robustness. Hence, the user must be aware of such a system's limitations, especially when using these models to replace people in decision-making.

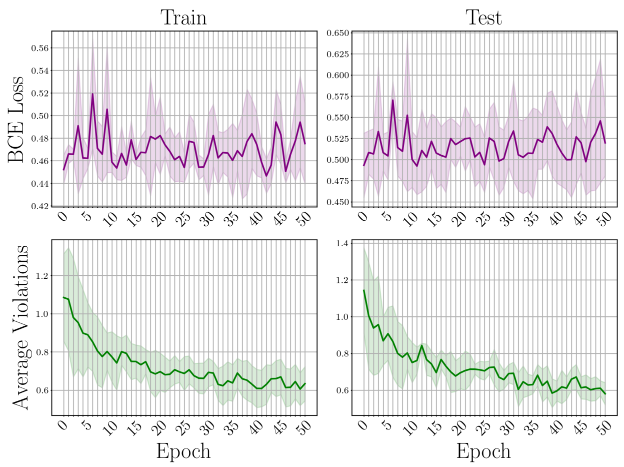

Fig. 3.3. German (FULL BATCH).

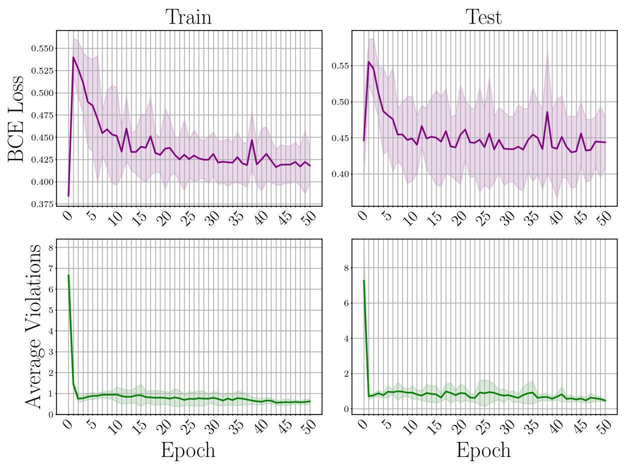

Fig. 3.4. IPUMS Adult (CE BATCH).

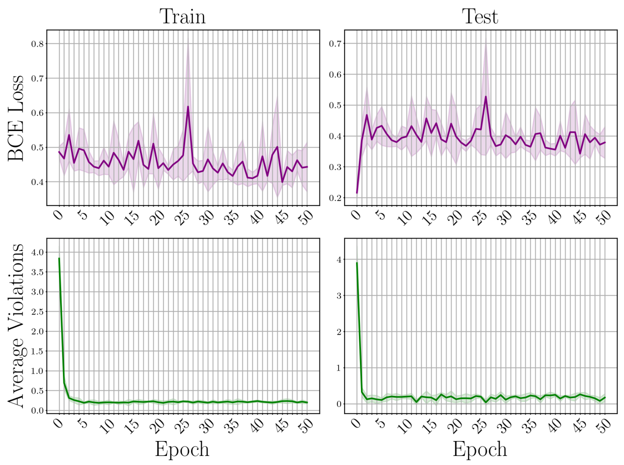

Fig. 3.5. Law School (CE BATCH).

Fig. 3.6. Loss and average violation curves for fairness-enforced training on different datasets. Each curve is the mean over 5 runs and the shaded area represents the std.

3.10. Additional Experiments

3.10.1. Comparison to the original *LCIFR* [69]

Here we report the results that are imported from the *LCIFR* [**69**] paper with their original implementation. Our overlapping datasets are German and Law School so here we only report those two.

Table 3.6 demonstrates these results. The results of *LCIFR* were obtained from their corresponding paper. We can see that at the same level of accuracy, we can guarantee fairness while *LCIFR* can only empirically demonstrate fairness. For the Law School dataset, the empirical unfairness for *LCIFR* is significantly large. Note in this experiment, we use more sensitive features for the German dataset which results in a more complex counterexample space.

Table 3.6. Comparison of FETA and LCIFR (original implementation)

Dataset	**Approach**	**Accuracy (%)**	**CE Rate (%)**
German	NN_b	76.70 ± 2.78	29.80 ± 2.15
	LCIFR [**69**]	75.53	0.0
	FETA(CE-Fair Train. + Pred.)	75.30 ± 3.65	0.0 ± 0.0
Law School	NN_b	82.72 ± 0.19	41.18 ± 2.05
	LCIFR [**69**]	84.4	48.9
	FETA(CE-Fair Train. + Pred.)	84.14 ± 0.97	0.0 ± 0.0

3.10.2. Counterexample-guided Counting vs. Naive Enumeration

We compare the performance of Counterexample-guided Counting for fair prediction (CE-Fair) to a naive enumeration method using the IPUMS Adult dataset, which has the most extensive range of sensitive features.

Table 3.7. Comparison of CE-guided Counting and Naive Enumeration in runtime

Dataset	**Approach**	**Average (s)**	**Total (s)**
IPUMS Adult	Enumeration	71.50	655,655
	CE-guided Counting	9.15	83,905

Chapter 4

Conclusion and Future Work

This work has targeted the problem of fairness in machine learning. There have been numerous examples that deploying machine learning models has gone wrong and people have been discriminated against due to biases that emerged or were amplified in the model. The community has come up with group-based or individual-based notions of fairness to remove or mitigate the bias in machine learning models. In this work, we focused on individual fairness that requires similar individuals to be treated similarly. The *similarity*, however, is a very general term and needs to be defined. We have adopted the causal discrimination notion from [**25**] that defines two individuals to be similar iff they differ only in sensitive attributes. We proposed a fairness verification algorithm along with in-processing and post-processing mitigation techniques. Specifically, we proposed 1) a MILP-based verification framework that searches for individual fairness counterexamples, 2) a counterexample-guided post-processing algorithm that tweaks the model decisions by computing a majority vote to satisfy and guarantee individual fairness, 3) a counterexample-guided in-processing approach that incorporates individual fairness counterexamples to fine-tune an already trained model toward being fairer; although it cannot guarantee fairness on its own, it can be combined with the post-processing approach to satisfy the individual fairness notion provably, 4) A tool called FETA that has all the algorithms implemented. We have shown the effectiveness of our approach through extensive experiments on three real-world datasets; FETA can improve and/or guarantee individual fairness with little to no damage to accuracy and model performance. We have studied FETA in the context of the related work and have compared it against the closest related work, LCIFR [**69**]. We have shown that FETA either outperforms related work or proposes guarantees lacking from related work. While having shown very promising results, we acknowledge the limitations of FETA and address them in this chapter in order of priority.

The individual fairness notion used in FETA has its own limitation; its implicit assumption of the independence of features might not be realistic. In real-world scenarios, features are usually correlated and there is an inherent *causal* mechanism underlying the data. The causal discrimination notion, unlike its name, neglects this important aspect. As an extension to FETA which is currently under active development, we are studying more sophisticated individual fairness notions that do take into account the causal structure underlying the data. We are leveraging Pearl's Structural Causal Model (SCM) framework [**61**] to define a new causal similarity measure based on *counterfactual balls.* We are building on top of prior work [**17, 22**] from the *algorithmic recourse* literature that proposes similarity balls around data points, that, unlike canonical similarity balls like the common ℓ_2 ball in robustness literature, does not have the implicit independence assumption; rather, it uses a novel form of *causal* perturbation that considers the causal structure of the data into account while forming a similarity ball around a sample.

More specifically, *hard* interventions are replaced by *soft* interventions in the Additive Noise Model (ANM) that allow for interventions in an additive form while maintaining causal relationships. Intuitively, in soft interventions, a vector may be added to a subset of variables without breaking causal connections, to study the downstream causal effect of that perturbation vector. Ultimately, this would result in the translation of, e.g., an ℓ_2 perturbation ball into a counterfactual perturbation ball. The main challenge, however, would be to handle the perturbation of the discrete variables in the SCM. For that purpose, we either use hard interventions for discrete variables and soft interventions for continuous variables or try to define the perturbation ball in the latent space, i.e., over exogenous variables of the SCM in the additive noise model. Once the theoretical properties of this similarity ball are established, we would have an individual fairness notion defined on top of this similarity measure and FETA can be used to enforce it.

Another potential future direction for FETA would be to address the scalability issue. FETA is providing provable guarantees for individual fairness in ReLU neural networks and that comes at a price. While the scale of NNs we have experimented with is more than sufficient for tabular datasets, FETA might suffer from runtime issues if applied to very deep networks. The scalability of neural network verifiers is an active line of research in the NN verification community. That being said, problem-specific remedies can be applied to boost performance. For example, since the underlying optimization engine uses Branch-and-Bound techniques, we can develop tighter bounds on the variables of the optimization problem to further limit the search space.

It is worth mentioning that there are numerous directions in which extensions to FETA are straightforward. For example, as for model architecture, neural networks that incorporate any arbitrary pice-wise linear activation functions can be encoded (e.g., max-pooling). Moreover, different expert constraints can be encoded in the form of MILP constraints; for instance, one can encode constraints limiting the range of change of age or salary for more realistic counterexamples.

Last but not least, we would like to acknowledge the complexity of the choice of a fairness notion in general. While FETA provides a practical tool that can ultimately mitigate bias in ML-powered decision-making scenarios, it is of utmost importance to be aware of the challenges of choosing the right fairness notion. FETA promises bias mitigation only with respect to the fairness notion studied in this work. There is a huge literature on various notions of fairness, and individual fairness is very context-dependent and should be defined relative to the task at hand. That being said, any individual fairness notion that can be encoded as MILP constraints would be supported by FETA and that also can be considered a potential future direction.

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[2] Aniya Agarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. Automated test generation to detect individual discrimination in ai models. *arXiv preprint arXiv:1809.03260*, 2018.

[3] Aws Albarghouthi, Loris D'Antoni, Samuel Drews, and Aditya V Nori. Fairsquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):1–30, 2017.

[4] Osbert Bastani, Xin Zhang, and Armando Solar-Lezama. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–27, 2019.

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

[6] BBC. Apple's 'sexist' credit card investigated by US regulator, 2019.

[7] Valeriu Beiu, Jan A Peperstraete, Joos Vandewalle, and Rudy Lauwereins. Vlsi complexity reduction by piece-wise approximation of the sigmoid function. In *ESANN*. Citeseer, 1994.

[8] Elias Benussi, Andrea Patane', Matthew Wicker, Luca Laurenti, and Marta Kwiatkowska. Individual fairness guarantees for neural networks. In Lud De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 651–658, 7 2022. Main Track.

[9] Reuben Binns. On the apparent conflict between individual and group fairness. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 514–524, 2020.

[10] Sumon Biswas and Hridesh Rajan. Fairify: Fairness verification of neural networks, 2022.

[11] Rudy Bunel, Ilker Turkaslan, Philip H.S. Torr, Pushmeet Kohli, and M. Pawan Kumar. A unified view of piecewise linear neural network verification. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 4795–4804, Red Hook, NY, USA, 2018. Curran Associates Inc.

[12] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 77–91. PMLR, 23–24 Feb 2018.

[13] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. *Advances in neural information processing systems*, 30, 2017.

[14] YooJung Choi, Golnoosh Farnadi, Behrouz Babaki, and Guy Van den Broeck. Learning fair naive bayes classifiers by discovering and eliminating discrimination patterns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10077–10084, 2020.

[15] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In *International Conference on Computer Aided Verification*, pages 154–169. Springer, 2000.

[16] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6478–6490. Curran Associates, Inc., 2021.

[17] Ricardo Dominguez-Olmedo, Amir H Karimi, and Bernhard Schölkopf. On the adversarial robustness of causal algorithmic recourse. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5324–5342. PMLR, 17–23 Jul 2022.

[18] Marcelo Carvalho dos Santos, Victor Henrique Cabral Pinheiro, Filipe Santana Moreira do Desterro, Renato Koga de Avellar, Roberto Schirru, Andressa dos Santos Nicolau, and Alan Miranda Monteiro de Lima. Deep rectifier neural network applied to the accident identification problem in a pwr nuclear power plant. *Annals of Nuclear Energy*, 133:400–408, 2019.

[19] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *UAI*, volume 1, page 3, 2018.

[20] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[21] Rüdiger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. *CoRR*, abs/1705.01320, 2017.

[22] Ahmad-Reza Ehyaei, Amir-Hossein Karimi, Bernhard Schölkopf, and Setareh Maghsudi. Robustness implies fairness in causal algorithmic recourse, 2023.

[23] Michael Feldman, Sorelle Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact, 2015.

[24] Ian Foster, Rayid Ghani, Ron S. Jarmin, Frauke Kreuter, and Julia Lane. *Data Science Methods and Tools for Research and Practice.* `https://textbook.coleridgeinitiative.org/`.

[25] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 498–510, 2017.

[26] Stephen Gillen, Christopher Jung, Michael Kearns, and Aaron Roth. Online learning with an unknown fairness metric. *Advances in neural information processing systems*, 31, 2018.

[27] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[28] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[29] Vincent Grari, Boris Ruf, Sylvain Lamprier, and Marcin Detyniecki. Achieving fairness with decision trees: An adversarial approach. *Data Sci. Eng.*, 5(2):99–110, 2020.

[30] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition.* Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2008.

[31] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2020.

[32] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29:3315–3323, 2016.

[33] Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

[34] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2266–2276, 2017.

[35] T. Hickey, Q. Ju, and M. H. Van Emden. Interval arithmetic: From principles to implementation. *J. ACM*, 48(5):1038–1068, September 2001.

[36] Hans Hofmann. Statlog (german credit data) data set, 1994.

[37] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *International Conference on Computer Aided Verification*, pages 3–29. Springer, 2017.

[38] Philips George John, Deepak Vijaykeerthy, and Diptikalyan Saha. Verifying individual fairness in machine learning models. In *Conference on Uncertainty in Artificial Intelligence*, pages 749–758. PMLR, 2020.

[39] Surya Mattu Julia Angwin, Jeff Larson and Lauren Kirchner. Machine Bias, 2016.

[40] Christopher Jung, Michael J Kearns, Seth Neel, Aaron Roth, Logan Stapleton, and Zhiwei Steven Wu. Eliciting and enforcing subjective individual fairness. 2019.

[41] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems*, 33(1):1–33, 2012.

[42] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 643–650. IEEE, 2011.

[43] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[44] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR, 2018.

[45] Haitham Khedr and Yasser Shoukry. Certifair: A framework for certified global fairness of neural networks, 2022.

[46] R Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. 12 1996.

[47] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[48] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *Advances in neural information processing systems*, 30, 2017.

[49] Matt J Kusner, Chris Russell, Joshua R Loftus, and Ricardo Silva. Causal interventions for fairness. *arXiv preprint arXiv:1806.02380*, 2018.

[50] Preethi Lahoti, Krishna P Gummadi, and Gerhard Weikum. ifair: Learning individually fair data representations for algorithmic decision making. In *2019 ieee 35th international conference on data engineering (icde)*, pages 1334–1345. IEEE, 2019.

[51] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. https://github.com/propublica/compas-analysis, 2016.

[52] Changliu Liu, Tomer Arnon, Christopher Lazarus, Clark W. Barrett, and Mykel J. Kochenderfer. Algorithms for verifying deep neural networks. *CoRR*, abs/1903.06758, 2019.

[53] Pranay K Lohia, Karthikeyan Natesan Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R Varshney, and Ruchir Puri. Bias mitigation post-processing for individual and group fairness. In *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 2847–2851. IEEE, 2019.

[54] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[55] Sagnik Maity and Arya Mazumdar. Statistical inference for individual fairness, 2021.

[56] Debmalya Mandal, Samuel Deng, Suman Jana, Jeannette Wing, and Daniel J Hsu. Ensuring fairness beyond the training data. *Advances in neural information processing systems*, 33:18445–18456, 2020.

[57] Kiarash Mohammadi, Amir-Hossein Karimi, Gilles Barthe, and Isabel Valera. Scaling guarantees for nearest counterfactual explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '21, page 177–187, New York, NY, USA, 2021. Association for Computing Machinery.

[58] Vedant Nanda, Samuel Dooley, Sahil Singla, Soheil Feizi, and John P Dickerson. Fairness through robustness: Investigating robustness disparity in deep learning. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 466–477, 2021.

[59] Department of Justice of the United States. Proving discrimination – disparate impact.

[60] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[61] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2009.

[62] Judea Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19(2), 2000.

[63] Felix Petersen, Debarghya Mukherjee, Yuekai Sun, and Mikhail Yurochkin. Post-processing for individual fairness. *Advances in Neural Information Processing Systems*, 34, 2021.

[64] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. *Advances in neural information processing systems*, 30, 2017.

[65] ProPublica. There's software used across the country to predict future criminals. and it's biased against blacks., 2016.

[66] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *In International Conference on Learning Representations*, 2018.

[67] Reuters. Amazon scraps secret ai recruiting tool that showed bias against women, 2018.

[68] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[69] Anian Ruoss, Mislav Balunović, Marc Fischer, and Martin Vechev. Learning certified individually fair representations. *arXiv preprint arXiv:2002.10312*, 2020.

[70] Hasan Sildir and Erdal Aydin. A mixed-integer linear programming based training and feature selection method for artificial neural networks using piece-wise linear approximations. *Chemical Engineering Science*, 249:117273, 2022.

[71] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–30, 2019.

[72] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. *In International Conference on Learning Representations*, 2018.

[73] Aishwarya Sivaraman, Golnoosh Farnadi, Todd Millstein, and Guy Van den Broeck. Counterexample-guided learning of monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:11936–11948, 2020.

[74] Armando Solar-Lezama, Liviu Tancau, Rastislav Bodik, Sanjit Seshia, and Vijay Saraswat. Combinatorial sketching for finite programs. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, pages 404–415, 2006.

[75] Vincent Tjeng and Russ Tedrake. Verifying neural networks with mixed integer programming. *CoRR*, abs/1711.07356, 2017.

[76] Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

[77] Caterina Urban, Maria Christakis, Valentin Wüstholz, and Fuyuan Zhang. Perfectly parallel fairness certification of neural networks. *Proceedings of the ACM on Programming Languages*, 4(OOPSLA):1–30, 2020.

[78] L. F. Wightman. Lsac national longitudinal bar passage study. lsac research report series., 1998.

[79] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

[80] Han Xu, Xiaorui Liu, Yaxin Li, Anil Jain, and Jiliang Tang. To be robust or to be fair: Towards fairness in adversarial training. In *International Conference on Machine Learning*, pages 11492–11501. PMLR, 2021.

[81] Lie Xu, Chiu-sing Choy, and Yi-Wen Li. Deep sparse rectifier neural networks for speech denoising. In *2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 1–5. IEEE, 2016.

[82] Mikhail Yurochkin, Amanda Bower, and Yuekai Sun. Training individually fair ml models with sensitive subspace robustness. *arXiv preprint arXiv:1907.00020*, 2019.

[83] Mikhail Yurochkin and Yuekai Sun. Sensei: Sensitive set invariance for enforcing individual fairness. *arXiv preprint arXiv:2006.14168*, 2020.

[84] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*, pages 1171–1180, 2017.

[85] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 325–333, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.